Good Codes From Quasi-Cyclic Codes Using ConstructionX

Dev Akre '23, Matthew Harrington '22, Saurav Pandey '23, Faculty Advisor: Nuh Aydin
Department of Mathematics and Statistics, Kenyon College, Gambier OH

What is Coding Theory?

Every day, approximately 2.5 quintillion bits of data are created and transferred. Every one of these channels of transfer has some non-zero amount of **noise**: that is, there exists a probability that a bit may be flipped from a 1 to a 0 or vice versa. The noisier the channel, the higher this probability. This could be anything from your phone connecting with a cell tower (very little noise) to NASA communicating with a deep space probe (extremely high noise). Coding theory is the study of how we can transmit data in such a way that the original information can be recovered in the event that it is corrupted by noise.

Linear Codes

A **linear code** is a **vector space** over a **finite field**. It can be thought of as a set of **codewords** (or **vectors**) that is closed under the operation of vector addition. Linear codes are preferable to non-linear codes for a number of reasons, but a big one is that they require far less storage, as you only need to store a **basis** rather than every word in the code (see the term *generator matrix* below).

A linear code C is defined by three parameters:

1. **Length** n
   The total number of "letters" in each codeword of C.
   (The total number of bits in a codeword.)
2. **Dimension** k $(\leq n)$
   The number of basis elements for C
   (The number of "information" bits in a codeword".)
3. **Minimum Distance** d
   $min\{d(u,v) : u, v \in C, u \neq v\}$
   The smallest number of differences (Hamming Distance) between the positions of any two distinct codewords in C. It determines the error correcting capability of the code.

These are known as $[n,k,d]_q$ codes, where q is the size of the finite field $\mathbb{F}_q$ over which C is a vector space. Often, we fix n and k, and try to maximize d. A code with the best known d for its n and k is known as a **Best Known Linear Code** (BKLC). In general, this is a very hard problem because computing the minimum distance is **computationally intractable**, and the number of codes for a given length and dimension increases incredibly quickly, so we must be smart with which codes and classes of codes we choose to examine. A database of BKLCs is kept at the website `codetables.de`.

Important Terminology

- A **generator matrix** of a code is a matrix whose rows form a **basis** for the code.
- A **finite field** is an algebraic structure which is a finite set that with operations of addition, multiplication, subtraction and division. The size q of a finite field is a power of a prime. There exists a unique finite field for every q, denoted $\mathbb{F}_q$
- A code C_{sub} is a **subcode** of a code C if every word of C_{sub} is also in C. This can also be stated as saying C is a **supercode** of C_{sub}.

Cyclic Codes

Cyclic codes are perhaps the single most important class of linear codes, for a number of reasons:

- Many record breaking codes are cyclic.
- They have a number of useful generalizations.
- Computationally easier to encode/decode.
- They form a strong link between Coding Theory and Algebra.

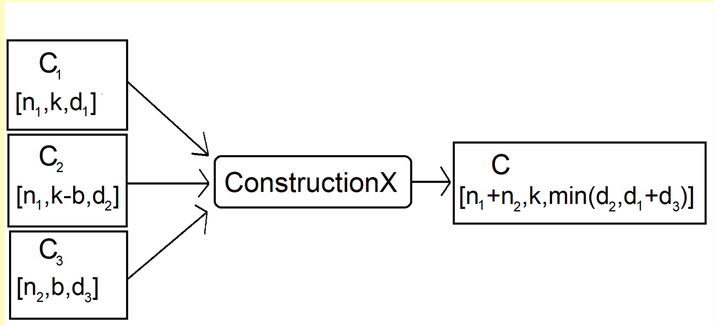
Definition 1.1 *A linear code $C \subseteq \mathbb{F}_q^n$ is said to be* **cyclic** *if it is closed under the* **cyclic shift**, *that is, if $(c_0, c_1, \cdots, c_{n-1}) \in C$, then $(c_{n-1}, c_0, c_1, \cdots, c_{n-2}) \in C$.*

For any cyclic code there exists a corresponding divisor of $x^n - 1$ which is unique. This is known as the **Generator Polynomial** of the code.

Abstract

One of the most important and challenging problems in coding theory is to construct codes with best possible parameters and properties. The class of quasi-cyclic (QC) codes is a generalization of cyclic codes known to be fertile to produce such codes. Using ConstructionX, a ternary operation on codes and some important properties of QC codes, we were able to find 33 codes with better parameters than any previously discovered code. In this presentation, we will explore what ConstructionX does, and how QC codes are well suited to it for use in producing new best known linear codes.

ConstructionX



ConstructionX is a **ternary operation** on codes. Given a code C_1 with parameters $[n_1, k_1, d_1]$, a subcode C_2 of it with parameters $[n_1, k_1 - b, d_2]$, and a third code C_3 with parameters $[n_2, b, d_3]$, ConstructionX attaches a codeword of C_3 onto each codeword of C_1 that is not in C_2. This results in a new code, C, with parameters $[n, k, d]$ such that $n = n_1 + n_3$, $k = k_1$, and $d_2 \geq d \geq \min\{d_2, d_1 + d_3\}$. C has a generator matrix of the form

$$\begin{bmatrix} G_1^* & G_3 \\ G_2 & 0 \end{bmatrix},$$

where G_2 is a gen matrix of C_2, G_3 is a gen matrix of C_3 and the rows of G_1^* are a set of linearly independent vectors of C_1 such that

$$\begin{bmatrix} G_1^* \\ G_2 \end{bmatrix}$$

generates C_1. The main problem in applying ConstructionX is finding the best possible C_2 while keeping b small. Smaller dimension codes will have better d than higher dimension codes, so there is a delicate balance between C_2 and C_3 that must be maintained. Generally, we find that the best codes come from a b between 1 and 6.

To find a good C_2, we must either start with C_1 and examine its subcodes, or start with a good code C_2 and examine its supercodes. However, finding good sub- or super- codes of a given code is difficult, as the number of sub- and super- codes a given code increases incredibly quickly. So we must find a smart way to go about examining sub- or super- codes.

Quasi-Cyclic Codes

Definition 1.2 *A linear code C is said to be ℓ-**quasi-cyclic** if for a positive integer ℓ, whenever $c = (c_0, c_1, ..., c_{n-1}) \in C$, it is also true that $(c_{n-\ell}, ..., c_{n-1}, c_0, c_1, ...c_{n-\ell-1}) \in C$.*

Quasi-Cyclic (QC) codes are known to contain many codes with good parameters.A QC code is made up of ℓ **blocks** of cyclic codes and each block has a **generator polynomial** that is a divisor of $x^n - 1$. We focus on QC codes with a generator of the form $(gf_1, gf_2, ..., gf_\ell)$ where $x^m - 1 = gh$. QC codes work well with ConstX beacuse we can easily obtain their sub and supercodes due to the following:

- For each divisor p of g, the QC code gen by $(g'f_1, g'f_2, ..., g'f_\ell)$ is a supercode of C, where $g' = \frac{g}{p}$
- For each divisor p of h, the QC code gen by $(pgf_1, pgf_2, ..., pgf_\ell)$ is a subcode of C.

The Algorithm

Algorithm 1: Finding ConstructionX Codes from Good QC Codes

```
Input: fs = [f_1, ..., f_ℓ];
Input: b;
g = gcd(fs);
C = QCcode(fs);
while Factors of g remain do
    factor = The next factor of g;
    if Degree(factor) ≠ b then
        continue;
    end
    newfs = [f/factor for f in fs];
    superC = QCcode(newfs);
    if MinimumDistance(Best) < MinimumDistance(superC) then
        Best = SuperC;
    end
end
for length to max do
    C3 = BKLC(length,b);
    CX = ConstructionX(C,Best,C3);
    print(CX);
end
Result: New high minimum distance codes from ConstructionX
```

Food for the Algorithm

As you may have noticed, this algorithm requires an input of QC codes. We obtained these in a number of different ways:

1. **ASR searches**. The ASR search algorithm was developed by Kenyon's own Prof. Aydin and is one of the best ways of finding QC codes. It has produced many record breakers in the past, so we did our own ASR searches and saved any codes with equal or better min dist than current record holders.
2. **Past years**. Many other students in previous summers had done ASR searches as well, and we had access to their data.
3. **Web scraping**. We wrote a web scraper in Python which combed `codetables.de` for record holding QC codes and downloaded them. We actually got our IP temporarily banned from the site at one point, since we were spamming them with around 500 requests per second and it looked identical to a DDOS attack.

Results

All told, the algorithm produced **33 new record breaking codes**, including **8 over the binary field**, which is way more than we were expecting. Below are a small sample of some of the record breaking codes we found and the codes that produced them.

Table 1: Record Breaking ConstructionX Codes

New Code	Original Code	Supercode	Third Code
$[98, 30, 26]_2$	$[96, 29, 26]_2$	$[96, 30, 24]_2$	$[2, 1, 2]_2$
$[97, 30, 25]_2$	$[96, 29, 26]_2$	$[96, 30, 24]_2$	$[1, 1, 1]_2$
$[99, 31, 26]_2$	$[96, 29, 26]_2$	$[96, 31, 24]_2$	$[3, 2, 2]_2$
$[98, 31, 25]_2$	$[96, 29, 26]_2$	$[96, 31, 24]_2$	$[2, 2, 1]_2$
$[177, 52, 41]_2$	$[170, 48, 42]_2$	$[170, 52, 38]_2$	$[7, 4, 3]_2$
$[178, 52, 42]_2$	$[170, 48, 42]_2$	$[170, 52, 38]_2$	$[8, 4, 4]_2$

A full list of codes, as well as all of the associated generators, can be obtained from our paper which is available on arxiv.org and the researchers will happily provide copies of it.

References

This project has too many references to be listed here. A full list can be obtained from the draft of the paper available at https://arxiv.org/abs/2108.06752