Algebraic lower bounds on the free distance of convolutional codes

Lally, Kristine

Lally, K. (2005). Algebraic lower bounds on the free distance of convolutional codes. Proceedings of the 2005 IEEE International Symposium on Information Theory, 2101–2110. https://doi.org/10.1109/TIT.2006.872980

Published Version: https://doi.org/10.1109/TIT.2006.872980

Algebraic Lower Bounds on the Free Distance of Convolutional Codes

Kristine Lally

***Abstract*—A new module structure for convolutional codes is introduced and used to establish further links with quasi-cyclic and cyclic codes. The set of finite weight codewords of an (n, k) convolutional code over F_q is shown to be isomorphic to an $F_q[x]$-submodule of $F_{q^n}[x]$, where $F_{q^n}[x]$ is the ring of polynomials in indeterminate x over F_{q^n}, an extension field of F_q. Such a module can then be associated with a quasi-cyclic code of index n and block length nL viewed as an $F_q[x]$-submodule of $F_{q^n}[x]/\langle x^L - 1\rangle$, for any positive integer L. Using this new module approach algebraic lower bounds on the free distance of a convolutional code are derived which can be read directly from the choice of polynomial generators. Links between convolutional codes and cyclic codes over the field extension F_{q^n} are also developed and Bose–Chaudhuri–Hocquenghem (BCH)-type results are easily established in this setting. Techniques to find the optimal choice of the parameter L are outlined.**

***Index Terms*—Convolutional codes, cyclic codes, free distance, lower bound, quasi-cyclic codes.**

I. Introduction

An (n, k) convolutional code is the set of all output sequences produced by a linear function which maps a (possibly infinite) input sequence $\boldsymbol{u} = (\boldsymbol{u}_0, \boldsymbol{u}_1, \boldsymbol{u}_2, \ldots)$ of k-tuples to a (possibly infinite) output sequence $\boldsymbol{v} = (\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots)$ of n-tuples, $n \geq k$, where each n-tuple output block $\boldsymbol{v}_l$ produced at time l depends on the corresponding k-tuple input block $\boldsymbol{u}_l$ and on some of the m previous input blocks $\boldsymbol{u}_{l-1}, \boldsymbol{u}_{l-2}, \ldots, \boldsymbol{u}_{l-m}$ that entered the encoder and were stored. The parameter m is called the memory order of the encoder.

There is a strong link between convolutional codes and (memoryless) block codes, in particular cyclic and quasi-cyclic block codes, shown in the work of Solomon and van Tilborg [20], Levy and Costello [10], and Tanner [21], and more recently by Rosenthal and York [15] and Smarandache *et al.* [19]. Lower bounds on the free distance of convolutional codes have been developed by Massey *et al.* [13], Costello [1] and Justesen [7], amongst others.

In this paper we develop a new module representation for convolutional codes. A similar idea was previously used by Séguin [17] to describe convolutional codes when k divides n, and by this author [9] and more recently by Séguin [18] to describe quasi-cyclic codes. Using our new module approach we establish further links between convolutional codes and cyclic and quasi-cyclic codes. Our main result is an algebraic lower bound on the free distance of a convolutional code which can be read directly from the choice of polynomial generators.

We conclude this first section with a brief introduction to the conventional algebraic structure of convolutional codes, and in particular the notion of a basic polynomial generator matrix which is needed throughout this paper. We refer the reader to [2], [6], [12], [14] for further details.

Generalizing the subspace structure of a linear block code, an (n, k) convolutional code C over F_q can be viewed (see [14]) as a k-dimensional vector subspace of $F_q(D)^n$, where $F_q(D)$ is the field of rational functions in indeterminate D over F_q, that is

$$F_q(D) = \left\{ \frac{p(D)}{q(D)} \mid p(D), q(D) \in F_q[D], q(D) \neq 0 \right\}.$$

A generator matrix (or transfer function matrix) for the code is a $k \times n$ matrix $G(D)$ over $F_q(D)$ whose rows form a basis for C. An input vector

$$\boldsymbol{u}(D) = [u^{(0)}(D), u^{(1)}(D), \ldots, u^{(k-1)}(D)] \in F_q(D)^k$$

is encoded to the output vector

$$\boldsymbol{v}(D) = [v^{(0)}(D), v^{(1)}(D), \ldots, v^{(n-1)}(D)] \in F_q(D)^n$$

by the mapping

$$\boldsymbol{u}(D)G(D) = \boldsymbol{v}(D)$$

where each rational function $u^{(j)}(D)$ and $v^{(j)}(D)$ has a unique expansion as a one-sided formal Laurent series. The code is the set

$$C = \{\boldsymbol{u}(D)G(D) \mid \boldsymbol{u}(D) \in F_q(D)^k\} \subseteq F_q(D)^n.$$

In practice, only generator matrices with causal rational entries (that is, realizable encoders) and input and output vectors with causal Laurent series are achievable. The weight of a vector $\boldsymbol{v}(D) \in F_q(D)^n$ is the sum of the weights of its component entries. The free distance of a convolutional code, denoted $d_{free}(C)$, is the minimum weight of any nonzero output vector obtained from a causal input vector (that is, one whose entries are all causal Laurent series of the form $u^{(j)}(D) = u_0^{(j)} + u_1^{(j)}D + u_2^{(j)}D^2 + \cdots$). The free distance of a convolutional code is an important parameter in determining the error correcting capability of the code.

Manuscript received February 7, 2005; revised November 18, 2005. The material in this paper was presented in part at the 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, September 2005.

The author is with the Department of Mathematics and Statistics, RMIT University, Melbourne, Australia (e-mail: kristine.lally@rmit.edu.au).

Communicated by C. Carlet, Associate Editor for Coding Theory.

Digital Object Identifier 10.1109/TIT.2006.872980

0018-9448/$20.00 © 2006 IEEE

Any generator matrix $G'(D)$ which is $F_q(D)$-row equivalent to $G(D)$ is also a generator matrix for C. It follows that every convolutional code has a polynomial generator matrix (PGM), that is, a $k \times n$ matrix of the form

$$G(D) = \begin{pmatrix} g_1^{(0)}(D) & g_1^{(1)}(D) & \cdots & g_1^{(n-1)}(D) \\ g_2^{(0)}(D) & g_2^{(1)}(D) & \cdots & g_2^{(n-1)}(D) \\ \vdots & \vdots & & \vdots \\ g_k^{(0)}(D) & g_k^{(1)}(D) & \cdots & g_k^{(n-1)}(D) \end{pmatrix} \quad (1)$$

with entries

$$g_i^{(j)}(D) = g_{i,0}^{(j)} + g_{i,1}^{(j)}D + g_{i,2}^{(j)}D^2 + \cdots + g_{i,m}^{(j)}D^m \in F_q[D],$$

$1 \leq i \leq k, 0 \leq j \leq n-1$. In this case, the maximum of the degrees of the polynomial entries is the memory order m of the encoder.

A PGM $G(D)$ can be expanded as the polynomial

$$G(D) = G_0 + G_1 D + \cdots + G_m D^m$$

with matrix coefficients $G_l = [g_{i,l}^{(j)}]_{k \times n}, 0 \leq l \leq m$. The matrix

$$\boldsymbol{G} = \begin{pmatrix} G_0 & G_1 & \cdots & \cdots & G_m & & \\ & G_0 & G_1 & \cdots & \cdots & G_m & \\ & & \ddots & \ddots & & & \ddots \end{pmatrix} \quad (2)$$

is called a semi-infinite scalar generator matrix for the code C and encodes the causal information sequences

$$\begin{aligned}\boldsymbol{u} &= (\boldsymbol{u}_0, \boldsymbol{u}_1, \boldsymbol{u}_2, \ldots) \\ &= (u_0^{(0)}u_0^{(1)} \ldots u_0^{(k-1)}, u_1^{(0)}u_1^{(1)} \ldots u_1^{(k-1)}, \ldots)\end{aligned}$$

to the causal codeword sequences

$$\begin{aligned}\boldsymbol{v} = \boldsymbol{u}\boldsymbol{G} &= (\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots) \\ &= (v_0^{(0)}v_0^{(1)} \ldots v_0^{(n-1)}, v_1^{(0)}v_1^{(1)} \ldots v_1^{(n-1)}, \ldots).\end{aligned}$$

Applying further $F_q(D)$-row operations to a PGM a more restrictive generator matrix for C can be obtained. We include the following definition from [14].

Definition 1: A $k \times n$ PGM $G(D)$ for a convolutional code is *basic* if and only if any of the following equivalent conditions is satisfied.

1) The gcd of all $k \times k$ minors of $G(D)$ is 1.
2) $G(D)$ has a right $F_q[D]$ inverse, that is, there exists an $n \times k$ polynomial matrix $B(D)$ such that $G(D)B(D) = I_k$.
3) if $\boldsymbol{v}(D) = \boldsymbol{u}(D)G(D)$ and $\boldsymbol{v}(D) \in F_q[D]^n$ then $\boldsymbol{u}(D) \in F_q[D]^k$, that is, polynomial output implies polynomial input.

Every (n,k) convolutional code C has a $k \times n$ basic PGM $G(D)$. It is well known that if $G(D)$ is a basic PGM for an (n,k) convolutional code C then there exists an $(n-k) \times n$ polynomial matrix $H(D)$ with rank $n-k$ such that $G(D)H(D)^T = \boldsymbol{0}$. $H(D)$ is called a parity check matrix for C and any $\boldsymbol{v}(D) \in F_q(D)^n$ satisfies $\boldsymbol{v}(D)H(D)^T = \boldsymbol{0}$ if and only if $\boldsymbol{v}(D) \in C$. Expanding this matrix

$$H(D)^T = \begin{pmatrix} h_1^{T(0)}(D) & h_1^{T(1)}(D) & \cdots & h_1^{T(n-k-1)}(D) \\ h_2^{T(0)}(D) & h_2^{T(1)}(D) & \cdots & h_2^{T(n-k-1)}(D) \\ \vdots & \vdots & & \vdots \\ h_n^{T(0)}(D) & h_n^{T(1)}(D) & \cdots & h_n^{T(n-k-1)}(D) \end{pmatrix} \quad (3)$$

with entry

$$h_i^{T(j)}(D) = h_{i,0}^{T(j)} + h_{i,1}^{T(j)}D + h_{i,2}^{T(j)}D^2 + \cdots + h_{i,m_s}^{T(j)}D^{m_s}$$

in $F_q[D]$, $1 \leq i \leq n$, $0 \leq j \leq n-k-1$, we can form the polynomial

$$H(D)^T = H_0^T + H_1^T D + \cdots + H_{m_s}^T D^{m_s}$$

with matrix coefficients $H_l^T = \left[h_{i,l}^{T(j)}\right]_{n \times (n-k)}, 0 \leq l \leq m_s$. The semi-infinite scalar parity check matrix

$$\boldsymbol{H}^T = \begin{pmatrix} H_0^T & H_1^T & \cdots & \cdots & H_{m_s}^T & & \\ & H_0^T & H_1^T & \cdots & \cdots & H_{m_s}^T & \\ & & \ddots & \ddots & & & \ddots \end{pmatrix}$$

satisfies

$$\boldsymbol{G}\boldsymbol{H}^T = \boldsymbol{0} \quad (4)$$

where $\boldsymbol{G}$ is given in (2). A sequence $\boldsymbol{v} = (\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots)$ of n-tuples satisfies $\boldsymbol{v}\boldsymbol{H}^T = \boldsymbol{0}$ if and only if $\boldsymbol{v} \in C$.

II. New Module Structure

Let C be an (n,k) convolutional code over F_q with a $k \times n$ basic PGM $G(D)$ over $F_q[D]$. We recall that C is the subspace of $F_q(D)^n$ spanned by the rows of $G(D)$. We first isolate an important polynomial submodule of C which is of key interest to us here. It is well known that a basic PGM is an noncatastrophic encoder, that is, a finite weight output sequence cannot be obtained from an infinite weight input sequence. Every finite weight Laurent series is a rational function of the form $p(D)/D^\ell$, where $p(D) \in F_q\,[D]$ and ℓ is a nonnegative integer. For our purposes here (determining lower bounds on d_{free}) we consider only output vectors produced from causal input vectors. It is a simple fact that a PGM encodes a causal input vector to a causal output vector. We note that every causal finite-weight Laurent series is in fact a polynomial. Thus it follows from Definition 1 part 3) that the set C' of all causal finite weight codewords in our convolutional code C is the set of all output vectors produced by input vectors with polynomial entries, that is

$$C' = \left\{\boldsymbol{v}(D) = \boldsymbol{u}(D)G(D) \mid \boldsymbol{u}(D) \in F_q[D]^k\right\} \subseteq F_q[D]^n$$

where $G(D)$ is a basic PGM. The free distance of C is therefore the minimum nonzero weight of the codewords in C'. Henceforth we ignore all infinite weight codewords and make no distinction between C and C'. A convolutional code C can, therefore, be viewed as an $F_q\,[D]$-submodule of $F_q\,[D]^n$ generated by the k rows of a basic PGM $G(D)$. We note that this view of a convolutional code was also adopted in recent papers such as [3] and [4].

We now introduce a new module structure for convolutional codes which follows naturally from our initial definition of a convolutional code given at the beginning of Section I, and is an isomorphic image of the polynomial module structure mentioned above. Let us consider a (finite weight causal) codeword

$$\boldsymbol{v}=(\boldsymbol{v}_0,\boldsymbol{v}_1,\ldots)=(v_0^{(0)}v_0^{(1)}\ldots v_0^{(n-1)},v_1^{(0)}v_1^{(1)}\ldots v_1^{(n-1)},\ldots)$$

over F_q as a sequence of blocks of length n, and associate each n-tuple $\boldsymbol{v}_l=(v_l^{(0)},v_l^{(1)},\ldots,v_l^{(n-1)})\in F_q^n$ with an element $v_l=v_l^{(0)}+v_l^{(1)}\alpha+\cdots+v_l^{(n-1)}\alpha^{n-1}$ of the field extension F_{q^n}, where $\{1,\alpha,\alpha^2,\ldots,\alpha^{n-1}\}$ is some fixed choice of basis of F_{q^n} as a vector space over F_q. We then associate the entire codeword $\boldsymbol{v}=(\boldsymbol{v}_0,\boldsymbol{v}_1,\boldsymbol{v}_2,\ldots)$ with the polynomial $v(D)=v_0+v_1D+v_2D^2+\cdots$ in $F_{q^n}[D]$ where as usual the increasing powers of the delay operator D indicate successive inputs over time.

This same association can be achieved from the causal vector

$$\boldsymbol{v}(D)=[v^{(0)}(D),v^{(1)}(D),\ldots,v^{(n-1)}(D)]\in F_q[D]^n$$

with

$$v^{(j)}(D)=v_0^{(j)}+v_1^{(j)}D+v_2^{(j)}D^2+\cdots$$

by the mapping

$$\begin{aligned}\phi(\boldsymbol{v}(D))&=v^{(0)}(D)+\alpha v^{(1)}(D)+\cdots+\alpha^{n-1}v^{(n-1)}(D)\\&=v_0+v_1D+v_2D^2+\cdots\\&=v(D)\in F_{q^n}[D].\end{aligned}$$

The mapping ϕ defines an $F_q[D]$-module isomorphism between $F_q[D]^n$ and $F_{q^n}[D]$ which preserves F_q-weight structure of the submodules. Henceforth, we fix $\{1,\alpha,\alpha^2,\ldots,\alpha^{n-1}\}$ as our choice of basis for F_{q^n}, and assume the association between elements $\boldsymbol{v}_l\in F_q^n$ and $v_l\in F_{q^n}$ described earlier, without further comment. It follows that an (n,k) convolutional code C can be viewed as an $F_q[D]$-submodule of $F_{q^n}[D]$, generated by the images under ϕ of the rows of a basic PGM $G(D)$, that is, the polynomials

$$\begin{aligned}g_i(D)&=g_i^{(0)}(D)+\alpha g_i^{(1)}(D)+\cdots+\alpha^{n-1}g_i^{(n-1)}(D)\\&=g_{i,0}+g_{i,1}D+\cdots+g_{i,m}D^m\in F_{q^n}[D],\\&\quad 1\le i\le k.\end{aligned}$$

The code C is the set

$$C=\Big\{u^{(0)}(D)g_1(D)+\cdots+u^{(k-1)}(D)g_k(D)\mid u^{(i)}(D)\in F_q[D],0\le i\le k-1\Big\}\subseteq F_{q^n}[D]$$

and as a vector space over F_q is generated by the set $\{D^sg_i(D),1\le i\le k,s=0,1,\ldots\}$ where $g_i(D)=g_{i,0}+g_{i,1}D+g_{i,2}D^2+\cdots+g_{i,m}D^m\in F_{q^n}[D]$ and each coefficient $g_{i,l}=g_{i,l}^{(0)}+g_{i,l}^{(1)}\alpha+\cdots+g_{i,l}^{(n-1)}\alpha^{n-1}\in F_{q^n}$.

A generator matrix for the code can be constructed from these k polynomial generators in the form

$$G=\begin{pmatrix}g_1(D)\\ \vdots\\ g_k(D)\\ Dg_1(D)\\ \vdots\\ Dg_k(D)\\ \vdots\end{pmatrix}
=\begin{pmatrix}g_{1,0}&g_{1,1}&\cdots&g_{1,m}&&\\ \vdots&\vdots&&\vdots&&\\ g_{k,0}&g_{k,1}&\cdots&g_{k,m}&&\\ &g_{1,0}&\cdots&g_{1,m-1}&g_{1,m}&\\ &\vdots&&&\vdots&\\ &g_{k,0}&\cdots&g_{k,m-1}&g_{k,m}&\\ &&\ddots&&\ddots&\ddots\end{pmatrix}$$

where each $g_{i,l}$ represents the vector $\boldsymbol{g}_{i,l}=\left(g_{i,l}^{(0)},g_{i,l}^{(1)},\ldots,g_{i,l}^{(n-1)}\right)\in F_q^n$, $1\le i\le k$, $0\le l\le m$. We call this G a *semi-infinite polynomial generator* matrix for the code C. The memory order m of the generator matrix is equal to $\max\limits_{1\le i\le k}\deg(g_i(D))$. The shift action by n places on the rows of the semi-infinite scalar generator matrix $\boldsymbol{G}$ given in (2) corresponds here to multiplication by D of the polynomial generator $g_i(D)\in F_{q^n}[D]$.

The rows of the semi-infinite parity check matrix $\boldsymbol{H}^T$ can be associated in a similar way with polynomials in $F_{q^{n-k}}[D]$ to form the semi-infinite polynomial parity check matrix

$$H^T=\begin{pmatrix}h_1^T(D)\\ \vdots\\ h_n^T(D)\\ Dh_1^T(D)\\ \vdots\\ Dh_n^T(D)\\ \vdots\end{pmatrix}$$

where

$$\begin{aligned}h_i^T(D)&=h_i^{T(0)}(D)+\beta h_i^{T(1)}(D)+\cdots\\&\quad+\beta^{n-k-1}h_i^{T(n-k-1)}(D)\\&=h_{i,0}^T+h_{i,1}^TD+h_{i,2}^TD^2+\cdots\\&\quad+h_{i,m_s}^TD^{m_s}\in F_{q^{n-k}}[D]\end{aligned}$$

for $1\le i\le n$, and each coefficient

$$h_{i,l}^T=h_{i,l}^{T(0)}+h_{i,l}^{T(1)}\beta+\cdots+h_{i,l}^{T(n-k-1)}\beta^{n-k-1}\in F_{q^{n-k}},$$

$0\le l\le m_s$, is a F_q-linear combination of the F_q-basis $\{1,\beta,\beta^2,\ldots,\beta^{n-k-1}\}$ of $F_{q^{n-k}}$. It follows from (4) that the polynomial $v(D)=v^{(0)}(D)+\alpha v^{(1)}(D)+\cdots+\alpha^{n-1}v^{(n-1)}(D)\in F_{q^n}[D]$ with $v^{(j)}(D)\in F_q[D]$, $0\le j\le n-1$, is a codeword in the convolutional code C if and only if $v^{(0)}(D)h_1^T(D)+v^{(1)}(D)h_2^T(D)+\cdots+v^{(n-1)}(D)h_n^T(D)=0$. We call the polynomials $h_i^T(D)$, $1\le i\le n$, *parity check polynomials* of the code.

III. Links to Quasi-Cyclic Codes

A quasi-cyclic code of index n and length $N = nL$ over F_q is a block linear code invariant under cyclic shifts by n places on its codewords, and is usually constructed as the rowspace of a block matrix consisting of rows of $L \times L$ circulant submatrices, and as such corresponds to an R-submodule of R^n where $R = F_q[x]/I$ and $I = \langle x^L - 1 \rangle$, [8], [16]. Links between convolutional codes and quasi-cyclic codes in this conventional 'circulant' form have been previously studied by many authors, for example, see [10], [20]. In this paper we establish links between convolutional codes and quasi-cyclic codes using our new alternative module approach and develop many useful results not forthcoming within the conventional context.

It has recently been shown in [9] that a quasi-cyclic code of index n and length nL can be viewed as an $F_q[x]$-submodule of $F_{q^n}[x]/I$. In this setting the rows of a generator matrix of a quasi-cyclic code are successive powers of x multiplied modulo $x^L - 1$ by each of the generating polynomials of the submodule in $F_{q^n}[x]/I$. The similarity to our new module representation of a convolutional code is immediately evident. Changing indeterminate D to x wherever previously used, we can associate an (n,k) convolutional code C, viewed as an $F_q[x]$-submodule of $F_{q^n}[x]$, with the quasi-cyclic block code C_L of index n and length nL, an $F_q[x]$-submodule of $F_{q^n}[x]/I$, for any positive integer L, by simply mapping each codeword $c(x) \in C$ to the codeword $c'(x) + I$ in C_L where $c'(x) = c(x) \bmod x^L - 1$. In this latter module we usually drop the coset notation, write $c'(x)$ for $c'(x) + I$, where $c'(x)$ is the unique polynomial of degree less than L in the coset $c'(x) + I$, and perform multiplication modulo $x^L - 1$. If the convolutional code C is generated by the polynomials $g_i(x) = g_{i,0} + g_{i,1}x + g_{i,2}x^2 + \cdots + g_{i,m}x^m \in F_{q^n}[x]$, $1 \le i \le k$, then the corresponding quasi-cyclic code is generated by $g'_i(x) = g_i(x) \bmod x^L - 1$, $1 \le i \le k$, where each $g'(x)$ is a polynomial in $F_{q^n}[x]/I$ of degree at most $L - 1$. The quasi-cyclic code C_L has dimension at most kL as a vector space over F_q and thus has rate at most $kL/nL = k/n$.

Using this link between a convolutional code C and the corresponding quasi-cyclic code C_L we now present a general lower bound on the free distance of C in the next theorem. This bound was previously derived by Tanner [21] using a polynomial parity check matrix $H(D)$ [given in (3)] to characterize C and the corresponding 'circulant' representation of C_L. We note however that finding the minimum distance of a quasi-cyclic code in this conventional context is not easy, no good algebraic lower bounds have been developed and good codes have largely been found by computationally intensive searching techniques (for example in [5]). Here we present the same general bound in the context of our new module representation for convolutional codes and the associated unconventional quasi-cyclic representation. The proof is a direct analogue of that given by Tanner. Our new module structure allows us to extend this connection between convolutional and block codes and establish further links to cyclic codes. Results derived in [9] for quasi-cyclic codes can then be adapted and applied to convolutional codes to develop more constructive lower bounds on $d_{\text{free}}(C)$, in particular a BCH-type bound derived from an associated cyclic code over an extension field. To achieve these additional results some aspects of our alternative proof of the following theorem are required in subsequent sections, and thus included here.

Let $c(x) = c_0 + c_1x + \cdots + c_tx^t \in F_{q^n}[x]$. We make the following distinction when considering the weight of $c(x)$. We denote by $wt_{q^n}(c(x))$ the number of nonzero coefficients of $c(x)$, that is, its weight as a polynomial in $F_{q^n}[x]$. We denote by $wt_q(c(x))$ the sum of the F_q-weights of the nonzero coefficients, where each coefficient is considered as a vector of length n in F_q^n.

Theorem 2: If C is an (n,k) convolutional code over F_q with basic PGM $G(x) = [g_i^{(j)}(x)]_{k\times n}$ over $F_q[x]$ and generators

$$\begin{aligned} g_i(x) &= g_i^{(0)}(x) + \alpha g_i^{(1)}(x) + \cdots + \alpha^{n-1} g_i^{(n-1)}(x) \\ &= g_{i,0} + g_{i,1}x + g_{i,2}x^2 + \cdots + g_{i,m}x^m \in F_{q^n}[x] \end{aligned}$$

for $1 \le i \le k$, as an $F_q[x]$-submodule of $F_{q^n}[x]$ then

$$d_{\text{free}}(C) \ge d_{\min}(C_L)$$

where C_L is the quasi-cyclic code over F_q of index n and length nL generated by $g_i(x) \bmod x^L - 1 \in F_{q^n}[x]/I$, $1 \le i \le k$, for any positive integer L.

Proof: If $c(x) = c_0 + c_1x + \cdots + c_tx^t \in C$ then $c'(x) = c(x) \bmod x^L - 1 \in C_L$. Each coefficient of $c'(x) = c'_0 + c'_1x + \cdots + c'_{L-1}x^{L-1} \in F_{q^n}[x]/I$ is formed as the sum $c'_l = \sum_{s\ge 0} c_{l+sL}$ of coefficients from $c(x)$, and thus it is easily seen that $wt_{q^n}(c(x)) \ge wt_{q^n}(c'(x))$. Furthermore the summation of elements in F_{q^n} is equivalent to the componentwise summation of the corresponding vector representation in F_q^n. Each component of the vector $\boldsymbol{c}'_l = (c_l'^{(0)}, c_l'^{(1)}, \ldots, c_l'^{(n-1)}) \in F_q^n$ is therefore formed as the sum $c_l'^{(j)} = \sum_{s\ge 0} c_{l+sL}^{(j)}$. It follows that $wt_q(c(x)) \ge wt_q(c'(x))$. Hence if $c'(x) \ne 0$ then $wt_q(c(x)) \ge d_{\min}(C_L)$. However if $c(x) \ne 0$ and $c'(x) = 0$ then $c(x) = k(x)(x^L - 1) \in F_{q^n}[x]$ for some nonzero multiple $k(x) \in F_{q^n}[x]$. Let $(x^L - 1)^\gamma$ be the highest (positive integer) power of $x^L - 1$ which divides $c(x)$. We write

$$\begin{aligned} c(x) &= (x^L - 1)^\gamma y(x) \\ &= (x^L - 1)^\gamma (y^{(0)}(x) + \alpha y^{(1)}(x) + \cdots \\ &\qquad + \alpha^{n-1} y^{(n-1)}(x)) \\ &= c^{(0)}(x) + \alpha c^{(1)}(x) + \cdots + \alpha^{n-1} c^{(n-1)}(x) \in F_{q^n}[x] \end{aligned}$$

where $y(x) \in F_{q^n}[x]$ is nonzero, $c^{(j)}(D) = (x^L - 1)^\gamma y^{(j)}(x) \in F_q[x]$, $0 \le j \le n-1$, and $(x^L - 1) \nmid y(x)$.

Since $G(x)$ is a basic PGM there exist parity check polynomials $h_i^T(x)$, $1 \le i \le n$ in $F_{q^{n-k}}[x]$. If $c(x) \in C$ then

$$\begin{aligned} &c^{(0)}(x)h_1^T(x) + c^{(1)}(x)h_2^T(x) + \cdots + c^{(n-1)}(x)h_n^T(x) \\ &= (x^L - 1)^\gamma y^{(0)}(x)h_1^T(x) + (x^L - 1)^\gamma y^{(1)}(x)h_2^T(x) + \cdots \\ &\quad + (x^L - 1)^\gamma y^{(n-1)}(x)h_n^T(x) \\ &= (x^L - 1)^\gamma \Big(y^{(0)}(x)h_1^T(x) + y^{(1)}(x)h_2^T(x) + \cdots \\ &\qquad\qquad + y^{(n-1)}(x)h_n^T(x) \Big) \\ &= 0. \end{aligned}$$

As $F_{q^{n-k}}[x]$ is an integral domain, this implies that $y^{(0)}(x)h_1^T(x) + \cdots + y^{(n-1)}(x)h_n^T(x) = 0$ and therefore $y(x) \in C$ also. It follows that $y'(x) = y(x) \bmod x^L - 1 \in C_L$ and since $(x^L - 1) \nmid y(x)$ we have $y'(x) \ne 0$. We now show that if $c(x) \ne 0$ and $c'(x) = 0$ then $wt_q(c(x)) \ge wt_q(y'(x))$

where $y'(x)$ is the nonzero codeword in C_L obtained as earlier. First applying the weight preserving property given in [13, Theorem 6.3] in $F_{q^n}[x]$ we find that

$$\begin{aligned} wt_{q^n}(c(x)) &= wt_{q^n}(y(x)(x^L-1)^\gamma) \\ &\geq wt_{q^n}((x-1)^\gamma) wt_{q^n}(y(x) \bmod (x^L-1)) \\ &\geq wt_{q^n}(y(x) \bmod (x^L-1)) = wt_{q^n}(y'(x)). \end{aligned}$$

Furthermore, we also have $c(x) = c^{(0)}(x) + \alpha c^{(1)}(x) + \cdots + \alpha^{n-1}c^{(n-1)}(x)$ with $c^{(j)}(x) \in F_q[x]$, $0 \leq j \leq n-1$, and now applying [13, Theorem 6.3] in $F_q[x]$ we find

$$\begin{aligned} wt_q(c(x)) &= \sum_{j=0}^{n-1} wt_q(c^{(j)}(x)) = \sum_{j=0}^{n-1} wt_q(y^{(j)}(x)(x^L-1)^\gamma) \\ &\geq \sum_{j=0}^{n-1} wt_q((x-1)^\gamma) wt_q(y^{(j)}(x) \bmod (x^L-1)) \\ &\geq \sum_{j=0}^{n-1} wt_q(y^{(j)}(x) \bmod (x^L-1)) \\ &= wt_q(y(x) \bmod (x^L-1)) = wt_q(y'(x)). \end{aligned}$$

Hence for any nonzero $c(x) \in C$ we have $wt_q(c(x)) \geq d_{\min}(C_L)$ and our theorem follows. □

IV. Single-Input ($k = 1$) Convolutional Codes

An $(n, 1)$ convolutional code over F_q has a $1 \times n$ basic PGM $G(x) = [g_1^{(j)}(x)]_{0 \leq j \leq n-1}$ over $F_q[x]$ and as an $F_q[x]$-submodule of $F_{q^n}[x]$ is generated by a single polynomial

$$\begin{aligned} g_1(x) &= g_1^{(0)}(x) + \alpha g_1^{(1)}(x) + \cdots + \alpha^{n-1} g_1^{(n-1)}(x) \\ &= g_{1,0} + g_{1,1}x + g_{1,2}x^2 + \cdots + g_{1,m}x^m \in F_{q^n}[x] \end{aligned}$$

where each coefficient $g_{i,l} = g_{i,l}^{(0)} + g_{i,l}^{(1)}\alpha + \cdots + g_{i,l}^{(n-1)}\alpha^{n-1} \in F_{q^n}$. The code C is the set

$$C = \left\{ u^{(0)}(x) g_1(x) \mid u^{(0)}(x) \in F_q[x] \right\} \subseteq F_{q^n}[x].$$

As an F_q-vector space C is generated by

$$\{ g_1(x), x g_1(x), x^2 g_1(x), \ldots \} \subseteq F_{q^n}[x]$$

and has a semi-infinite polynomial generator matrix of the form

$$G = \begin{pmatrix} g_1(x) \\ x g_1(x) \\ x^2 g_1(x) \\ \vdots \end{pmatrix} = \begin{pmatrix} g_{1,0} & g_{1,1} & \cdots & g_{1,m} & & & \\ & g_{1,0} & g_{1,1} & \cdots & g_{1,m} & & \\ & & g_{1,0} & g_{1,1} & \cdots & g_{1,m} & \\ & & & \ddots & \ddots & & \ddots \end{pmatrix}. \tag{5}$$

The PGM $G(x)$ is basic and so the corresponding generator $g_1(x) \in F_{q^n}[x]$ is not arbitrary as the following theorem shows.

Theorem 3: The $(n, 1)$ convolutional code C over F_q has basic PGM $G(x)$ over $F_q[x]$ if and only if the polynomial generator $g_1(x) \in F_{q^n}[x]$, as an $F_q[x]$-submodule of $F_{q^n}[x]$, has no monic divisor in $F_q[x]$ other than 1.

Proof: From Definition 1 part 1), we know that the $1 \times n$ matrix $G(x) = [g_1^{(j)}(x)]_{0 \leq j \leq n-1}$ is a basic PGM if and only if $\gcd(g_1^{(0)}(x), g_1^{(1)}(x), \ldots, g_1^{(n-1)}(x)) = 1$. Let $d(x) = \gcd(g_1^{(0)}(x), g_1^{(1)}(x), \ldots, g_1^{(n-1)}(x))$. Since $d(x) \in F_q[x]$ divides each $g_1^{(j)}(x)$, $0 \leq j \leq n-1$, it follows that $d(x)$ divides $g_1(x) = g_1^{(0)}(x) + \alpha g_1^{(1)}(x) + \cdots + \alpha^{n-1} g_1^{(n-1)}(x) \in F_{q^n}[x]$. If the largest monic divisor of $g_1(x)$ in $F_q[x]$ is 1 then $d(x) = 1$ and $G(x)$ is a basic PGM.

Now for the converse, suppose $e(x) \in F_q[x]$ and $e(x) \mid g_1(x)$. Then $g_1(x) = e(x)f(x)$ for some $f(x) \in F_{q^n}[x]$. Writing

$$f(x) = f^{(0)}(x) + \alpha f^{(1)}(x) + \cdots + \alpha^{n-1} f^{(n-1)}(x)$$

with $f^{(j)}(x) \in F_q[x]$, $0 \leq j \leq n-1$, we have

$$\begin{aligned} g_1(x) &= e(x)\left(f^{(0)}(x) + \alpha f^{(1)}(x) + \cdots + \alpha^{n-1} f^{(n-1)}(x)\right) \\ &= e(x)f^{(0)}(x) + \alpha\left(e(x)f^{(1)}(x)\right) + \cdots \\ &\quad + \alpha^{n-1}\left(e(x)f^{(n-1)}(x)\right) \end{aligned}$$

which implies that $g_1^{(j)}(x) = e(x)f^{(j)}(x)$, $0 \leq j \leq n-1$. It follows that $e(x) \in F_q[x]$ divides $g_1^{(j)}(x)$, $0 \leq j \leq n-1$, and therefore $e(x)$ divides $d(x)$. If $d(x) = 1$ then we must have $e(x) = 1$ also. □

Henceforth we call such a polynomial in $F_{q^n}[x]$ a *basic polynomial generator* for the single-input convolutional code C. It can be easily seen that the constant coefficient of the polynomial is nonzero in this case. As aforementioned, the convolutional code C can be associated, by the reduction mapping modulo $x^L - 1$, with the quasi-cyclic code C_L of index n and length nL.

Corollary 4: If C is an $(n, 1)$ convolutional code over F_q with basic polynomial generator $g_1(x) \in F_{q^n}[x]$ as an $F_q[x]$-submodule of $F_{q^n}[x]$, then the associated quasi-cyclic code C_L of index n and length nL generated by $g_1(x) \bmod x^L - 1 \in F_{q^n}[x]/I$, for any positive integer L, has dimension L.

Proof: We know from Theorem 3 that the largest monic divisor of $g_1(x) \in F_{q^n}[x]$ in $F_q[x]$ is 1. It follows that the largest monic divisor of both $g_1(x)$ and $x^L - 1$ in $F_q[x]$ is 1 for any positive integer L, and so applying [9, Theorem 2] we see that the dimension of the code C_L is $k' = L - \deg(1) = L$. □

A quasi-cyclic code C_L of index n, length nL and dimension L has rate $1/n$, which is the maximum rate possible for a 1-generator quasi-cyclic code.

We now derive some further connections between C, C_L and an associated cyclic code. As described in [9], $I = \langle x^L - 1 \rangle$ is the annihilator ideal of the $F_q[x]$-module $F_{q^n}[x]/I$ and so every quasi-cyclic code can also be viewed as an $F_q[x]/I$- submodule of $F_{q^n}[x]/I$. As such the code C_L is a subset subcode of the cyclic code $\widetilde{C}_L$ over F_{q^n} generated as an $F_{q^n}[x]/I$-submodule of $F_{q^n}[x]/I$ (an ideal in $F_{q^n}[x]/I$) by the same set of generators. These observations establish a link between a convolutional code C over the field F_q and an associated cyclic code $\widetilde{C}_L$ defined over the extension field F_{q^n} of F_q and leads us to the following simple result.

Lemma 5: If C is an $(n, 1)$ convolutional code over F_q with basic polynomial generator $g_1(x) \in F_{q^n}[x]$ as an $F_q[x]$-submodule of $F_{q^n}[x]$, then

$$d_{\text{free}}(C) \geq d_{\min}(\widetilde{C}_L)$$

where $\widetilde{C}_L$ is the cyclic code over F_{q^n} of length L with generator polynomial $g(x) = \gcd(g_1(x), x^L - 1) \in F_{q^n}[x]/I$, for any positive integer L.

Proof: Applying Theorem 2 and [9, Lemma 1] we have $d_{\mathrm{free}}(C) \geq d_{\min}(C_L) \geq d_{\min}(\widetilde{C}_L)$. The cyclic code $\widetilde{C}_L$ is generated by $g_1'(x) = g_1(x) \bmod x^L - 1 \in F_{q^n}[x]/I$ and, therefore, has generator polynomial $g(x) = \gcd(g_1'(x), x^L - 1) = \gcd(g_1(x), x^L - 1)$. □

Previous work has been done in [7] and [19] associating a subclass of convolutional codes to cyclic codes over the same field F_q and lower bounds on d_{free} obtained when $\gcd(n, q) = 1$. Our results here can be applied to any convolutional code, for any parameters n and q, and as we see later any k.

However it is easily seen that this initial result is useful only when n is small. The weight of a codeword $c'(x) = c_0' + c_1'x + \cdots + c_{L-1}'x^{L-1} \in F_{q^n}[x]/I$ in the cyclic code $\widetilde{C}_L$ is the number of nonzero coefficients in F_{q^n}. In the convolutional code however each nonzero coefficient $c_l \in F_{q^n}$ of a codeword $c(x) = c_0 + c_1x + c_2x^2 + \cdots + c_tx^t \in F_{q^n}[x]$ represents a vector in F_q^n and thus can contribute up to weight n to the F_q-weight of the convolutional codeword. We now develop an improved lower bound on $d_{\mathrm{free}}(C)$ which also arises from our new module representation.

We can see from the shifting nature of our semi-infinite polynomial generator matrix G (given in (5)) that each coefficient $c_l \in F_{q^n}$ of a convolutional codeword is an F_q-linear combination of the subset of the entries $g_{1,0}, g_{1,1}, \ldots, g_{1,m} \in F_{q^n}$ that appear in the l^{th} column of this generator matrix. It follows that the corresponding vector representation $\boldsymbol{c_l} \in F_q^n$ of this coefficient is a codeword in the linear block code of length n generated by $\{\boldsymbol{g}_{1,0}, \boldsymbol{g}_{1,1}, \cdots, \boldsymbol{g}_{1,m}\} \subseteq F_q^n$, where each $\boldsymbol{g}_{1,l}$, $0 \leq l \leq m$, is the vector equivalent of the coefficient $g_{1,l} \in F_{q^n}$.

Theorem 6: If C is an $(n, 1)$ convolutional code over F_q with basic polynomial generator

$$g_1(x) = g_{1,0} + g_{1,1}x + g_{1,2}x^2 + \cdots + g_{1,m}x^m \in F_{q^n}[x]$$

as an $F_q[x]$-submodule of $F_{q^n}[x]$, then

$$d_{\mathrm{free}}(C) \geq d_{\min}(\widetilde{C}_L)d_{\min}(\mathcal{G})$$

where $\widetilde{C}_L$ is the cyclic code of length L over F_{q^n} with generator polynomial $g(x) = \gcd(g_1(x), x^L - 1) \in F_{q^n}[x]/I$ for any positive integer L, and $\mathcal{G}$ is the linear block code of length n over F_q generated by the set $\{\boldsymbol{g}_{1,0}, \boldsymbol{g}_{1,1}, \cdots, \boldsymbol{g}_{1,m}\} \subseteq F_q^n$.

Proof: The proof of Theorem 1 also showed that each nonzero codeword $c(x) \in C$ satisfies $wt_{q^n}(c(x)) \geq wt_{q^n}(y'(x))$ for some nonzero codeword $y'(x) \in F_{q^n}[x]/I$ in C_L. Every $y'(x) \in C_L$ is also in $\widetilde{C}_L$ and so every nonzero $c(x) \in C$ has at least $d_{\min}(\widetilde{C}_L)$ nonzero coefficients as a polynomial in $F_{q^n}[x]$. In turn each such nonzero coefficient $c_l \in F_{q^n}$ of a codeword $c(x) \in C$ when viewed as a vector $\boldsymbol{c_l} \in F_q^n$ is a codeword in the block linear code generated by $\{\boldsymbol{g}_{1,0}, \boldsymbol{g}_{1,1}, \cdots, \boldsymbol{g}_{1,m}\} \subseteq F_q^n$. □

The coefficients $g_{1,l} \in F_{q^n}$, $0 \leq l \leq m$, in the theorem can be read directly from the choice of basic polynomial generator $g_1(x) \in F_{q^n}[x]$, and are not subject to the effects of reduction modulo $x^L - 1$, (as a direct application of Theorem 2 and [9, Theorem 3] would require).

The minimum distance of a cyclic code plays an important part in the lower bound for $d_{\mathrm{free}}(C)$ given earlier. Various bounds for cyclic codes can be further applied here. For example, when $\gcd(L, q) = 1$ the BCH lower bound on minimum distance can be applied to the generator polynomial $g(x)$ of the cyclic code $\widetilde{C}_L$, which can be readily computed from the choice of generator $g_1(x)$ for C.

If t is the smallest positive integer such that L divides $q^{nt} - 1$ then $F_{q^{nt}}$ is the smallest extension field of F_{q^n} which contains all L^{th}-roots of unity. Let β be a primitive L^{th}-root of unity in $F_{q^{nt}}$. The cyclic code $\widetilde{C}_L$ of length L over F_{q^n} has BCH designed minimum distance $\#CR_{q^n}(g(x)) + 1$ where $\#CR_{q^n}(g(x))$ denotes the largest number of consecutive powers of β in $F_{q^{nt}}$ which are roots of the generator polynomial $g(x) \in F_{q^n}[x]$.

Corollary 7: If C is an $(n, 1)$ convolutional code over F_q with basic polynomial generator $g_1(x) = g_{1,0} + g_{1,1}x + g_{1,2}x^2 + \cdots + g_{1,m}x^m \in F_{q^n}[x]$ as an $F_q[x]$-submodule of $F_{q^n}[x]$, then

$$d_{\mathrm{free}}(C) \geq (\#CR_{q^n}(g(x)) + 1)d_{\min}(\mathcal{G})$$

where $g(x) = \gcd(g_1(x), x^L - 1) \in F_{q^n}[x]$ for any positive integer L, and $\mathcal{G}$ is the linear block code of length n over F_q generated by the set $\{\boldsymbol{g}_{1,0}, \boldsymbol{g}_{1,1}, \cdots, \boldsymbol{g}_{1,m}\} \subseteq F_q^n$.

We note that this lower bound is often equal to the actual free distance of a convolutional code, as the following example shows.

Example 8: Let $k = 1$, $n = 3$, and $q = 2$. Let C be a binary $(3, 1)$ convolutional code with basic PGM

$$G(x) = \begin{pmatrix} x & 1 + x + x^2 & 1 + x^2 \end{pmatrix}.$$

The corresponding polynomial generator as an $F_2[x]$-submodule of $F_8[x]$ is

$$\begin{aligned} g_1(x) &= x + \alpha\left(1 + x + x^2\right) + \alpha^2\left(1 + x^2\right) \\ &= \alpha + \alpha^2 + (1 + \alpha)x + \left(\alpha + \alpha^2\right)x^2 \in F_8[x] \end{aligned}$$

where α is a root of $x^3 + x^2 + 1$ and thus a primitive element of F_8, and $\{1, \alpha, \alpha^2\}$ is a F_2-basis for F_8. The smallest value of L such that $g_1(x)$ divides $x^L - 1$ is $L = 9$. It follows that all roots of $g_1(x)$ are 9th-roots of unity. F_{2^6} is the smallest field extension of F_8 which contains a primitive 9th-root of unity $\zeta = \beta^7$, where β is a primitive element in F_{2^6}, (taken as a root of the primitive polynomial $x^6 + x + 1$). Here $\alpha = \beta^9$. The polynomial $g_1(x)$ splits over F_{2^6} and choosing $L = 9$ we have

$$g(x) = \gcd(g_1(x), x^9 - 1) = (x + \zeta^4)(x + \zeta^5)$$

with $\#CR_8(g(x)) + 1 = 3$. The nonzero coefficients of $g_1(x)$ expressed in vector form are $g_{1,0} = g_{1,2} = (0, 1, 1)$, $g_{1,1} = (1, 0, 1)$. The linear block code $\mathcal{G} = \langle (0, 1, 1), (1, 0, 1) \rangle$ has minimum distance 2. It follows that $d_{\mathrm{free}}(C) \geq 6$. The F_2-weight of the generator $g_1(x)$ itself is 6 and so we have $d_{\mathrm{free}}(C) = 6$. A generator matrix for the code C is

$$G = \begin{pmatrix} g_1(x) \\ xg_1(x) \\ \vdots \end{pmatrix} = \begin{pmatrix} 011 & 110 & 011 & & \\ & 011 & 110 & 011 & \\ & & \ddots & \ddots & \ddots \end{pmatrix}.$$

We recall that if an $(n, 1)$ convolutional code C has a basic polynomial generator $g_1(x) = g_{1,0} + g_{1,1}x + g_{1,2}x^2 + \cdots + g_{1,m}x^m \in F_{q^n}[x]$ with memory order m then both $g_{1,0}$ and $g_{1,m}$ are nonzero. It is easily seen from the shift action on the rows of the semi-infinite polynomial generator matrix G that the initial and final coefficient of any nonzero codeword must be an F_q-scalar multiple of the trailing coefficient $g_{1,0}$ and leading coefficients $g_{1,m}$, respectively.

Corollary 9: If C is an $(n,1)$ convolutional code over F_q with basic polynomial generator $g_1(x) = g_{1,0}+g_{1,1}x+g_{1,2}x^2+\cdots+g_{1,m}x^m \in F_{q^n}[x]$ as an $F_q[x]$-submodule of $F_{q^n}[x]$, and memory order m, then

$$d_{\text{free}}(C) \geq (\#CR_{q^n}(g(x))-1)d_{\min}(\mathcal{G}) + d_{\min}(\mathcal{G}_0) + d_{\min}(\mathcal{G}_m)$$

where $g(x) = \gcd(g_1(x), x^L-1) \in F_{q^n}[x]$ for any positive integer L, and $\mathcal{G}_0$, $\mathcal{G}_m$, and $\mathcal{G}$ are the linear block code of length n over F_q generated by the sets $\{\boldsymbol{g}_{1,0}\}, \{\boldsymbol{g}_{1,m}\}$ and $\{\boldsymbol{g}_{1,0}, \boldsymbol{g}_{1,1}, \ldots, \boldsymbol{g}_{1,m}\} \subseteq F_q^n$, respectively.

In the binary case $(q = 2)$ we have $d_{\min}(\mathcal{G}_r) = wt_2(\boldsymbol{g}_{1,r}), r = 0$ or m. When the polynomial generator has memory order $m = 1$ the unit memory $(n,1)$ binary convolutional code satisfies

$$d_{\text{free}}(C) = wt_2(\boldsymbol{g}_{1,0}) + wt_2(\boldsymbol{g}_{1,1}) = wt_2(g_1(x))$$

which, of course, can be read directly from the choice of polynomial generator $g_1(x)$ for the code.

V. Multiple-Input$(k > 1)$ Convolutional Codes

An (n,k) convolutional code C over F_q has a $k \times n$ basic PGM $G(x) = [g_i^{(j)}(x)]_{k\times n}$ over $F_q[x]$ and as an $F_q[x]$-submodule of $F_{q^n}[x]$, is generated by the k polynomials

$$\begin{aligned} g_i(x) &= g_i^{(0)}(x) + \alpha g_i^{(1)}(x) + \cdots + \alpha^{n-1} g_i^{(n-1)}(x) \\ &= g_{i,0} + g_{i,1}x + g_{i,2}x^2 + \cdots + g_{i,m}x^m \in F_{q^n}[x], \end{aligned}$$

$1 \leq i \leq k$. The code is the set

$$C = \left\{ u^{(0)}(x)g_1(x) + \cdots + u^{(k-1)}(x)g_k(x) \mid u^{(i)}(x) \in F_q[x], 0 \leq i \leq k-1 \right\} \subseteq F_{q^n}[x].$$

The matrix $G(x)$ is a basic PGM and so the polynomial generators $g_i(x) \in F_{q^n}[x]$, $1 \leq i \leq k$, also satisfy restrictive properties.

Theorem 10: If C is an (n,k) convolutional code over F_q with basic PGM $G(x) = [g_i^{(j)}(x)]_{k\times n}$ over $F_q[x]$ then each of the generators $g_i(x) \in F_{q^n}[x]$, $1 \leq i \leq k$, as an $F_q[x]$-submodule of $F_{q^n}[x]$, has no monic divisor in $F_q[x]$ other than 1.

Proof: By definition, the PGM $G(x)$ is basic if and only if the gcd of all $k \times k$ minors of $G(x)$ is 1. A $k \times k$ minor of $G(x)$ is the sum of all signed elementary products from a $k \times k$ submatrix of $G(x)$ and every such elementary product contains a term from each of the k rows of $G(x)$. It follows that the gcd of the entries in the i^{th} row, that is, $d_i(x) = \gcd(g_i^{(0)}(x), g_i^{(1)}(x), \ldots, g_i^{(n-1)}(x))$, $1 \leq i \leq k$, divides every $k \times k$ minor. Since $G(x)$ is basic and $d_i(x)$ divides the gcd of all $k \times k$ minors, we must have $d_i(x) = 1, 1 \leq i \leq k$. The remainder of the proof follows as for Theorem 3. □

As before, reducing all codewords module $x^L - 1$ for any positive integer L, our (n,k) convolutional code can be associated with a k-generator quasi-cyclic code C_L of index n and length nL, generated as a submodule of $F_{q^n}[x]/I$ by $g_i'(x) = g_i(x) \bmod x^L - 1 \in F_{q^n}[x]/I$, $1 \leq i \leq k$. The quasi-cyclic code C_L is a subset subcode of the cyclic code $\widetilde{C}_L$ over F_{q^n} with generator polynomial $g(x) = \gcd(g_1(x), g_2(x), \ldots, g_k(x), x^L - 1) \in F_{q^n}[x]/I$. A lower bound similar to that given in Theorem 6 can be derived in this multi-input case as follows.

Theorem 11: If C is an (n,k) convolutional code over F_q with basic PGM $G(x) = [g_i^{(j)}(x)]_{k\times n}$ over $F_q[x]$ and generators $g_i(x) = g_{i,0} + g_{i,1}x + g_{i,2}x^2 + \cdots + g_{i,m}x^m \in F_{q^n}[x]$, $1 \leq i \leq k$, as an $F_q[x]$-submodule of $F_{q^n}[x]$, then

$$d_{\text{free}}(C) \geq d_{\min}(\widetilde{C}_L)d_{\min}(\mathcal{G})$$

where $\widetilde{C}_L$ is cyclic code of length L over F_{q^n} with generator polynomial $g(x) = \gcd(g_1(x), g_2(x), \ldots, g_k(x), x^L-1) \in F_{q^n}[x]/I$ for any positive integer L, and $\mathcal{G}$ is the linear block code of length n over F_q generated by the set $\{\boldsymbol{g}_{i,l}, i = 1,2,\ldots,k, l = 0,1,\ldots,m\} \subseteq F_q^n$.

We note that all polynomials $g_i(x)$ and coefficients $g_{i,l}$ required can be found straightforwardly from the polynomial entries in a basic PGM for the convolutional code C.

Example 12: Let $k = 2, n = 4$, and $q = 2$. Let C be a binary $(4,2)$ convolutional code with basic PGM

$$G(x) = \begin{pmatrix} x & 1+x^2 & 0 & 1+x+x^2 \\ x & 1 & 1 & x \end{pmatrix}.$$

Then the corresponding polynomial generators as an $F_2[x]$-submodule of $F_{16}[x]$ are $g_1(x) = x + \alpha\left(1+x^2\right) + \alpha^3\left(1+x+x^2\right) = \alpha + \alpha^3 + \left(1+\alpha^3\right)x + \left(\alpha+\alpha^3\right)x^2$ and $g_2(x) = x + \alpha(1) + \alpha^2(1) + \alpha^3(x) = \alpha + \alpha^2 + \left(1+\alpha^3\right)x$ in $F_{16}[x]$, where α is a root of x^4+x+1 and, thus, a primitive element of F_{16}, and $\left\{1,\alpha,\alpha^2,\alpha^3\right\}$ is a basis for F_{16} over F_2. The $\gcd(g_1(x), g_2(x)) = (x+\alpha^6)$ splits in $F_{16}[x]$, and so choosing $L = 15$ we have

$$g(x) = \gcd(g_1(x), g_2(x), x^{15}-1) = (x+\alpha^6)$$

with $\#CR_{16}(g(x)) + 1 = 2$. The nonzero coefficients of the generators expressed in vector form are $g_{1,0} = g_{1,2} = (0,1,0,1), g_{1,1} = (1,0,0,1), g_{2,0} = (0,1,1,0)$, $g_{2,1} = (1,0,0,1)$. The linear block code

$$\mathcal{G} = \langle(0,1,0,1),(1,0,0,1),(0,1,1,0)\rangle$$

has minimum distance 2. It follows that $d_{\text{free}}(C) \geq 4$. Since the F_2-weight of the generator $g_2(x)$ is 4 we have $d_{\text{free}}(C) = 4$. A generator matrix for the code C is

$$G = \begin{pmatrix} g_1(x) \\ g_2(x) \\ xg_1(x) \\ xg_2(x) \\ \vdots \end{pmatrix} = \begin{pmatrix} 0101 & 1001 & 0101 & & \\ 0110 & 1001 & 0000 & & \\ & 0101 & 1001 & 0101 & \\ & 0110 & 1001 & 0000 & \\ & & \ddots & \ddots & \ddots \end{pmatrix}.$$

VI. Choosing L to Maximize Our Lower Bound

For a given (n,k) convolutional code C with basic PGM, and polynomial generators $g_1(x), g_2(x), \ldots, g_k(x) \in F_{q^n}[x], k \geq 1$, with memory order m, varying the value of L can change the lower bound we obtain in Theorem 11. The choice of L determines the length of the cyclic code $\widetilde{C}_L$ generated by $f(x) = \gcd(g_1(x), g_2(x), \ldots, g_k(x)) \in F_{q^n}[x]$, a polynomial of degree at most m, and thus of course influences the minimum distance of this code. Since n (the length of the linear code $\mathcal{G}$) is in practice usually a lot smaller than m, $d_{\min}(\widetilde{C}_L)$ is usually the dominant factor in our lower bound. In this section we apply the BCH lower bound to the generator polynomial $g(x) = \gcd(f(x), x^L - 1) \in F_{q^n}[x]$ of the cyclic code $\widetilde{C}_L$ and examine how to choose L satisfying $\gcd(L, q) = 1$ to maximize $\#CR_{q^n}\left(\gcd(f(x), x^L - 1)\right)$ and hence maximize the lower bound on $d_{\mathrm{free}}(C)$ we achieve.

If $\gcd(L, q) = 1$ then only distinct roots of $f(x) \in F_{q^n}[x]$ can be preserved in $\gcd(f(x), x^L - 1) \in F_{q^n}[x]$. As a first consideration we now aim to choose L large enough to ensure that all distinct roots of $f(x)$ are roots of $\gcd(f(x), x^L - 1)$. Ignoring the trivial case we assume that $\deg(f(x)) \geq 1$. From Theorem 3 and 10 we see that $f(0) \neq 0$ and so all roots of $f(x)$ are nonzero.

The order of $f(x)$ over F_{q^n} is the smallest positive integer $e \leq (q^n)^m - 1$ such that $f(x)$ divides $x^e - 1$ and can be easily found by direct checks. If $q = p^s$ with p prime then $f(x)$ has no multiple roots in any extension field of F_{q^n} if and only if $\gcd(e, p) = 1$. Let $e = p^r e'$ with $\gcd(e', p) = 1$ and $r \geq 0$. Then the splitting field of $f(x) \in F_{q^n}[x]$ over F_{q^n} is $F_{q^{nt}}$ where t is the smallest positive integer such that $e' \mid ((q^n)^t - 1)$ or equivalently $(x^{e'} - 1) \mid (x^{q^{nt}-1} - 1)$. It follows that $\gcd(f(x), x^{e'} - 1)$ is the product of all distinct irreducible factors of $f(x)$ over F_{q^n} and its roots comprise all the distinct roots of $f(x)$, each an e'^{th}-root of unity in the field $F_{q^{nt}}$. Furthermore, e' is the smallest positive integer for which this is true. Among these roots we now count the largest number of consecutive powers of a primitive e'^{th}-root of unity in $F_{q^{nt}}$.

We note that choosing $L > e'$ will not increase and usually decreases our count, as consecutive e'^{th}-roots of unity preserved in $\gcd(f(x), x^L - 1)$ are spread out as L^{th}-roots of unity. For this reason we relabel e' as $\ell_{\max}$, the largest candidate for our choice of L.

We now consider smaller values of L which may provide a larger count of consecutive powers among all the distinct roots of $f(x) \in F_{q^n}[x]$. We know that each root of $f(x)$ has order dividing $\ell_{\max}$. For each positive integer ℓ dividing $\ell_{\max}$ we count $\#CR_{q^n}\left(\gcd(f(x), x^{\ell} - 1)\right)$, that is, the largest number of consecutive powers of a primitive ℓ^{th}-root of unity which are roots of $f(x) \in F_{q^n}[x]$.

It is easily shown (from basic finite field theory, see [11]) that

$$J_i = \left\{i, iq^n, i\left(q^n\right)^2, \ldots, i\left(q^n\right)^{s-1}\right\}$$

is the cyclotomic coset of i modulo $\ell_{\max}$ over F_{q^n}, corresponding to an irreducible factor of $\gcd(f(x), x^{\ell_{\max}} - 1) \in F_{q^n}[x]$, if and only if, for any $d > 0$ dividing i

$$J_{\frac{i}{d}} = \left\{\frac{i}{d}, \frac{i}{d}q^n, \frac{i}{d}\left(q^n\right)^2, \ldots, \frac{i}{d}\left(q^n\right)^{s-1}\right\}$$

is the cyclotomic coset of $\frac{i}{d}$ modulo $\ell = \frac{\ell_{\max}}{d}$ over F_{q^n}, corresponding to an irreducible factor $\gcd(f(x), x^{\ell} - 1) \in F_{q^n}[x]$. Each term in J_i corresponds to a power of a primitive $\ell_{\max}^{th}$-root of unity whereas each term in $J_{\frac{i}{d}}$ corresponds to a power of a primitive ℓ^{th}-root of unity. It follows that for each ℓ dividing $\ell_{\max}$ counting $\#CR_{q^n}\left(\gcd(f(x), x^{\ell} - 1)\right)$ can be easily achieved as the largest number of consecutive terms in the union of all such cyclotomic cosets $J_{\frac{i}{d}}$ modulo $\ell = \frac{\ell_{\max}}{d}$ formed by modifying the cosets J_i modulo $\ell_{\max}$ previously known from $\gcd(f(x), x^{\ell_{\max}} - 1)$.

At last, we choose our L as that value of ℓ, a divisor of $\ell_{\max}$, for which this count is maximum. We illustrate this technique in the following example.

Example 13: Let $k = 1$, $n = 4$, and $q = 2$. Let C be a the binary $(4, 1)$ convolutional code with basic PGM

$$G(x) = \left(\, x^3 \quad 1 + x + x^3 \quad 0 \quad 1 + x \,\right).$$

The corresponding polynomial generator as an $F_2[x]$-submodule of $F_{16}[x]$ is

$$\begin{aligned} g_1(x) &= x^3 + \alpha\left(1 + x + x^3\right) + \alpha^3\left(1 + x\right) \\ &= \alpha + \alpha^3 + \left(\alpha + \alpha^3\right)x + (1 + \alpha)\, x^3 \in F_{16}[x] \end{aligned}$$

where α is a root of $x^4 + x + 1$ and thus a primitive element of F_{16}, and $\left\{1, \alpha, \alpha^2, \alpha^3\right\}$ is a F_2-basis for F_{16}. The order of $g_1(x)$ is $e = e' = 15$. It follows that $g_1(x)$ splits over F_{16} and all its roots are 15th-roots of unity. Hence $\ell_{\max} = 15$ and we have

$$\gcd(g_1(x), x^{15} - 1) = (x + \alpha^5)(x + \alpha^6)(x + \alpha^9)$$

with $\#CR_{16}\left(\gcd(g_1(x), x^{15} - 1)\right) = 2$. The corresponding cyclotomic cosets modulo 15 over F_{16} are $\{5\}, \{6\}$ and $\{9\}$. Taking $d = 3$ the cyclotomic cosets preserved modulo-5 are $\{2\}$ and $\{3\}$. Let $\beta = \alpha^3$ be a primitive fifth-root of unity in F_{16}. Hence when $\ell = 5$ we have

$$\gcd(g_1(x), x^5 - 1) = (x + \beta^2)(x + \beta^3)$$

with $\#CR_{16}\left(\gcd(g_1(x), x^5 - 1)\right) = 2$. Taking $d = 5$ the only cyclotomic coset preserved modulo 3 is $\{1\}$. Let $\gamma = \alpha^5$ be a primitive third-root of unity in F_{16}. Hence when $\ell = 3$ we have

$$\gcd(g_1(x), x^3 - 1) = x + \gamma$$

with $\#CR_{16}\left(\gcd(g_1(x), x^3 - 1)\right) = 1$. Our maximum number of consecutive roots is found when $L = 5$ or 15 The nonzero coefficients of $g_1(x)$ expressed in vector form are $g_{1,0} = g_{1,1} = (0, 1, 0, 1), g_{1,3} = (1, 1, 0, 0)$. The linear block code $\mathcal{G} = \langle (0, 1, 0, 1), (1, 1, 0, 0) \rangle$ has minimum distance 2. It follows that $d_{\mathrm{free}}(C) \geq (2 + 1)(2) = 6$. Again it is easily seen that in fact $d_{\mathrm{free}}(C) = 6$.

As with any method of determining the actual free distance of an (n, k) convolutional code C (see discussion [12, p. 538]), obtaining our lower bound on $d_{\mathrm{free}}(C)$ is in general more computationally intensive for larger values of m. For example, using a modified version of the Viterbi algorithm to determine free distance requires q^v state metrics to be stored, where v is the overall constraint length of the encoder given by $v = \sum_{1 \leq i \leq k} \deg\left(g_i(x)\right)$ and $v \geq m$. However the difficulty of applying our lower bound depends more so on the nature of $f(x) = \gcd(g_1(x), g_2(x), \ldots, g_k(x)) \in F_{q^n}[x]$, a polynomial of degree at most m, and in particular on its order (directly linked to the number and degree of its irreducible factors and the order of the roots of these irreducible factors). Computing the order e of $f(x)$ (for example by trial division in $F_{q^n}[x]$) can involve up to $(q^n)^m - 1$ steps, and finding the roots of $\gcd(f(x), x^{e'} - 1)$ in the splitting field $F_{q^{nt}}$ (for example by a simple exhaustive search) is impractical when t is large. However, as the following example shows, if the values of e, e'

and t are found to be small, then determining our lower bound on $d_{\text{free}}(C)$ is efficient for many codes with large m where direct computation of free distance is known to be infeasible, such as when $m > 30$ [12].

Example 14: Let $k = 1, n = 2$, and $q = 2$. Let C be a binary $(2, 1)$ convolutional code with basic polynomial generator, as an $F_2[x]$-submodule of $F_4[x]$, given by

$$\begin{aligned} g_1(x) = {} & x^{34} + x^{33} + x^{32} + x^{31} + (\alpha+1)x^{29} + \alpha x^{27} \\ & + x^{28} + x^{26} + x^{25} + x^{24} + (\alpha+1)x^{23} \\ & + x^{22} + (\alpha+1)x^{21} + x^{20} + \alpha x^{18} + \alpha x^{15} \\ & + \alpha x^{13} + x^{14} + \alpha x^{11} + x^{12} + (\alpha+1)x^{9} + x^{7} \\ & + x^{6} + x^{5} + x^{3} + x^{2} + (\alpha+1)x + 1 \in F_4[x] \end{aligned}$$

where α is a root of $x^2 + x + 1$ and thus a primitive element of F_4, and $\{1, \alpha\}$ is a F_2-basis for F_4. The memory order of the generator matrix is $m = \deg(g_1(x)) = 34$. The order of $g_1(x)$ is $e = e' = 85$. It follows that all roots of $g_1(x)$ are 85th-roots of unity. F_{2^8} is the smallest field extension of F_4 which contains a primitive 85th-root of unity $\zeta = \beta^3$, where β is a primitive element in F_{2^8}, (taken as a root of the primitive polynomial $x^8 + x^4 + x^3 + x^2 + 1$). Here $\alpha = \beta^{85}$. Now $g(x) = \gcd(g_1(x), x^{85} - 1) \in F_4[x]$ has root ζ^i, for each

$$\begin{aligned} i \in \{&1,3,4,9,12,13,14,16,22,30,34,35,36,38,42,46,48, \\ &50,51,52,53,54,55,56,57,58,59, \\ &62, 64, 66, 67,77,78,83\} \end{aligned}$$

and therefore $\#CR_4(g(x)) + 1 = 11$. The nonzero coefficients of $g_1(x)$ expressed in vector form are $(0, 1), (1, 0)$ and $(1, 1)$. The linear block code $\mathcal{G} = \langle(0, 1), (1, 0), (1, 1)\rangle$ has minimum distance 1. It follows that $d_{\text{free}}(C) \geq 11$. The actual free distance may of course be higher.

VII. Constructing Good Convolutional Codes

For given values of n, k and m, our new lower bound suggests a method of algebraically constructing convolutional codes with good designed free distance.

For example when $k = 1$, one way to construct good $(n, 1)$ convolutional codes with basic polynomial generator of memory order m, is to form the cyclotomic cosets modulo L over F_{q^n} for various values of $L \geq m$, satisfying $\gcd(L, q) = 1$, and consider the union of subsets of these cyclotomic cosets of size m, which contain a large number of consecutive terms. The basic property can be easily checked by ensuring this union contains no complete cyclotomic coset modulo L over F_q. The corresponding polynomial has degree m, and as a divisor of $x^L - 1$ can be taken as $g(x) \in F_{q^n}[x]$ in Corollary 7. Good results here depend on the existence of long BCH codes over F_{q^n} with high designed minimum distance. When $m \leq q^n - 1$ we can set $L = q^n - 1$ and consider Reed–Solomon codes of length L over F_{q^n}, choosing our union of size m with m consecutive terms (subject to the basic criteria being met). Finally we can choose a nonzero multiple $e_1 \in F_{q^n}$ so that the coefficients of $g_1(x) = e_1 g(x) \in F_{q^n}[x]$ generates a linear block code $\mathcal{G}$ of length n with high minimum distance.

Since our polynomial generator $g_1(x) \in F_{q^n}[x]$ has no monic divisors in $F_q[x]$ other than 1, its nonzero coefficients cannot all be the same nor can they all be F_q-multiples of a single element of F_{q^n}. It follows that the block code $\mathcal{G}$ cannot have minimum distance n. Hence by Corollary 7, we can ensure a lower bound on d_{free} of at most $(m + 1)(n - 1)$. It follows that, no matter how large we choose our alphabet field F_q, when $k = 1$ our lower bound can never guarantee the existence of maximum distance separable (MDS) codes, that is, those with $d_{\text{free}} = n_A = n(m + 1)$.

TABLE I
Some Binary $(2, 1)$ Convolutional Codes Obtained From $L < 20$

m	d_{free}	L	J_i	d_{free}/n_A
1	≥ 2	3	1	≥ 0.5
2	≥ 3	5	2	≥ 0.5
3	≥ 3	15	$1, 5$	≥ 0.38
4	≥ 3	9	$1, 3$	≥ 0.3
5	≥ 4	11	1	≥ 0.33
6	≥ 5	13	2	≥ 0.36
7	≥ 6	15	$2, 5, 6, 7$	≥ 0.38
8	≥ 7	17	$2, 6$	≥ 0.39
9	≥ 5	19	1	≥ 0.25

TABLE II
Some Binary $(2, 1)$ Convolutional Codes Obtained From $L = 85$

m	d_{free}	J_i	d_{free}/n_A
22	≥ 10	$2, 7, 18, 29, 30, 34$	≥ 0.22
24	≥ 9	$2, 7, 18, 19, 29, 30$	≥ 0.18
26	≥ 11	$2, 7, 18, 19, 29, 30, 34$	≥ 0.2
28	≥ 9	$1, 9, 13, 14, 30, 42, 57$	≥ 0.16
30	≥ 11	$1, 9, 13, 14, 30, 34, 42, 57$	≥ 0.18

When $q = 2$ we can obtain a lower bound on d_{free} of at most $m+1, 2(m+1)$ and $2(m+1)$ for binary $(2, 1), (3, 1)$ and $(4, 1)$ convolutional codes, respectively. Comparing these values to the tables of optimal codes given in [12], (obtained by exhaustive search for small values of m), we see that our lower bound cannot in general ensure optimal codes. However our approach does provide a method of algebraically constructing convolutional codes with additional BCH-type structure for any value of n, and can often guarantee a good designed free distance.

For example taking $n = 2$ and searching odd values of L up to 20, we can construct binary $(2, 1)$ convolutional codes with basic polynomial generator of memory order $m < 10$ and lower bounds on d_{free} given in Table I. The representatives i of the cyclotomic cosets J_i modulo L over F_{2^2} which contribute to the choice of generator are listed in the second last column. Searching higher values of L we can construct codes with larger m. For example when $L = 85$ binary $(2, 1)$ codes can be constructed for even values of m from 10 to 20 with ratios d_{free}/n_A in the range 0.18 to 0.23. Results for even values of m from 22 to 30 are given in Table II. In each case the actual free distance could of course be higher. We also note that when $n = 2$ we always have $d_{\min}(\mathcal{G}) = 1$ and so no computations in $F_{q^n}[x]$ are required to obtain our lower bound.

Lower bounds on the free distance of a best code are given by Costello in [1]. It was shown that there exists at least one time-varying $(2, 1)$ convolutional code with large m and $d_{\text{free}}/n_A \geq 0.39$, and at least one fixed convolutional code with large m and $d_{\text{free}}/n_A \geq 0.22$. The ratios obtained in Table I are comparable to the former bound (showing the existence of some fixed binary $(2, 1)$ codes which meet this bound for time-varying codes), whereas codes given in Table II are comparable to the latter bound. Searching larger values of L may enable the construction of codes with higher m and improved values of our lower bound on d_{free}.

VIII. Conclusion

Representing a convolutional code over F_q as an $F_q[x]$-submodule of $F_{q^n}[x]$ allows us to develop new links to quasi-cyclic and cyclic block codes. In this setting good algebraic lower bounds on free distance can be established. BCH-type results for convolutional codes are also derived. Such lower bounds suggest a method of constructing good convolutional codes with high designed free distance.

Acknowledgment

The author would like to thank the anonymous reviewers for their valuable comments and suggestions that helped to improve the paper.

References

[1] D. J. Costello Jr., "Free distance bounds for convolutional codes," *IEEE Trans. Inf. Theory*, vol. IT-20, pp. 356–365, 1974.

[2] G. D. Forney Jr., "Convolutional codes I: Algebraic structure," *IEEE Trans. Inf. Theory*, vol. IT-16, pp. 268–278, 1970.

[3] H. Gluesing-Luerssen and W. Schmale, "On cyclic convolutional codes," *Acta Appl. Math.*, vol. 82, pp. 183–237, 2004.

[4] H. Gluesing-Luerssen, J. Rosenthal, and R. Smarandache, "Strongly-MDS convolutional codes," *IEEE Trans. Inf. Theory*, vol. 52, pp. 584–598, 2006.

[5] T. A. Gulliver and V. K. Bhargava, "Some best rate $1/p$ and rate $(1-p)/p$ systematic quasi-cyclic codes," *IEEE Trans. Inf. Theory*, vol. 37, pp. 552–555, 1991.

[6] R. Johannesson and K. S. Zigangirov, *Fundamentals of Convolutional Coding*. Piscataway, NJ: IEEE, 1999.

[7] J. Justesen, "New convolutional code constructions and a class of asymptotically good time-varying codes," *IEEE Trans. Inf. Theory*, vol. IT-19, pp. 220–225, 1973.

[8] K. Lally and P. Fitzpatrick, "Algebraic structure of quasicyclic codes," *Discrete Appl. Math.*, vol. 111, no. 1–2, pp. 157–175, 2001.

[9] K. Lally, "Quasicyclic codes of index ℓ over F_q viewed as $F_q[x]$-submodules of $F_{q^\ell}[x]/\langle x^m-1\rangle$," in *Proc. AAECC-15*, vol. 2643, Lect. Notes in Comp. Sci, 2003, pp. 244–253.

[10] Y. Levy and D. J. Costello Jr, "An algebraic approach to constructing convolutional codes from quasicyclic codes," *DIMACS Ser. Discr. Math. Theor. Comput. Sci.*, vol. 14, pp. 189–198, 1993.

[11] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and Their Applications*. Cambridge, U.K.: Cambridge University Press, 1986.

[12] S. Lin and D. J. Costello Jr, *Error Control Coding: Fundamentals and Applications*, 2nd ed. Upper Saddle River, NJ: Pearson Prentice-Hall, 2004.

[13] J. L. Massey, D. J. Costello Jr, and J. Justesen, "Polynomial weights and code constructions," *IEEE Trans. Inf. Theory*, vol. IT-19, pp. 101–110, 1973.

[14] R. J. McEliece, "The algebraic theory of convolutional codes," in *Handbook of Coding Theory*, V. Pless and W. Huffman, Eds. Amsterdam, The Netherlands: Elsevier Science, 1998, vol. 1, pp. 1065–1138.

[15] J. Rosenthal and E. V. York, "BCH convolutional codes," *IEEE Trans. Inf. Theory*, vol. 45, pp. 1833–1844, 1999.

[16] G. E. Séguin and G. Drolet, "The Theory of 1-Generator Quasi-Cyclic Codes" (in preprint), in Dept. Elect. Comput. Eng., Royal Military College of Canada, Kingston, Canada, 1990.

[17] G. Séguin, "On a class of convolutional codes," *IEEE Trans. Inf. Theory*, vol. IT-29, pp. 215–223, 1983.

[18] ——, "A class of 1-generator quasi-cyclic codes," *IEEE Trans. Inf. Theory*, vol. 50, pp. 1745–1753, 2004.

[19] R. Smarandache, H. Gluesing-Luerssen, and J. Rosenthal, "Constructions of MDS-convolutional codes," *IEEE Trans. Inf. Theory*, vol. 47, pp. 2045–2049, 2001.

[20] G. Solomon and H. C. A. van Tilborg, "A connection between block and convolutional codes," *SIAM J. Appl. Math.*, vol. 37, pp. 358–369, 1979.

[21] R. M. Tanner, "Convolutional Codes From Quasicyclic Codes: A Link Between the Theories of Block and Convolutional Codes," Univ. Calif., Santa Cruz, CA, UCSC-CRL-87-21, 1987.