Journal of Computer Science

Research Article

Real MDS Codes for Error Correction: Construction and an Application on Federated Learning Pre Aggregation Validation

Ravi Varma B.[1], N. Suresh Babu[2], Santhosh Kumar K. P.[3] and Shailesh Sivan[4]

[1] *Department of Mathematics, S.N. College, University of Kerala, Kollam, Kerala, India*
[2] *Department of Mathematics, University College, University of Kerala, Thiruvananthapuram, Kerala, India*
[3] *Department of Computer Science, Sacred Heart College (Autonomous), Ernakulam, Kerala, India*
[4] *Department of Computer Science, Cochin University of Science and Technology, Ernakulam, Kerala, India*

Article history
Received: 15-07-2024
Revised: 31-10-2024
Accepted: 14-11-2024

Corresponding Author:
Ravi Varma B.
Department of Mathematics, S.N. College, University of Kerala, Kollam, Pin 691001, Kerala, India
Email: ravivarmarrr@gmail.com

Abstract: Real number codes play a crucial role in correcting errors within communication systems that involve the transmission of analog signals or continuous data. Among these codes, Maximum Distance Separable (MDS) codes stand out as highly efficient for error correction. The present research paper is on the construction of a class of real number MDS codes for any desired dimension. The main objective is to enhance the error-detection and correction capability of a communication system through the application of a real-number code which has the maximum distance and hence the maximum error-detection and correction capability. The paper introduces a practical and systematic approach for encoding and decoding of messages which are in the form of real arrays using the proposed real MDS code. Also, an application of a real MDS code in federated learning is provided and illustrated with a case study.

Keywords: Error Correcting Code, Real Number Code, MDS Code, Federated Learning, Machine Learning

Introduction

In the realm of communication systems, particularly those involving analog signals or continuous data transmission, real number error correcting codes play a pivotal role. The efficacy of error detection and correction using a code is intricately tied to its minimum distance, making codes with the maximum possible "minimum distance" particularly desirable. These codes, named Maximum Distance Separable (MDS) codes, are highly sought after for their unparalleled effectiveness in error detection and correction. The construction of an MDS code with a given length and dimension poses a significant research challenge in the theory of error correction coding. In fact, finite field MDS codes are rare and they exist only for certain parameters. This research paper addresses the construction of a real-number MDS code for any desired dimension. A novel construction method is used to obtain a linear MDS code over the real field for any given dimension. The contribution extends beyond the construction process, as a decoding procedure that is not only theoretically robust but also straight forward and practical for implementation is also provided. The main objective of this work is to maximize the error-detection and correction capability of a communication system by using a real number MDS code. This research has broader implications for improving the reliability of the communication systems, especially those dealing with analog signals or continuous data, where the robustness of error correction is utmost important. In today's world, data is everywhere. Data can provide valuable insights, but privacy issues and communication limits frequently prohibit us from consolidating it. Federated Learning (FL) is a new approach in machine learning which solves this problem by training a model jointly across many devices or servers without sharing the data directly. Each server trains a local model with own data and then sends only model changes to a central server, which combines them to enhance the global model.

The aggregation process is the heart of federated learning. It's where the local knowledge from individual servers is combined to create a more vigorous global model. There are different ways to perform aggregation, such as averaging or weighted averaging the updates from all servers. An application of the real MDS code in federated learning is provided in this work. Real MDS code is used to detect and correct errors introduced in the weight matrices of client models transmitted through a noisy communication channel.

Literature Survey

Research on error control codes for communication systems began with Shannon (1948). This work sparked the field of coding theory, dedicated to enhancing data transmission over noisy channels by recovering corrupted messages through advanced encoding and decoding techniques.

SCIENCE Publications

Error correcting codes were constructed to realize Shannon's fundamental theorem's ideas. The invention of Hamming codes (Hamming, 1950), was the turning point in the evolution of coding theory. These codes possess the ability to correct single errors and detect double errors. In his influential work, Hamming introduced fundamental concepts such as linear parity-check matrices and metrics, laying the foundation for modern error-correcting code theory.

Subsequently, Golay (1949), achieved a breakthrough by developing two codes with enhanced error correction capabilities, capable of correcting multiple bits simultaneously. The Reed-Muller (RM) codes were introduced by Muller (1954), and shortly afterwards Reed (1954) constructed an algorithm for decoding a message by correcting to half of its minimum distance. Afterwards, a major advancement in the theory of coding was independently made by Bose and Ray-Chaudhuri (1960); Hocquenghem (1959) when they introduced a generic technique for creating binary codes that could correct multiple random errors. Error-correcting codes have developed as a result of practical requirements in communication systems, where information is frequently sent across noise and interference. While the codes discussed thus far have primarily been defined over finite fields F_q, there are numerous applications where error correction codes based on real number field are essential.

The study of codes over the real number field is relatively new. The initial connection between codes over finite fields and real number analysis techniques was established in Wolf (1967). Although he didn't introduce codes over real number at this time, Wolf later revisited this work in subsequent studies. Marshall (1981) is credited with being the first to actively explore the concept of real number codes. Their paper initiated the basic groundwork for this field. Marshall found several advantages for real number codes over their finite field equivalents. First, he highlighted the growing availability of signal-processing technology capable of effectively handling real numbers. Marshall also realized the possibilities for combining source-channel coding with real number approaches. Marshall developed the notion of "bandwidth compression", which closely corresponds with the solutions mentioned in this article for decreasing overall quantization noise levels. Marshall (1984) presents the existence and qualities of codes over the real number field. Nair and Abraham (1990) explored the use of codes over real number field for fault-tolerant matrix operations. This paper illustrates how real number codes can reduce roundoff noise during these processes. Key contributions to the understanding and development of real number codes can be found in Chen (2009); Jose *et al*. (2006); Suresh *et al.* (2023); Raviv *et al*. (2018); Suresh Babu *et al*. (2024); Zhang and Pfister (2008) each offering valuable insights into this evolving field.

The Griesmer bound (Griesmer, 1960) establishes the maximum length of a linear code, thereby determining the existence of such a code. Subsequently, Solomon and Stiffler (1965); Belov (1974) identified simplex codes that satisfy the conditions of the Griesmer bound. Conversely, the Singleton bound, developed by Singleton (1964), provides a straightforward upper limit and defines a family of codes called Maximum Distance Separable (MDS) codes, including notable examples such as RS-codes. These results play a fundamental role in the theory of coding, for the design and analysis of codes for error correction used in various communication systems.

Fundamentals of Error Correction

This section provides the essential definitions and foundational results that are prerequisite for the subsequent sections, laying the groundwork for a comprehensive understanding of the topics explored in this paper. We refer to Hamming (1950); Ling and Xing (2003); Van Lint (1971) for these basic results.

Definition 1. Let F^n be a linear space of dimension n over a field F. An $[n, k]$ linear code over F is a k-dimensional subspace of F^n.

An $[n, k]$ linear code C over F is categorized as a finite field code when F is a finite field, and as a real number code when F is the real field $\mathbb{R}$.

Definition 2. If C is an $[n, k]$ linear code over F, the dual code of C denoted as $C^\perp$ is the orthogonal complement of C.

Clearly $C^\perp$ is itself a linear $[n, n - k]$ code over F.

Remark 1:

(i) A generator matrix of a linear code C is a matrix G whose rows form a basis for C
(ii) A parity-check matrix H of a linear code C is a generator matrix of the dual code $C^\perp$

Definition 3: If C is an $[n, k]$-linear code, then a generator matrix for C must be a $k \times n$ matrix and a parity-check matrix for C must be an $(n - k) \times n$ matrix.

Definition 4:

(i) A generator matrix of the form $(I_k \mid X)$ is said to be in standard form
(ii) A parity-check matrix in the form $(Y \mid I_{n-k})$ is said to be in standard form

Remark 2. If the generator matrix of C is in the standard form as $G = (I_k \mid X)$, then the parity check matrix is $H = (-X^T \mid I_{n-k})$.

Definition 5. Let C be an $[n, k]$ linear code. Then the (minimum) distance of C, denoted by d, is:

$$d = \min \{d_H (v, w): v, w \in C, v \neq \mathrm{w}\}$$

where d_H denotes the Hamming distance.

An $[n, k]$ linear code C with minimum distance d is usually denoted as $C\,[n, k, d]$.

Lemma 1. Let C be a linear code and let H be a parity-check matrix for C. Then the following statements are equivalent:

(i) C has distance d
(ii) any $d - 1$ columns of H are linearly independent and H has d columns that are linearly dependent

Lemma 2. Let C be an $[n, k, d]$ linear code. Then, C is a $(d - 1)$-error detecting and $\left[\frac{d-1}{2}\right]$-error correcting code.

The Singleton bound dictates that $d \leq n - k + 1$, and codes achieving equality with this bound are termed Maximum Distance Separable (MDS) codes (Singleton, 1964).

Definition 6. An $[n, k, d]$ linear error correcting code with $d = n - k + 1$ is said to be a maximum distance separable (MDS) code.

Thus, an MDS linear $[n, k, d]$ code detects $n - k$ errors and corrects $\left[\frac{n-k}{2}\right]$ errors in a received word.

Materials and Methods

This section presents a succinct procedural framework for constructing a Real number Maximum Distance Separable (MDS) code for any desired dimension. The result given below plays a vital role in the construction of a real MDS code.

Theorem 1. Let $1 \leq k \leq n$. Consider the matrix:

$$A = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 2 & 3 & \cdots & 1-k \\ 1^2 & 2^2 & 3^3 & \cdots & (n-k)^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1^{k-1} & 2^{k-1} & 3^{k-1} & \cdots & (n-k)^{k-1} \end{pmatrix}_{k \times (n-k)}$$

Let $H = (-A^T | I_{n-k})$, where I_{n-k} is the identity matrix. Then:

(i) Every square submatrix of A is non-singular
(ii) The rows of H are linearly independent
(iii) Any $(n - k)$ columns of H are linearly independent
(iv) H has $(n - k + 1)$ columns that are linearly dependent

Proof:

(i) Note that A is a specific instance of a Vandermonde matrix, and since each square submatrix of a Vandermonde matrix is non-singular, it follows that each square submatrix of A is also non-singular
(ii) The linear independence of the rows of the matrix H is evident from its structure
(iii) To prove that any $n - k$ columns of H are linearly independent, consider the following three cases:

- Case (i): If all $(n - k)$ columns belong to $-A^T$
  As every submatrix of A is invertible, these $(n - k)$ columns of $-A^T$ are inherently linearly independent
- Case (ii) If all $(n - k)$ columns belong to I_{n-k}
  This case follows a similar rationale to Case (i)
- Case (iii) If $(n - k - h)$ columns are chosen from $-A^T$ and h columns are chosen from I_{n-k} and $1 \leq h \leq (n - k - 1)$

Without loss of generality, let $v_1, v_2, ..., v_{n-k-h}$ be columns from $-A^T$ and $w_1, w_2, ..., w_h$ be columns from I_{n-k}. It is to be demonstrated that the columns $v_1, v_2, ..., v_{n-k-h}$, $w_1, w_2, ..., w_h$ are linearly independent. For that, on contrary, let us assume that $v_1, v_2, \ldots, v_{n-k-h}$, $w_1, w_2, ..., w_h$ are linearly dependent. In this case, there exist nonzero constants $\beta_1, \beta_2, ..., \beta_{n-k-h}, \gamma_1, \gamma_2, ... \gamma_h$ such that:

$$\beta_1 v_1 + \beta_2 v_2 + \cdots + \beta_{n-k-h} v_{n-k-h} + \gamma_1 w_1 + \cdots + \gamma_h w_h = 0$$

$$i.e. \beta_1 v_1 + \beta_2 v_2 + \cdots + \beta_{n-k-h} v_{n-k-h} = -\gamma_1 w_1 - \gamma_2 w_2 - \cdots - \gamma_h w_h$$

Since $w_1, w_2, ..., w_h$ are columns of I_{n-k}, at most h components of the vector of length $(n - k)$ in the right side are non-zero, and the other $n - k - h$ must be zero. Hence, we can construct an $(n - k - h) \times (n - k - h)$ sub matrix of A which is not invertible, leading to a contradiction. Therefore, our assumption is wrong. Thus, every $n - k$ columns of $H(\lambda_1)$ are linearly independent.

(iv) Now it is to be proved that there exists $n - k + 1$ columns in H which are linearly dependent.

If we write $H = (-A^T I_{n-k}) = (C_1 C_2 \cdots C_n)$, then we have:

$$C_2 + C_{k+1} + 2.C_{k+2} + 3.C_{k+3} + \cdots + (n - k).C_n = 0,$$

i.e. the above $n - k + 1$ columns $C_2, C_{k+1}, \cdots, C_n$, of H are linearly dependent.

Construction Procedure

In the realm of finite field codes, not every parameter set $[n, k]$ admits the existence of MDS codes. But, in the context of real number codes, *MDS* codes exist for any given parameters $[n, k]$. The following is the procedure for construction of a real number *MDS* code for any desired length and dimension.

Let n and k be the length and dimension respectively of the real number *MDS* code. We choose the parity check matrix H through which the desired *MDS* code C $[n, k, n - k + 1]$ is constructed. Let $H = (-A^T \ I_{n-k})$, where I_{n-k} is the identity matrix of order $n - k$ and:

$$A = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 2 & 3 & \cdots & (n-k) \\ 1^2 & 2^2 & 3^2 & \cdots & (n-k)^2 \\ 1^3 & 2^3 & 3^3 & \cdots & (n-k)^3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1^{k-1} & 2^{k-1} & 3^{k-1} & ... & (n-k)^{k-1} \end{pmatrix}_{k \times (n-k)} \qquad (1)$$

Proposition 1. Let C be the real linear $[n, k, d]$ code whose parity check matrix $H = (-A^T \ I_{n-k})$, where I_{n-k} is the identity matrix of order $n - k$ and A is the matrix in Equation 1. Then C is an MDS code over the real field.

Proof. We have:

$$\mathrm{H} = (-\mathrm{A}^{\mathrm{T}}\ \mathrm{I}_{\mathrm{n-k}})$$

$$= \begin{pmatrix} -1 & -1 & -1 & \cdots & -1 & 1 & 0 & 0 & 0 & \cdots & 0 \\ -1 & -2 & -2^2 & \cdots & -2^{k-1} & 0 & 1 & 0 & 0 & \cdots & 0 \\ -1 & -3 & -3^2 & \cdots & -3^{k-1} & 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ -1 & -(n-k) & -(n-k)^2 & \cdots & -(n-k)^{k-1} & 0 & 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

Clearly C is a linear code over R with parity check matrix H. By Lemma 1 and Theorem 1, C is an MDS code.

Error Correction Using Real MDS Codes

Investigating different directions in coding theory is an ongoing effort to enhance a code's error detection and correction capabilities. Each time a new error-correcting method is developed, researchers strive to improve the effectiveness of the code while exploring better encoding and decoding techniques. "Random errors" are those that frequently occur during transmission and are characterized by their independent and unpredictable nature. These issues are particularly prevalent in scenarios such as data storage and satellite communication channels. The error rates experienced depend on the specific characteristics of the communication channel.

Given a communication channel with an error rate 'p', we present the implementation procedure of a real MDS code corresponding to the channel. MDS codes are widely recognized for their optimal error-correction capabilities, providing maximum reliability in correcting random errors under various channel conditions. These codes are essential in ensuring data integrity and robustness in communication systems.

The implementation of an MDS code involves careful design and selection of parameters to meet the requirements of the communication channel. By utilizing the properties of MDS codes, one can achieve efficient error detection and correction, even if a significant amount of noise and interference is present in the communication channel. This section explores the practical aspects of implementing MDS codes and their significance in enhancing the reliability and performance of communication systems.

Procedure for Choosing the Generator and Parity Check Matrices of the Real MDS Code

Consider a communication channel with an error rate 'p'. To ensure a robust error-correction mechanism, assume that p<0.5. Suppose $p = \frac{s}{m}$, where $s < m$ and $(s, m) = 1$. This condition implies that at most s digits in a block of m digits of a transmitted word may be corrupted. Given message length as 'k', we aim to construct an MDS code 'C' with length '$n = mk$' and dimension 'k', capable of correcting '$t = ks$' errors. The generator matrix G of C is constructed as follows:

$$G = (I_k\ A) = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 1 & 1 & 1 & ... & 1 \\ 0 & 1 & 0 & 0 & \cdots & 1 & 2 & 3 & ... & (n-k) \\ 0 & 0 & 1 & 0 & \cdots & 1 & 2^2 & 32 & ... & (n-k)^2 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots & \vdots & ... & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1^{k-1} & 2^{k-1} & 3^{k-1} & \cdots & (n-k)^{k-1} \end{pmatrix}_{k \times n}$$

The corresponding parity check matrix H is given by:

$$H = (-A^T\ I_{n-k})$$

$$= \begin{pmatrix} -1 & -1 & -1 & \cdots & -1 & 1 & 0 & 0 & 0 & \cdots & 0 \\ -1 & -2 & -2^2 & \cdots & -2^{k-1} & 0 & 1 & 0 & 0 & \cdots & 0 \\ -1 & -3 & -3^2 & \cdots & -3^{k-1} & 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ -1 & -(n-k) & -(n-k)^2 & \cdots & -(n-k)^{k-1} & 0 & 0 & 0 & 0 & \cdots & 1 \end{pmatrix}_{(n-k) \times n}$$

This procedure ensures the design of an MDS code tailored to the specific characteristics of the communication channel, maximizing its error-correction capabilities while minimizing the impact of symbol errors during transmission.

Encoding Using Real MDS Code C

Consider the real MDS code, $C\ [n, k, d]$. Let G and H represent the corresponding generator and parity check matrices of C respectively. For the given message vector

$U = [u_1 u_2 \cdots u_k] \in \mathbb{R}^k$, the expression below represents a code word in *C:*

$$\begin{aligned} X &= UG \\ &= [x_1\ x_2 \cdots x_n\ ] \end{aligned}$$

Conversely, any code word *X* belonging to *C* can be uniquely expressed as *X* = *UG*, where $U = [u_1 u_2 \cdots u_k] \in \mathbb{R}^k$. Consequently, each word *U* in $\mathbb{R}^k$ can be encoded as *X* = *UG*. This process, wherein the elements *U* of $\mathbb{R}^k$ are represented as code words *X* = *UG* within the code *C*, is referred to as encoding.

Decoding With Real MDS Code C

The efficiency of a code heavily relies on the effectiveness of its decoding mechanism. This section, presents a straightforward yet sophisticated approach for decoding a real MDS code. Consider a scenario where the vector *X* is corrupted in at most t places, and the indices of corruption are represented by the set $I \subset \{1, ..., n\}$. The received vector *Y* can be expressed as:

$$Y = X + e$$

Where *e* is an error vector with at most *t* non-zero values at positions indicated by *I*. Upon multiplying the received vector *Y* by $H^{T,}$ we obtain:

$$S = YH^T = (X + e)H^T = UGH^T + eH^T = eH^T$$

The vector *S* is termed as the syndrome, and owing to the orthogonality of the columns of *H* with the rows of *G*, the syndrome is solely dependent on the error vector *e*. Given the received vector *Y*, we can calculate the syndrome *S*.

If *S* = 0, we infer that there is no error; however, $S \neq 0$ indicates the presence of errors, facilitating error detection. The error detection process involves reconstructing the error vector e by solving the linear system of equations:

$$eH^T = S \tag{2}$$

The uniqueness of the solution and, consequently, the uniqueness of the error vector can be ensured if:

$$t < \frac{d}{2} = \frac{n-k+1}{2}$$

Remark 3. The real [*n*, *k*, *d*] MDS code *C* has the capability to detect $n - k$ errors and correct up to $\left\lfloor \frac{n-k}{2} \right\rfloor$ errors.

An Application of the Real Number MDS Code

This section provides an application of a real number *MDS* Code in Federated Learning.

Federated Learning

Federated Learning (FL) emerges as a transformative paradigm in machine learning, offering a collaborative approach to model training while safeguarding data privacy. Unlike traditional centralized methods where data is aggregated for training, FL empowers individual devices or participants, referred to as clients, to train models locally on their own data without directly sharing it. This approach unlocks significant advantages in scenarios where data privacy is paramount, encompassing healthcare, finance, and Internet-of-Things (IoT) applications.

The FL architecture (Fig. 1) typically revolves around three key entities: Clients, a coordinator, and a server. Clients, encompassing devices like smartphones, wearables, sensors, or any device capable of local computations, hold the decentralized data. The coordinator, acting as the central entity, orchestrates the training process. It broadcasts the initial model to the clients, aggregates their local updates, and iteratively updates the global model. The server facilitates communication between the coordinator and clients, enabling the exchange of model updates and necessary information.

The learning process in nodes unfolds in a well-defined sequence. Firstly, the coordinator broadcasts the initial model or the current model state to all participating clients. Each client then leverages its local data to train the downloaded model locally, updating the model parameters based on its specific data distribution. Subsequently, clients generate local updates encapsulating the information learned from their data, typically involving gradients or compressed representations of the model changes. These local updates are then securely uploaded to the server. Upon receiving local updates from participating clients, the server initiates the aggregation process. Depending on the chosen FL algorithm, various privacy-preserving techniques like averaging, secure aggregation, or federated averaging are employed to aggregate the received updates. This aggregated update is then utilized to update the global model at the coordinator. Finally, the coordinator transmits the updated global model back to the clients, and the entire process iterates for a predefined number of rounds or until convergence is achieved. This iterative process of local training, update generation, aggregation, and global model update fosters collaborative learning of a robust model while meticulously preserving individual data privacy. Federated learning paves the way for advancements in various domains, including healthcare, where collaborative analysis of patient data can lead to improved diagnoses and treatment plans without compromising patient privacy. In finance, FL can enable the development of personalized financial models while safeguarding sensitive financial data. Furthermore, FL holds immense potential in the realm of IoT, allowing devices to collaboratively learn and improve their functionalities without compromising the privacy of the data they collect. As research in FL continues to flourish, novel algorithms, communication protocols, and 8 privacy-enhancing techniques are being developed to address the challenges associated with scalability, communication efficiency, and robustness against adversarial attacks. Federated learning stands poised to revolutionize the landscape of machine learning, empowering collaborative model development while upholding the critical principle of data privacy.

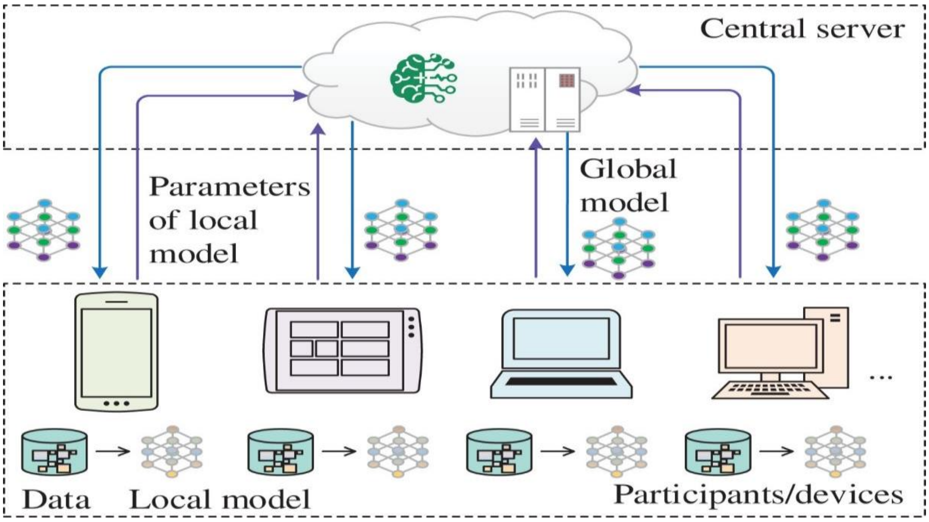

Fig. 1: Federated learning model

Model Aggregation in Federated Learning

Federated Learning (FL) empowers collaborative model training while preserving data privacy. Aggregation, where local updates from clients are combined, is crucial. For shallow neural networks at each client, various techniques exist:

Federated Averaging (Fed Avg)

Averages local weight and bias updates across clients for each layer, offering simplicity and efficiency but susceptible to poisoning attacks and potentially suboptimal for non-IID data.

Secure Aggregation

Employs cryptography (homomorphic encryption or secure multi-party computation) to perform aggregation on encrypted updates, enhancing privacy but increasing computational cost and communication overhead.

Federated Distillation

Utilizes knowledge distillation, where the global model acts as a teacher and local models as students. Clients train on their data and soft labels from the global model, potentially improving performance for non-IID data but introducing additional communication overhead. Choosing the right technique depends on privacy requirements, data distribution, computational resources, and communication overhead. Careful consideration of these factors and network architecture is essential for selecting the most suitable aggregation technique for your specific *FL* application.

Model Errors: Cause and Effects

Federated learning, while offering collaborative model training with data privacy, introduces vulnerabilities during weight matrix transmission between clients and the server. These vulnerabilities can be exploited through various attacks like eavesdropping, data poisoning, and man-in-the-middle attacks, potentially leading to compromised model performance and privacy breaches. Attacks can manifest as corrupted values, noise injection, or data leakage within the weight matrices, ultimately degrading the global model’s performance (reduced accuracy, increased generalization error) and potentially causing complete training failure. Additionally, sensitive client data might be leaked, raising ethical and legal concerns. Disrupted communication due to malicious attacks can hinder the entire training process.

To mitigate these risks, secure communication protocols like homomorphic encryption or secure multi-party computation can safeguard weight matrix confidentiality. Differential privacy, by adding controlled noise to updates, enhances model robustness against data poisoning. Furthermore, client authentication and monitoring mechanisms can help identify and potentially prevent malicious activities. By understanding these vulnerabilities and implementing appropriate mitigation strategies, researchers and practitioners can bolster the security and robustness of federated learning models, ensuring the integrity of the training process and protecting the privacy of participating clients.

Case Study

Detection and Correction for Pre-Aggregation Validation of Client Model Weights

After learning a local model with the data collected through the local client, the model weight needs to be sent to the central server as weight matrices. The chance of different attacks in the communication channel is often observed in such scenarios. These attacks usually try to manipulate the weight matrices to corrupt the model aggregation. So, server needs to validate these weight matrices before performing the model aggregation. The proposed method can be utilized for this purpose as shown in Figure 2.

Example 1. Consider a Neural network (Fig. 3) having 3 nodes each for input and output.

The corresponding weight matrix is a 3×3 matrix given by $M = \begin{bmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \\ W_{31} & W_{32} & W_{33} \end{bmatrix}$ which is to be transmitted through a noisy communication channel.

As a particular case, let $M = \begin{bmatrix} -1 & 7 & 3 \\ 10 & 3.5 & 0 \\ 8 & -2 & 4.3 \end{bmatrix}$ be the weight matrix to be transmitted through the communication channel and assume that error rate $p = 0.4$. We'll utilize the error correction method described earlier by constructing a real MDS code to transmit M.

Consider the rows of the weight matrix M as the message vectors. Represent:

$$M = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \end{bmatrix} = \begin{bmatrix} -1 & 7 & 3 \\ 10 & 3.5 & 0 \\ 8 & -2 & 4.3 \end{bmatrix}$$

Note that, here, the message vectors are U_1, U_2, and U_3 of length 3 such that $U_1 = [-1\ 7\ 3]$, $U_2 = [10\ 3.5\ 0]$ and U_3 $= [8 - 2\ 4.3]$. Additionally, with $p = 0.4 = \frac{2}{5}$, we have s $= 2$ and $m = 5$. Since $k = 3$, the length of the code is n = km = 3×5 = 15. Consequently, we construct a [15, 3] real MDS code (i.e., $d = 13$) to transmit M by sending a code word of length 15 for each row (message) one by one. Sine C is *MDS*, it can be used to detect up to 12 errors and correct up to 6 errors in any received word.

The generator matrix G is constructed as:

$$G = (I_3\ A) = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ 0 & 0 & 1 & 1 & 4 & 9 & 16 & 25 & 36 & 49 & 64 & 81 & 100 & 121 & 144 \end{pmatrix}$$

and so, the parity check matrix H is given by:

$$H = (-A^T\ I_{12}) = \begin{pmatrix} -1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -2 & -4 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -3 & -9 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -4 & -16 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -5 & -24 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -6 & -36 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -7 & -49 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & -8 & -64 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & -9 & -81 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -1 & -10 & -100 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & -11 & -121 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & -12 & -144 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Encoding

The message vectors $U_1 = [-1\ 7\ 3]$ $, U_2 = [10\ 3.5\ 0]$, and $U_3 = [8\ \text{-}2\ 4.3]$ can be encoded as X_1, X_2, and X_3, respectively, using the generator matrix G as in shown in Table 1.

Table 1: Encoding

Message Vector U_i	Code word $X_i = U_iG$
U_1 = [-1 7 3]	X_1 = [−1 7 3 9 25 47 75 109 149 195 247 305 369 439 515]
U_2 = [10 3.5 0]	X_2 = [10 3.5 0 13.5 17 20.5 24 27.5 31 34.5 38 41.5 45 48.5 52]
U_3 = [8 -2 4.3]	X_3 = [8 −2 4.3 10.3 21.2 40.7 68.8 105.5 150.8 204.7 267.2 338.3 418 506.3 603.2]

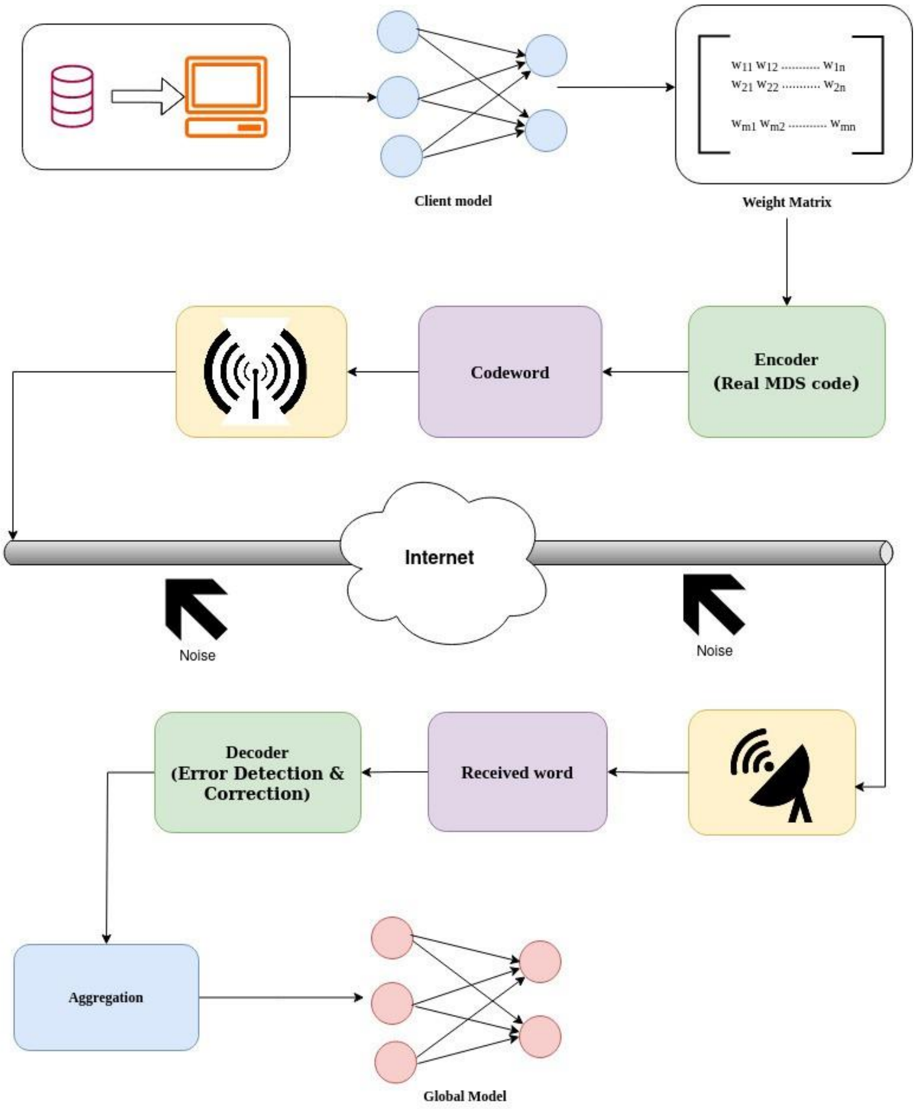

Fig. 2: Process flow of Pre-aggregation validation

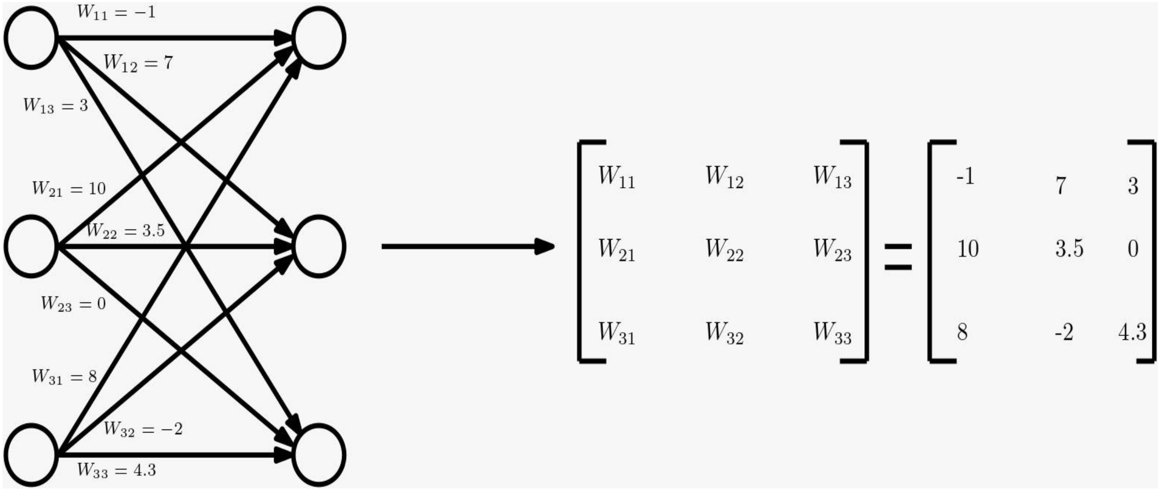
$W_{11} = -1$
$W_{12} = 7$
$W_{13} = 3$
$W_{21} = 10$
$W_{22} = 3.5$
$W_{23} = 0$
$W_{31} = 8$
$W_{32} = -2$
$W_{33} = 4.3$
$\begin{bmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \\ W_{31} & W_{32} & W_{33} \end{bmatrix} = \begin{bmatrix} -1 & 7 & 3 \\ 10 & 3.5 & 0 \\ 8 & -2 & 4.3 \end{bmatrix}$

Fig. 3: A sample neural network and weight matrix

The code words X_1, X_2, and X_3 are to be transmitted through the channel. Upon reception, let's denote the received vectors as Y_1, Y_2, and Y_3, respectively.

As a particular case suppose:

$$Y_1 = [-1\ 7\ 3\ 9\ 25\ 47\ 75\ 109\ 149\ 195\ 247\ 305\ 369\ 439\ 515]$$

$$Y_2 = [10\ 4.5\ 0\ 13.5\ 17\ 22\ 24\ 27.5\ 31\ 34.5\ 38\ 41.5\ 45\ 45.5\ 52]$$

$$Y_3 = [6\ 0\ 4.3\ 13.3\ 21.2\ 41\ 68.8\ 105.5\ 150.8\ 204.7\ 267.2\ 339\ 418\ 506.3\ 602]$$

Note that Y_1 has no error, Y_2 has 3 errors occurring at the 2nd, 6th and 14th positions and Y_3 has 6 errors occurring at the 1st, 2nd, 4th, 6th, 12th and 15th positions.

Decoding

Decoding involves two main steps: syndrome calculation (for error detection) and error correction.

Step I: Syndrome Calculation (Error Detection)

The first step in decoding is to calculate the syndrome for each received vector. The syndrome is a vector obtained by multiplying the received vector by the transpose of the parity check matrix *H*. It helps to detect whether errors have occurred during transmission. The syndromes of the received vectors are shown in Table 2.

Table 2: Decoding

Received Vector Y_i	Syndrome $S_i = Y_iH^T$
Y_1 = [−1 7 3 9 25 47 75 109 149 195 247 305 369 439 515]	S_1 = [0 0 0 0 0 0 0 0 0 0 0 0]
Y_2 = [10 4.5 0 13.5 17 22 24 27.5 31 34.5 38 41.5 45 45.5 52]	S_2 = [-1 -2 -1.5 -4 -5 -6 -7 -8 -9 -10 -14 -12]
Y_3 = [6 0 4.3 13.3 21.2 41 68.8 105.5 150.8 204.7 267.2 339 418 506.3 602]	S_3 = [3 -2 -3.7 -6 -8 -10 -12 -14 -15.3 -18 -20 -23.2]

From the syndromes, we can infer the following:

- Since $S_1 = 0$, the received word Y_1 has no error. Consequently, we can conclude that Y_1 itself represents the transmitted message vector X_1
- On the other hand, S_2 and S_3 are not zero vectors, indicating that errors have occurred in both. Therefore, we need to employ the error correction procedure to decode both Y_2 and Y_3

Step II: Error Correction

To correct the errors in Y_2 and Y_3, we need to find the error vector '*e*'.

The syndrome of the error, denoted as S_{error}, is obtained by multiplying the error vector 'e' by the transpose of the parity check matrix *H*. The syndrome helps to identify the error pattern. Once we have the syndrome S_{error}, we can use it to determine the error pattern and correct the received word. If the error vector is:

$$e = [e_1\ e_2\ e_3\ e_4\ e_5\ e_6\ e_7\ e_8\ e_9\ e_{10}\ e_{11}\ e_{12}\ e_{13}\ e_{14}\ e_{15}]$$

Then the syndrome of the error is:

$S_{\text{error}} = \text{e}H^{\text{T}} =$

$$\begin{matrix} [-e_1 - e_2 - e_3 + e_4 & -e_1 - 2e_2 - 4e_3 + e_5 & -e_1 - 3e_2 - 9e_3 + e_6 \\ -e_1 - 4e_2 - 16e_3 + e_7 & -e_1 - 5e_2 - 25e_3 + e_8 & -e_1 - 6e_2 - 36e_3 + e_9 \\ -e_1 - 7e_2 - 49e_3 + e_{10} & -e_1 - 8e_2 - 64e_3 + e_{11} & -e_1 - 9e_2 - 81e_3 + e_{12} \\ -e_1 - 10e_2 - 100e_3 + e_{13} & -e_1 - 11e_2 - 121e_3 + e_{14} & -e_1 - 12e_2 - 144e_3 + e_{15}]_{1X12} \end{matrix}$$

Error Correction in Y2

By equating S_{error} and S_2, we get:

$$\begin{matrix} -e_1 - e_2 - e_3 + e_4 = -1 & -e_1 - 2e_2 - 4e_3 + e_5 = -2 \\ -e_1 - 3e_2 - 9e_3 + e_6 = -1.5 & -e_1 - 4e_2 - 16e_3 + e_7 = -4 \\ -e_1 - 5e_2 - 25e_3 + e_8 = -5 & -e_1 - 6e_2 - 36e_3 + e_9 = -6 \\ -e_1 - 7e_2 - 49e_3 + e_{10} = -7 & -e_1 - 8e_2 - 64e_3 + e_{11} = -8 \\ -e_1 - 9e_2 - 81e_3 + e_{12} = -9 & -e_1 - 10e_2 - 100e_3 + e_{13} = -10 \\ -e_1 - 11e_2 - 121e_3 + e_{14} = -14 & -e_1 - 12e_2 - 144e_3 + e_{15} = -12 \end{matrix} \tag{3}$$

Solving the above system of equations, we find:

$$\begin{matrix} e_1 = 66x_1 - 120x_2 + 55x_3 - 360 & e_2 = -11.5x_1 + 22x_2 - 10.5x_3 + 67 \\ e_3 = 0.5x_1 - x_2 + 0.5x_3 - 3 & e_4 = 55x_1 - 99x_2 + 45x_3 - 297 \\ e_5 = 45x_1 - 80x_2 + 36x_3 - 240 & e_6 = 36x_1 - 63x_2 + 28x_3 - 187.5 \\ e_7 = 28x_1 - 48x_2 + 21x_3 - 144 & e_8 = 21x_1 - 35x_2 + 15x_3 - 105 \\ e_9 = 15x_1 - 24x_2 + 10x_3 - 72 & e_{10} = 10x_1 - 15x_2 + 6x_3 - 45 \\ e_{11} = 6x_1 - 8x_2 + 3x_3 - 24 & e_{12} = 3x_1 - 3x_2 + x_3 - 9 \\ e_{13} = x_1 & e_{14} = x_2 \\ e_{15} = x_3 & \end{matrix} \tag{4}$$

Since $t = 6$, at most six e_i's are non-zero. Solving the system we get the required solution:

$$e_1 = 0, e_2 = 1, e_3 = 0, e_4 = 0, e_5 = 0, e_6 = 1.5, e_7 = 0, e_8 = 0$$
$$e_g = 0, e_{10} = 0, e_{11} = 0, e_{12} = 0, e_{13} = 0, e_{14} = -3, e_{15} = 0$$

This provides the transmitted code word:

$X_2 = Y_2 - \text{e}$

$$= [10\ 4.5\ 0\ 13.5\ 17\ 22\ 24\ 27.5\ 31\ 34.5\ 38\ 41.5\ 45\ 45.5\ 52\ ] -$$
$$[0\ 1\ 0\ 0\ 0\ 1.5\ 0\ 0\ 0\ 0\ 0\ 0\ 0 - 3\ 0\ ]$$
$$= [10\ 3.5\ 0\ 13.5\ 17\ 20.5\ 24\ 27.5\ 31\ 34.5\ 38\ 41.5\ 45\ 48.5\ 52\ ]$$

Error Correction in *Y*3

By equating S_{error} and S_3, we get:

$$\begin{matrix} -e_1 - e_2 - e_3 + e_4 = 3 & -e_1 - 2e_2 - 4e_3 + e_5 = -2 \\ -e_1 - 3e_2 - 9e_3 + e_6 = -3.7 & -e_1 - 4e_2 - 16e_3 + e_7 = -6 \\ -e_1 - 5e_2 - 25e_3 + e_8 = -8 & -e_1 - 6e_2 - 36e_3 + e_9 = -10 \\ -e_1 - 7e_2 - 49e_3 + e_{10} = -12 & -e_1 - 8e_2 - 64e_3 + e_{11} = -14 \\ -e_1 - 9e_2 - 81e_3 + e_{12} = -15.3 & -e_1 - 10e_2 - 100e_3 + e_{13} = -18 \\ -e_1 - 11e_2 - 121e_3 + e_{14} = -20 & -e_1 - 12e_2 - 144e_3 + e_{15} = -23.2 \end{matrix} \tag{5}$$

Solving the above system of equations, we find:

$$\begin{aligned}
&e_1 = 66x_1 - 120x_2 + 55x_3 + 64 && e_2 = -11.5x_1 + 22x_2 - 10.5x_3 - 10.6\\
&e_3 = 0.5x_1 - x_2 + 0.5x_3 + 0.6 && e_4 = 55x_1 - 99x_2 + 45x_3 + 57\\
&e_5 = 45x_1 - 80x_2 + 36x_3 + 43.2 && e_6 = 36x_1 - 63x_2 + 28x_3 + 33.9\\
&e_7 = 28x_1 - 48x_2 + 21x_3 + 25.2 && e_8 = 21x_1 - 35x_2 + 15x_3 + 18\\
&e_9 = 15x_1 - 24x_2 + 10x_3 + 12 && e_{10} = 10x_1 - 15x_2 + 6x_3 + 7.2\\
&e_{11} = 6x_1 - 8x_2 + 3x_3 + 3.6 && e_{12} = 3x_1 - 3x_2 + x_3 + 1.9\\
&e_{13} = x_1 && e_{14} = x_2\\
&e_{15} = x_3
\end{aligned} \tag{6}$$

Since t = 6, at most six e_i's are non-zero. Solving the system we get the required solution:

$$\begin{gathered}
e_1 = -2, e_2 = 2, e_3 = 0, e_4 = 3, e_5 = 0, e_6 = 0.3, e_7 = 0, e_8 = 0\\
e_9 = 0, e_{10} = 0, e_{11} = 0, e_{12} = 0.7, e_{13} = 0, e_{14} = 0, e_{15} = -1.2
\end{gathered}$$

This provides the transmitted code word:

$$\begin{aligned}
&\mathrm{X}_3 = \mathrm{Y}_3 - \mathrm{e}\\
&= [6\ \ 0\ \ 4.3\ \ 13.3\ \ 21.2\ \ 41\ \ 68.8\ \ 105.5\ \ 150.8\ \ 204.7\ \ 267.2\ \ 339\ \ 418\ \ 506.3\ \ 602] -\\
&[-2\ \ 2\ \ 0\ \ 3\ \ 0\ \ 0.3\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0.7\ \ 0\ \ 0\ \ -1.2]\\
&= [8\ \ -2\ \ 4.3\ \ 10.3\ \ 21.2\ \ 40.7\ \ 68.8\ \ 105.5\ \ 150.8\ \ 204.7\ \ 267.2\ \ 338.3\ \ 418\ \ 506.3\ \ 603.2]
\end{aligned}$$

Hence, the received vectors Y_1, Y_2 and Y_3 are respectively decoded correctly as:

$$\begin{aligned}
X_1 &= [-1\ \ 7\ \ 3\ \ 9\ \ 25\ \ 47\ \ 75\ \ 109\ \ 149\ \ 195\ \ 247\ \ 305\ \ 369\ \ 439\ \ 515],\\
X_2 &= [10\ \ 3.5\ \ 0\ \ 13.5\ \ 17\ \ 20.5\ \ 24\ \ 27.5\ \ 31\ \ 34.5\ \ 38\ \ 41.5\ \ 45\ \ 48.5\ \ 52],\\
X_3 &= [8\ \ -2\ \ 4.3\ \ 10.3\ \ 21.2\ \ 40.7\ \ 68.8\ \ 105.5\ \ 150.8\ \ 204.7\ \ 267.2\ \ 338.3\ \ 418\ \ 506.3\ \ 603.2]
\end{aligned}$$

Discussion

The reliability of a communication system fundamentally depends on its error detection and correction capabilities. The capabilities are determined by the various parameters of the code. One of the most important parameters is minimum distance 'd' of the code, which is the minimum number of changes needed to convert one codeword into another. Larger the minimum distance, the maximum is the system's capability to detect and correct errors, making it desirable to construct codes with the maximum possible 'd' to ensure a reliable communication.

In this work, a real-number Maximum Distance Separable (MDS) code has been constructed, designed to maximize the error detection and correction. For a communication system characterized by an error probability $p(=\frac{s}{m})$ and a message length k, we developed a real-number [*n*, *k*, *d*] code, where *n* = km and $d = n - k + 1$. The result is a code that achieves the MDS property, meaning it can detect and correct the highest number of errors theoretically possible for the given parameters. This construction can be utilized to systems which transmit continuous or analog signals, encoding any message of length k into a real n-vector.

The properties of the constructed code ensure that it can detect up to $n - k$ random errors and correct up to $\left[\frac{n-k}{2}\right]$ random errors in a received word of length n. This feature makes the code optimal among all [*n, k*] linear codes, as it attains the maximum error detection and correction capability. The practical implications of this include increased reliability and data integrity in communication systems, even when noise or interference is present.

To demonstrate the applicability of real-number MDS codes, a case study was conducted in the context of federated learning, a domain where communication reliability is paramount. In this example, a real weight matrix of dimension 3×3 was transmitted through a noisy channel, with the error probability set at $p=0.4\left(=\frac{2}{3}\right)$ Here, the constructed real-number code was a [15, 3, 13] linear code, implying that the code length n was 15, the message length k was 3, and the minimum distance 'd' was 13. This configuration allowed the detection of up to 12 random errors and the correction of up to 6 random errors in a received word of length 15.

The simulation explored three scenarios:

- No Error Scenario: The received word had no error, and the decoding confirmed accurate data transmission
- Moderate Error Scenario: The received word contained 3 errors, which were successfully corrected, validating the code's detection and correction accuracy
- High Error Scenario: The received word had 6 errors the maximum that the code could correct and all errors were successfully recovered, demonstrating the code's optimal performance

These results underscore the robustness of the real-number MDS code in maintaining data integrity even in high-noise environments, which is critical for applications requiring reliable communication.

Limitations and Future Directions

While the proposed real-number MDS code offers optimal error detection and correction, the construction inherently results in longer code lengths. The larger code length could be seen as a drawback in scenarios where bandwidth is limited or when lower-latency communication is needed. However, given the significant computational power available today, this length is not a critical barrier. Future research could look into finding new constructions that minimize code length while preserving the same level of error correction, thus increasing efficiency without sacrificing performance.

Further research could also design specific coding strategies that adapt to varying channel conditions, which will maximize the communication system's performance under different noise environments. Additionally, the intersection of real-number MDS codes with emerging fields such as quantum communication and machine learning-based signal processing presents intriguing possibilities for enhancing data security and error resilience.

Overall, while the proposed MDS code framework has demonstrated strong theoretical and practical performance, there is ample room for innovation in refining its efficiency, exploring its implementation challenges, and expanding its applicability to new domains.

Conclusion

Real number codes are used for detecting and correcting errors in communication systems that handle analog signals or continuous data. Real number Maximum Distance Separable (MDS) codes are used for correcting maximum number of errors in such communication systems. This work proposes a class of real number MDS codes.

Through the design of a practical and systematic methods for encoding and decoding messages using the proposed real MDS codes, this work contributes significantly to the existing theory of error-correction codes in communication systems transmitting continuous data. Furthermore, the application of these real MDS codes is explored in federated learning to validate the real data transmitted. This underscores their versatility and relevance in contemporary technological landscapes.

As technology continues to evolve, the development and refinement of error-correction mechanisms remain paramount. The insights gleaned from this research not only offer theoretical advancements but also pave the way for practical implementations, fostering enhanced reliability and efficiency in communication systems and beyond. In essence, the exploration of real-number codes, particularly MDS codes, opens new vistas for innovation and optimization in the realm of error detection/correction methodologies, promising a more robust and resilient communication infrastructure for the future.

Acknowledgment

The author (s) would like to thank the anonymous reviewers for their valuable suggestions and constructive feedback, which have significantly contributed to the improvement of this manuscript.

Funding Information

The authors did not receive any monetary assistance for the investigation, composition, and/or publication of this manuscript.

Author’s Contributions

Ravivarma B.: Proved the results, carried out data analysis, constructed examples, and drafted and finalized the manuscript.

N. Suresh Babu: Conceived the research concepts, designed the research plan, verified the proofs of the results, and contributed to manuscript corrections.

Santhosh Kumar K. P.: Contributed to developing the revised methodology, performed additional data analysis, provided critical insights for revisions, and participated in the final proofreading.

Shailesh Sivan: Applied the constructed codes in Federated Learning, contributed to result validation, and assisted in manuscript preparation.

Ethics

This manuscript is an original work. The corresponding author certifies that co-authors have reviewed and approved the final version of the manuscript. No ethical concerns are associated with this submission.

Conflict of Interest

The authors hereby declare their complete independence from any organization or entity that may have a financial or non-financial interest in the subject matter or materials discussed in this manuscript.

References

Belov, B. I. (1974). A conjecture on the griesmer bound, Optimization Methods and Their Applications, (Russian). Optimization Methods and Their Applications (in Russian), 100–106.

Bose, R. C., & Ray-Chaudhuri, D. K. (1960). Further results on error correcting binary group codes. Information and Control, 3(3), 279–290. https://doi.org/10.1016/s0019-9958(60)90870-6

Chen, Z. (2009). Optimal real number codes for fault tolerant matrix operations. Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis, 1–10. https://doi.org/10.1145/1654059.1654089

Golay, M. J. E. (1949). Notes on digital coding. Proc IEEE 37, 657.

Griesmer, J. H. (1960). A bound for error-correcting codes. IBM Journal of Research and Development, 4(5), 532–542. https://doi.org/10.1147/rd.45.0532

Hamming, R. W. (1950). Error Detecting and Error Correcting Codes. Bell System Technical Journal, 29(2), 147–160. https://doi.org/10.1002/j.1538-7305.1950.tb00463.x

Hocquenghem, A. (1959). Codes correcteurs d’erreurs. Chiffers, 2, 147–156.

Jacobus, H. L. (1971). Coding theory. 201. Springer Nature Link. https://doi.org/10.1007/978-3-540-36657-7

Jose, V., Dorabella, S., & JSG, F. P. (2006). Error detection with real-number codes based on random matrices. IEEE, 526–530.

Ling, S., & Xing, C. (2003). Coding theory: a first course.

Marshall, T. G. (1981). Real number transform and convolutional codes. Proceedings 24th Midwest Symposium on Circuits and Systems, 650–653.

Marshall, T. (1984). Coding of Real-Number Sequences for Error Correction: A Digital Signal Processing Problem. IEEE Journal on Selected Areas in Communications, 2(2), 381–392. https://doi.org/10.1109/jsac.1984.1146063

Muller, D. E. (1954). Application of Boolean algebra to switching circuit design and to error detection. Transactions of the I.R.E. Professional Group on Electronic Computers, EC-3(3), 6–12. https://doi.org/10.1109/irepgelc.1954.6499441

Nair, V. S. S., & Abraham, J. A. (1990). Real-number codes for fault-tolerant matrix operations on processor arrays. IEEE Transactions on Computers, 39(4), 426–435. https://doi.org/10.1109/12.54836

Raviv, N., Tamo, I., Tandon, R., & Dimakis, A. G. (2020). Gradient Coding from Cyclic MDS Codes and Expander Graphs. IEEE Transactions on Information Theory, 66(12), 7475–7489. https://doi.org/10.1109/tit.2020.3029396

Reed, I. (1954). A class of multiple-error-correcting codes and the decoding scheme. Transactions of the IRE Professional Group on Information Theory, 4(4), 38–49. https://doi.org/10.1109/tit.1954.1057465

Singleton, R. (1964). Maximum distance q-nary codes. IEEE Transactions on Information Theory, 10(2), 116–118. https://doi.org/10.1109/tit.1964.1053661

Shannon, C. E. (1948). A Mathematical Theory of Communication. Bell System Technical Journal, 27(3), 379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

Solomon, G., & Stiffler, J. J. (1965). Algebraically punctured cyclic codes. Information and Control, 8(2), 170–179. https:/doi.org/10.1016/s0019-9958(65)90080-x

Suresh Babu, N., Ravivarma, B., Elsayed, E. M., & Sreekumar, K. G. (2025). Spectral codes of real symmetric operators for error correction. Discrete Mathematics, Algorithms and Applications, 17(04). https://doi.org/10.1142/s1793830924500551

Suresh, B., Ravivarma, B., Elsayed, E. M., & Sreekumar, K. G. (2023). Construction of a Class of Real Array Rank Distance Codes. Journal of Electrical and Computer Engineering, 2023, 1–9. https://doi.org/10.1155/2023/9952813

Wolf, J. (1967). Decoding of Bose-Chaudhuri-Hocquenhem codes and Prony's method for curve fitting (Corresp.). IEEE Transactions on Information Theory, 13(4), 608–608. https://doi.org/10.1109/tit.1967.1054056

Zhang, F., & Pfister, H. D. (2008). Compressed sensing and linear codes over real numbers. 2008 Information Theory and Applications Workshop, 558–561. https://doi.org/10.1109/ita.2008.4601055