Designs, Codes and Cryptography
https://doi.org/10.1007/s10623-022-01124-1

A generalization of cyclic code equivalence algorithm to constacyclic codes

Dev Akre[1] · Nuh Aydin[1] · Matthew Harrington[1] · Saurav Pandey[1]

Received: 27 February 2022 / Revised: 28 June 2022 / Accepted: 15 September 2022

Abstract
Recently, a new algorithm to test equivalence of two cyclic codes has been introduced which is efficient and produced useful results. In this work, we generalize this algorithm to constacyclic codes. As an application of the algorithm we found many constacyclic codes with good parameters and properties. In particular, we found 22 new codes that improve the minimum distances of best known linear codes (BKLCs).

Keywords Quasi-cyclic codes · Best known codes · Reversible codes · LCD codes · Self-orthogonal codes

Mathematics Subject Classification 94B05 · 94B15

1 Introduction and motivation

A linear code C over a finite field $\mathbb{F}_q$ is a vector subspace of $\mathbb{F}_q^n$ and it has three fundamental parameters: the length (n), the dimension (k), and the minimum distance (d), and such a code is referred to as a $[n, k, d]_q$ code. One most of the most important problems in coding theory is the optimization of the minimum distance of a linear code. That is, for a given n and k, we seek the highest possible d. There exist theoretical upper bounds on d. A code attaining the upper bound for minimum distance is called (distance) optimal. It should be noted that currently best known theoretical upper bounds may actually be unattainable. One objective in

Communicated by G. Ge.

✉ Nuh Aydin
aydinn@kenyon.edu

Dev Akre
akre1@kenyon.edu

Matthew Harrington
harrington1@kenyon.edu

Saurav Pandey
pandey1@kenyon.edu

[1] Department of Mathematics and Statistics, Kenyon College, Gambier, OH 43022, USA

Published online: 10 October 2022

coding theory is to find codes whose minimum distances get as close to the optimal distance as possible. These are called BKLCs (best known linear codes). The online database [14] gives information about BKLCs over small finite fields $\mathbb{F}_q$, for $q \leq 9$ up to certain lengths, including lower and upper bounds on d. The upper bounds are theoretical, and lower bounds are obtained by explicit constructions. In general, optimal codes are known when either k or $n - k$ is small. This is because calculating the minimum distance is computationally intractable [24], and the number of linear codes of a given length and dimension is very large. Consequently, exhaustive searches are not feasible. Hence, we focus on special classes of codes that are promising.

In this paper, we focus on constacyclic (CC) codes which are a generalization of cyclic codes. Cyclic codes have a prominent place in coding theory for both theoretical and practical reasons. They provide a fundamental link between coding theory and algebra. Both cyclic and CC codes are used as building blocks in various search algorithms, particularly the ASR search algorithm, which have produced numerous record-breaking quasi-cyclic (QC) [5], quasi-twisted (QT) [4, 6, 8, 16], and multi-twisted (MT) codes [1, 2]. Furthermore, they are also used to produce quantum codes with good parameters [12, 18].

All of these algorithms benefit from having CC codes with high minimum weights. In a comprehensive implementation of the ASR algorithm, we start with examining all cyclic or CC codes of a given length. Since equivalent codes have the same parameters, it is redundant to use CC codes that are equivalent to each other. Testing the equivalence between two arbitrary linear codes is computationally expensive. However, an efficient algorithm that is specifically designed for cyclic codes based on cyclotomic cosets has been recently presented [7]. In this work, we generalize that algorithm to CC codes. Like the cyclic case, the resulting algorithm is faster than the general purpose equivalence test algorithm that is available in computer algebra systems like Magma. This enables us to conduct more extensive searches on QC, QT and MT codes.

We ran exhaustive searches for CC codes up to a certain dimension for all finite fields of size $\leq$ 9. We obtained a large number of CC codes with better parameters than the best known QT codes given in the database [10]. A significant number of these codes have additional desirable properties such as reversibility, self-orthogonality, and having linear complementary dual (LCD). Furthermore, we found a new code over $GF(7)$ with minimum distance 3 units higher than the current BKLC and obtained 20 additional new codes using the standard constructions on it. Another constacyclic code over $GF(5)$ produced a new code using construction X.

2 Basic definitions

We begin by defining constacyclic (CC) codes. Note that we will be using the usual convention of representing the codewords and vectors in $\mathbb{F}_q^n$ as polynomials in $\mathbb{F}_q[x]$:

$$\vec{c} = (c_0, c_1, \ldots, c_{n-1}) \in \mathbb{F}_q^n \leftrightarrow c(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1} \in \mathbb{F}_q[x].$$

Definition 1 A linear code C over $\mathbb{F}_q$ that is closed under a constacyclic shift π_a by a nonzero element $a \in \mathbb{F}_q$ is called a constacyclic code, that is, for any $c = (c_0, c_1, \ldots, c_{n-1}) \in C$, $\pi_a(c) = (a \cdot c_{n-1}, c_0, c_1, \ldots, c_{n-2}) \in C$ as well.

The CC shift of a codeword $c(x)$ corresponds to $x \cdot c(x) \mod x^n - a$. It follows that a CC code is an ideal in the quotient ring $\mathbb{F}_q[x]/\langle x^n - a\rangle$ which is a principal ideal ring. For each CC code C, there exists a unique monic generator polynomial $g(x) \in \mathbb{F}_q[x]$ of least degree such that $\langle g(x)\rangle = C$. Hence, $x^n - a = g(x)h(x)$ and there is a one-to-one correspondence between CC codes of length n with shift constant a over $\mathbb{F}_q$ and divisors of $x^n - a$ over $\mathbb{F}_q$. The polynomial $h(x)$ is called the check polynomial of C. A CC code is uniquely determined by either the generator polynomial or the check polynomial. For the special case when the shift constant a is 1, we obtain a cyclic code. Thus, CC codes are generalizations of cyclic codes. CC codes are in turn are a special case of QT codes.

The concept of code equivalence is important in various contexts in coding theory. In the case of computer searches for new linear codes, since equivalent codes have the same parameters, it is unnecessary to examine codes that are equivalent. Two linear codes over $\mathbb{F}_q$ are called equivalent if one can be obtained from the other by any combination of the following transformations.

1. A permutation of the coordinates.
2. Multiplication of elements in a fixed position by a non-zero scalar in $\mathbb{F}_q$.
3. Applying a field automorphism $\sigma : \mathbb{F}_q \to \mathbb{F}_q$ to each component of a vector.

If only (1) is used, then the codes are called permutation equivalent. This is a very important special case since it arises most commonly. Moreover, for binary codes it is the only form of equivalence. We can summarize all of these conditions in the following way.

Definition 2 [17] Two linear codes $C_1, C_2 \subseteq \mathbb{F}_q^n$ are equivalent if there exists a monomial matrix M and an automorphism σ over $\mathbb{F}_q$ such that $C_1 = C_2 M\sigma$.

In an implementation of a computer search algorithm, checking for equivalence with codes that have already been examined before calculating the minimum distance of a new code might save computational time provided this check is fast enough. There exists a polynomial time reduction from the graph isomorphism problem to code equivalence and thus equivalence checking is not NP-complete [22]. However, in practice these checks a long time, especially for codes of larger lengths. The special case of testing the equivalence of two cyclic code can be faster as demonstrated in [3]. Our goal in this paper is to generalize this algorithm to CC codes.

Through our exhaustive searches, we found many codes that are as good as the currently BKLCs and they additional desirable properties. We define these properties here. For any linear code C, its dual code is defined as $C^\perp = \{v \in \mathbb{F}_q^n : v \cdot c = 0 \text{ for all } c \in C\}$ where $v \cdot c$ is the standard inner product in $\mathbb{F}_q^n$. If the dimension of C is k, then the dimension of $C^\perp$ is $n - k$. A code C is self-orthogonal if $C \subseteq C^\perp$, i.e., for any two codewords $a, b \in C$, $a \cdot b = 0$. An $[n, k]_q$ code C is self-dual if $C = C^\perp$. Note that in this case, the dimensions of C and $C^\perp$ need to be equal, thus $k = n/2$. A code C is dual-containing if $C^\perp \subseteq C$. All of these properties of codes have been used extensively to find optimal quantum error-correcting codes [9, 13, 23].

An $[n, k]_q$ code C is linear complementary dual (LCD) if $C \cap C^\perp = \{0\}$. They were first introduced by Massey [21], and were seen to have an optimal solution for a two-user binary adder channel as well as decoding algorithms that are less complex than that for general linear codes. They are also useful in cryptography by protecting the information managed by sensitive devices, particularly against fault invasive attacks and side-channel attacks (SCA) [19].

A code C is reversible if for any codeword $(c_0, c_1, \ldots, c_{n-2}, c_{n-1}) \in C$ its reverse $(c_{n-1}, c_{n-2}, \ldots, c_1, c_0)$ is also a codeword. Reversible codes are useful in cases where the code might be read from any direction [20]. They are also very important for the study of DNA codes.

Let C be a linear code and $w(c)$ denote the Hamming weight of codeword $c \in C$. If w takes at most two distinct nonzero values for all $c \in C$, we call C a two-weight code. These codes have important applications in secret-sharing schemes, and are mathematically related to strongly related graphs [11].

3 On equivalence of constacyclic codes

Cyclotomic cosets are useful in the study of cyclic codes in many ways. They give much information about a cyclic code and they are particularly useful for certain types of cyclic codes such as BCH codes. Recently, sufficient conditions for two cyclic codes to be equivalent are obtained from cyclotomic cosets [3, 6, 7]. In this work, we generalize some of these results to CC codes.

Definition 3 [4] Let $\gcd(n, q) = 1$. For any $i \in \mathbb{Z}_n$, the q-cyclotomic coset of n containing i is the set $S_i = \{iq^j \mod n : j \in \mathbb{N}\}$.

It is well known that in the case $\gcd(n, q) = 1$ (simple root cyclic codes), there is a one-to-one correspondence between cyclotomic cosets $\mod n$ and irreducible divisors of $x^n - 1$. Each divisor $g(x)$ of $x^n - 1$ corresponds to a union $S_{g(x)}$ of cyclotomic cosets $\mod n$. Following results about equivalence of cyclic codes are obtained based on cyclotomic cosets in [7].

Theorem 1 [7] *Let $g_1(x)$ and $g_2(x)$ be the standard generators of cyclic codes of length n over $\mathbb{F}_q$ and assume $gcd(e, n) = 1$. Then the isometry $\phi : \mathbb{F}_q[x]/\langle x^n - 1\rangle \mapsto \mathbb{F}_q[x]/\langle x^n - 1\rangle$ given by $x \mapsto x^e \mod (x^n - 1)$ has the property $g_2(x) = \phi(g_1(x)))$ if and only if the map $\phi : S_{g_1} \mapsto S_{g_2}$ given by $\phi(z) = e^{-1}z \mod n$, where e^{-1} is the multiplicative inverse of e* mod *n, is a bijection.*

Theorem 2 [7] *Let $g_1(x)$ be the standard generator of a cyclic code of length n over $\mathbb{F}_q$ where $gcd(n, q) = 1$, and let $\delta = \alpha^{-b}$ where α is a primitive nth root of unity, such that n divides $b \cdot deg(g_1(x)) \cdot (q - 1)$. Let K be an extension field of $\mathbb{F}_q$ that contains δ. Then the isometry $\phi : K[x]/\langle x^n - 1\rangle \mapsto K[x]/\langle x^n - 1\rangle$ defined by $\phi(f(x)) = f(\delta x) \mod (x^n - 1)$ has the property that $\phi(g_1(x)) \in \mathbb{F}_q[x]$ and generates a cyclic code of length n over $\mathbb{F}_q$ if and only if the map $\phi : \mathbb{Z}_n \mapsto \mathbb{Z}_n$ defined by $\phi(z) = z + b \mod n$ is a bijection such that $\phi(Sg_1) = S\phi(g_1)$.*

In a recent work [3], this correspondence is extended to the repeated root case by considering multisets. Suppose $\gcd(n, q) \neq 1$, where q is a power of p. We first write $n = n'p^t$ such that $\gcd(p, n') = 1$. Next we find the cyclotomic cosets $\mod n'$. Define a function P which takes cyclotomic cosets to polynomials. Let α be an n'^{th} root of unity, and S be a cyclotomic coset $\mod n'$. We define $P(S) = \prod_{i \in S}(x - \alpha^i)$. Then we use a multiset to describe unions where if an irreducible factor of $x^n - 1$ appears multiple times (say m times) in a divisor, then the elements of the cyclotomic coset that corresponds to that divisor appear m times in the multiset. Hence, a multiset MS is a union of not necessarily distinct cyclotomic cosets $S_1, S_2, \ldots, S_k$ and it corresponds to the polynomial $P(MS) = P(S_1) \cdot P(S_2) \cdots P(S_k)$. Based on this approach, an algorithm to test equivalence of cyclic codes is given in [3].

We now generalize these results to CC codes. The following observation is very useful for us to be able to generalize them for constacyclic codes.

Consider the polynomial $x^n - a$ over $\mathbb{F}_q$, where p is the characteristic of $\mathbb{F}_q$. Given n, write $n = p^t n'$ such that p does not divide n'. Then, since the map $x \to x^{p^t}$ is a bijection (even an automorphism) on $\mathbb{F}_q$, there exists $b \in \mathbb{F}_q$ such that $a = b^{p^t}$, hence we can write $x^n - a = (x^{n'} - b)^{p^t}$.

Lemma *Given a and b as above, we have $|a| = |b|$, where $|\theta|$ denotes the order of θ in the multiplicative group $\mathbb{F}_q^*$.*

Proof Let α be a primitive element of $\mathbb{F}_q$. Then for some integer j, $b = \alpha^j$, and subsequently $a = \alpha^{jp^t}$. Thus we want to show that $|\alpha^j| = |\alpha^{jp^t}|$. The proof is based on the following well-known theorem from group theory. In a finite cyclic group generated by g, the order of a power of g is given by $|g^i| = \dfrac{|g|}{\gcd(|g|, i)}$

From this and the fact that $|\alpha| = q - 1$ it follows that $|\alpha^j| = \dfrac{q-1}{\gcd(q-1, j)}$ and $|\alpha^{jp^t}| = \dfrac{q-1}{\gcd(q-1, jp^t)}$ As p is the characteristic of $\mathbb{F}_q$, we have $q = p^m$ for some positive integer m, and therefore p^t is relatively prime to $q-1$. Hence $\gcd(q-1, j) = \gcd(q-1, jp^t)$, and $|\alpha^j| = |\alpha^{jp^t}|$. Thus, $|a| = |b|$. □

Theorem 3 *Let $g_1(x)$, $g_2(x)$ be generators of constacyclic codes of length n over $\mathbb{F}_q$ with shift constant a (hence $g_1(x)$, $g_2(x)$ are divisors of $x^n - a$). If there is a bijection m of the form $m(x) = ex + b$, where* $\gcd(e, n) = 1$ *and b is as given in* [7, Theorem 3]*, between cyclotomic cosets* mod *nr corresponding to $g_1(x)$ and $g_2(x)$, then the constacyclic codes $\langle g_1(x)\rangle$ and $\langle g_2(x)\rangle$ are equivalent.*

Proof We know that $g_1(x)|(x^n - a)|x^{nr} - 1$ and $g_2(x)|(x^n - a)|x^{nr} - 1$. Hence, $g_1(x)$ and $g_2(x)$ generate cyclic codes of length nr over $\mathbb{F}_q$. Let K be the extension of $\mathbb{F}_q$ as in [7, Theorem 3]. Writing $m(x) = m_2(m_1(x))$ where $m_1(x) = ex$ and $m_2(x) = x + b$, we observe that there is an isometry Φ from $K[x]/\langle x^{nr} - 1\rangle$ to $K[x]/\langle x^{nr} - 1\rangle$ such that $g_2(x) = \Phi(g_1(x))$ and cyclic codes of length nr generated by $g_1(x)$ and $g_2(x)$ are equivalent by Theorems 2, 3, and the remark that follows them in [7]. Since $\mathbb{F}_q[x]\langle x^n - a\rangle$ is a subring of $K[x]/\langle x^{nr} - 1\rangle$, Φ induces an isometry from $\mathbb{F}_q[x]\langle x^n - a\rangle$ to itself such that $g_2(x) = \Phi(g_1(x))$. Hence the CC codes of length n generated by $g_1(x)$ and $g_2(x)$ are equivalent. □

4 The generalized algorithm

We now describe our approach in developing an algorithm for checking equivalence based on the theory discussed in the last section. We first obtain $r = Ord_q(a)$, $p = char(\mathbb{F}_q)$ and n' such that $n = n' \cdot p^t$. After finding an $n' \cdot r$th root of unity (δ), for $i = 0, 1, \ldots, n' - 1$ we form cyclotomic cosets mod $n'r$ of the exponents of δ of the form $1 + i \cdot r$. We then take unions of multi-sets of elements of these cosets where the multiplicity of an element is between 0 and p^t. Each multiset corresponds to a polynomial.

Checking for a linear map is computationally expensive. The most straight-forward approach is to try all values of $a, b \in \{0, \ldots, n'\}$ such that $\gcd(a, n) = 1$ and for each $x \in C_1, ax + b \in C_2$. If such pair of values exist then the codes are equivalent. The complexity of this process is $O(n^3)$. We save a lot of time by checking if the sum of multiplicities of the elements as well as their frequency distribution are identical before starting to check for a linear map.

Another matter of note is that the choice of the root of unity δ affects the code that will be stored from an equivalence class. Since different choices for δ give equivalent codes from the same classes, this is not a problem for our purposes.

Algorithm 1 Algorithm to decide equivalence between two CC codes based on cosets

```
Input: F (Finite Field of size $q$), $n$ (Length), $a$ (Shift Constant), $g_1$ and $g_2$
    (generator polynomials of $C_1$ and $C_2$)
Output: True (if algorithm detects equivalence), False (otherwise)
Function CC_CosetEq($F, n, a, g_1, g_2$):
    $r = Order(a)$  $p = Characteristic(F)$  $n'$ such that $n = n' \cdot (p)^t$ for highest possible $t \in \mathbb{N}$
    $EF = ExtensionField(F)$ defined by irreducible polynomial in $F[x]$ of degree $(Order(n' \cdot r \mod q))$
    elements = $[1 + i \cdot r$: $i$ from 0 to $n' - 1]$  $rou = (n'r)^{th}$ root of unity in $EF$
    for $i$ in elements do
        coset1[i] is the largest integer $y$ such that $(x - rou^i)^y | g_1$  coset2[i] is the largest integer $y$ such that
        $(x - rou^i)^y | g_2$
    end
    Equivalent = false  if Sum(coset1) == Sum(coset2) then
        if FrequencyDistribution(coset1) == FrequencyDistribution(coset2) then
            if existsLinearMap(coset1,coset2) then
                Equivalent = true
            end
        end
    end
    return Equivalent
```

5 Performance and limitations

The following table compares our CC_CosetEq Function with Magma's IsEquivalent Function. It can be seen that our method is always faster, considerably so in cases where the codes are actually equivalent. However, Magma's function is more versatile while our method is tailor-made for CC codes. Since Magma has no other version of testing code equivalence available, we will make the comparison between our algorithm and Magma's algorithm. In the tables below, a polynomial is represented as a list containing only coefficients in order to save space. The ordering is such that coefficients of lowest degree term is in the left-most position. For instance, the polynomial $1 \cdot x + 2 \cdot x^2 + 3 \cdot x^3$ will be represented as [0123]. The online Magma calculator [15] is used for the comparison. It has a time limit of 120 seconds and memory limit of about 360 MB. The entries in the table with "DNF" refers to the programs that did not finish either due to the online calculator's time or memory limit. The Magma IsEquivalent Function also only works for small prime fields or fields of size less than or equal to 4. In Table 1, A is a root of the irreducible polynomial $x^2 + x + 1$ over $\mathbb{F}_2$ and a primitive element of $\mathbb{F}_4$.

It is important to note that the algorithm only checks for a sufficient condition of equivalence. This means that the function might return False even if the codes are actually equivalent. For instance, consider the constacyclic codes with $n = 32, a = 1$ with generators $g_1 = 222120111202021, g_2 = 222112021022021$. Our algorithm does not detect equivalence between these codes even though they actually are equivalent. However, computational evidence seems to suggest that this is a rare occurrence. The next section about partitioning CC codes into equivalence classes furthermore shows that in many cases the number of codes that need to be searched is reduced by a large amount. We can also check for equivalence in $GF(8)$ and $GF(9)$, which was not possible with Magma's function. Thus,

Table 1 Performance comparison of CosetEq method vs inbuilt IsEquivalent function

q	n	g(s)	a	equiv	CC_CosetEq		IsEquivalent	
					CPU time(s)	Memory (MB)	CPU time(s)	Memory (MB)
2	210	[11100110011001001] [11000110011010101]	1	True	0.120	32	101.300	32
3	90	[12011] [11021]	2	False	0.000	32	DNF-Memory limit	
4	60	[A001001] [A00A001]	A	True	0.000	32	0.700	32
5	68	[11434131132402021] [14424434122103031]	3	True	0.020	32	DNF-Time limit	
7	53	[43034063550606030363504603 1] [46015060403430623020505206 1]	4	False	0.020	32	DNF-Time limit	

for the goal of executing an exhaustive search on CC codes, we find our method more viable even with the potential for rare occurrence of redundancies.

6 Applications of the algorithm

We can use the the algorithm given in Sect. 4 to partition all CC codes of a given length and shift constant into equivalence classes. It is purely based on cyclotomic cosets and their combinations. We break the list of all relevant exponents of the root of unity (δ) into the component cyclotomic cosets $\mod n' \cdot r$. Then we take unions of not necessarily distinct cosets up to p^t times. Using the new algorithm we check if the codes generated by this union of cosets is equivalent to any previously seen code. In the end, we convert the multisets to generator polynomials using the map P we defined in Sect. 3 and store them.

Algorithm 2 Algorithm that returns list of unequivalent generators for CC codes of length n, shift constant a

```
Input: q (size of finite field), n (Length), a (Shift Constant)
Output: generatorList (Unequivalent generators dividing $x^n - a$)
$F = FiniteField(q)$
$r = Order(a)$
$p = Characteristic(F)$
$n'$ such that $n = n' \cdot (p)^t$ for highest possible $t \in \mathbb{N}$
$EF = ExtField(F)$ defined by irreducible polynomial in $F[x]$ of degree $(Order(n' \cdot r \mod q))$
CycCosets = []
elements = $[1 + i \cdot r$: $i$ from 0 to $n' - 1]$
for $i$ in elements do
    if $i$ not in CycCosets then
        CycCosets+= $\{iq^j : j = 0, 1, \ldots\}$
    end
end
$numCosets = \#CycCosets$
$rou = (n'r)^{th}$ root of unity in $EF$
$totalnum = (p^t + 1) \wedge numCosets - 2$; (Non-trivial divisors of $x^n - a$)
UneqCosets = [ ], generatorList = [ ]
for i from 1 to totalnum do
    TempCoset = {}; (Multi-Set)
    for j from 1 to numCosets do
        TempCoset += CycCosets[$j$] $\wedge$ ($j^{th}$ $Digit(i)$)
    end
    Equivalent = false
    for CheckCoset in UneqCosets do
        if Sum(CheckCoset) == Sum(TempCoset) then
            if Distribution(CheckCoset) == Distribution(TempCoset) then
                if existsLinearMap(CheckCoset,TempCoset) then
                    Equivalent = true
                end
            end
        end
    end
    if Equivalent == true then
        UneqCosets+= TempCoset
        generator = 1
        for j in TempCoset do
            generator*= $(x - rou^j)^{(Multiplicity(j))}$
        end
        generatorList+= generator
    end
end
Print(generatorList)
```

Table 2 shows the effectiveness of our constacyclic partition algorithm for some sample code lengths. Here, q is the size of the finite field, n represents the length of the code, a represents the shift constant, *total* represents the total number of divisors of $x^n - a$, *new*

Table 2 Reduction in the number of codes from our algorithm

q	n	a	Total	New	Net	Percent decrease
2	93	1	16,382	2798	13,584	82.92
2	105	1	32,766	9598	23,168	70.71
2	120	1	59,047	32,803	26,244	44.45
2	124	1	78,123	13,173	64,950	83.14
3	146	2	8190	536	7654	93.46
3	122	2	8190	455	7735	94.44
3	130	2	32,766	969	31,797	97.04
5	124	2	2046	26	2020	98.73
5	90	2	7774	3074	4700	60.46
5	52	2	8190	1380	6810	83.15
5	104	2	8190	469	7721	94.27
5	52	4	16,382	2129	14,253	87.00
5	108	4	16,382	1269	15,113	92.25
5	60	4	46,654	12,839	33,815	72.48
5	120	4	46,654	696	45,958	98.51
7	76	6	16,382	1126	15,256	93.13
7	90	6	32,766	1519	31,247	95.36
7	86	6	32,766	655	32,111	98.00

represents the number of polynomials generated by our algorithm, and *net* represents the difference between *total* and *new*. Here, *net* is the reduction in the number of codes due to code equivalence and thus is a good indication of a possible benefit of our algorithm when considering a code of a given length. The final column, *Percent decrease*, shows the percentage of reduction in the total number of codes due to our algorithm. This value is simply the ratio of *net* to *total* multiplied by 100.

7 Results

This section contains our findings from an implementation of the partition algorithm. Tables 3, 4, 5, 6, 7, 8, 9, 10, 11 and 12 below show CC codes obtained from our searches using the new algorithm. These codes are as good as the current BKLCs [14], and better than currently known QT Codes [10]. Furthermore, the ones listed here have additional properties by which they are classified into tables. For brevity's sake, we only list some of all (638) such codes we obtained in Table 3 through Table 13.

The first column specifies the parameters of the code, the second lists the shift constant, and the third column gives either the generator g or the parity check polynomial h, whichever is more concise. For $GF(4)$, $GF(8)$ and $GF(9)$, the primitive polynomials x^2+x+1, $x^3+x+1 \in GF(2)[x]$ and $x^2+2x+2 \in GF(3)[x]$ are used to extend the field. For all non-prime fields, A is the primitive element, that is, the root of the primitive polynomial we used for extensions.

We have found 453 more codes that are better than best known QT codes and as good as current BKLCs without any additional properties. However, they are very simple to construct. Many of the BKLCs with the same parameters as our codes have complicated constructions

Table 3 CC codes that are self-orthogonal (19 of 122)

$[n, k, d]_q$	a	h
$[91, 39, 20]_2$	1	1100001011011010010010100011110110111111
$[79, 39, 16]_2$	1	1110110000010110101111001111011100011001
$[223, 37, 72]_2$	1	11111000101011001111101100111000101101
$[83, 41, 21]_3$	1	221200201021221100222200221210011200121201
$[164, 26, 72]_3$	2	2221200212010201220122100021
$[82, 24, 30]_3$	2	1222011211212222111201211
$[19, 9, 8]_4$	1	$1A^20A^2A^2AA0A1$
$[129, 21, 64]_4$	1	$111A0A^2A^211AAA^2A^211AA0A^2111$
$[38, 18, 13]_5$	1	4142302342133022111
$[52, 14, 25]_5$	4	322133301433401
$[31, 12, 14]_5$	1	1403040341241
$[47, 23, 17]_7$	1	654323415250330435200061
$[50, 20, 20]_7$	6	112364342641233364261
$[85, 16, 48]_7$	1	12105062226642241
$[79, 13, 49]_8$	1	$1A^41A^4A^5A^6A^300A^6AA^5A^61$
$[19, 9, 10]_9$	1	$2A^52A^2A^3A^5A^21A^31$
$[31, 15, 12]_9$	1	$2A^5A^3A^7A^32A^6A^7AA^61A^5AA^5A^31$
$[37, 18, 14]_9$	1	$1A^310A^6A^5A12221A^3A^7A^201A1$

Table 4 CC codes that are dual containing (16 of 139)

$[n, k, d]_q$	a	g
$[133, 112, 6]_2$	1	1110101111110000110001
$[151, 106, 13]_2$	1	1010100111001100110111000110110101001010111001
$[93, 48, 14]_2$	1	1001111001101000011101000000001100100000101100001
$[109, 82, 10]_3$	1	2220110212021200001101101101
$[82, 58, 10]_3$	2	110021000202110100022002 1
$[133, 112, 8]_4$	1	$1A^20A^2A^2A^2A^2AAA^2A^2110AA001AA^21$
$[71, 51, 10]_5$	1	10340202144003240213 1
$[52, 34, 10]_5$	4	3103324404332410421
$[44, 23, 12]_5$	1	20441424100121324034 01
$[58, 44, 8]_7$	1	650012241422041
$[40, 28, 8]_7$	6	1060426323511
$[47, 24, 16]_7$	1	610005243044025263454321
$[79, 66, 8]_8$	1	$1A^6A^5AA^600A^3A^6A^5A^41A^41$
$[37, 28, 7]_9$	1	$2A^61A^6AA^7A^62A^61$
$[31, 16, 11]_9$	1	$2A^7AA^5A2A^2A^5A^3A^21A^7A^3A^7A1$
$[37, 19, 13]_9$	1	$1A10A^2A^7A^312221AA^5A^601A^31$

Table 5 CC codes that are LCD (12 of 79)

$[n, k, d]_q$	a	g or h
$[57, 30, 14]_4$	A	$111AA^2A^2AAA^2A11111111A^2AA^2A^2AAA^2111$
$[105, 84, 8]_4$	A	$A^2000AA^2A^200A^2A^2A0A^2A01A^2AA01$
$[171, 18, 96]_4$	A	$h = A^200AA^201A^20A01A^201A001$
$[68, 52, 8]_5$	2	11040231132244121
$[52, 24, 17]_5$	2	$h = 40230141401024134402041 01$
$[46, 24, 13]_5$	2	31443110010104003414241
$[86, 72, 8]_7$	3	440151543452041
$[40, 20, 14]_7$	3	$h = 262441353161263215461$
$[50, 16, 26]_7$	3	$h = 24632222431542661$
$[10, 6, 5]_9$	A	$A^2A^7A^2A^61$
$[34, 26, 6]_9$	A	$2A^3A^5A^71A^6A^311$
$[58, 14, 33]_9$	A	$h = A^7A^6A^501A^6A^7AA^62A^50111$

Table 6 Codes that are reversible (19 of 29)

$[n, k, d]_q$	a	g or h
$[204, 191, 4]_2$	1	10101100110101
$[180, 166, 4]_2$	1	111010101010111
$[168, 154, 4]_2$	1	100110000011001
$[72, 61, 4]_2$	1	110101101011
$[30, 23, 4]_3$	1	11122111
$[12, 7, 4]_3$	1	101101
$[6, 3, 3]_3$	1	$h = 1221$
$[34, 20, 8]_4$	1	$111A^2A01010AA^2111$
$[68, 61, 4]_4$	1	$1A^2A00AA^21$
$[65, 58, 4]_5$	1	40141401
$[30, 25, 4]_5$	1	142241
$[15, 10, 4]_5$	1	424131
$[56, 50, 4]_7$	1	1124211
$[56, 45, 6]_7$	1	166534435661
$[18, 11, 6]_8$	1	$1A^51A^6A^61A^51$
$[18, 14, 4]_8$	1	$1A^20A^21$
$[36, 31, 4]_8$	1	$1A^6A^2A^2A^61$
$[30, 25, 4]_9$	1	$1A^311A^31$

Table 7 CC codes that are self-dual

$[n, k, d]_q$	a	h
$[28, 14, 9]_3$	2	221211000122221
$[8, 4, 5]_7$	6	15221

Table 8 CC codes that are self-orthogonal and reversible

$[n, k, d]_q$	a	h
$[10, 3, 6]_4$	1	$1A^2A^21$
$[7, 3, 5]_7$	1	6341
$[18, 3, 14]_8$	1	$1A^3A^31$

Table 9 CC codes that are dual-containing and reversible (4 of 9)

$[n, k, d]_q$	a	g
$[10, 7, 3]_5$	1	4411
$[56, 52, 3]_7$	1	16361
$[56, 51, 4]_7$	1	102201
$[28, 23, 4]_7$	1	134431

Table 10 CC codes that are LCD and reversible (20 of 245)

$[n, k, d]_q$	a	g or h
$[171, 134, 10]_2$	1	100100000010001101111011000100000001001
$[129, 87, 13]_2$	1	101111101100110011101110111001100110111101
$[65, 40, 10]_2$	1	10001101101011010110110001
$[146, 122, 8]_3$	1	1122121011100011101212211
$[82, 49, 14]_3$	1	1211200010200021001200020100021121
$[74, 38, 16]_3$	2	1101011222112200022200022112221101011
$[29, 15, 11]_4$	1	$1A0AA^21A^2AA^21A^2A0A1$
$[65, 33, 16]_4$	1	$1A^2A^20AA01A^2100A^21101011A^2001A^210AA0A^2A^21$
$[241, 228, 6]_4$	1	$1A^21A00AA00A1A^21$
$[67, 23, 27]_5$	1	$h = 4211340302114430201244 31$
$[67, 22, 28]_5$	1	$h = 14324002204340220042341$
$[41, 21, 13]_5$	1	100203331020133302001
$[29, 14, 12]_5$	1	$h = 144224030422441$
$[50, 21, 20]_7$	1	$h = 6515262441166335152621$
$[29, 15, 11]_7$	1	104516141615401
$[57, 13, 33]_8$	1	$h = 1A^2A^3A^5A^60A^3A^30A^6A^5A^3A^21$
$[65, 52, 8]_8$	1	$1A^6A^3A^4A^2A^311A^3A^2A^4A^3A^61$
$[29, 14, 12]_9$	1	$h = 1A1A^2A^32A^2A^2A^22A^3A^21A1$
$[41, 24, 12]_9$	1	$2A^6A^7A^5A^6A^6A^70A^2A^60A^3A^2A^2AA^3A^21$
$[73, 12, 47]_9$	1	$h = 1A^50AA^60A^30A^6A0A^51$

Table 11 CC codes that are self-orthogonal and two-weight (3 of 6)

$[n, k, d]_q$	a	h
$[7, 3, 4]_4$	1	1011
$[22, 5, 12]_3$	1	102221
$[12, 4, 6]_3$	2	11221

Table 12 CC codes that are LCD and two-weight

$[n, k, d]_q$	a	h
$[17, 4, 12]_4$	1	$11A11$
$[26, 4, 20]_5$	2	41331

Table 13 CC codes that are self-orthogonal, two-weight and reversible

$[n, k, d]_q$	a	h
$[34, 4, 24]_4$	1	$11A11$
$[10, 4, 4]_2$	1	11111

involving multiple steps. For instance, consider the BKLC $[68, 52, 8]_5$ from http://www.codetables.de [14] is constructed in 7 seven steps. Our construction for a code just as good is just one step- CC code with length 68, shift constant 2 and generator 11040231132244121. This construction is less complicated and the code is easier to replicate. Thus, our codes are better alternatives than the ones listed in the database with the same parameters. Additionally, we have found a total of 23 new linear codes with higher minimum distances than the currently BKLCs listed in [14].

We found a new $[65, 51, 8]_5$ code from our search results using construction X. This code is better than currently known linear codes and can be constructed as follows:

1 : [63, 51, 6] Constacyclic Code over GF(5) $a = 1$, $g = 1133013103311$
2 : [63, 50, 8] Constacyclic Code over GF(5) $a = 1$, $g = 40303432120201$
3 : [2, 1, 2] Cyclic Linear Code over GF(5) RepetitionCode of length 2
4 : [65,51,8] Linear Code over GF(5) Construct X using [1], [2] and [3]

We also found a $[93, 15, 58]_7$ code whose minimum distance is 3 units larger than the current BKLC having same length and dimension. [93,15,58] Constacyclic Code over GF(7) $a = 2, g = 434026354222142014153662356345646414114150215061021463420301224620135213654061 1$

8 Recursive standard constructions

In the course of the search, we found codes that beat the currently best known minimum distances by more than one unit. In the case of the $[93, 15, 58]_7$ code, its minimum distance was 3 higher than the BKLC that preceded it. This means that there was high potential for other codes derived from this code to produce additional record breakers by using such standard constructions as extension, puncturing, and shortening. All of these constructions are implemented in Magma [15] by the help of the following algorithm:

Through the use of this algorithm, we found 20 new codes stemming from the $[93, 15, 58]_7$ code, C_1. Any code derived from C_1 or its derivative by extension, puncturing or shortening is the name of original code appended by ’e’,’p’ or ’s’ respectively. For instance C1ees is C1 extended twice and then shortened once. Puncturing and shortening is done from the best position possible. We also indicate, for each code, the improvement on the minimum distance of the previously best known linear code with the same length and dimension. For example, $C1 : [93, 15, 58], +3$ means the previous record was [93, 15, 55] and our code $C1$ improves the minimum distance by 3 units.

Algorithm 3 Recursive Code Modification

```
Input: C: A good code with parameters $[n, k, d]_q$ Input: ShortenLimit: A constant that will determine how many places we can shorten at once  Function RecursivelyModify(C,foundparams):
    Function Check(C,foundparams):
        if Cprime is better than the corresponding BKLC and Parameters(Cprime) not in foundparams then
            Print(Cprime)  return Concatenate(foundparams,RecursivelyModify(Cprime,foundparams));
        end
        return [ ]
    foundparams = Append(foundparams,Parameters(C))  Cprime=ExtendCode(C)  foundparams+= Check(Cprime,foundparams)
    CprimeP=CprimeS= [1,1,1] trivial code  for s from 1 to n do
        CtempP=PunctureCode(C,i) CtempS=ShortenCode(C,s)  if CtempP is better than CprimeP then
            CprimeP=CtempP
        end
        if CtempS is better than CprimeS then
            CprimeS=CtempS
        end
    end
    foundparams+= Check(CprimeP,foundparams)  foundparams+= Check(CprimeS,foundparams)  return foundparams
RecursivelyModify(C,[ ]);
```

C1: [93,15,58], +3

C1e: [94,15,58], +2

C1ee: [95,15,58], +1

C1ees: [94,14,58], +1

C1eesp: [93,14,58], +1

C1eespp: [92,14,58], +2

C1eespppp: [90,14,56], +1

C1eesppps: [90,13,57], +1

C1eeppps: [91,13,58], +2

C1eesps: [92,13,58], +1

C1p: [92,15,57], +3

C1pp: [91,15,56], +3

C1ppp: [90,15,56], +3

C1pppp: [89,15,56], +4

C1ppppp: [88,15,56], +5

C1pppppp: [87,15,56], +5

C1ppppppp: [86,15,56], +6

C1pppps: [88,14,54], +1

C1sppp: [89,14,55]*[a]

C1ppss: [89,13,56], +1

C1ppssp: [88,13,55], +1

[a] We originally found a [89,14,54]-code. In the process of verifying our codes, M. Grassl found this one

This algorithm is especially useful for producing new codes from a good code which beats the corresponding minimum distance record by more than 1 unit. A Magma file to execute it can be obtained by contacting the authors of this paper.

References

1. Aydin N., Guidotti T., Liu P.: Good classical and quantum codes from multi-twisted codes. CoRR arXiv:2008.07037 (2020)
2. Aydin N., Halilović A.: A generalization of quasi-twisted codes: multi-twisted codes. Finite Fields Appl. **45**, 96–106 (2017). https://doi.org/10.1016/j.ffa.2016.12.002.
3. Aydin N.O., Vandenberg R.O.: A new algorithm for equivalence of cyclic codes and its applications. Appl. Algebra Eng. Commun. Comput. (2021). https://doi.org/10.1007/s00200-021-00525-4.
4. Aydin N., Siap I., Ray-Chaudhuri D.K.: The structure of 1-generator quasi-twisted codes and new linear codes. Des. Codes Cryptogr. **24**(3), 313–326 (2001). https://doi.org/10.1023/a:1011283523000.
5. Aydin N., Connolly N., Murphree J.: New binary linear codes from quasi-cyclic codes and an augmentation algorithm. Appl. Algebra Eng. Commun. Comput. **28**(4), 339–350 (2017). https://doi.org/10.1007/s00200-017-0327-x.

6. Aydin N., Connolly N., Grassl M.: Some results on the structure of constacyclic codes and new linear codes over GF(7) from quasi-twisted codes. Adv. Math. Commun. **11**(1), 245–258 (2017). https://doi.org/10.3934/amc.2017016.
7. Aydin N., Lambrinos J., Vandenberg O.: On equivalence of cyclic codes, generalization of a quasi-twisted search algorithm, and new linear codes. Des. Codes Cryptogr. **87**(10), 2199–2212 (2019). https://doi.org/10.1007/s10623-019-00613-0.
8. Aydin N., Guidotti T.H., Liu P., Shaikh A.S., Vandenberg R.O.: Some generalizations of the ASR search algorithm for quasitwisted codes. Involve J. Math. **13**(1), 137–148 (2020). https://doi.org/10.2140/involve.2020.13.137.
9. Calderbank A.R., Rains E.M., Shor P.W., Sloane N.J.A.: Quantum error correction and orthogonal geometry. Phys. Rev. Let. **78**(3), 405–408 (1997). https://doi.org/10.1103/physrevlett.78.405.
10. Chen E.: Quasi-cyclic codes: bounds on the parameters of of QC codes. http://www.tec.hkr.se/~chen/research/codes/qc.htm. Accessed Aug 2021
11. Ding K., Ding C.: A class of two-weight and three-weight codes and their applications in secret sharing. IEEE Trans. Inf. Theory **61**(11), 5835–5842 (2015). https://doi.org/10.1109/tit.2015.2473861.
12. Dinh H.Q., Bag T., Upadhyay A.K., Ashraf M., Mohammad G., Chinnakum W.: Quantum codes from a class of constacyclic codes over finite commutative rings. J. Algebra Appl. **19**(12), 2150003 (2019). https://doi.org/10.1142/s0219498821500031.
13. Gottesman D.: Class of quantum error-correcting codes saturating the quantum hamming bound. Phys. Rev. A **54**(3), 1862–1868 (1996). https://doi.org/10.1103/physreva.54.1862.
14. Grassl M.: Code tables: bounds on the parameters of codes. http://www.codetables.de/. Accessed Aug 2021
15. Group M.: Magma computer algebra system. http://magma.maths.usyd.edu.au/calc/. Accessed Aug 2021
16. Gulliver T.A., Venkaiah V.C.: Construction of quasi-twisted codes and enumeration of defining polynomials. J. Algebra Comb. Discret. Struct. Appl. (2019). https://doi.org/10.13069/jacodesmath.645015.
17. Huffman W.C., Pless V.: Fundamentals of Error-Correcting Codes. Cambridge University Press, Cambridge (2003) https://doi.org/10.1017/cbo9780511807077.
18. Koroglu M., Siap I.: A class of constacyclic codes from group algebras. Filomat **31**(10), 2917–2923 (2017). https://doi.org/10.2298/fil1710917k.
19. Lu L., Li R., Fu Q., Xuan C., Ma W.: Optimal ternary linear complementary dual codes. CoRR arXiv:2012.12093 (2020)
20. Massey J.L.: Reversible codes. Inf. Control **7**(3), 369–380 (1964). https://doi.org/10.1016/s0019-9958(64)90438-3.
21. Massey J.L.: Linear codes with complementary duals. Discret. Math. **106–107**, 337–342 (1992). https://doi.org/10.1016/0012-365x(92)90563-u.
22. Petrank E., Roth R.: Is code equivalence easy to decide? IEEE Trans. Inf. Theory **43**(5), 1602–1604 (1997). https://doi.org/10.1109/18.623157.
23. Steane A.M.: Error correcting codes in quantum theory. Phys. Rev. Let. **77**(5), 793–797 (1996). https://doi.org/10.1103/physrevlett.77.793.
24. Vardy A.: The intractability of computing the minimum distance of a code. IEEE Trans. Inf. Theory **43**(6), 1757–1766 (1997). https://doi.org/10.1109/18.641542.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.