On the Optimal Source Key Size of Secure Gradient Coding

Yang Zhou*, Wenbo Huang*, Kai Wan*, Robert Caiming Qiu*
*Huazhong University of Science and Technology, 430074 Wuhan, China,
{hust_zhou, eric_huang, kai_wan, caiming}@hust.edu.cn

***Abstract*—With gradient coding, a user node can efficiently aggregate gradients from server nodes processing local datasets, achieving low communication costs and maintaining resilience against straggling servers. This paper considers secure gradient coding problem, where a user aims to compute the sum of the gradients from K datasets with the assistance of N distributed servers. The user should recover the sum of gradients by the transmissions from any $\mathsf{N_r}$ servers, where each dataset is assigned to $\mathsf{N} - \mathsf{N_r} + \mathsf{m}$ datasets. The security constraint guarantees that even if the user receives the transmission messages from all servers, it cannot obtain any other information about the datasets beyond the sum of gradients. It was shown in the literature that the security constraint will not increase the optimal communication cost of the gradient coding problem, if enough source keys are shared among the servers. However, the minimum required source key size to guarantee the security while achieving this optimal communication cost has only been studied for the case $\mathsf{m} = 1$. In this paper, we focus on the more general case $\mathsf{m} \geq 1$, and aim to characterize the minimum required source key size for the above purpose. A new information-theoretic converse bound on the source key size and a new achievable scheme with smartly designed are proposed. Our proposed scheme outperforms the optimal scheme with the widely-used cyclic data assignment and coincides with the converse bound under some system parameters.**

I. Introduction

Secure aggregation is a well-established topic in distributed computation, referring to the multiparty computation problem where the task is to compute a multiparty sum while no party reveals its update in the clear, not even to the aggregator [1]. The concept of secret sharing was first introduced in [2], where a secret is divided into multiple shares, and only participants meeting a threshold can reconstruct the secret. The complexity of verifiable secret sharing and its extension to multiparty computation were further studied in [3]–[6].

In this paper, we consider the secure aggregation in gradient coding, called secure gradient coding. The gradient coding problem was first proposed in [7]. In a general gradient coding model, a user wants to compute a sum of gradients on K datasets by N servers. During the data assignment phase, each dataset is assigned to $\mathsf{M} = \mathsf{N} - \mathsf{N_r} + \mathsf{m}$ servers. During the computing phase, each server first computes gradients of the assgined datasets and then sends a coded message on these gradients to the user. During the decoding phase, the user should recover the computation task result using the transmissions from any $\mathsf{N_r}$ servers. The objective is to minimize the number of transmissions (i.e., communication cost). In [8], the optimal communication cost under linear encoding, was characterized. Furthermore, the authors in [9] considers heterogeneous (and arbitrarily fixed) data assignment, where each dataset may be assigned to different number of servers, and characterizes the optimal communication cost under linear encoding.

The secure aggregation model for gradient coding was first proposed in [10].[1] In this model, a key distribution server assigns a common shared key to all servers, in order to ensure security. It was proven in [10] that the security constraint will not increase the optimal communication cost of the gradient coding problem, if enough source keys are shared among the servers. Then for the case $\mathsf{m} = 1$ (i.e., when M is minimum to tolerate $\mathsf{N} - \mathsf{N_r}$ stragglers), an information-theoretic lower bound on the source key size and several secure distributed coded computation schemes for specific data distribution patterns (e.g., cyclic, repetitive, and mixed) were proposed in [10], while achieving the optimal communication cost with different requirement on the source key size.

Note that, recently, secure aggregation for federated learning has also gained significant attention [11]–[15]. In secure aggregation for federated learning, each distributed computing node holds its own local data to compute the gradient; thus secure aggregation against stragglers (or user dropouts) requires two-round transmissions. In contrast, in the gradient coding problem, an additional data assignment phase for servers exists, and because of the redundancy in data assignment, one-round transmission is enough for secure aggregation.

Main Contributions: In this paper, we aim to characterize the minimum source key size for the case $\mathsf{m} > 1$, while achieving the optimal communication cost under linear encoding. Our main contributions are as follows.

- For a given data assignment, we provide an information-theoretic converse bound on the source key size, while achieving the optimal communication cost under linear encoding.
- To reduce the source key size, we introduce a secure gradient coding scheme with a new data assignment strategy. This scheme outperforms the widely used cyclic assignment scheme and can achieve the converse bound under some system parameters.

Notation Convention: Calligraphic symbols denote sets, bold lower-case letters denote vector, bold upper-case letters denote matrices, and sans-serif symbols denote system

[1]The secure distributed linearly separable computation was formulated in [10]. When the user only requests one linear combination of the gradients (or input vectors), the problem reduces to secure gradient coding.

parameters. We use $|\cdot|$ to represent the cardinality of a set or the length of a vector; $[a:b] := \{a, a+1, \ldots, b\}$; $[n] := [1:n]$; $\mathbb{F}_{\mathsf{q}}$ represents a finite field with order q; $\mathbf{M}^{\mathsf{T}}$ and $\mathbf{M}^{-1}$ represent the transpose and the inverse of matrix $\mathbf{M}$, respectively; the matrix $[a;b]$ is written in a Matlab form, representing $[a,b]^{\mathsf{T}}$; $(\mathbf{M})_{m\times n}$ represents the dimension of matrix $\mathbf{M}$ is $m \times n$; $\text{Mod}(b,a)$ represents the modulo operation on b with integer divisor a and in this paper we let $\text{Mod}(b,a) \in \{1,\ldots,a\}$ (i.e., we let $\text{Mod}(b,a) = a$ if a divides b).

II. System Model

This section describes the $(\mathsf{K}, \mathsf{N}, \mathsf{N}_{\rm r}, \mathsf{m})$ secure gradient coding problem. We consider the framework that a user computes the sum of gradients on K independent datasets $D_1, \ldots, D_{\mathsf{K}}$ with the help of N servers. Compared to distributed computing without considering secure aggregation, there is another trusted server to distribute keys to the computing servers. It generates a set of random variables $\mathcal{Q} = \{Q_1, \ldots, Q_{\mathsf{N}}\}$ on $\mathbb{F}_{\mathsf{q}}$ indepndent of the datasets,

$$I(Q_1,\ldots,Q_{\mathsf{N}}; D_1,\ldots,D_{\mathsf{K}}) = 0. \tag{1}$$

The source key size η measures the total amount of randomness among all keys, i.e.,

$$\eta = H(Q_1,\ldots,Q_{\mathsf{N}})/\mathsf{L}. \tag{2}$$

The secure gradient coding framework comprises three phases, *data assignment, computing, and decoding*.

Data assignment phase. The data center assigns each dataset to $\mathsf{M} = \mathsf{N} - \mathsf{N}_{\rm r} + \mathsf{m}$ servers, where m is computation cost factor (which is also called communication reduction factor in [8]). The index set of datasets assigned to server $n \in [\mathsf{N}]$ is denoted as $\mathcal{Z}_n \in [\mathsf{K}]$. In addition, the trusted key server allocates the key $\mathcal{Q}_i$ to each server $i \in [\mathsf{N}]$.

Computing phase. Each server $n \in [\mathsf{N}]$ first computes the message $g_k = \nabla(D_k)$ for each $k \in \mathcal{Z}_n$. Then it sends the coded message to the user,

$$X_n = \psi_n\left(\{g_k : k \in \mathcal{Z}_n\}, Q_n\right), \tag{3}$$

where ψ_n is a function of messages $\{g_k : k \in \mathcal{Z}_n\}$,

$$\psi_n : \mathbb{F}_{\mathsf{q}}^{|\mathcal{Z}_n|\mathsf{L}} \times |\mathcal{Q}| \to \mathbb{F}_{\mathsf{q}}^{\mathsf{T}_n}, \tag{4}$$

and T_n represents the length of X_n. m is the computation cost factor because in a distributed system, the complexity of computing the linear combinations of the gradients is much lower than computing the gradients.

Decoding phase. We define $\mathsf{N}_{\rm r}$ as the number of minimum available servers, and the identity of the surviving servers is not known until the decoding phase. Thus the user should be able to use coded messages from any $\mathsf{N}_{\rm r}$ servers to decode the computation tasks. Thus, for each subset of servers $\mathcal{A} \subseteq [\mathsf{N}]$ where $|\mathcal{A}| = \mathsf{N}_{\rm r}$, by defining $X_{\mathcal{A}} := (X_n : n \in \mathcal{A})$, it should satisfy that

$$H(g_1 + \ldots + g_K | X_{\mathcal{A}}) = 0. \tag{5}$$

We define

$$\mathsf{R} := \max_{\mathcal{A}\subseteq[\mathsf{N}]:|\mathcal{A}|=\mathsf{N}_{\rm r}} \frac{\sum_{n\in\mathcal{A}} \mathsf{T}_n}{\mathsf{L}} \tag{6}$$

as the communication cost, which represents the maximum normalized number of symbols received from any $\mathsf{N}_{\rm r}$ servers to recover the computational task.

The secure aggregation protocol imposes that

$$I\left(g_1,\ldots,g_{\mathsf{K}}; X_{[\mathsf{N}]} | g_1 + \ldots + g_{\mathsf{K}}\right) = 0, \tag{7}$$

where $X_{[\mathsf{N}]}$ presents the whole messages from N servers which may be received by the user. This equation ensures that, except for the computation task, the user cannot obtain any information about the datasets.

If the source key size is large enough, the optimal communication cost under linear encoding (i.e., ψ_n is a linear function), can be characterized by directly combining [8], [10].[2]

Theorem 1 ([8], [10]). *For the $(\mathsf{K}, \mathsf{N}, \mathsf{N}_{\rm r}, \mathsf{m})$ secure gradient coding problem, the optimal communication cost under linear encoding is $\frac{\mathsf{N}_{\rm r}}{\mathsf{m}}$.*

Our main objective in this paper is to minimize the source key size η while maintaining the optimal communication cost under linear encoding for the case $\mathsf{m} \geq 1$.

III. Main Results

We first introduce a novel converse bound on η for a fixed assignment, whose proof can be found in Appendix A.

Theorem 2. *For the $(\mathsf{K}, \mathsf{N}, \mathsf{N}_{\rm r}, \mathsf{m})$ secure gradient coding problem, for a fixed assignment $\mathbf{Z} = (\mathcal{Z}_1, \ldots, \mathcal{Z}_{\mathsf{N}})$, if there exists an ordered set of servers in $[\mathsf{N}]$, denoted by $\mathsf{s} = (s_1, \ldots, s_{|\mathsf{s}|})$, where each server contains at least one dataset that appears in the datasets of its preceding servers at most $\mathsf{m}-1$ times, this can be expressed as:*

$$\forall i \in [|\mathsf{s}|], \exists x \in \mathcal{Z}_{s_i} \text{ such that } \sum_{j=1}^{i-1} \delta(x \in \mathcal{Z}_{s_j}) \leq \mathsf{m} - 1, \tag{8}$$

where $\delta(x \in \mathcal{Z}_{s_j})$ equals 1 if $x \in \mathcal{Z}_{s_j}$, and 0 otherwise.

Under the symmetric transmission and linear coding, it must hold that

$$\eta \geq \frac{|\mathsf{s}|}{\mathsf{m}} - 1. \tag{9}$$

Theorem 2 indicates that minimizing source key size η is a combinatorial problem that depends on the data assignment. Further, we can establish a min-max problem to characterize the minimum $\eta^\star$ as follows.

Corollary 1. *For the $(\mathsf{K}, \mathsf{N}, \mathsf{N}_{\rm r}, \mathsf{m})$ secure gradient coding problem, we have*

$$\eta^\star \geq \min_{\mathbf{Z}} \max_{\mathsf{s}\subseteq[\mathsf{N}],\ \textit{subject to } (8)} \frac{|\mathsf{s}|}{\mathsf{m}} - 1. \tag{10}$$

[2]Without security, the authors in [8] characterized the optimal communication cost under linear encoding is $\mathsf{N}_{\rm r}/\mathsf{m}$. In addition, [10, Theorem 1] showed that for any $(\mathsf{K}, \mathsf{N}, \mathsf{N}_{\rm r}, \mathsf{m})$ non-secure gradient coding problem, the coding scheme can be made secure without increasing the communication cost.

The essence of min-max problems lies in balancing extreme values across all chosen s.In order to simplify the computation, we provide a loosen converse bound on $\eta^\star$, whose proof could be found in Appendix A.

Corollary 2. *For the* $(\mathsf{K},\mathsf{N},\mathsf{N_r},\mathsf{m})$ *secure gradient coding problem, it must hold that*

$$\eta^\star \geq \frac{\left\lceil \frac{\mathsf{mN}}{\mathsf{N}-\mathsf{N_r}+\mathsf{m}} \right\rceil}{\mathsf{m}} - 1. \tag{11}$$

Next, we propose a novel secure gradient coding scheme that employs a new data assignment strategy, improving the commonly used cyclic assignment.

Theorem 3. *For the* $(\mathsf{K},\mathsf{N},\mathsf{N_r},\mathsf{m})$ *secure gradient coding problem, the source key size* $\eta = \frac{h(\mathsf{N},\mathsf{M})}{\mathsf{m}} - 1$ *is achievable, where the output of the function* $h(\cdot,\cdot)$ *is given by the recursive algorithm shown in Fig. 1 with the following properties:*

- *When* $\mathsf{N} > 2\mathsf{M}$*, by Scheme 1 described in Section IV-A we have*
$$h(\mathsf{N},\mathsf{M}) = h\big(\mathsf{N} - \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor \mathsf{M}, \mathsf{M}\big) + \mathsf{m} \lfloor \mathsf{N}/\mathsf{M} - 1 \rfloor . \tag{12}$$
- *When* $1.5\mathsf{M} \leq \mathsf{N} < 2\mathsf{M}$ *and* M *is even, with* $\mathsf{M} \geq 2\mathsf{m}$*, by Scheme 2 described in Section IV-B, we have*
$$h(\mathsf{N},\mathsf{M}) = h\left(\mathsf{N}-\mathsf{M}, \frac{\mathsf{M}}{2}\right) + \mathsf{m}. \tag{13}$$
- *When* $1.5\mathsf{M} \leq \mathsf{N} < 2\mathsf{M}$ *and* M *is odd, with* $\mathsf{M} \geq 2\mathsf{m}+1$*, by Scheme 3 described in Section IV-C we have*
$$h(\mathsf{N},\mathsf{M}) = \mathsf{N} - \frac{3\mathsf{M}-1}{2} + 2\mathsf{m}; \tag{14}$$
- *When* $\mathsf{M} < \mathsf{N} < 1.5\mathsf{M}$*, with* $\mathsf{M} \geq 2\mathsf{m}$*, by Scheme 4 described in Section IV-D we have*
$$h(\mathsf{N},\mathsf{M}) = h(\mathsf{M}, 2\mathsf{M}-\mathsf{N}). \tag{15}$$

□

Remark 1 (The source key size in extreme cases)**.** For the $(\mathsf{K},\mathsf{N},\mathsf{N_r},\mathsf{m})$ secure gradient coding problem with $\mathsf{M} = \mathsf{N} - \mathsf{N_r} + \mathsf{m}$, and M divides N, the minimum source key size is

$$\eta^\star = \frac{\mathsf{N}}{\mathsf{M}} - 1, \tag{16}$$

which matches the converse (2) and is achievable using fractional repetition assignment.

Under the cyclic assignment constraint, the minimum source key size required for optimal communication cost is

$$\eta^\star_{\text{cyc}} = \frac{\mathsf{N_r}}{\mathsf{m}} - 1. \tag{17}$$

At the end of this section, we present numerical evaluations comparing the source key size in converse in (11), in the cyclic assignment scheme, and in the scheme from Theorem 3. The results demonstrate that the combined scheme requires a significantly smaller source key size than the cyclic assignment scheme and coincides with the converse at some points.

IV. New Achievable Schemes For Theorem 3

For any secure distributed linearly separable scheme, [10, Theorem 1] provides a transfer method from a non-secure scheme to a secure scheme while maintaining the communication cost. The required source key size is $\frac{\lambda}{\mathsf{m}} - 1$, where λ represents the number of linearly independent messages transmitted by all the servers, i.e., $H(X_1,\ldots,X_{\mathsf{K}})/\mathsf{L}$. Hence, to design secure gradient coding scheme achieving the optimal communication cost with source key size as small as possible, we can design non-secure gradient coding scheme achieving the optimal communication cost, while minimizing the number of linearly independent messages. Note that when $\mathsf{N}|\mathsf{K}$, we can transfer the problem into $\mathsf{K} = \mathsf{N}$ using [10, Remark 1]. Thus, we focus on minimizing λ in the non-secure scheme where $\mathsf{K} = \mathsf{N}$. To elucidate the coding scheme, we present an illustrative example in Scheme 3, where the general description is given in Appendix B.

We mainly solve two challenges introduced by the transition from $\mathsf{m} = 1$ to $\mathsf{m} > 1$: when dividing the servers into groups and assigning different datasets to different groups, datasets within the same group should be stored at least m times, we design a more complex structure; to reduce λ, we do not use the coding scheme in [9], but jointly design the coding scheme across different groups using transmitter Interference Alignment [16]–[18]. For details, readers can refer to Appendix B.

A. Scheme 1 for (12)

In this section, we consider the case where $\mathsf{N} > 2\mathsf{M}$. We partition servers and datasets into intervals of length M, i.e., $\big[(i-1)\mathsf{M}+1 : i\mathsf{M}\big]$ for $i \in \left[\left\lfloor \frac{\mathsf{N}}{\mathsf{M}} - 1 \right\rfloor\right]$, and assign them via the fractional repetition strategy, yielding $\mathsf{m}\left\lfloor \frac{\mathsf{N}}{\mathsf{M}} - 1 \right\rfloor$ linearly independent combinations. We then assign the remaining $\mathsf{N} - \left\lfloor \frac{\mathsf{N}}{\mathsf{M}} - 1 \right\rfloor \mathsf{M}$ datasets and servers, thereby reducing the original (N,M) problem to a smaller subproblem $\big(\mathsf{N} - \lfloor \frac{\mathsf{N}}{\mathsf{M}} - 1 \rfloor \mathsf{M}, \mathsf{M}\big)$.

B. Scheme 2 for (13)

We consider the case $1.5\mathsf{M} \leq \mathsf{N} < 2\mathsf{M}$ with even $\mathsf{M} \geq 2\mathsf{m}$. The servers are divided into three groups: those in $[1, \frac{\mathsf{M}}{2}]$ (first group), those in $[\frac{\mathsf{M}}{2}+1, \mathsf{M}]$ (second group), and the remaining $\mathsf{N}-\mathsf{M}$ servers (third group). The first two groups each provide m linearly independent linear combinations, while the third group employs the $(\mathsf{N}-\mathsf{M}, \frac{\mathsf{M}}{2})$ scheme to recover $h(\mathsf{N}-\mathsf{M}, \frac{\mathsf{M}}{2})$ combinations.

We first consider the data assignment for the servers in $[\mathsf{M}]$, whose structure is as follows.

server	1	$\cdots$	$\frac{\mathsf{M}}{2}$	$\frac{\mathsf{M}}{2}+1$	$\cdots$	M
dataset	D_1	$\cdots$	D_1	D_1	$\cdots$	D_1
	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
	D_{y}	$\cdots$	D_{y}	D_{y}	$\cdots$	D_{y}
	$D_{\mathsf{y}+1}$	$\cdots$	$D_{\mathsf{y}+1}$	$D_{\mathsf{M}+1}$	$\cdots$	$D_{\mathsf{M}+1}$
	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
	D_{M}	$\cdots$	D_{M}	D_{N}	$\cdots$	D_{N}

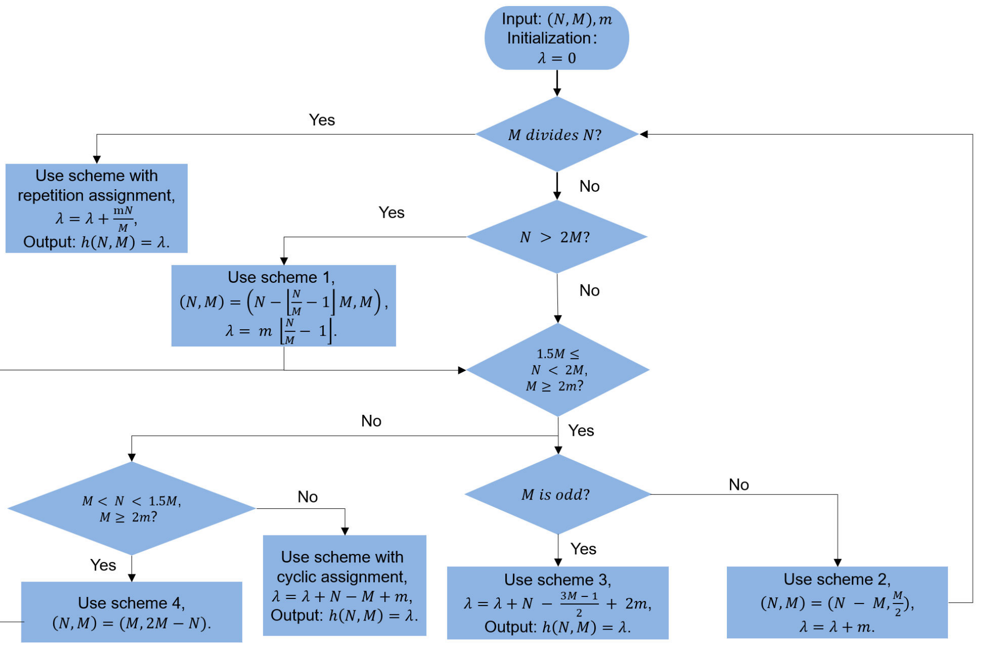
Input: $(N,M), m$
Initialization:
$\lambda = 0$
M divides N?
Yes
No
Use scheme with repetition assignment,
$\lambda = \lambda + \frac{mN}{M}$,
Output: $h(N,M) = \lambda$.
$N > 2M$?
Yes
No
Use scheme 1,
$(N,M) = \left(N - \left\lfloor \frac{N}{M} - 1 \right\rfloor M, M\right)$,
$\lambda = m \left\lfloor \frac{N}{M} - 1 \right\rfloor$.
$1.5M \le N < 2M, M \ge 2m$?
No
Yes
$M < N < 1.5M, M \ge 2m$?
No
Yes
Use scheme with cyclic assignment,
$\lambda = \lambda + N - M + m$,
Output: $h(N,M) = \lambda$.
Use scheme 4,
$(N,M) = (M, 2M - N)$.
M is odd?
Yes
No
Use scheme 3,
$\lambda = \lambda + N - \frac{3M-1}{2} + 2m$,
Output: $h(N,M) = \lambda$.
Use scheme 2,
$(N,M) = (N - M, \frac{M}{2})$,
$\lambda = \lambda + m$.

Fig. 1: Flow diagram of the combined scheme in Theorem 3.

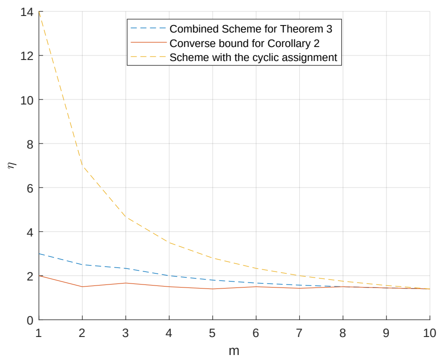

Fig. 2: Numerical evaluations for the secure gradient coding with $\mathsf{K} = \mathsf{N} = 24, \mathsf{M} = 10$.

In this setup, we assign D_1 to D_y to all servers in $[\mathsf{M}]$, while each D_k for $k \in [\mathsf{y}+1 : \mathsf{N}]$ is assigned to $\frac{\mathsf{M}}{2}$ servers in $[\mathsf{M}]$. We then allocate the remaining $\mathsf{N}-\mathsf{M}$ servers. Since $\mathsf{N}-\mathsf{y} = 2(\mathsf{N}-\mathsf{M})$ datasets (from $[\mathsf{y}+1 : \mathsf{N}]$) must be distributed among $\mathsf{N}-\mathsf{M}$ servers, each dataset is assigned to $\frac{\mathsf{M}}{2}$ servers, and each server receives M datasets.

To implement this, we divide the datasets in $[\mathsf{y}+1 : \mathsf{N}]$ into $\frac{\mathsf{N}-\mathsf{y}}{2} = \mathsf{N}-\mathsf{M}$ pairs $\mathcal{P}_i = \{\mathsf{y}+i, \mathsf{M}+i\}$ for $i \in [\mathsf{N}-\mathsf{M}]$. We then use the assignment phase of the $\left(\mathsf{N}-\mathsf{M}, \frac{\mathsf{M}}{2}\right)$ scheme, assigning each pair to $\frac{\mathsf{M}}{2}$ servers, with each server receiving $\frac{\mathsf{M}}{2}$ pairs.

C. *Scheme 3 for* (14)

We provide an example to illustrate the main idea.

Example 1. We consider the $(\mathsf{N},\mathsf{M}) = (12,7)$ non-secure problem, where $\mathsf{m} = 2$.

Data assignment phase. We assign the datasets as follows.

server	1	2	3	4	5	6	7
	D_1	D_1	D_1	D_1	D_1	D_1	D_1
	D_2	D_2	D_2	D_2	D_2	D_2	D_2
	D_3	D_3	D_3	D_3	D_3	D_3	D_3
dataset	D_4	D_4	D_8	D_9	D_{10}	D_{11}	D_{12}
	D_5	D_5	D_9	D_{10}	D_{11}	D_{12}	D_8
	D_6	D_6	D_{10}	D_{11}	D_{12}	D_8	D_9
	D_7	D_7	D_{11}	D_{12}	D_8	D_9	D_{10}

8	9	10	11	12
D_4	D_4	D_4	D_4	D_4
D_5	D_5	D_5	D_5	D_5
D_6	D_6	D_6	D_6	D_6
D_7	D_7	D_7	D_7	D_7
D_8	D_9	D_{10}	D_{11}	D_{12}
D_9	D_{10}	D_{11}	D_{12}	D_8
D_{10}	D_{11}	D_{12}	D_8	D_9

Computing phase. To minimize communication cost, each message W_k, for $k \in [\mathsf{K}]$, is divided into m equal-length sub-messages, denoted as $W_k = \{W_{k,j} : j \in [\mathsf{m}]\}$, with each sub-message containing $\frac{\mathsf{L}}{\mathsf{m}} = \frac{\mathsf{L}}{2}$ symbols in $\mathbb{F}_\mathsf{q}$. Each server sends a linear combination of messages, allowing the user to recover $\mathbf{F}[W_{1,1}; \ldots; W_{12,2}]$ from any $\mathsf{N}_\mathrm{r} = 7$ servers, where $\mathbf{F}$ is shown at the top of the page in (18). The design of $\mathbf{f}_5$ and $\mathbf{f}_6$ will be detailed later. We define that $[F_1; F_2; F_3; F_4; F_5; F_6] = \mathbf{F}\ [W_{1,1}; \ldots; W_{12,2}]$.

We divide the servers into three groups based on the data assignment: servers 1 and 2 form the first group, those in $[3:7]$ **form the second, and those in** $[8:12]$ **form the third. We require that the** $\mathsf{m} = 2$ **linear combinations**

$$\mathbf{F} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{f}_3 \\ \mathbf{f}_4 \\ \mathbf{f}_5 \\ \mathbf{f}_6 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 4 & 2 & 4 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 3 & 4 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 4 & 4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 3 & 1 & 1 & -5 \end{bmatrix} \tag{18}$$

$(F_1 - F_3)$ **and** $(F_2 - F_4)$ **be recovered from the first group. From the second group, we recover** $(2F_1 - F_3)$**,** $(2F_2 - F_4)$**,** F_5**, and** F_6**, which together form** $\mathsf{N} - \frac{3\mathsf{M}-1}{2} + \mathsf{m} = 4$ **linear combinations, and from the third group, we recover** F_3**,** F_4**,** F_5**, and** F_6**, another** 4 **combinations. Hence, we can recover** $\mathsf{N} - \frac{3\mathsf{M}-1}{2} + 2\mathsf{m} = 6$ **linear combinations in total, matching the result in** (14)**. The following outlines the specific procedure**

We focus on servers 1 and 2, which share the same datasets and exclude those from [8:11]. server 1 computes

$$X_1 = (F_1 - F_3) + (F_2 - F_4),$$

and server 2 computes

$$X_2 = (F_1 - F_3) - (F_2 - F_4).$$

For servers handling [8:12], we design transmissions so that F_3, F_4, F_5, and F_6 can be recovered from any four responses. Specifically, for $n \in$ [8:12], let server n compute

$$\mathbf{s}_n \big[\mathbf{f}_3; \mathbf{f}_4; \mathbf{f}_5; \mathbf{f}_6\big] \big[W_{1,1}; \ldots; W_{12,2}\big],$$

We design $\mathbf{s}_n$ according to the following steps. We focus on the datasets in $[8:12]$ under cyclic assignment, extracting the corresponding columns from the matrix to form a submatrix $\mathbf{F}'_1$.

Notice that server n cannot compute 2 of the datasets in $[8:12]$. We extract the corresponding columns from the matrix to form a submatrix $\overline{\mathbf{F}'_1}$.

Our required $\mathbf{s}_n$ is the left null vector of $\overline{\mathbf{F}'_1}$. There should be a linear combination of the columns in $\overline{\mathbf{F}'_1}$ leading to one rank deficiency, with the form

$$\mathbf{F}'_1 \mathbf{e}_n^{\mathrm{T}} = \mathbf{0}_{4\times 1}, \tag{19}$$

for the first row of $\overline{\mathbf{F}'_1}$, since $\begin{bmatrix} 1 & 1 \end{bmatrix} [1, -1]^{\mathrm{T}} = \mathbf{0}_{1\times 1}$, the corresponding elements in $\mathbf{e}_n$ should be a multiple of $(1, -1)$, and we choose a random multiple. Similarly, the same applies to the second row of $\overline{\mathbf{F}'_1}$. Repeating the above steps, we obtain the matrix $\mathbf{E}$, where

$$\mathbf{E} = \begin{bmatrix} \mathbf{e}_1 \\ \vdots \\ \mathbf{e}_5 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & -1 & 1 \\ 2 & 0 & 0 & 0 & -2 & -1 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 & 0 & 2 & -2 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 2 & -2 & 0 \end{bmatrix} \tag{20}$$

By (19) and (20), we have

$$\mathbf{F}'_1 \mathbf{E}^{\mathrm{T}} = \mathbf{0}_{4\times 5}, \tag{21}$$

where $\mathbf{E}^{\mathrm{T}}$ is 10×5 and full rank. Thus, its left null space contains 5 linearly independent vectors, spanning the rows of $\mathbf{F}'_1$. The first two rows lie in the null space of $\mathbf{E}^{\mathrm{T}}$, and any two vectors from the remaining three null space vectors complete $\mathbf{F}'_1$, given as:

$$\mathbf{F}'_1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 2 & 4 & 2 & 4 & 2 & 2 & 1 & 3 & 4 & 2 \\ 2 & 2 & 4 & 4 & -2 & 3 & 3 & 1 & 1 & -5 \end{bmatrix}.$$

For $\mathbf{s}_n$, satisfying $\mathbf{s}_n \overline{\mathbf{F}'_1} = \mathbf{0}_{1\times 4}$, the resulting matrix is:

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_1 \\ \vdots \\ \mathbf{s}_5 \end{bmatrix} = \begin{bmatrix} 2 & 11/4 & -3/4 & 1/4 \\ 4 & 4 & -2 & 0 \\ 2 & 3 & 0 & -1 \\ 8 & 16/3 & -4/3 & -4/3 \\ 6 & 3/2 & 0 & -3/2 \end{bmatrix}.$$

For $n \in [8, 12]$, server n computes

$$X_n = \mathbf{s}_n \big[F_3; F_4; F_5; F_6\big].$$

Next, servers in [3:7], excluding datasets from [4:7], ensure that responses from any 4 servers recover $(2F_1 - F_3)$, $(2F_2 - F_4)$, F_5, and F_6. For $n \in [3, 7]$, server n computes

$$X_n = \mathbf{s}_n \big[2\mathbf{f}_1 - \mathbf{f}_3; 2\mathbf{f}_2 - \mathbf{f}_4; \mathbf{f}_5; \mathbf{f}_6\big] \big[W_{1,1}; \ldots; W_{12,2}\big].$$

For datasets in [8:12], cyclic assignment forms $\mathbf{F}'_2$, and it is evident that $\mathbf{F}'_2 = \mathbf{F}'_1$. Since datasets missing from server $n \in$ [3:7] are included in those missing from $n+5$, we have $\mathbf{S}' = \mathbf{S}$. Thus, server n computes

$$X_n = \mathbf{s}_n \big[2F_1 - F_3; 2F_2 - F_4; F_5; F_6\big].$$

In conclusion, under the condition $\mathsf{m} = 2$, the number of totally transmitted linearly independent combinations is $h(12, 7) = 12 - 10 + 4 = 6$, which coincides with (14). Moreover, our calculations show that if

$$\mathsf{m}(\mathsf{N} - \mathsf{M} - 1) \ \leq \ \big[(\mathsf{N} - \mathsf{M})(\mathsf{m} - 1) + 1\big] \Big(\mathsf{N} - \tfrac{3}{2}\mathsf{M} - \tfrac{1}{2}\Big),$$

transmitter Interference Alignment [16]–[18] is required.

D. Scheme 4 for (15)

In this section, we consider the case $\mathsf{M} < \mathsf{N} < 1.5\mathsf{M}$ with $\mathsf{M} \geq 2\mathsf{m}$ and transform the (N, M) problem into $(\mathsf{M}, 2\mathsf{M} - \mathsf{N})$ by dividing the servers into two groups: those in $[\mathsf{M}]$ and those in $[\mathsf{M} + 1 : \mathsf{N}]$. We assign $D_1, \ldots, D_{\mathsf{N}-\mathsf{M}}$ to each server in $[\mathsf{M}]$, and $D_{\mathsf{N}-\mathsf{M}+1}, \ldots, D_{\mathsf{N}}$ to each server in $[\mathsf{M}+1 : \mathsf{N}]$. Thus, every server in $[\mathsf{M}+1 : \mathsf{N}]$ holds M datasets, while each server in $[\mathsf{M}]$ has $\mathsf{N} - \mathsf{M}$. Since every dataset in $[\mathsf{N} - \mathsf{M} + 1 : \mathsf{N}]$ is assigned to $\mathsf{N} - \mathsf{M}$ servers, we further assign each D_k (for $k \in [\mathsf{N} - \mathsf{M} + 1 : \mathsf{N}]$) to $2\mathsf{M} - \mathsf{N}$ additional servers in $[\mathsf{M}]$, giving each such server $(2\mathsf{M} - \mathsf{N})$ datasets. This allocation follows the assignment phase of the $(\mathsf{M}, 2\mathsf{M} - \mathsf{N})$ non-secure problem.

References

[1] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1175–1191. [Online]. Available: https://doi.org/10.1145/3133956.3133982

[2] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, p. 612–613, Nov. 1979. [Online]. Available: https://doi.org/10.1145/359168.359176

[3] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Annual international cryptology conference*. Springer, 1991, pp. 129–140.

[4] R. Cramer, I. Damgård, and S. Dziembowski, "On the complexity of verifiable secret sharing and multiparty computation," in *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, 2000, pp. 325–334.

[5] D. Chaum, C. Crépeau, and I. Damgard, "Multiparty unconditionally secure protocols," in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, ser. STOC '88. New York, NY, USA: Association for Computing Machinery, 1988, p. 11–19. [Online]. Available: https://doi.org/10.1145/62212.62214

[6] X. Yue, H. Wang, D. Jin, M. Li, and W. Jiang, "Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control," *Journal of medical systems*, vol. 40, pp. 1–8, 2016.

[7] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding: Avoiding stragglers in distributed learning," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 3368–3376. [Online]. Available: https://proceedings.mlr.press/v70/tandon17a.html

[8] M. Ye and E. Abbe, "Communication-computation efficient gradient coding," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5610–5619.

[9] T. Jahani-Nezhad and M. A. Maddah-Ali, "Optimal communication-computation trade-off in heterogeneous gradient coding," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 3, pp. 1002–1011, 2021.

[10] K. Wan, H. Sun, M. Ji, and G. Caire, "On secure distributed linearly separable computation," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 3, pp. 912–926, 2022.

[11] H. Fereidooni, S. Marchal, M. Miettinen, A. Mirhoseini, H. Möllering, T. D. Nguyen, P. Rieger, A.-R. Sadeghi, T. Schneider, H. Yalame, and S. Zeitouni, "Safelearn: Secure aggregation for private federated learning," in *2021 IEEE Security and Privacy Workshops (SPW)*, 2021, pp. 56–62.

[12] B. Choi, J.-y. Sohn, D.-J. Han, and J. Moon, "Communication-computation efficient secure aggregation for federated learning," *arXiv preprint arXiv:2012.05433*, 2020.

[13] J. So, C. He, C.-S. Yang, S. Li, Q. Yu, R. E. Ali, B. Guler, and S. Avestimehr, "Lightsecagg: a lightweight and versatile design for secure aggregation in federated learning," in *Proceedings of Machine Learning and Systems*, D. Marculescu, Y. Chi, and C. Wu, Eds., vol. 4, 2022, pp. 694–720. [Online]. Available: https://proceedings.mlsys.org/paper_files/paper/2022/file/6c44dc73014d66ba49b28d483a8f8b0d-Paper.pdf

[14] T. Jahani-Nezhad, M. A. Maddah-Ali, S. Li, and G. Caire, "Swiftagg: Communication-efficient and dropout-resistant secure aggregation for federated learning with worst-case security guarantees," in *2022 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2022, pp. 103–108.

[15] Y. Zhao and H. Sun, "Information theoretic secure aggregation with user dropouts," *IEEE Transactions on Information Theory*, vol. 68, no. 11, pp. 7471–7484, 2022.

[16] V. R. Cadambe and S. A. Jafar, "Interference alignment and degrees of freedom of the k-user interference channel," *IEEE Transactions on Information Theory*, vol. 54, no. 8, pp. 3425–3441, 2008.

[17] S. A. Jafar and M. J. Fakhereddin, "Degrees of freedom for the mimo interference channel," *IEEE Transactions on Information Theory*, vol. 53, no. 7, pp. 2637–2642, 2007.

[18] W. Huang, K. Wan, H. Sun, M. Ji, R. C. Qiu, and G. Caire, "Fundamental limits of distributed linearly separable computation under cyclic assignment," in *2023 IEEE International Symposium on Information Theory (ISIT)*, 2023, pp. 2296–2301.

[19] ——, "Fundamental limits of distributed linearly separable computation under cyclic assignment," in *2023 IEEE International Symposium on Information Theory (ISIT)*, 2023, pp. 2296–2301.

Appendix A
Proof of Theorem 2

By the security constraint in (7), the user can only obtain $W_1+\cdots+W_{\mathsf{K}}$ without accessing any additional information about the messages, even after receiving answers from all servers. Let $X_{\mathcal{S}}=\{X_n : n\in\mathcal{S}\}$. From (7), we derive:

$$0 = I(W_1,\ldots,W_{\mathsf{K}};X_{[\mathsf{N}]}|W_1+W_2+\cdots+W_{\mathsf{K}}) \tag{22a}$$
$$= H(X_{[\mathsf{N}]}|W_1+W_2+\cdots+W_{\mathsf{K}}) - H(X_{[\mathsf{N}]}|W_1,\ldots,W_{\mathsf{K}}) \tag{22b}$$
$$\geq H(X_{[\mathsf{N}]}|W_1+W_2+\cdots+W_{\mathsf{K}}) - H(Q,W_1,\ldots,W_{\mathsf{K}}|W_1,\ldots,W_{\mathsf{K}}) \tag{22c}$$
$$= H(X_{[\mathsf{N}]}|W_1+W_2+\cdots+W_{\mathsf{K}}) - H(Q) \tag{22d}$$
$$= H(X_{[\mathsf{N}]}) - I(X_{[\mathsf{N}]};W_1+W_2+\cdots+W_{\mathsf{K}}) - H(Q) \tag{22e}$$
$$\geq H(X_{[\mathsf{N}]}) - H(W_1+W_2+\cdots+W_{\mathsf{K}}) - H(Q), \tag{22f}$$

where (22c) follows from the fact that $X_{[\mathsf{N}]}$ is a function of Q and $W_1,\ldots,W_{\mathsf{K}}$, and (22d) relies on Q being independent of $W_1,\ldots,W_{\mathsf{K}}$.

From (22f), we further obtain:

$$\eta\mathsf{L} \geq H(Q) \geq H(X_{[\mathsf{N}]}) - H(W_1+\cdots+W_{\mathsf{K}}) \tag{23a}$$
$$\geq H(X_{[\mathsf{N}]}) - \mathsf{L} \tag{23b}$$
$$\geq H(X_{s_1},\ldots,X_{s_{|\mathsf{s}|}}) - \mathsf{L} \tag{23c}$$
$$\geq \frac{|\mathsf{s}|\mathsf{L}}{\mathsf{m}} - \mathsf{L}, \tag{23d}$$

where (23b) follows from the independence of K messages, each uniformly i.i.d. over $[\mathbb{F}_{\mathsf{q}}]^{\mathsf{L}}$.

Next, I will prove (23d) under the symmetric transmission and linear coding. Note that server s_i always contains at least one message W_{s_i} which can not be computed by m servers from $\{s_1,\ldots,s_{i-1}\}$. Consider the case that apart from $\mathsf{N}-\mathsf{N}_{\mathrm{r}}$ stragglers, there are exactly m servers who can compute W_{s_i}. Under the symmetric transmission and linear coding, each server should contain $\frac{1}{m}$ independent part of W_{s_i}. Thus, we have the message of server s_i is independent to messages from $\{s_1,\ldots,s_{i-1}\}$, because of the independent part of W_{s_i}. Since the length of each message should be larger than $\frac{\mathsf{L}}{\mathsf{m}}$, we can obtain (23d) under the symmetric transmission and linear coding.

Appendix B
The general scheme of Theorem 3

A. general scheme 1

Data assignment phase. We partition the entire system into $\lfloor \mathsf{N}/\mathsf{M}\rfloor$ blocks. For each $i\in[\lfloor \mathsf{N}/\mathsf{M}\rfloor-1]$, the i^{th} block includes the datasets $\{D_k : k\in[(i-1)\mathsf{M}+1:i\mathsf{M}]\}$ and the servers in the range $[(i-1)\mathsf{M}+1:i\mathsf{M}]$. The last block includes the datasets and servers in the range $[\lfloor \mathsf{N}/\mathsf{M}-1\rfloor\mathsf{M}+1:\mathsf{N}]$.

For the first $[\lfloor \mathsf{N}/\mathsf{M}\rfloor-1]$ blocks, we use the fractional repetition assignment to distribute the data. As for the last block, it contains $\mathsf{N}-\lfloor \mathsf{N}/\mathsf{M}-1\rfloor\mathsf{M}$ servers and $\mathsf{N}-\lfloor \mathsf{N}/\mathsf{M}-1\rfloor\mathsf{M}$ datasets, with each server being able to compute M datasets. Therefore, we can directly apply the existing data assignment scheme designed for solving the $(\mathsf{N}-\lfloor \mathsf{N}/\mathsf{M}-1\rfloor\mathsf{M},\mathsf{M})$ non-secure problem to this last block.

Computing phase. For each $i\in[\lfloor \mathsf{N}/\mathsf{M}\rfloor-1]$, each server in the i^{th} block computes m linear combinations $\sum_{k\in[(i-1)\mathsf{M}+1:i\mathsf{M}]} W_{k,j}$, and each server sends a random linear combination of these m computed combinations.

Next, we focus on the last block. The computation phase in this block still follows the $(\mathsf{N}-\lfloor \mathsf{N}/\mathsf{M}-1\rfloor\mathsf{M},\mathsf{M})$ non-secure scheme, which results in a total of $h(\mathsf{N}-\lfloor \mathsf{N}/\mathsf{M}-1\rfloor\mathsf{M},\mathsf{M})$ linearly independent combinations being sent.

Decoding phase. The user receives information from $\mathsf{N}_{\mathrm{r}}=\mathsf{N}-\mathsf{M}+\mathsf{m}$ servers, meaning that responses from $\mathsf{M}-\mathsf{m}$ servers will not be received. For the first $\lfloor \mathsf{N}/\mathsf{M}-1\rfloor$ blocks, each block has M servers, so at least m servers' responses from each block are received by the user. Thus, from each block, the user can recover $\sum_{k\in[(i-1)\mathsf{M}+1:i\mathsf{M}]} W_{k,j}$, where j ranges over $[\mathsf{m}]$. For the last block, at least $\mathsf{N}-\lfloor \mathsf{N}/\mathsf{M}-1\rfloor\mathsf{M}-\mathsf{M}+\mathsf{m}$ servers will respond. By carefully designing the coding scheme, we can recover $\sum_{k\in[\lfloor \mathsf{N}/\mathsf{M}-1\rfloor\mathsf{M}+1:\mathsf{N}]} W_k$ from the information sent by any $\mathsf{N}-\lfloor \mathsf{N}/\mathsf{M}-1\rfloor\mathsf{M}-\mathsf{M}+\mathsf{m}$ servers in the last block. together with the transmissions of the first $\lfloor \mathsf{N}/\mathsf{M}-1\rfloor$ blocks, the user can recover $W_1+\cdots+W_{\mathsf{N}}$.

In conclusion, we have demonstrated that $h(\mathsf{N},\mathsf{M}) = \mathsf{m}\lfloor \mathsf{N}/\mathsf{M}-1\rfloor + h(\mathsf{N}-\lfloor \mathsf{N}/\mathsf{M}-1\rfloor\mathsf{M},\mathsf{M})$, which aligns with equation (12). Furthermore, it is evident that $\mathsf{M}<\mathsf{N}-\lfloor \mathsf{N}/\mathsf{M}-1\rfloor\mathsf{M}<2\mathsf{M}$ when $\mathsf{N}>2\mathsf{M}$ and M does not evenly divide N.

B. general scheme 2

We now focus on the case where $1.5\mathsf{M}\leq\mathsf{N}<2\mathsf{M}$, and M is even, with $\mathsf{M}\geq 2\mathsf{m}$. For this case, we consider the general Scheme 2 for the (N,M) non-secure problem to prove (13). We assume that a specific feasible scheme already exists for the $\left(\mathsf{N}-\mathsf{M},\frac{\mathsf{M}}{2}\right)$ non-secure problem and that this scheme can transmit a total of $h\left(\mathsf{N}-\mathsf{M},\frac{\mathsf{M}}{2}\right)$ linearly independent combinations of messages.

Data assignment phase. We first consider the dataset assignment for the servers in $[\mathsf{M}]$, which is structured as follows:

server	1	$\cdots$	$\frac{\mathsf{M}}{2}$	$\frac{\mathsf{M}}{2}+1$	$\cdots$	M
dataset	D_1	$\cdots$	D_1	D_1	$\cdots$	D_1
	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
	D_{y}	$\cdots$	D_{y}	D_{y}	$\cdots$	D_{y}
	$D_{\mathsf{y}+1}$	$\cdots$	$D_{\mathsf{y}+1}$	$D_{\mathsf{M}+1}$	$\cdots$	$D_{\mathsf{M}+1}$
	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
	D_{M}	$\cdots$	D_{M}	D_{N}	$\cdots$	D_{N}

In this setup, we assign datasets D_1 through D_{y} to all servers in $[\mathsf{M}]$, while each dataset D_k for $k\in[\mathsf{y}+1:\mathsf{N}]$ is distributed among $\frac{\mathsf{M}}{2}$ servers in $[\mathsf{M}]$.

Next, we consider the assignment for the last $\mathsf{N}-\mathsf{M}$ servers. We need to allocate $\mathsf{N}-\mathsf{y}=2(\mathsf{N}-\mathsf{M})$ datasets (from $[\mathsf{y}+1:\mathsf{N}]$) to a total of $\mathsf{N}-\mathsf{M}$ servers. Each dataset is assigned to $\frac{\mathsf{M}}{2}$ servers, and each server receives M datasets.

Datasets in $[y+1:N]$ are divided into $\frac{N-y}{2}=N-M$ pairs, where the i^{th} pair is defined as $\mathcal{P}_i=\{y+i, M+i\}$ for $i\in[N-M]$. Thus, we apply the assignment phase of the scheme for the $\left(N-M,\frac{M}{2}\right)$ non-secure problem, assigning each pair to $\frac{M}{2}$ servers, with each server being allocated $\frac{M}{2}$ pairs.

Computing phase. We divide each message W_k for $k\in[K]$ into m equal-length, non-overlapping sub-messages, denoted as $W_k=\{W_{k,j}:j\in[m]\}$. We first focus on the servers in $[M]$. By constructing the messages sent by these servers, we can recover the following $2m$ linear combinations:

$$\begin{aligned}
F_1 &= W_{1,1}+W_{2,1}+\cdots+W_{N,1},\\
&\vdots\\
F_m &= W_{1,m}+W_{2,m}+\cdots+W_{N,m},\\[1em]
F_{m+1} &= 2(W_{y+1,1}+\cdots+W_{M,1})+W_{M+1,1}+\cdots+W_{N,1},\\
&\vdots\\
F_{2m} &= 2(W_{y+1,m}+\cdots+W_{M,m})+W_{M+1,m}+\cdots+W_{N,m}.
\end{aligned}$$

We group the m linear combinations $(F_1-F_{m+1}),\ldots,(F_m-F_{2m})$ into one group, and the m linear combinations $(2F_1-F_{m+1}),\ldots,(2F_m-F_{2m})$ into another group. Since the servers in $[M/2]$ are not assigned any data from $[M+1:N]$, we let each server in this range compute a random linear combination of the m linear combinations in the first group. The servers in $[M/2+1:M]$, who are not assigned any data from $[y+1:M]$, are assigned to compute a random linear combination of the m linear combinations in the second group.

We then focus on the servers in $[M+1:N]$. For each dataset pair $\mathcal{P}_i=\{y+i,M+i\}$ where $i\in[N-M]$, we define $P_i=2W_{y+i}+W_{M+i}$. Therefore, we can express F_{m+1} to F_{2m} as $P_{1,1}+\cdots+P_{N-M,1},\ldots,P_{1,m}+\cdots+P_{N-M,m}$. Thus, we can apply the computing phase of the proposed scheme to the $\left(N-M,\frac{M}{2}\right)$ non-secure problem.

Decoding phase.

The user receives $N_r=N-M+m$ server responses, which can be categorized into two cases.

First, we consider the case where at least $N-1.5M+m$ responses come from servers in $[M+1:N]$. In this scenario, F_{m+1} to F_{2m} can be recovered. Additionally, at least m responses from servers in $[M]$ are received, providing at least m linearly independent combinations. Together with the combinations from F_{m+1} to F_{2m}, this gives a total of $2m$ linearly independent combinations, enabling recovery of F_1 to F_{2m}. We then consider the second case, where the servers in $[M+1:N]$ respond with $N-1.5M+m-x$ responses, providing at least $h(N-M,M/2)-x$ linearly independent combinations. Meanwhile, the servers in $[M]$ will have $M/2+x$ responses, contributing at least $m+x$ linearly independent combinations. In total, these responses provide $h(N-M,M/2)$ linearly independent combinations, which are sufficient to recover F_1 to F_{2m}.

The number of linearly independent combinations transmitted by servers in $[M+1:N]$ is $h\left(N-M,\frac{M}{2}\right)$, which spans the space containing F_{m+1} to F_{2m}. Additionally, servers in $[M]$ transmit $2m$ linearly independent combinations, also spanning the space of F_{m+1} to F_{2m}. Therefore, the total number of transmitted linearly independent combinations is $h(N,M)=2m+h\left(N-M,\frac{M}{2}\right)-m=h\left(N-M,\frac{M}{2}\right)+m$, which matches (13).

C. general scheme 3

We now consider the (N,M) non-secure problem, where $1.5M\le N<2M$, M is odd, and $M\ge 2m+1$. Our goal is to construct a scheme (Scheme 3) to prove (14). We also define that $N=2M-y$.

Data assignment phase. The assignment is shown at the top of the next page. In this assignment, we split the N datasets into three sections. The first section includes $D_1,\ldots,D_t$ (The reason for choosing $t=\frac{M-1}{2}$ is provided in [10].), and these datasets are assigned to servers in $[M]$. The second section, containing $D_{t+1},\ldots,D_M$, is assigned to servers in $[y]\cup[M+1:N]$. The third section, consisting of $D_{M+1},\ldots,D_N$, is cyclically allocated to servers in $[y+1:M]$, where each server receives $M-t$ adjacent datasets within $\left[\frac{M-1}{2}+1:N\right]$. Additionally, datasets $D_{M+1},\ldots,D_N$ are assigned to servers in $[M+1:N]$ in a cyclic manner such that each server receives t consecutive datasets within $\left[\frac{M-1}{2}+1:N\right]$.

Computing phase. We first divide each message W_k for $k\in[K]$ into m equal-length, non-overlapping sub-messages, denoted as $W_k=\{W_{k,j}:j\in[m]\}$. We design the computing phase so that the total number of linearly independent combinations computed by all servers is $\frac{M+1}{2}-y+2m$. We let each server send a linear combination of messages so that the user can recover $\mathbf{F}'\,[W_{1,1};\ldots;W_{N,m}]$ from the responses of any $N_r=N-M+m$ servers, where $\mathbf{F}'$ is shown at the top of the next page in (24).

We design the demand matrix

$$\mathbf{F}=\begin{array}{ccc}\left[\begin{array}{c:c:c} 1,\ldots,1 & 1,\ldots,1 & 1,\ldots,1\\ 0,\ldots,0 & 2,\ldots,2 & 1,\ldots,1\end{array}\right]\\ \begin{array}{ccc}\mathbf{C}_1 & \quad\mathbf{C}_2\quad & \mathbf{C}_3\end{array}\end{array}\tag{25}$$

We divide matrix $\mathbf{F}$ into three column-wise sub-matrices: $\mathbf{C}_1$, containing $\frac{M-1}{2}$ columns, corresponding to the messages in $\left[\frac{M-1}{2}\right]$, $\mathbf{C}_2$, containing $\frac{M+1}{2}$ columns, corresponding to the messages in $\left[\frac{M+1}{2}:M\right]$, and $\mathbf{C}_3$, containing $(N-M)$ columns, corresponding to the messages in $[M+1:N]$.

Let the matrix $\mathbf{V}_i$, for $i\in[m]$, be defined as

$$\mathbf{V}_i=\begin{array}{c}\left[\begin{array}{c:c:c} 0,\ldots,0 & 0,\ldots,0 & *,\ldots,*\\ \vdots & \vdots & \vdots\\ 0,\ldots,0 & 0,\ldots,0 & *,\ldots,*\end{array}\right]\\ \begin{array}{ccc}\mathbf{C}_1 & \quad\mathbf{C}_2\quad & \mathbf{C}_3\end{array}\end{array}\tag{26}$$

In the matrix $\mathbf{V}_i$, the entries denoted by $*$ are the unknowns that we need to design. And $\mathbf{C}_1$, $\mathbf{C}_2$, and $\mathbf{C}_3$ have column counts consistent with the definitions above.

server	1	$\cdots$	y	$\mathsf{y}+1$	$\mathsf{y}+2$	$\cdots$	M	M+1	$\mathsf{M}+2$	$\cdots$	N
	D_1	$\cdots$	D_1	D_1	D_1	$\cdots$	D_1	$D_{\frac{\mathsf{M}-1}{2}+1}$	$D_{\frac{\mathsf{M}-1}{2}+1}$	$\cdots$	$D_{\frac{\mathsf{M}-1}{2}+1}$
	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
dataset	$D_{\frac{\mathsf{M}-1}{2}}$	$\cdots$	$D_{\frac{\mathsf{M}-1}{2}}$	$D_{\frac{\mathsf{M}-1}{2}}$	$D_{\frac{\mathsf{M}-1}{2}}$	$\cdots$	$D_{\frac{\mathsf{M}-1}{2}}$	D_{M}	D_{M}	$\cdots$	D_{M}
	$D_{\frac{\mathsf{M}-1}{2}+1}$	$\cdots$	$D_{\frac{\mathsf{M}-1}{2}+1}$	$D_{\mathsf{M}+1}$	$D_{\mathsf{M}+2}$	$\cdots$	D_{N}	$D_{\mathsf{M}+1}$	$D_{\mathsf{M}+2}$	$\cdots$	D_{N}
	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
	D_{M}	$\cdots$	D_{M}	$D_{\frac{3\mathsf{M}+1}{2}}$	$D_{\frac{3\mathsf{M}+1}{2}+1}$	$\cdots$	$D_{\frac{\mathsf{M}+1}{2}-1}$	$D_{\frac{3\mathsf{M}-1}{2}}$	$D_{\frac{3\mathsf{M}-1}{2}+1}$	$\cdots$	$D_{\frac{\mathsf{M}-1}{2}-1}$

$$\mathbf{F}' = \begin{bmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_{\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}} \end{bmatrix} = \begin{bmatrix} (\mathbf{F})_{2\times\mathsf{N}} & \mathbf{0}_{2\times\mathsf{N}} & \cdots & \mathbf{0}_{2\times\mathsf{N}} \\ \mathbf{0}_{2\times\mathsf{N}} & (\mathbf{F})_{2\times\mathsf{N}} & \cdots & \mathbf{0}_{2\times\mathsf{N}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{2\times\mathsf{N}} & \mathbf{0}_{2\times\mathsf{N}} & \cdots & (\mathbf{F})_{2\times\mathsf{N}} \\ (\mathbf{V}_1)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times\mathsf{N}} & (\mathbf{V}_2)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times\mathsf{N}} & \cdots & (\mathbf{V}_{\mathsf{m}})_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times\mathsf{N}} \end{bmatrix}_{(\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m})\times\mathsf{Nm}} . \tag{24}$$

$$\mathbf{F_s} = \begin{bmatrix} \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} \\ \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} \\ (\mathbf{V}'_1)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} & (\mathbf{V}'_2)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} & \cdots & (\mathbf{V}'_{\mathsf{m}})_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} \end{bmatrix}_{(\frac{\mathsf{M}+1}{2}-\mathsf{y}+\mathsf{m})\times\mathsf{m}(\mathsf{N}-\mathsf{M})} \tag{28}$$

Now, we focus on the servers in [y], which have the same datasets and do not contain datasets from $[\mathsf{M}+1:\mathsf{N}]$. From these servers, we aim to recover:

$$F_1 - F_2 = W_{1,1} + \cdots + W_{t,1} - W_{t+1,1} - \cdots - W_{\mathsf{N},1},$$
$$\vdots$$
$$F_{2\mathsf{m}-1} - F_{2\mathsf{m}} = W_{1,\mathsf{m}} + \cdots + W_{t,\mathsf{m}} - W_{t+1,\mathsf{m}} - \cdots - W_{\mathsf{N},\mathsf{m}}.$$

Thus, we let the servers in [y] each compute a random linear combination of the m linearly independent combinations $(F_1 - F_2), \ldots, (F_{2\mathsf{m}-1} - F_{2\mathsf{m}})$.

For the servers in $[M+1:N]$ that do not contain datasets from $[t]$, we design their transmissions so that the user can recover $F_2, F_4, \ldots, F_{2\mathsf{m}}, F_{2\mathsf{m}+1}, \ldots, F_{\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}}$ from any $\frac{\mathsf{M}+1}{2} - \mathsf{y} + \mathsf{m}$ responses. For $n \in [\mathsf{M}+1, \mathsf{N}]$, we let server n compute

$$\mathbf{s}_n[F_2; F_4; \ldots; F_{2\mathsf{m}}; F_{2\mathsf{m}+1}; \ldots; F_{\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}}] \times [W_{1,1}; W_{2,1}; \ldots; W_{\mathsf{N},\mathsf{m}}]. \tag{27}$$

We design $\mathbf{s}_n$ as follows. Notice that $W_1, \ldots, W_{\frac{\mathsf{M}-1}{2}}$ can be computed by server n; and that in the linear combination (27), the coefficients of $W_{\frac{\mathsf{M}-1}{2}+1}, \ldots, W_{\mathsf{M}}$ are 0. Hence, in order to guarantee that in (27) the coefficients of the messages which server n cannot compute are 0, we only need to consider the messages in $W_{\mathsf{M}+1}, \ldots, W_{\mathsf{N}}$, corresponding to the columns in $\mathbf{C}_3$ in the matrix, with a total of $\mathsf{m}(\mathsf{N}-\mathsf{M})$ columns. By extracting the corresponding rows and columns from the matrix $\mathbf{F}'$, we obtain $\mathbf{F_s}$, as shown at the top of the next page in (28), where $\mathbf{V}'_i$ is the sub-matrix formed by the last $\mathsf{N}-\mathsf{M}$ columns of $\mathbf{V}_i$. Notice that the datasets in $[\mathsf{M}+1:\mathsf{N}]$ are placed in a cyclic assignment among the servers in $[\mathsf{M}+1:\mathsf{N}]$. Each server cannot compute $\mathsf{N}-\mathsf{M}-\frac{\mathsf{M}-1}{2}$ of the datasets in $[\mathsf{M}+1:\mathsf{N}]$. By extracting the corresponding columns from the matrix $\mathbf{F_s}$, we form a submatrix $\mathbf{F}_{\mathbf{s}_n}$, with dimensions $\left(\frac{\mathsf{M}+1}{2} - \mathsf{y} + \mathsf{m}\right) \times \mathsf{m}\left(\mathsf{N} - \frac{3\mathsf{M}-1}{2}\right)$. Our desired vector s_n is then the left null vector of this matrix. If the number of rows in the matrix $\mathbf{F}_{\mathbf{s}_n}$ is greater than the number of columns, we can directly assign random values to the entries denoted by $*$ as in [10] to obtain $\mathbf{s}_n$. If the number of rows is less than or equal to the number of columns, we apply the interference alignment strategy from [19] to design the values of the $*$ entries in order to obtain $\mathbf{s}_n$.

Finally, we focus on the servers in $[\mathsf{y}+1:\mathsf{M}]$. We design their transmissions so that the user can recover $2F_1 - F_2, 2F_3 - F_4, \ldots, 2F_{2\mathsf{m}-1} - F_{2\mathsf{m}}, F_{2\mathsf{m}+1}, \ldots, F_{\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}}$ from any $\frac{\mathsf{M}+1}{2} - \mathsf{y} + \mathsf{m}$ responses. For $n \in [\mathsf{y}+1, \mathsf{M}]$, we also let server n compute

$$\mathbf{s}_n[2F_1 - F_2; \ldots; 2F_{2\mathsf{m}-1} - F_{2\mathsf{m}}; F_{2\mathsf{m}+1}; \ldots; F_{\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m}}] \times [W_{1,1}; W_{2,1}; \ldots; W_{\mathsf{N},\mathsf{m}}]. \tag{29}$$

We design $\mathbf{s}_n$ as follows. Notice that $W_1, \ldots, W_{\frac{\mathsf{M}-1}{2}}$ can be computed by server n; and that in the linear combination (29) the coefficients of $W_{\frac{\mathsf{M}-1}{2}+1}, \ldots, W_{\mathsf{M}}$ are 0. Hence, in order to guarantee that in (29) the coefficients of the messages which server n cannot compute are 0, we only need to consider the messages in $W_{\mathsf{M}+1}, \ldots, W_{\mathsf{N}}$,corresponding to the columns in $\mathbf{C}_3$ in the matrix, with a total of $\mathsf{m}(\mathsf{N}-\mathsf{M})$ columns.By extracting the corresponding rows and columns from the matrix $\mathbf{F}'$, we obtain $\mathbf{F}'_{\mathbf{s}}$, as shown at the top of the next page in (30). It is easy to observe that $\mathbf{F}'_{\mathbf{s}} = \mathbf{F_s}$.Notice that the datasets in $[\mathsf{M}+1, \mathsf{N}]$ are also cyclically assigned

$$\mathbf{F}'_{\mathbf{s}} = \left[\begin{array}{c:c:c:c} \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} \\ \hdashline \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} \\ \hdashline \vdots & \vdots & \ddots & \vdots \\ \hdashline \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \cdots & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} \\ \hdashline (\mathbf{V}'_1)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} & (\mathbf{V}'_2)_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} & \cdots & (\mathbf{V}'_{\mathsf{m}})_{(\frac{\mathsf{M}+1}{2}-\mathsf{y})\times(\mathsf{N}-\mathsf{M})} \end{array}\right]_{(\frac{\mathsf{M}+1}{2}-\mathsf{y}+\mathsf{m})\times\mathsf{m}(\mathsf{N}-\mathsf{M})}. \tag{30}$$

to the servers in $[\mathsf{y}+1:\mathsf{M}]$. For $n \in [\mathsf{y}+1:\mathsf{M}]$, server n cannot compute $\mathsf{N}-\frac{3\mathsf{M}+1}{2}$ datasets from $[\mathsf{M}+1,\mathsf{N}]$, and these datasets are also not computable by server $n+\mathsf{M}-\mathsf{y}$. Since $\mathbf{F}'_{\mathbf{s}} = \mathbf{F}_{\mathbf{s}}$, we can set $\mathbf{s}_n = \mathbf{s}_{n+\mathsf{M}-\mathsf{y}}$.

Decoding phase. The user receives responses from $N_{\mathrm{r}} = \mathsf{N}-\mathsf{M}+\mathsf{m}$ servers, of which x responses are from servers in $[\mathsf{y}]$. Depending on the value of x, we can categorize the scenarios into the following two cases:

- *Case 1:* $\mathsf{x} \leq \mathsf{m}$. In this case, the user receives x linearly independent combinations sent by the servers in $[\mathsf{y}]$. The user still needs to collect an additional $2\mathsf{m}-\mathsf{x}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations from any $(\mathsf{N}-\mathsf{M}-\mathsf{x}+\mathsf{m})$ servers in $[\mathsf{y}+1:\mathsf{N}]$. In the worst-case scenario, the servers in one of the groups, either $[\mathsf{y}+1:\mathsf{M}]$ or $[\mathsf{M}+1:\mathsf{N}]$, all respond, transmitting a total of $\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations. Additionally, $\mathsf{m}-\mathsf{x}$ servers from the other group respond, providing another $\mathsf{m}-\mathsf{x}$ linearly independent combinations.
  Together, this results in a total of $2\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations, sufficient to fully recover the demand matrix $\mathbf{F}'$.
- *Case 2:* $\mathsf{m} < \mathsf{x} \leq \mathsf{y}$. In this case, the user receives m linearly independent combinations sent by the servers in $[\mathsf{y}]$. We consider the worst-case scenario, where $x = \mathsf{y}$, meaning that the user still needs to collect an additional $\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations from any $2\mathsf{N}-3\mathsf{M}+\mathsf{m}$ servers in $[\mathsf{y}+1:\mathsf{N}]$. Since $\mathsf{N} \geq \frac{3\mathsf{M}+1}{2}$, it follows that $2\mathsf{N}-3\mathsf{M}+\mathsf{m} \geq \mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$. Additionally, any $\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ servers in either of the groups $[\mathsf{y}+1:\mathsf{M}]$ or $[\mathsf{M}+1:\mathsf{N}]$ compute linearly independent combinations. Therefore, it is possible to receive $\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations from any $2\mathsf{N}-3\mathsf{M}+\mathsf{m}$ servers in $[\mathsf{y}+1:\mathsf{N}]$.

Thus, the user obtains a total of $2\mathsf{m}+\mathsf{N}-\frac{3\mathsf{M}-1}{2}$ linearly independent combinations, which suffices to reconstruct the demand matrix $\mathbf{F}'$.

By the above scheme, the number of linearly independent transmissions by all servers is equal to the number of rows in $\mathbf{F}'$, i.e., $\frac{\mathsf{M}+1}{2}-\mathsf{y}+2\mathsf{m} = \mathsf{N}-\frac{3\mathsf{M}-1}{2}+2\mathsf{m}$, coinciding with (14).

D. general scheme 4

In this section, we examine the case where $\mathsf{M} < \mathsf{N} < 1.5\mathsf{M}$, with $\mathsf{M} \geq 2\mathsf{m}$, and construct a recursive scheme (Scheme 4) that aims to demonstrate (15). This recursive scheme builds on the solution for the $(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$ non-secure problem, assuming that this solution has been previously developed, and provides a total of $h(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$ linearly independent combinations of messages.

Data assignment phase. We begin by allocating datasets $D_1, \ldots, D_{\mathsf{N}-\mathsf{M}}$ to each server in $[\mathsf{M}]$. Subsequently, datasets $D_{\mathsf{N}-\mathsf{M}+1}, \ldots, D_{\mathsf{N}}$ are assigned to each server in $[\mathsf{M}+1:\mathsf{N}]$, ensuring that each server in $[\mathsf{M}+1:\mathsf{N}]$ has access to M datasets, while each server in $[\mathsf{M}]$ holds fewer than M datasets, specifically $\mathsf{N}-\mathsf{M}$. Additionally, each dataset in $[\mathsf{N}-\mathsf{M}+1:\mathsf{N}]$ is assigned to $\mathsf{N}-\mathsf{M} < \mathsf{M}$ servers. Therefore, in the next step, we assign each dataset D_k for $k \in [\mathsf{N}-\mathsf{M}+1:\mathsf{N}]$ to $2\mathsf{M}-\mathsf{N}$ servers in $[\mathsf{M}]$, such that each server in $[\mathsf{M}]$ obtains $2\mathsf{M}-\mathsf{N}$ datasets in $[\mathsf{N}-\mathsf{M}+1:\mathsf{N}]$. This allocation follows the assignment phase of the proposed scheme for the $(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$ non-secure problem.

Computing phase. Each message W_k for $k \in [\mathsf{K}]$ is divided into m equal-length, non-overlapping sub-messages, noted as $W_k = \{W_{k,j} : j \in [\mathsf{m}]\}$. First, we focus on the $(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$ non-secure problem, where the messages are defined as $W'_1, \ldots, W'_{\mathsf{M}}$. Under this scheme, each server computes a linear combination of these M messages. Letting the total number of linearly independent combinations across all servers be $h(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$, we can represent these $h(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$ combinations as

$$\mathbf{F}' \; [W'_{1,1}; \ldots; W'_{\mathsf{M},\mathsf{m}}]. \tag{31}$$

Each transmission from a server $n' \in [\mathsf{M}]$ is represented as

$$\mathbf{s}_{n'} \; \mathbf{F}' \; [W'_{1,1}; \ldots; W'_{\mathsf{M},\mathsf{m}}].$$

Returning to the (N, M) non-secure problem, we set up the server transmissions so that, together, they produce $h(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$ linearly independent combinations, represented as $\mathbf{F}[W_{1,1}; \ldots; W_{\mathsf{N},\mathsf{m}}]$, where $\mathbf{F}$ is shown at the top of the next page in (32). Each row in the submatrix $\mathbf{A}$ contains identical random numbers. For each $i \in [\mathsf{m}]$, $\mathbf{V}_i$ shares elements with the corresponding elements in matrix $\mathbf{F}$, allowing matrix $\mathbf{F}$ to be viewed as a submatrix extracted from $\mathbf{F}'$ based on the columns in each $\mathbf{V}_i$.

For each server $n \in [\mathsf{M}]$, by the construction in the assignment phase, it has access to datasets $D_1, \ldots, D_{\mathsf{N}-\mathsf{M}}$, while the assignment for datasets $D_{\mathsf{N}-\mathsf{M}+1}, \ldots, D_{\mathsf{N}}$ is based on the assignment for the $(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$ non-secure problem. Thus, server n computes $\mathbf{s}_n \; \mathbf{F} \; [W_1; \ldots; W_{\mathsf{N}}]$, where $\mathbf{s}_n$ corresponds to the transmission vector of server n in the $(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$ problem.

For each server $n \in [\mathsf{M}+1:\mathsf{N}]$, it cannot compute $W_1, \ldots, W_{\mathsf{N}-\mathsf{M}}$, which corresponds to the columns of submatrix $\mathbf{A}$ in $\mathbf{F}$. Extracting these columns creates a matrix $\mathbf{F}''$ of size $h(\mathsf{M}, 2\mathsf{M}-\mathsf{N}) \times \mathsf{m}(\mathsf{N}-\mathsf{M})$, whose rank is m. Consequently, the left null space of $\mathbf{F}''$ includes $h(\mathsf{M}, 2\mathsf{M}-\mathsf{N})-\mathsf{m}$ independent vectors. We let the transmission vector $\mathbf{s}_n$ for

$$\mathbf{F} = \left[\begin{array}{c:c:c:c:c:c:c} \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{1}_{1\times\mathsf{M}} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} \\ \hdashline \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{1}_{1\times\mathsf{M}} & \cdots & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} \\ \hdashline \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \hdashline \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} & \mathbf{0}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{0}_{1\times\mathsf{M}} & \cdots & \mathbf{1}_{1\times(\mathsf{N}-\mathsf{M})} & \mathbf{1}_{1\times\mathsf{M}} \\ \hdashline (\mathbf{A})_{(h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times(\mathsf{N}-\mathsf{M})} & (\mathbf{V}_1)_{(h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times\mathsf{M}} & (\mathbf{A})_{(h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times(\mathsf{N}-\mathsf{M})} & (\mathbf{V}_2)_{(h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times\mathsf{M}} & \cdots & (\mathbf{A})_{(h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times(\mathsf{N}-\mathsf{M})} & (\mathbf{V}_\mathsf{m})_{(h(\mathsf{M},2\mathsf{M}-\mathsf{N})-\mathsf{m})\times\mathsf{M}} \end{array}\right]_{h(\mathsf{M},2\mathsf{M}-\mathsf{N})\times\mathsf{m}\mathsf{N}}. \tag{32}$$

each server n be a random linear combination of these vectors, using uniformly i.i.d. coefficients from $\mathbb{F}_\mathsf{q}$, so that each server n computes $\mathbf{s}_n\mathbf{F}[W_1;\ldots;W_\mathsf{N}]$.

Decoding phase. Suppose $\mathcal{A}$ denotes the set of responding servers with $|\mathcal{A}| = \mathsf{N}_\mathrm{r}$, $\mathcal{A}_1 = \mathcal{A}\cap[\mathsf{M}]$, and $\mathcal{A}_2 = \mathcal{A}\setminus[\mathsf{M}]$, where $|\mathcal{A}_2| \le \mathsf{N}-\mathsf{M} = \mathsf{N}_\mathrm{r}-\mathsf{m}$. When $\mathsf{M} \ge \mathsf{N}-\mathsf{M}+\mathsf{m} = \mathsf{N}_\mathrm{r}$, if $|\mathcal{A}_2| = 0$, the user can recover the first m rows of $\mathbf{F}$, leveraging the decodability of the $(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$ scheme.

When $0 < |\mathcal{A}_2| \le \mathsf{N}-\mathsf{M}$ and $\mathcal{A}_1$ alone does not suffice, the user receives at least m independent responses from $\mathcal{A}_1$. Let the linearly independent combinations from $\mathcal{A}_1$ be denoted by λ_1. In addition to the responses from $\mathcal{A}_1$, the user will receive responses from at least $h(\mathsf{M}, 2\mathsf{M}-\mathsf{N})-\lambda_1$ servers in $[\mathsf{M}+1 : \mathsf{N}]$. Since each server sends an independent combination of $h(\mathsf{M}, 2\mathsf{M}-\mathsf{N})-\mathsf{m}$, any $h(\mathsf{M}, 2\mathsf{M}-\mathsf{N})-\lambda_1$ responses in $[\mathsf{M}+1 : \mathsf{N}]$ are linearly independent with high probability. Combining with λ_1, this totals $h(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$ combinations, allowing the user to recover $\mathbf{F}\left[W_{1,1};\ldots;W_{\mathsf{N},\mathsf{m}}\right]$.

In summary, $h(\mathsf{N},\mathsf{M}) = h(\mathsf{M}, 2\mathsf{M}-\mathsf{N})$, consistent with (15).

Appendix C
The applicable range of Interference Alignment in general scheme 3

Recall that in (28), $\mathbf{F}_\mathbf{s}$ is a matrix of dimensions $\left(\frac{\mathsf{M}+1}{2}-\mathsf{y}+\mathsf{m}\right)\times\mathsf{m}(\mathsf{N}-\mathsf{M})$. We use $\mathsf{N}-\mathsf{M}$ workers to compute this required matrix, where each worker can compute $\frac{\mathsf{M}-1}{2}$ datasets. If we proceed with interference alignment individually for each worker as in Example 1, we should generate $(\mathsf{m}-1)\left(\mathsf{N}-\frac{3\mathsf{M}}{2}-\frac{1}{2}\right)$ linearly independent linear equations for each worker. Thus, in total, we generate $(\mathsf{N}-\mathsf{M})(\mathsf{m}-1)\left(\mathsf{N}-\frac{3\mathsf{M}}{2}-\frac{1}{2}\right)$ linear equations. The resulting matrix $\mathbf{E}$ has dimensions $\left[(\mathsf{N}-\mathsf{M})(\mathsf{m}-1)\left(\mathsf{N}-\frac{3\mathsf{M}}{2}-\frac{1}{2}\right)\right]\times\mathsf{m}(\mathsf{N}-\mathsf{M})$. By construction, these $(\mathsf{N}-\mathsf{M})(\mathsf{m}-1)\left(\mathsf{N}-\frac{3\mathsf{M}}{2}-\frac{1}{2}\right)$ linear equations are linearly independent, and thus, $\mathbf{E}$ is of full rank. The left null space of $\mathbf{E}^\mathsf{T}$ contains only $\mathsf{m}(\mathsf{N}-\mathsf{M})-(\mathsf{N}-\mathsf{M})(\mathsf{m}-1)\left(\mathsf{N}-\frac{3\mathsf{M}}{2}-\frac{1}{2}\right)$ linearly independent vectors. However, the number of rows of $\mathbf{F}_\mathbf{s}$ is $\left(\frac{\mathsf{M}+1}{2}-\mathsf{y}+\mathsf{m}\right) = \mathsf{N}-\frac{3\mathsf{M}}{2}+\frac{1}{2}+\mathsf{m}$. Therefore, when

$$\mathsf{m}(\mathsf{N}-\mathsf{M})-(\mathsf{N}-\mathsf{M})(\mathsf{m}-1)\left(\mathsf{N}-\frac{3\mathsf{M}}{2}-\frac{1}{2}\right) < \mathsf{N}-\frac{3\mathsf{M}}{2}+\frac{1}{2}+\mathsf{m},$$

i.e.,

$$\mathsf{m}(\mathsf{N}-\mathsf{M}-1) \le [(\mathsf{N}-\mathsf{M})(\mathsf{m}-1)+1]\left(\mathsf{N}-\frac{3}{2}\mathsf{M}-\frac{1}{2}\right),$$

we need to use the interference alignment computing scheme, which jointly aligns the interferences across different workers.