(Re)constructing code loops

Ben Nagy and David Michael Roberts

Abstract. Write later.

The theory of codes makes for a fascinating study. At their heart, codes are 'merely' subspaces of vector spaces over some small finite field, with certain combinatorial properties. Why do such things exist? Like a lot of exceptional objects in combinatorics, it can come down to: "because". This makes constructing codes sometimes more of an art than something systematic. In this paper, we are going to consider the construction of certain [algebro-combinatoric] structures closely related to codes, called *code loops*. We will also only be considering the case where the base field is $\mathbb{F}_2 = \{0, 1\}$, and refer to its elements as *bits*.

The first published example of a code loop appeared as a step in John Conway's construction of the Monster sporadic simple group [**1**]. The code loop Conway used originally appeared in an unpublished manuscript by Richard Parker. A general study of code loops was then made by Robert Griess [**2**]. Griess also proved the existence of code loops by an algorithmic construction, starting from a particular type of code. More recent approaches will be discussed below.

Recall that the elements (or *words*) in a code $\mathcal{C}$, being vectors, can be combined by addition—this is a group operation and hence associative. The elements of a code loop consist of a pair: a code word and one extra bit. The extra bit *twists* the addition so that combination of code loop elements is a *non-associative operation*: $(xy)z \neq x(yz)$.

More specifically, while addition of words in a code is performed by coördinatewise addition in $\mathbb{F}_2$—bitwise XOR—the algebraic operation in a code loop is not so easily described. The code loop operation can be reconstructed from a function $\mathcal{C} \times \mathcal{C} \to \mathbb{F}_2$ satisfying certain identities, called a *twisted cocycle*. It is the computation and presentation of this function that will mainly concern us in this article, using Griess's algorithm [**2**, proof of Theorem 10]. As a result, we will observe some curious features of the Parker loop, obtained via experimentation and, it seems, previously unknown.

1. EXTENSIONS AND COCYCLES. As a warm-up, we will describe a more familiar structure using the techniques that will be used later. Recall that the *quaternion group* Q_8 is the group consisting of the positive and negative basis quaternions:

$$Q_8 = \{1, i, j, k, -1, -i, -j, -k\}$$

The elements of Q_8 satisfy the identities

$$i^2 = j^2 = k^2 = -1, \quad ij = k.$$

There is a surjective group homomorphism $\pi\colon Q_8 \to \mathbb{F}_2 \times \mathbb{F}_2 =: V$, sending i to $(1, 0)$ and j to $(0, 1)$, and the kernel of π is the subgroup $\{1, -1\} \simeq \mathbb{F}_2$.

Moreover, this kernel is the *center* of Q_8, the set of all elements that commute with every other element of the group.

Now Q_8 is a nonabelian group, but both $\mathbb{F}_2$ and V are abelian groups. One might think that it shouldn't be possible to reconstruct Q_8 from the latter two groups, but it is! That is, if we are given some extra information that uses only the two abelian groups. There is an obvious function $s\colon V \to Q_8$, sending $(0,0)$ to 1, $(1,0)$ to i, $(0,1)$ to j and $(1,1)$ to k. This almost looks like a group homomorphism, but it is not, as $(1,0)+(1,0) = (0,0)$ in V, but $i^2 \neq 1$ in Q_8. We can measure the failure of s to be a group homomorphism by considering the two-variable function

$$d\colon V \times V \to \mathbb{F}_2$$

defined by $(-1)^{d(v,w)} = s(v)s(w)s(v+w)^{-1}$. It is a nice exercise to see that $s(v)s(w)s(v+w)^{-1}$ is always ± 1, so that this definition makes sense. The values of $d(v,w)$ are given as:

$v \setminus w$	00	10	01	11
00	0	0	0	0
10	0	1	1	0
01	0	0	1	1
11	0	1	0	1

where $00 = (0,1)$, $10 = (1,0)$ etc. If s *were* a homomorphism, d would be constant at 0. One can check that d satisfies the *cocycle identities*:

$$d(v,w) - d(u+v,w) + d(u,v+w) - d(u,v) = 0$$

for all triples $u, v, w \in V$. It is also immediate from the definition that $d(0,0) = 0$. An alternative visualisation is given in Figure 1.

The reason for this somewhat mysterious construction is that we can build a bijection of sets using s and the isomorphism $\mathbb{F}_2 \simeq \{1, -1\}$, namely

$$\mathbb{F}_2 \times V \simeq (\{1\} \times V) \cup (\{-1\} \times V) \xrightarrow{\phi} \{1, i, j, k\} \cup \{-1, -i, -j, -k\} = Q_8$$

If we define a new product operation on the *set* $\mathbb{F}_2 \times V$ by

$$(s,v) *_d (t,w) := (s + t + d(v,w), v + w),$$

then the cocycle identities ensure that this is in fact associative and further, a group operation. Finally, ϕ can be checked to be a homomorphism for the group operation on Q_8 and for $*_d$, hence is a group isomorphism.

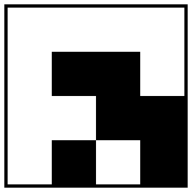


Figure 1. A 4×4 array giving the values of the cocycle $d\colon V \times V \to \mathbb{F}_2$, with white = 0, black = 1.

Thus we can reconstruct, at least up to isomorphism, the nonabelian group Q_8 from the two abelian groups V and $\mathbb{F}_2$, together with the *cocycle* $d\colon V \times V \to \mathbb{F}_2$. A table of the values of d is shown in Figure 1, where $00 = (0,0)$, $10 = (1,0)$ etc. If we didn't know about the group structure of Q_8 already we could construct it from scratch using d. This is what we aim to do to construct the Parker loop, using a similar approach.

2. TWISTED COCYCLES AND LOOPS. The construction in the previous section is a fairly typical case of reconstructing a central extension from a cocycle (although in general one does not even need the analogue of the group V to be abelian). However, we wish to go one step further, and construct a structure with a *non-associative* product from a pair of abelian groups: the group $\mathbb{F}_2$ and a vector space V. Instead of a cocycle, we use a *twisted cocycle*: a function $\alpha\colon V \times V \to \mathbb{F}_2$ like d that instead satisfies

$$\alpha(v,w) - \alpha(u+v,w) + \alpha(u,v+w) - \alpha(u,v) = f(u,v,w),$$

for a special *twisting function* $f\colon V \times V \times V \to \mathbb{F}_2$. We will assume that α satisfies $\alpha(0,v) = \alpha(v,0) = 0$ for all $v \in V$, a property that holds for d in the previous section. From a twisted cocycle the set $\mathbb{F} \times V$ can be given a binary operation

$$(s,v) *_\alpha (t,w) := (s + t + \alpha(v,w), v + w).$$

We denote $\mathbb{F} \times V$ equipped with this binary operation by $\mathbb{F} \times_\alpha V$.

Recall that a *loop* is a set L with a binary operation $\star\colon L \times L \to L$, a unit element $e \in L$ such that $e \star x = x \star e = x$ for all $x \in L$, and such that the functions $(-) \star z\colon L \to L$ and $z \star (-)\colon L \to L$ are bijections for every $z \in L$. Informally, this means that every element $z \in L$ has a left inverse and a right inverse for $\star$, and these are unique—but may be different in general. The following is a cute exercise is using the twisting function and the assumption on $\alpha(0,v)$ and $\alpha(v,0)$ stated in the previous paragraph.

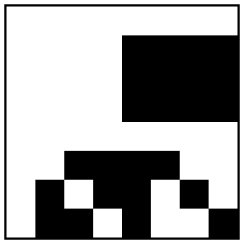
Lemma 1. *The operation $*_\alpha$ makes $\mathbb{F} \times_\alpha V$ into a loop, with identity element $(0,\underline{0})$, for $\underline{0}$ the zero vector in V.*

Groups are examples of loops, but they are in a sense the uninteresting case. Arbitrary loops are quite badly behaved: their product is non-associative in general. But there is a special non-associative case, introduced by Ruth Moufang [3], where one has better algebraic properties.

Definition. A *Moufang loop* is a loop $(L,\star)$ satisfying the identity $x \star (y \star (x \star z)) = ((x \star y) \star x) \star z$ for all choices of elements $x,y,z \in L$.

The most famous example of a Moufang loop is probably the non-zero octonions $\mathbb{O}^\times$ under multiplication. A key property of a Moufang loop L is that any subloop $\langle x,y\rangle < L$ generated by a pair of elements x,y is in fact a group. As a corollary, powers of a *single* element are well-defined, and do not require extra bracketing: $x \star (x \star x) = (x \star x) \star x =: x^3$, for example. Additionally, the left and right inverses *always* agree in a Moufang loop, so that for each $x \in L$, there is a unique x^{-1} such that $x \star x^{-1} = x^{-1} \star x = e$. Importantly for us, code loops, defined below as a special case of the construction of $\mathbb{F} \times_\alpha V$, turn out to be Moufang.

Example 1. The finite Moufang loop $M := M_{16}(C_2 \times C_4)$ of [] (classification paper, ?Moufang Loops of Small Order. I?), with 16 elements, is an extension $\mathbb{F}_2 \to M \to V = \mathbb{F}_2^3$ arising from a twisted cocycle $V \times V \to \mathbb{F}_2$ given by the 8×8 array



where again, white = 0, black = 1. The order of the row/colum labels is 000, 100, 010, 110, 001, 101, 011, 111. Notice in particular that the first four columns/rows correspond to the subspace $U \subset V$ spanned by 10 and 01, and that the restriction $U \times U \to \mathbb{F}_2$ is identically zero (i.e. white). This means that the restriction of M to the subspace U—the subloop of elements that map to elements of U—is in fact the direct product $\mathbb{F}_2 \times U$, and in particular a group.

3. CODES AND CODE LOOPS. To describe the twisting function f for our code loops, we need to know about some extra operations that exist on vector spaces over the field $\mathbb{F}_2$. For W an n-dimensional vector space over $\mathbb{F}_2$ and vectors $v, w \in W$, there is a new vector $v \,\&\, w \in W$ given by

$$v \,\&\, w := (v_1 w_1,\, v_2 w_2,\, \ldots,\, v_n w_n).$$

If we think of such vectors as binary words, then this is bitwise AND. Note that if we take a code $\mathcal{C} \subset (\mathbb{F}_2)^n$, then $\mathcal{C}$ is not guaranteed to be closed under this operation. The other operation takes a vector $v \in W$ and returns its *weight*: the sum, as an integer, of its entries: $|v| := v_1 + \cdots + v_n$. Equivalently, it is the number of nonzero entries in v.

The desired twisting function is a combination of these two, namely $f(u, v, w) := |u \,\&\, v \,\&\, w|$. However, as alluded to above, we also are going to ask that further identities hold. For these identities to make sense we need to start with a code with the special property of being *doubly even*.

Definition. A code $\mathcal{C} \subset (\mathbb{F}_2)^n$ is *doubly even* if for every word $v \in \mathcal{C}$, $|v|$ is divisible by 4.

Example 2. The *Hamming (8,4) code* is doubly even, and is given by the subspace of $\mathbb{F}_2^8$ spanned by the rows of the matrix

$$\begin{pmatrix} 1&0&0&0&0&1&1&1 \\ 0&1&0&0&1&0&1&1 \\ 0&0&1&0&1&1&0&1 \\ 0&0&0&1&1&1&1&0 \end{pmatrix}$$

A more substantial example is given by the Golay code.

Example 3. The Golay code $\mathcal{G} \subset \mathbb{F}_2^{24}$ is the span of the 12 rows in the following two matrices

$$\begin{pmatrix} 000110000000010110100011 \\ 101001111101101111110001 \\ 000100000000100100111110 \\ 010000000010000110101101 \\ 000000000010010101010111 \\ 100000000000100111110001 \end{pmatrix} \qquad \begin{pmatrix} 101001011100111001111111 \\ 100000011100001001001100 \\ 000001000000111001001110 \\ 100000001000111000111000 \\ 100000000100101000010111 \\ 011011000001111011111111 \end{pmatrix}$$

This is a different basis from the usual one [**5**], which can be built using the complement of the incidence matrix of the icosahedron. This basis, however, allows us to demonstrate some interesting properties below.

The inclusion/exclusion formula applied to counting nonzero entries allows us to show that, for all v and w in any doubly even code $\mathcal{C}$,

$$|v + w| + |v \,\&\, w| = |v| + |w| - |v \,\&\, w|\,.$$

In other words: $|v \,\&\, w| = \frac{1}{2}(|v| + |w| - |v + w|)$, which implies that $|v \,\&\, w|$ is divisible by 2. Thus for words v, w in a doubly even code, both $\frac{1}{4}|v|$ and $\frac{1}{2}|v \,\&\, w|$ are integers.

Definition (Griess [2]). Let $\mathcal{C}$ be a doubly even code. A *code cocycle* $\alpha\colon \mathcal{C} \times \mathcal{C} \to \mathbb{F}_2$ is a function satisfy the identities

$$\alpha(v,w) - \alpha(u+v,w) + \alpha(u,v+w) - \alpha(u,v) = |u \,\&\, v \,\&\, w| \pmod 2 \tag{1}$$

$$\alpha(v,w) + \alpha(w,v) = \tfrac{1}{2}|v \,\&\, w| \pmod 2 \tag{2}$$

$$\alpha(v,v) = \tfrac{1}{4}|v| \pmod 2 \tag{3}$$

Remark. What we call a code cocycle, Griess actually calls a 'factor set'. Given that a code cocycle is an example of a twisted cocycle, we prefer a name that indicates this.

It is not obvious, on first consideration, that code cocycles even exist, or how many there are for a given doubly even code. However, Griess gave an proof that inductively constructs code cocycles, and counts how many arbitrary choices can be made along the way, proving that code cocycles do indeed exist. The growth of the number of possible code cocycles with $\dim \mathcal{C}$ is rather fearsome: $2^{2^{k-k-1}}$, for $k = \dim \mathcal{C}$ ([**2**, Theorem 10]). For the 4-dimensional Hamming code given above, this is 512, but for the extended Golay code below there are 2^{4083} possible code cocycles, a number with 1230 digits in base 10.

4. GRIESS'S ALGORITHM AND ITS OUTPUT. The algorithm that Griess gives in [**2**, Theorem 10] to construct code cocycles for a code $\mathcal{C}$ takes as input an ordered basis $\{b_1, \ldots, b_k\}$ for $\mathcal{C}$. The code cocycle is then built up inductively over larger and larger subspaces $\operatorname{span}\{b_1, \ldots, b_m\}$, at each stage applying the identities to define the growing code cocycle on a larger domain.

More accurately, Griess outlines the algorithm, using steps like 'determine the cocycle on such-and-such subset using identity X', where X refers to one

Data: Basis $B = \{b_0, b_1, \ldots, b_{n-1}\}$ for the code C
Result: Code cocycle $\theta\colon C \times C \to \mathbb{F}_2$, encoded as a square array of elements from $\mathbb{F}_2$, with rows and columns indexed by C

```
// Initialise
```
forall $c_1, c_2 \in C$ **do**
  $\theta(c_1, c_2) \leftarrow 0$
end
$\theta(b_0, b_0) \leftarrow \frac{1}{4}|b_0|$

forall $1 \le k \le \text{length}(B)$ **do**
  Define $V_k := \text{span}\{b_0, \ldots, b_{k-1}\}$
  `// (D1) define theta on {bk} x Vk then deduce on Vk x {bk}`
  forall $v \in V_k$ **do**
    if $v \neq 0$ **then**
      $\theta(b_k, v) \leftarrow$ random `// In practice, random = 0`
      $\theta(v, b_k) \leftarrow \frac{1}{2}|v \,\&\, b_k| + \theta(b_k, v)$
    else
      `//` $\theta(b_k, v)$ `is already set to 0`
      $\theta(v, b_k) \leftarrow \frac{1}{2}|v \,\&\, b_k|$
    end
  end
  `// (D2) deduce theta on {bk} x Wk and Wk x {bk}`
  forall $v \in V_k$ **do**
    $\theta(b_k, b_k + v) \leftarrow \frac{1}{4}|b_k| + \theta(b_k, v)$
    $\theta(b_k + v, b_k) \leftarrow \frac{1}{2}|b_k \,\&\, (b_k + v)| + \frac{1}{4}|b_k| + \theta(b_k, v)$
  end
  `// (D3) deduce theta on Wk x Wk`
  forall $v_1 \in V_k$ **do**
    forall $v_2 \in V_k$ **do**
      $w \leftarrow b_k + v_2$
      $a \leftarrow \theta(v_1, b_k)$
      $b \leftarrow \theta(v_1, b_k + w)$
      $c \leftarrow \theta(w, b_k)$
      $r \leftarrow \frac{1}{2}|v_1 \,\&\, w| + a + b + c$
      $\theta(w, b_k + v_1) \leftarrow r$
    end
  end
  `// (D4) deduce theta on Wk x Vk and Vk x Wk`
  forall $v_1 \in V_k$ **do**
    forall $v_2 \in V_k$ **do**
      $w \leftarrow b_k + v_2$
      $a \leftarrow \theta(w, v_1 + w)$
      $\theta(w, v_1) \leftarrow \frac{1}{4}|w| + a$
      $\theta(v_1, w) \leftarrow \frac{1}{2}|v \,\&\, w| + \frac{1}{4}|w| + a$
    end
  end
end

Algorithm 1: Reverse engineered from proof of Theorem 10 in [2].

of (1), (2), (3), or corollaries of these. We have reconstructed the process in detail in Algorithm 1.

We implemented Algorithm 1 in the language Go [4], as well as diagnostic tools, for instance for veryifying the result is Moufang. The output of the algorithm is a matrix of zeroes and ones with rows and columns labelled by words in the Golay code, and can be displayed as an array of black and white pixels. The pixel colours correspond to ones and zeroes, as in Example 1; for the Golay code the image looks like 16 million pixels (more accurately, 4096×4096) of almost random noise.

As a combinatorial object, the code cocycle $\theta\colon \mathcal{C} \times \mathcal{G} \to \mathbb{F}_2$ constructed from Algorithm 1 using the basis in Example 3 is too large and unwieldy to examine for any interesting structure. Moreover, to define with the Parker loop $\mathcal{P} := \mathbb{F}_2 \times_\theta \mathcal{G}$ one needs to know all 16 million or so values of θ. Thus, if one could reconstruct θ by a method shorter than just running Algorithm 1, then one is ahead of the game.

Lemma 2. *Let $\mathcal{C}$ be a doubly even code, α a code cocycle on it, and $\mathcal{C} = V \oplus W$ a decomposition into complementary subspaces. Then for $v_1, v_2 \in V$ and $w_1, w_2 \in W$,*

$$\begin{aligned}\alpha(v_1 + w_1, v_2 + w_2) := {} & \alpha(v_1, v_2) + \alpha(w_1, w_2) + \alpha(v_1, w_1) \\ & + \alpha(w_2, v_2) + \alpha(v_1 + v_2, w_1 + w_2) \\ & + \tfrac{1}{2}|v_2 \,\&\, (w_1 + w_2)| + |v_1 \,\&\, v_2 \,\&\, (w_1 + w_2)| \\ & + |w_1 \,\&\, w_2 \,\&\, v_2| + |v_1 \,\&\, w_1 \,\&\, (v_2 + w_2)| \pmod 2 .\end{aligned} \tag{4}$$

Proof. sorry. ∎

Observe that in Lemma 2, on the RHS the code cocycle α is only ever evaluated on vectors from the subset $V \cup W \subset \mathcal{C}$. This means that if we throw away all of the array encoding the values of α except those positions with labels coming from $V \cup W$, then we can still reconstruct arbitrary values of α using (4). If we assume that $\mathcal{C}$ is $2k$-dimensional, and that V and W are both k-dimensional, then the domain of the restricted α has $(2^k + 2^k - 1)^2 = 2^{2(k+1)} - 2^{k+2} + 1 = O((2^k)^2)$ elements. Compare this to the full domain of α, which has $2^{2k} \times 2^{2k} = (2^k)^4$ elements, giving a roughly square-root saving.

And, now it should be clear why the Golay code basis in Example 3 was partitioned into two lists of six vectors: we can reconstruct θ, and hence the Parker loop multiplication, from a mere $2^{14} - 2^8 + 1 = 16,129$ values. The span of the rows of the left matrix in Example 3 give the subspace $V \subset \mathcal{G}$, and the span of the rows of the right matrix give $W \subset \mathcal{G}$.

In practice, we double count slightly when displaying the values of the restricted code cocycle, and use the *disjoint* union $V \sqcup W$, rather than $V \cup W$, to label the colums and rows of the resulting array. The top left quadrant then contains to the restriction of α to $V \times V$, and the bottom right quadrant the restriction to $W \times W$. The off-diagonal quadrants contain the values of α restricted to $V \times W$ and $W \times V$.

Remarkably, the restriction of the Parker loop $\mathcal{P}$ to $V \subset \mathcal{G}$ reduces to a direct product: $\theta\big|_{V \times V}$ is identically zero! Moreover, the restriction of $\mathcal{P}$ to W is the direct product $\mathbb{F}_2^3 \times M$, where M is the Moufang loop from Example 1.

ACKNOWLEDGMENT. DMR is supported by the Australian Research Council's Discovery

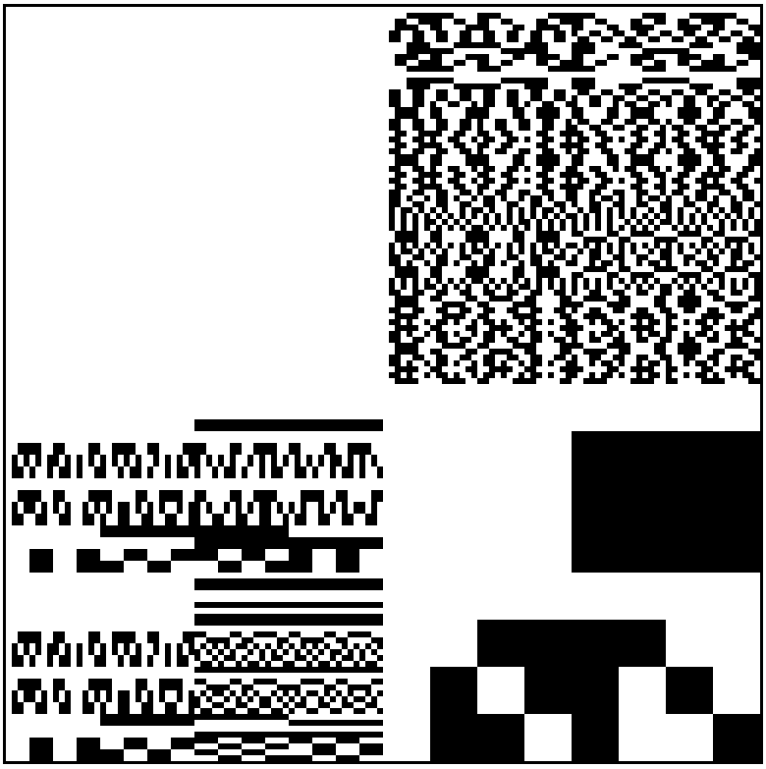


Figure 2. The restriction of the Parker loop code cocycle θ to $(V \sqcup W)^2$

Projects scheme, (project number DP180100383), funded by the Australian Government. **ADD MORE HERE, IF NEEDED**

REFERENCES

1. John H. Conway, A simple construction for the Fischer–Griess monster group *Invent. math.* **79** (1985) 513–540.
2. Robert L. Griess Jr., Code loops *J. Algebra* **100** (1986) 224–234.
3. Moufang, R. (1935), "Zur Struktur von Alternativkörpern", Math. Ann., 110: 416âĂŞ430, doi:10.1007/bf01448037
4. Ben Nagy and David Michael Roberts, `codeloops`, GitHub repository, (2019) `https://github.com/bnagy/codeloops`
5. Thomas M. Thompson (1983). From Error Correcting Codes through Sphere Packings to Simple Groups. Carus Mathematical Monographs. **21**. Mathematical Association of America.

BEN NAGY is an MPhil. candidate at the University of Adelaide, researching computational stylistics for Latin Poetry.
Department of Classics, Archaeology & Ancient History, The University of Adelaide
benjamin.nagy@adelaide.edu.au

DAVID MICHAEL ROBERTS is a mathematician specialising in category theory, geometry and a hodge-podge of other random topics.
School of Mathematical Sciences, The University of Adelaide
david.roberts@adelaide.edu.au