Coding Scheme Extractor: A Tool for Constructing Codes Abiding by the Gilbert-Varshamov Bound

Name: Anshit, Rupesh, Tanmay (**ART**)

Roll No: ***2203304, 2203106,2203133***

Abstract

Error-correcting codes are fundamental for ensuring reliable data transmission and storage. The Gilbert-Varshamov (GV) bound provides a theoretical limit on the maximum achievable code size for a given alphabet size q, code length n, and minimum distance d. However, efficiently constructing codes that meet or exceed this bound remains a challenge. This project presents a Coding Scheme Extractor that constructs codes satisfying or exceeding the GV bound using various strategies, including exhaustive search, greedy approach and some heuristics. While brute-force methods are infeasible for large q and n, there are other techniques could reduce time but the algorithm still remains an NP Hard problem to solve and can't quickly solve for large q and n.

Contents

1. Introduction

Error-correcting codes introduce redundancy in data transmission to detect and correct errors, making them essential in digital communication, data storage, and cryptography . A fundamental problem in coding theory is to construct a code $C \subseteq F_q^n$ with the largest possible size $|C|$ while maintaining a minimum Hamming distance d between codewords. The Gilbert-Varshamov (GV) bound provides a lower bound on the maximum possible size of such a code, but efficiently constructing these codes remains a challenge.

1.1. Overview of the Problem Statement

Given:

- A finite field F_q of size q,
- Code length n,
- Minimum Hamming distance d,

Objective: Find the largest code C such that for all $C_i, C_j \in C$, $d(C_i, C_j) \geq d$.

To achieve this, we explore various techniques ahead. But before we dive deep into the topic. Let's discuss about the kind of question we came across in the process of implementing for this problem. First question what is the GV bound? Is there some easier way to implement this problem? How long would this problem take? Why is it that we need to search over the whole space is there not some general algorithm out there? What should we test against? Why are we doing this ? Does this help in some way to anyone? and all such weird questions.

2. Timeline

The extraction of a coding scheme abiding by the Gilbert-Varshamov (GV) bound requires a structured approach to ensure efficiency within the given timeframe. The following timeline outlines the planned tasks done weekly by each of us:

Week	**Tasks**
Week 1	- Problem selection and understanding the domain - Idea brainstorming - Resolving conceptual doubts
Week 2	- Literature review and paper reading - Algorithm implementation - Development phase
Week 3	- Performance evaluation - Testing and debugging - Finalizing documentation

Table 1: Timeline for Coding Scheme Extraction

Weekly meetings were held for discussing the development of the project so as to complete it on time before reaching the deadline. Tanmay and Anshit were assigned with the task of resolving the doubts, implementation of algorithms, reading research paper and doing the documentation. Rupesh was mainly tasked with development, testing and making the user-friendly interface of the problem.

3. The Gilbert-Varshamov Bound

3.1. Definition

The Gilbert-Varshamov bound states that there exists a code C over F_q^n of size $M = |C|$ satisfying:

$$M \geq \frac{q^n}{\sum_{i=0}^{d-1} \binom{n}{i}(q-1)^i} \tag{1}$$

which translates into a rate bound:

$$R = \frac{k}{n} \geq 1 - H_q\left(\frac{d}{n}\right) \tag{2}$$

where $H_q(x)$ is the q-ary entropy function:

$$H_q(x) = x \log_q(q-1) - x \log_q x - (1-x)\log_q(1-x) \tag{3}$$

This result implies that for sufficiently large n, codes exist that approach this bound .

3.2. Proof of the GV Bound

Let C be a code of length n and minimum Hamming distance d having maximal size:

$$|C| = A_q(n, d).$$

Then for all $x \in \mathbb{F}_q^n$, there exists at least one codeword $c_x \in C$ such that the Hamming distance $d(x, c_x)$ satisfies:

$$d(x, c_x) \leq d - 1$$

since otherwise, we could add x to the code whilst maintaining the code's minimum Hamming distance d—a contradiction on the maximality of $|C|$.

Hence, the whole of $\mathbb{F}_q^n$ is contained in the union of all balls of radius $d-1$ having their centre at some $c \in C$:

$$\mathbb{F}_q^n = \bigcup_{c \in C} B(c, d-1).$$

Now, each ball has size:

$$\sum_{j=0}^{d-1} \binom{n}{j} (q-1)^j$$

since we may allow (or choose) up to $d-1$ of the n components of a codeword to deviate (from the value of the corresponding component of the ball's centre) to one of

$(q-1)$ possible other values. Recall that the code is q-ary, meaning it takes values in $\mathbb{F}_q^n$. Hence, we deduce:

$$q^n = |\mathbb{F}_q^n| = \left| \bigcup_{c\in C} B(c, d-1) \right| \leq \sum_{c\in C} |B(c, d-1)| = |C| \sum_{j=0}^{d-1} \binom{n}{j} (q-1)^j.$$

That is:

$$A_q(n, d) = |C| \geq \frac{q^n}{\sum_{j=0}^{d-1} \binom{n}{j} (q-1)^j}.$$

4. Strategies for Constructing Codes Near the GV Bound

4.1. Exhaustive Search (Brute Force Approach)

This method systematically examines all subsets of possible codewords and checks which satisfy the minimum distance constraint and outputs the one with the largest size.

Complexity: $\mathcal{O}(nq^{2n(1-H_q(d/n))}2^{q^n})$, practically infeasible for large q, n.

4.2. Greedy Algorithm

A faster approach that iteratively selects codewords while ensuring they maintain the minimum distance constraint.

Algorithm:

1. Generate all possible codewords in F_q^n.
2. Sort them based on weight or diversity heuristics.
3. Iteratively add the best codewords, ensuring each maintains the minimum distance d.

Complexity: $\mathcal{O}(q^n \log q^n)$, significantly faster than brute force, but still exponential.

4.3. Altruistic Algorithm

The **Altruistic Algorithm** is an iterative method that refines the codebook by progressively removing elements to maximize the volume of spheres of radius $d-1$ in the search space. This ensures an optimal coding scheme under the Gilbert-Varshamov bound.

Algorithm:

1. Generate all possible codewords in F_q^n.
2. Sort them based on weight or diversity heuristics.
3. Iteratively add the best codewords, ensuring each maintains the minimum distance d and once you add this codeword remove all the codewords at distance less than d from this codeword , decreasing the search space.

4.4. Random Sampling Algorithm(Pre-seeding + Greedy Search)

The **Random Sampling Algorithm** is a straightforward method that generates candidate codes randomly and selects those that satisfy the Gilbert-Varshamov bound.

Algorithm:

1. Start with a pre-seed codebook containing codewords at minimum distance d from each other.

2. Randomly generate new codewords and check if they are at least distance d away from all existing codewords in the codebook.

3. If the generated codeword satisfies the distance condition, add it to the codebook; repeat until the desired codebook size is achieved.

The complexity of the algorithm is still exponential in q.

4.5. Algorithm for Linear Coding Scheme Using Randomization

This algorithm constructs a linear code over $\mathbb{F}_q$ (where q is a **prime power**) with parameters n (code length), k (dimension), and minimum distance $\geq d$, ensuring that it satisfies the Gilbert-Varshamov (GV) bound. .

Algorithm:

1. Randomly generate an $(n-k) \times k$ matrix P over the finite field F_q, and form the parity-check matrix $H = [P \mid I_{n-k}]$, where I_{n-k} is the $(n-k) \times (n-k)$ identity matrix.

2. Ensure all sets of r columns of H are linearly independent for $1 \leq r \leq d-1$. This guarantees that the smallest dependent set has at least d columns, ensuring the minimum distance satisfies $d_{\min} \geq d$.

3. Compute the generator matrix G as:

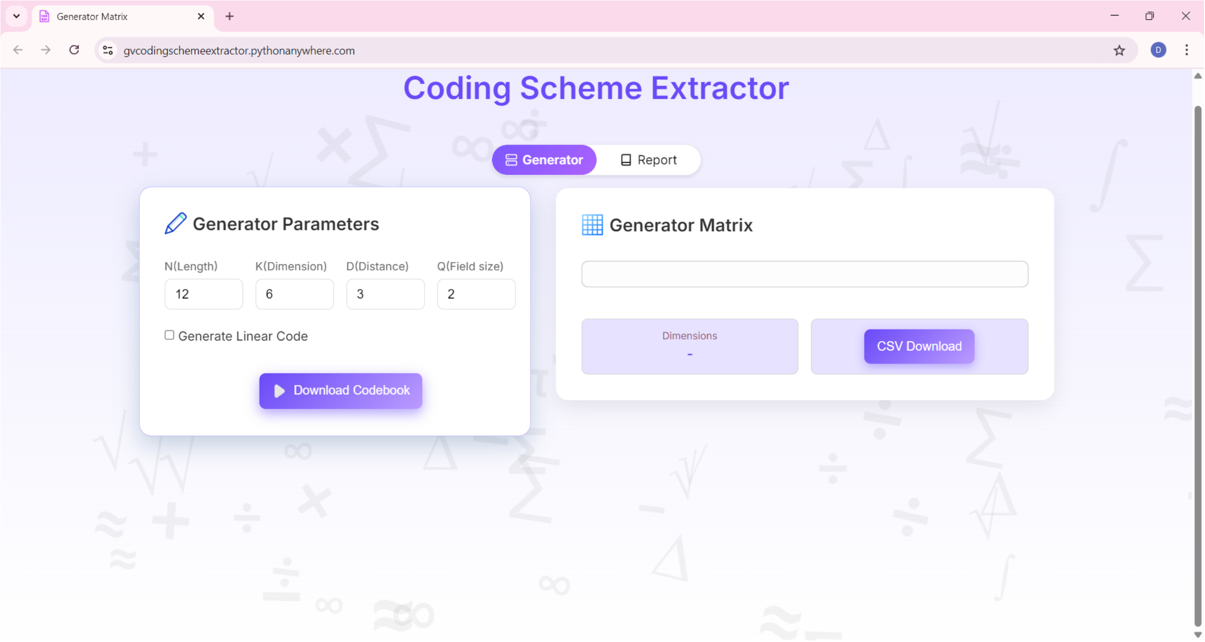
$$G = [I_k \mid -P^T],$$

   verifying $G \cdot H^T \equiv 0 \pmod{q}$.

Out of all the above mentioned strategies we came across, we decided to implement the last two algorithms for our problem statement as they are better than the other three. In the first three we would need to generate q^n codewords and save somewhere which will take a lot of space (exponential!!!), which we really don't want to. The Randomized algorithms are better in this sense that we don't have to atleast track this thing and can perform sometimes better. But there is a drawback the other three guaranteed a solution but the randomized ones don't within the given time(number of iterations). Randomized algorithms are still efficient than the other three mentioned.

Therefore, we selected Algorithms 4.4 and 4.5, where the latter is restricted to linear codes (requiring q to be a prime power) and the former applies to general codes (which may be linear or nonlinear). The reason for choosing a different algorithm for linear codes is that we can easily leverage the properties of linear codes to generate a generator matrix, thus avoiding the need to store q^k codewords.

5. Output



Generator Parameters

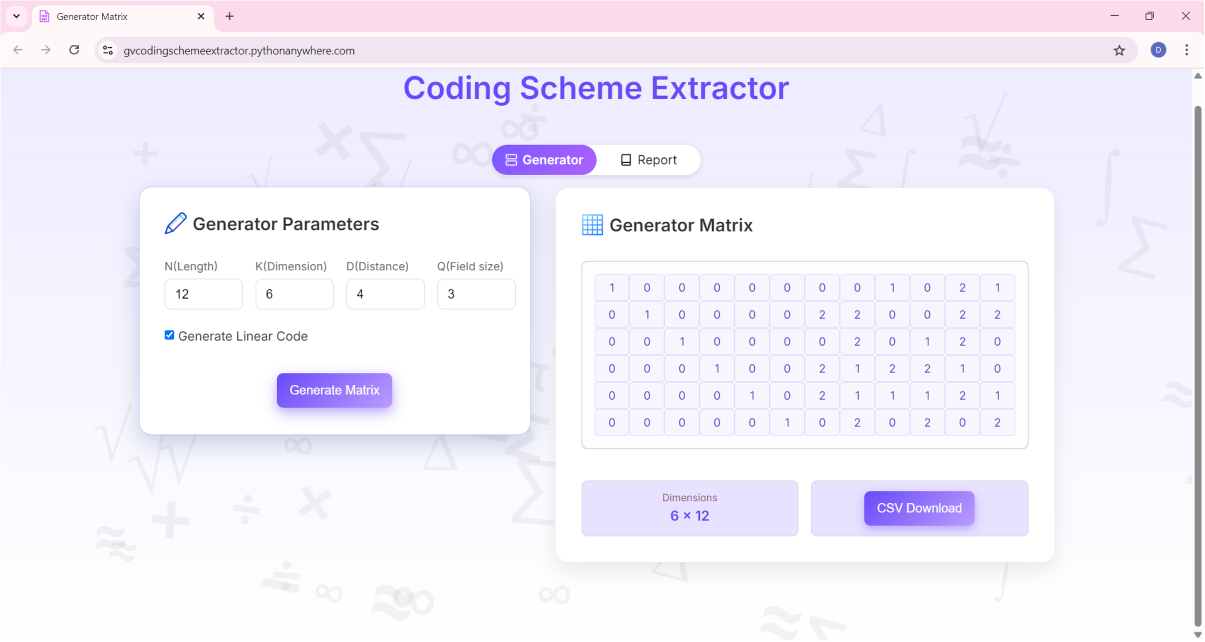
N(Length): 12
K(Dimension): 6
D(Distance): 3
Q(Field size): 2

☐ Generate Linear Code

Generator Matrix

Dimensions
-

(a) Generating general code



Generator Parameters

N(Length): 12
K(Dimension): 6
D(Distance): 4
Q(Field size): 3

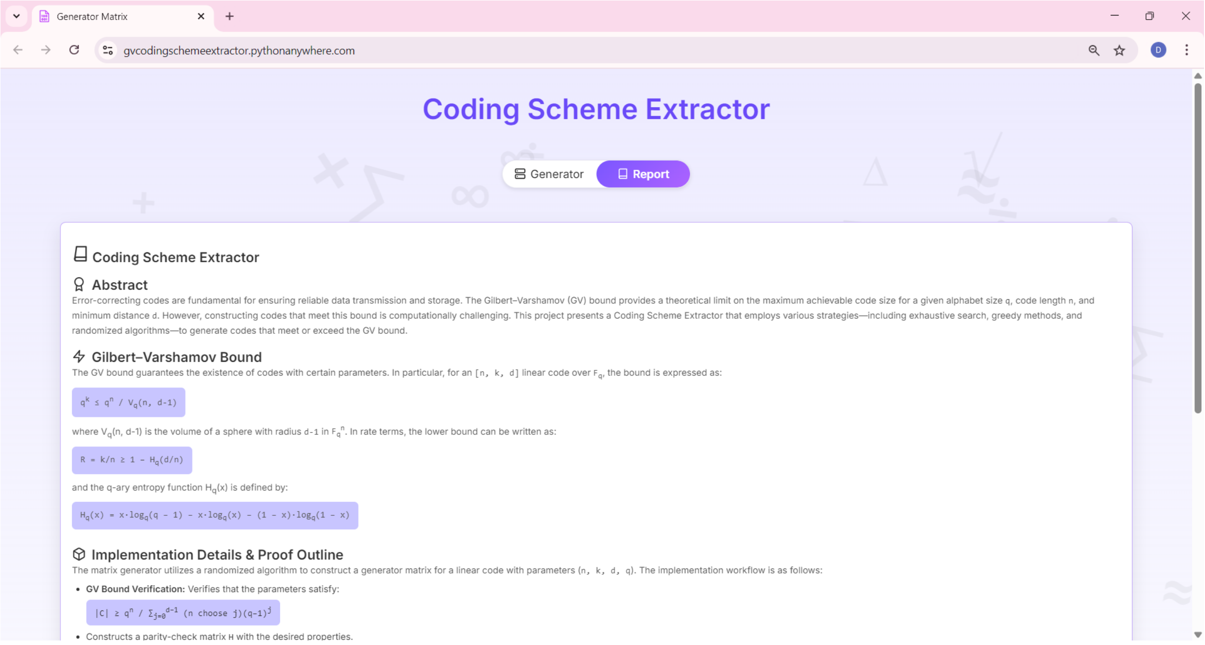
☑ Generate Linear Code

Generator Matrix

1	0	0	0	0	0	0	0	1	0	2	1
0	1	0	0	0	0	2	2	0	0	2	2
0	0	1	0	0	0	0	2	0	1	2	0
0	0	0	1	0	0	2	1	2	2	1	0
0	0	0	0	1	0	2	1	1	1	2	1
0	0	0	0	0	1	0	2	0	2	0	2

Dimensions
6 × 12

(b) Generating linear code



Coding Scheme Extractor

Abstract

Error-correcting codes are fundamental for ensuring reliable data transmission and storage. The Gilbert–Varshamov (GV) bound provides a theoretical limit on the maximum achievable code size for a given alphabet size q, code length n, and minimum distance d. However, constructing codes that meet this bound is computationally challenging. This project presents a Coding Scheme Extractor that employs various strategies—including exhaustive search, greedy methods, and randomized algorithms—to generate codes that meet or exceed the GV bound.

Gilbert–Varshamov Bound

The GV bound guarantees the existence of codes with certain parameters. In particular, for an [n, k, d] linear code over F_q, the bound is expressed as:

$$q^k \leq q^n / V_q(n, d-1)$$

where $V_q(n, d-1)$ is the volume of a sphere with radius d-1 in F_q^n. In rate terms, the lower bound can be written as:

$$R = k/n \geq 1 - H_q(d/n)$$

and the q-ary entropy function $H_q(x)$ is defined by:

$$H_q(x) = x \cdot \log_q(q - 1) - x \cdot \log_q(x) - (1 - x) \cdot \log_q(1 - x)$$

Implementation Details & Proof Outline

The matrix generator utilizes a randomized algorithm to construct a generator matrix for a linear code with parameters (n, k, d, q). The implementation workflow is as follows:

- **GV Bound Verification:** Verifies that the parameters satisfy:

  $$|C| \geq q^n / \Sigma_{j=0}^{d-1} (n \text{ choose } j)(q-1)^j$$

- Constructs a parity-check matrix H with the desired properties.

Figure 2: Report Section

Click here to visit our fully functional website.

6. Applications of Coding Schemes Abiding by the Gilbert-Varshamov Bound

The Gilbert-Varshamov (GV) bound provides a theoretical limit on the parameters of error-correcting codes, guiding the construction of codes with optimal trade-offs between code length, rate, and error correction capability. Coding schemes that adhere to the GV bound have numerous applications across different domains:

- Error Correction in Communication Systems
- Data Storage and Retrieval
- Cryptography and Secure Communications
- Biometric Authentication and Identification
- Coding Theory Research and Combinatorial Optimization

The construction of codes meeting the GV bound is fundamental to these applications, as it ensures a balance between redundancy and efficiency, making coding schemes both practical and theoretically optimal.

7. Conclusion and Future Scope

This report presents several algorithms for constructing error-correcting codes that satisfy the Gilbert-Varshamov (GV) bound. Among these, randomized strategies such as the Randomized Coding Schemes offer practical alternatives to brute-force methods, balancing performance with computational feasibility.

While deterministic and greedy methods guarantee correctness, they become impractical for large q and n. Randomized algorithms, in contrast, are easier to implement, scalable, and often yield good approximations with reduced overhead. These advantages make them powerful tools for generating codes within the GV bound.

Future research may explore hybrid strategies such as metaheuristic and genetic algorithms, which were beyond the scope of this project and therefore not implemented.

References

[1] J. H. van Lint, *Introduction to Coding Theory*, Springer-Verlag, 1999.

[2] Venkatesan Guruswami, Atri Rudra, Madhu Sudan, *Essential Coding Theory*, Cambridge University Press, 2012.

[3] *Gilbert-Varshamov Bound*, Available: https://en.wikipedia.org/wiki/Gilbert%E2%80%93Varshamov_bound

[4] *Gilbert-Varshamov Bound for Linear Codes*, Available: https://en.wikipedia.org/wiki/Gilbert%E2%80%93Varshamov_bound_for_linear_codes.

[5] Venkatesan Guruswami, *Coding Theory Lecture Notes*, Available: https://www.cs.cmu.edu/~venkatg/teaching/codingtheory/notes/notes2.pdf.

[6] Jian Gu and Tom Fuja, "A generalized Gilbert-Varshamov bound derived via analysis of a code-search algorithm," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 1089-1093, 1993.

[7] John Orth, "The Salmon Algorithm—A New Population Based Search Metaheuristic," 2012.

[8] Wolfgang Haas and Sheridan Houghten, "Evolutionary Algorithms for Optimal Error-Correcting Codes," *Brock University*, 2005.