Data-Syndrome codes project

Weilei Zeng, Alexey Ashikhmin, and Leonid Pryadko

(Dated: June 1, 2018)

I. READING FORNEY-GRASSL-GUHA-2007 ON QCC[1]

Quantum Convolutional Code (QCC)

Arbitrary number of blocks of length n; shift-symmetric by n. Support of a generator occupies no more than $\nu+1$ consecutive blocks. ν is called the *constraint length*, or *memory length*.

Example: generator $g_1 = (\ldots|000|111|1\omega\bar{\omega}|000|\ldots)$ with $n=3$, $\nu=1$. It is clearly self-orthogonal. Such a QCC code has distance $d^\perp = 3$. Orthogonal generator for minimum-weight codewords: $g_2 = (\ldots|000|\omega\bar{\omega}1|000|\ldots)$.

Polynomial notations: use D as a shift operator, then the above generator corresponds to $g_1(D) = \{1+D, 1+\omega D, 1+\bar{\omega}D\}$.

Canonical generator $g(D)$: e.g., the smallest degree. Given polynomials $c(D) = \{c_1(D),\ldots,c_m(D)\}$, min-degree polynomial generator is $g(D) = c(D)/d(D)$, where $d(D) = \gcd c(D)$. In this case the constraint length $\nu = \deg g(D)$ (maximum degree of any of the polynomials in the set).

Orthogonality. Introduce the *cross-correlation sequence* polynomial

$$R_{gh}(D) = \sum_{j=1}^{n} g_j^\dagger(D^{-1}) h_j(D)$$

. A sequence $a(D)$ is orthogonal to all shifts $D^\ell b(D)$ of a sequence $b(D)$ iff $R_{ab}(D) = 0$.

With a single generator, we need to ensure $R_{gg}(D) = 0$.

Symmetries (preserve orthogonality and weights)

- Multiplication of any $g_j(D)$ by any monomial αD^ℓ, $\alpha \neq 0$, $\ell \in \mathbb{Z}$. $\rightarrow$ We can choose all $g_j(D)$ to be monic, that is, have smallest coefficient $g_{j0} = 1$.
- Conjugation $g(D) \rightarrow g^\dagger(D)$.
- Time-reversal $g(D) \rightarrow D^\nu g(D^{-1})$.
- Modulation $g(D) \rightarrow g(\alpha D)$ for any $0 \neq \alpha \in \mathbb{F}_4$.
- Permutation of components $g_j(D)$.

To get $d^\perp \geq 3$, $g_j(D)$ must be different (as long as they are monic).

To construct such a code, we just list polynomials of low degree, and write the corresponding $R_{aa}(D)$ (positive coefficients only). For rate-$1/n$ self-orthogonal convolutional code (CC), we need to find a subset of size n such that the corresponding sum of polynomials is zero.

The corresponding tail-biting codes of sufficiently large length have the same minimal distance, see[2].

statement: Let g_i, $i = 1, \ldots, m$ of degrees ν_i and $\sum_i \nu_i = \nu$ and h_j, $j = 1, \ldots, \tilde{m}$ of degrees $\tilde{\nu}_j$ be the minimum-degree generators of two orthogonal convolutional codes, $R_{g_j h_j}(D) = 0$. Then $\tilde{\nu} = \nu$, see[3]. Also see[4] for the analysis of the dual codes.

Matrix form of multiple-generator codes (matrix of dimension $m \times n$)

$$G \equiv G(D) = \begin{pmatrix} g_1(D) \\ g_2(D) \\ \vdots \\ g_m(D) \end{pmatrix},$$

where m is the number of generators. Generator $H = H(D)$ for dual convolutional code satisfies[4] $GH^T = 0$ (**or is it** $\bar{G}(D)H^T(D^{-1}) = 0$**?**) Here $\bar{G}$ denotes conjugation in GF(4).

Question: Is it true that $P(D)G(D) = 0$ would be a condition for a parity check matrix $P = P(D)$ of the syndrome code? Here $G(D)$ is the stabilizer generator matrix (with redundancy).

II. GENERATOR WEIGHTS AND TWO-GENERATOR CODES

Single-generator QCC codes in[1] have one drawback: the distances of orthogonal codes are small. Indeed, let $\nu = \deg g_1(D)$ be the memory of the original single-generator code (self-orthogonal). The corresponding dual generators $h_1(D)$, $h_2(D)$,... $h_{n-1}(D)$ have degrees $\tilde{\nu}_i$ that sum to ν. The larger n, the smaller degree polynomials one encounters. Generally, small degrees imply small(ish) distances.

Consider self-orthogonal QCCs $\mathcal{C}_1$ generated by $g_1(D)$ only, and $\mathcal{C}_2$ generated by $g_1(D)$ and $g_2(D)$. The latter code has larger ν, which might imply a better distance of the corresponding stabilizer code.

III. CYCLIC (AND QUASI-CYCLIC) CODES.

Consider a tail-biting code from a QCC, with mutually-orthogonal generator polynomials $g_i(D)$. The code will have the same distance if the block length is large enough (does not matter how large), and the rate is going to be the same.

In comparison, if we consider a quantum cyclic code[5] (single-generator additive cyclic codes introduced in[6]), with the same generator polynomial(s), increasing the length of the code by a factor of m will increase the distance by a factor of m, leaving the number of encoded qubits the same.

I believe that (even though the codes appear to be very similar) the Vitterbi decoding algorithm will not work for cyclic codes. **Is this correct?** It looks like there are some other trellis-search algorithms which are more or less efficient with block codes. See[7,8].

In any case, let us also consider the single-generator cyclic codes, see Sec. VI in Ref. 6. Statement is, these are relatively easy to construct, if we restrict ourselves to generator polynomials of small weight. If we look at an additive cyclic code of block length n with one generator of the form $g(x) = p(x)\omega + q(x)$, the conditions are (a) $p(x)$ divides $x^n - 1$, and (b) the code is self-orthogonal if[5]

$$p(x)\tilde{q}(x) = \tilde{p}(x^n)q(x) \pmod{x^n - 1},$$

where $\tilde{p}(x) = x^{\deg p(x)} p(1/x)$ is the corresponding reciprocal polynomial. (I think there should be also a condition on $q(x)$; in Ref. 6 we had a specific form $q(x) = p(x)r(x)$ but I am not sure whether this gives the most general case).

In any case, an easy prescription for searching such LDPC codes is to enumerate all polynomials of weight $w = 4, 5, \ldots$ and up to some maximum degree (for $w < 4$ one necessarily gets a code with $k = 0$), and then for each polynomial try to find an n which gives a valid cyclic code, and finally calculate the corresponding distances.

I did a search like this for Ref. 6 but the distance-finding routine was very slow; as a result, few interesting codes have been found (the only simple family of codes that we could identify was for rotated toric codes with $w = 4$ and parameters $[[5,1,3]]$, $[[13,1,5]]$, $[[25,1,7]]$, $[[41,1,9]]$, $\ldots$; generally $[[t^2 + (t+1)^2, 1, 2t+1]]$.

The advantage of the cyclic construction is that for an $[[n,k,d]]$ code only $n-k$ rows of the circulant matrix generated by $g(x)$ are going to be independent, while the remaining k rows will be redundant — thus, potentially useful to define a DS code. Moreover, I think the distance of the data portion of such a code would be equal to the weight of the generator (otherwise, there should be a smaller-weight generator).

Question: is there a class of codes which are in between cyclic and tail-biting codes, say, if one of the generator polynomials divides $D^m - 1$, where mn is the block length? As I understand, these would produce "catastrophic" QCCs, but could nevertheless contain codes with smaller weight of the stabilizer generators.. . .

Examples: Rotated hypergraph-product codes[9] can be rendered into one- or two-generator cyclic codes. So are the hypercubic codes.

Not sure about cubic codes constructed by Haah[10].

IV. QUASICYCLIC/TAIL-BITING CODES

Consider a general class of codes of block length $n = \ell m$, which are symmetric with respect to distance ℓ shifts: if $c = (c_1 \ldots c_{\ell m})$ is a codeword, then $Tc = (c_{\ell+1} c_{\ell+2} \ldots c_{\ell m} c_1 c_2 \ldots c_\ell)$ is also a codeword. It is convenient to represent the codewords as m blocks of length ℓ. Then a cyclic shift of a codeword in each block will give a codeword. Polynomial representation:

$$c = (c_{01} \ldots c_{0\ell} | \ldots | c_{m-1,1} \ldots c_{m-1,\ell}) \quad \Leftrightarrow \quad \mathbf{g}(x) = (g_1(x), g_2(x), \ldots g_\ell(x)),$$

where $g_j(x) = c_{0j} + x c_{1j} + \ldots x^{m-1} c_{m-1,j}$, $j \in \{1, \ldots, \ell\}$. Periodic shift correspond to a simultaneous replacement $g_j(x) \to x g_j(x) \bmod x^m - 1$, for all j.

Generally, a code may have several generators, $\mathbf{g}_1(x)$, $\ldots$, $\mathbf{g}_r(x)$. Use these as rows to form a polynomial $r \times \ell$ generator matrix $G(x)$, with matrix elements $g_{ij}(x)$, $1 \le i \le r$, $1 \le j \le \ell$.

Statement 1 (Orthogonality) *Consider polynomials with coefficients in GF(4), with the usual map to Pauli operators. Then the operators corresponding to polynomials $\mathbf{g}(x)$ and $\mathbf{h}(x)$ commute iff*

$$R_{\mathbf{gh}}(x) \equiv \sum_{j=1}^{\ell} \bar{g}_j(1/x) h_j(x) - \bar{h}_j(1/x) g_j(x) = 0 \bmod x^m - 1.$$

Here $\bar{h}(x)$ represents the usual conjugation of the coefficients, $\omega \to \bar{\omega} = \omega + 1$, and $g(1/x)$ is the (shifted) reciprocal polynomial; $\tilde{g}(x) = x^{\deg g(x)} g(1/x)$.

Notice that for tail-biting (TB) codes, we just have $R_{\mathbf{gh}}(x) = 0$, without the need for taking $\pmod{x^n - 1}$.

Statement 2 (Code equivalence) *The following gives equivalent QC/TB codes preserving orthogonality and weight spectrum:*

1. *Reversal* $G(x) \to \tilde{G}(x) = x^{n-1} G(1/x)$.
2. *Multiplication of any column of* $G(x)$ *by* αx^s, $0 \neq \alpha \in \mathrm{GF}(4)$; $s \in \mathbb{Z}$. *Explicitly,* $g_{ij}(x) \to \alpha x^s g_{ij}(x) \bmod x^m - 1$, *for some* $1 \leq j \leq \ell$ *and all* $i \in \{1, \ldots, r\}$.
3. *Modulation of any column of* $G(x)$ *by replacing* $x \to \alpha x$, $0 \neq \alpha \in GF(4)$. *Explicitly,* $g_{ij}(x) \to g_{ij}(\alpha x)$, *for some* $1 \leq j \leq \ell$ *and all* $i \in \{1, \ldots, r\}$.
4. *Column permutations*
5. *Conjugation* $G(x) \to \bar{G}(x)$
6. *Multiplication of any row of* $G(x)$ *by* αx^s, $0 \neq \alpha \in \mathrm{GF}(4)$; $s \in \mathbb{Z}$. *Explicitly, one has* $g_{ij}(x) \to \alpha x^s g_{ij}(x) \bmod x^m - 1$, *for some* $1 \leq i \leq r$ *and all* $j \in \{1, \ldots, \ell\}$.
7. *Nonsingular row transformations:*
   (a) *Row permutations*
   (b) $\mathbf{g}_i(x) \to \mathbf{g}_i(x) + \alpha x^s \mathbf{g}_{i'}(x) \bmod x^m - 1$, $i \neq i'$.
   (c) $\mathbf{g}_i(x) \to p(x) \boldsymbol{g}_i(x)$, *where* $\gcd(p(x), x^m - 1) = 1$

Only the transformations 1 to 6 and 7(a) preserve the weights of the generators.

Only the transformation 6 should be used to "canonicize" a given row without breaking the orthogonality with other rows.

Searching for such codes:

- Symmetry: implement search of a "canonical" form of the generators, which gives the alphabetically first generator matrix (row by row).
  - Encode ("packed form") a polynomial into a pair of integers (a, b), with the integer a giving the position of coefficients (e.g., rightmost byte c_0, second byte c_1, etc), and b listing the coefficients (two bits per coefficient). This way, a 16-byte (128 bit) unsigned integer a can store positions of up to 16 non-zero coefficients with $n \leq 127$. This should suffice for enumeration of codes, since the search space would be huge (I do not think we should bother with generators of weight more than 8 or 9).
  - Implement comparison of two polynomials alphabetically, first a_1 and a_2, then (if these are equal) b_1 and b_2. This way polynomials of lower degree will go first.
  - Implement search for the "canonical" form of a given polynomial using transformations 1 to 5, for given ℓ and m. A polynomial will only be used as the first generator if it is in the canonical form (given $g(x)$, no combination of these transformations make a polynomial which goes before $g(x)$ alphabetically).

- Implement enumeration of polynomials of a given weight, so that alphabetically first polynomials would appear first (ideally, only canonical ones would appear).
- Implement reduced search for "canonical" form of second and subsequent generators using transformation 6 only (complexity $3w$, where w is the number of non-zero coefficients).
- Implement calculation of $R_{\mathbf{g},\mathbf{h}}(x)$.
- Implement pretty printout, conversion (e.g., of a polynomial array to a matrix), and input/output to a file.

- Single-generator: given m and ℓ, enumerate small-weight polynomials $\mathbf{g}(x)$ which are self-orthogonal and also are in the canonical form. Only keep those that give k large enough (distance can be increased by adding generators).
- Two- or more-generator codes: start with a single self-orthogonal polynomial. Form the corresponding generator matrix, search for small-weight codewords in the orthogonal code which are also self-orthogonal (e.g., by running Random Window for a while and reducing every found codeword that is self-orthogonal to canonical form using transformation 6 only; keep it only if it goes alphabetically after the last of the already chosen generators); form new codes by adding one of found codewords to the list of generators; do it recursively to find codes with $r = 2$, 3, etc generators. This way the rows will automatically be canonically ordered — transformation 7(a) not needed!
- Calculate the code parameters (dimension, distance, etc.) and whatever else properties needed; keep only "good enough" codes which satisfy certain constraints set (e.g., on k, d, or on the parameters of the syndrome code).

In the special case of TB codes use the reduced list of transformations; only list generators up to a given maximum degree. I suppose if the polynomial $\mathbf{g}(x)$ is irreducible, this guarantees that at large enough m distance will be equal to the distance of the corresponding QCC.

In the special case of cyclic codes Theorem 14 in Ref. 5 states that two generators should suffice. (It also gives some additional properties of such polynomials which I think we do not need to check since we are only interested in small-weight generators). I think with $\ell > 1$, generally, one should use no more than 2ℓ generators.

Think more: (1) how to search for small-weight dual polynomials in the case of TB codes. (2) how to define the "packed form" that would be invariant to changes of m (and possibly ℓ). E.g., list the position of the non-zero coefficient for x^s in $g_j(x)$ as $j + \ell s$.

---

[1] G. D. Forney, M. Grassl, and S. Guha, "Convolutional and tail-biting quantum error-correcting codes," IEEE Transactions on Information Theory **53**, 865–880 (2007).

[2] M. Handlery, S. Höst, R. Johannesson, and V. V. Zyablov, "A distance measure tailored to tailbiting codes," Problems of Information Transmission **38**, 280–295 (2002), [Translated from Problemy Peredachi Informatsii, **38**, 37-55 (2002)].

[3] G. Forney, "Convolutional codes I: Algebraic structure," IEEE Transactions on Information Theory **16**, 720–738 (1970).

[4] G. Forney, “Structural analysis of convolutional codes via dual codes,” IEEE Transactions on Information Theory **19**, 512–518 (1973).

[5] A. R. Calderbank, E. M. Rains, P. M. Shor, and N. J. A. Sloane, “Quantum error correction via codes over GF(4),” IEEE Trans. Info. Theory **44**, 1369–1387 (1998).

[6] A. A. Kovalev, I. Dumer, and L. P. Pryadko, “Design of additive quantum codes via the code-word-stabilized framework,” Phys. Rev. A **84**, 062319 (2011).

[7] Y. S. Han, C. R. P. Hartmann, and K. G. Mehrotra, “Decoding linear block codes using a priority-first search: performance analysis and suboptimal version,” IEEE Transactions on Information Theory **44**, 1233–1246 (1998).

[8] Yunghsiang S. Han, C. R. P. Hartmann, and Chih-Chieh Chen, “Efficient maximum-likelihood soft-decision decoding of linear block codes using algorithm A*,” in *Proceedings. IEEE International Symposium on Information Theory* (1993) pp. 27–27.

[9] A. A. Kovalev and L. P. Pryadko, “Quantum Kronecker sum-product low-density parity-check codes with finite rate,” Phys. Rev. A **88**, 012311 (2013).

[10] Jeongwan Haah, “Local stabilizer codes in three dimensions without string logical operators,” Phys. Rev. A **83**, 042330 (2011), arXiv:1101.1962.