This article presents the results of a study of a class of error-correcting codes called partial-unit-memory convolutional codes, or PUM codes for short. This class of codes, though not entirely new, has until now remained relatively unexplored. This article shows that it is possible to use the well-developed theory of block codes to construct a large family of promising PUM codes. Indeed, at the end of the article the performances of several specific PUM codes are compared with that of the Voyager standard (2, 1, 6) convolutional code. It was found that these codes can outperform the Voyager code with little or no increase in decoder complexity. This suggests that there may very well be PUM codes that can be used for deep-space telemetry that offer both increased performance and decreased implementational complexity over current coding systems.

I. Introduction

This article gives a general construction for, and several interesting examples of, partial-unit-memory (PUM) convolutional codes. First, some definitions and notation are established.

A convolutional code $\mathcal{C}$ of length $n$ and dimension $k$ over a field $F$ is defined by an encoder

$$G(D) = G_0 + G_1 D + \cdots + G_M D^M$$  \hspace{1cm} (1)

where $G_0, G_1, \ldots, G_M$ are $k \times n$ matrices with entries from $F$. The ratio $R = k/n$ is called the rate of the code, and $M$ is the code's memory. If $u(D) = u_0 + u_1 D + u_2 D^2 + \cdots$ is the input to the encoder (where the $u_i$'s are elements of $F$, and $D$ is an indeterminate), then $x(D) = u(D)G(D)$ is the output, which is also called a codeword. The encoder is said to be noncatastrophic if no infinite-weight input produces a finite-weight output. The free distance, $d_{\text{free}}$, of a convolutional code $\mathcal{C}$ is defined to be the minimum weight of any nonzero codeword $x(D) \in \mathcal{C}$. If the encoder $G(D)$ is noncatastrophic, then $d_{\text{free}}$ is the minimum weight of all codewords $u(D)G(D)$ generated by inputs $u(D)$ of finite weight. All other things being equal, it is generally desirable to have the quantity

$\text{In this article, it is always assumed that } F = GF(2), \text{ but most or all of the results generalize easily to other finite fields.}$
$Q = Rd_{\text{free}}$ as large as possible, since $Q$ is the asymptotic coding gain of the code, which is a good measure of the communications improvement afforded by the use of the code.

A convolutional code $C$ has state complexity $m$ if the sum of the maximum degrees of the rows of $G(D)$ is $m$. This terminology reflects the fact that a physical encoder for $C$ based on $G(D)$ has $2^m$ states. It is desirable to have $m$ as small as possible, since the computational complexity of the Viterbi decoding algorithm for $C$ is proportional to $2^m$.

The notation "$[n,k,m,d]_C$ code" is introduced to describe a convolutional code of length $n$, dimension $k$, state complexity $m$, and free distance $d$. The notation "$[n,k,m]_G$" code is sometimes used to describe the same code without explicitly referring to its free distance. For example, in this notation, an $[n,k,d]$ block code, which can be viewed as a convolutional code with $M = 0$, is both an $[n,k,0,d]$ and an $(n,k,0)$ convolutional code. For a given $n$, $k$, and $m$, a code for which $d_{\text{free}}$ is as large as possible is said to be an optimal (more properly, distance optimal) code.

A convolutional code with $M = 0$ is just a block code. A convolutional code with $M = 1$ is called a unit-memory convolutional code. Unit-memory codes seem to form a class that lies halfway between block and convolutional codes. They were first studied seriously by Lee [5], who found a number of interesting examples of unit-memory convolutional codes. Thommesen and Justesen [10] have obtained bounds on the performance of unit-memory codes, and Justesen, Paaske, and Ballan [3] have constructed a class of unit-memory codes which they call quasi-cyclic codes. This article studies another subclass of unit-memory codes called partial-unit-memory codes.

For a unit-memory convolutional code, the state complexity is just the number of nonzero rows of $G_1$. If some of the rows of $G_1$ are zero, i.e., if $m < k$, then it is said that the code is a partial-unit-memory (PUM) convolutional code. Partial-unit-memory codes were introduced by Lauer [4], who constructed several optimal PUM codes. Some general constructions and further examples of PUM codes (under the name finite-state codes) were given in [7] and [8]. This article should be viewed as a continuation of these earlier studies, in which, among other things, it is shown that many of these earlier results follow from the authors’ methods. (For example, Lauer’s equidistant PUM codes appear in Example 2 in Section III of this article, and the general construction of Theorem 5 in [7] appears as Corollary 4 in Section II.)

Here is a summary of this article. Section II gives a general construction for PUM codes based on the existence of certain block codes. Roughly speaking, the main result is that if there exist two distinct $[n,k,d_0]$ block codes with a common $[n,k^*,d^*]$ subcode, then there exists a noncatastrophic $[n,k,k^*-d^*,d]$ PUM code with $d \geq \min(d^*, 2d_0)$. (As a point of comparison, Theorem 5 in [7] shows that if there exists a single $[n,k,d_0]$ block code with an $[n,k^*,d^*]$ subcode, then there exists a noncatastrophic $[n,k-1,k^*-1,d]$ PUM code with $d \geq \min(d^*, 2d_0)$.) Since there is a huge existing catalog of block codes, what this means is that it is possible to construct a very large number of interesting PUM codes. This is illustrated in Section III with several examples called Hamming, Reed-Muller, and Golay PUM codes. While these examples are possibly interesting and potentially important, the authors believe that they have only scratched the surface, and hope that future authors, using these techniques, or ones of their own devising, will unearth many more examples.

II. Main Results

Theorem 1. Suppose that $C_0$ is an $[n,k,d_0]$ linear block code, $C_1$ is an $[n,k,d_1]$ linear block code, and $C_0 \neq C_1$. Suppose further that $C_0$ and $C_1$ contain a common subcode $C^*$, which is an $[n,k^*,d^*]$ code. Then there exists a noncatastrophic $[n,k,m,d]$ PUM convolutional code, with $m = k - k^*$ and $d \geq \min(d^*, d_0 + d_1)$.

Proof: Begin by choosing $k \times n$ generator matrices $G_0$ and $G_1$ for $C_0$ and $C_1$ of the form

\[
G_0 = \begin{bmatrix} K^* \\ K_0 \end{bmatrix} \quad G_1 = \begin{bmatrix} K^* \\ K_1 \end{bmatrix}
\]

(2)

where $K^*$ is a $k^* \times n$ generator matrix for $C^*$. Note that both $K_0$ and $K_1$ are $m \times n$ matrices; for future reference, let $C_0^*$ and $C_1^*$ be the corresponding codes, i.e.,

\[
C_0^* = \langle K_0 \rangle \\
C_1^* = \langle K_1 \rangle
\]

(3)

(4)

Next, define the matrix $G_0^*$ as

\[
G_0^* = \begin{bmatrix} 0 \\ K_1 \end{bmatrix}
\]

(5)
where $O$ is a $k^* \times n$ matrix of 0's. Then define a $k \times n$ polynomial matrix $G(D)$ as follows:

$$G(D) = G_0 + G_1 D$$  \(6\)

Plainly, $G(D)$ is the generator matrix for an $(n, k, m) \text{PUM} \text{code}$. The proof will be complete when the following two things are shown: (1) $d_{\text{free}} > \min(d^*, d_0 + d_1)$, and (2) $G(D)$ is noncatastrophic. Begin with the assertion about $d_{\text{free}}$.

Assume that $u(D) = u_0 + u_1 D + u_2 D^2 + \ldots$ is a finite nonzero input sequence, where $u_0 \neq 0$ and each $u_i$ is a $k$-dimensional row vector. Then the corresponding output sequence is $z(D) = u(D)G(D) = z_0 + z_1 D + z_2 D^2 + \ldots$, where

$$
\begin{align*}
    z_0 &= u_0 G_0 \\
    z_i &= u_i G_0 + u_{i-1} G_1, & \text{for } i \geq 1
\end{align*}
$$  \(7\)

If $u = (\mu_1, \mu_2, \ldots, \mu_k)$ is a $k$-dimensional vector, $u^L$ (the left part of $u$) and $u^R$ (the right part of $u$) are defined as follows:

$$
\begin{align*}
    u^L &= (\mu_1, \ldots, \mu_{k^*}) \\
    u^R &= (\mu_{k^*+1}, \ldots, \mu_k)
\end{align*}
$$  \(8\)

Then [see Eq. (2)], for any vector $u$, one has

$$u G_0 = u^L K^* + u^R K_0, \quad u G_1 = u^R K_1$$  \(9\)

After combining Eq. (8) with Eq. (6), one has

$$
\begin{align*}
    z_0 &= u_0^L K^* + u_0^R K_0 \\
    z_i &= u_i^R K_0 + [u_i^L, u_{i-1}^R] G_1, & \text{for } i \geq 1
\end{align*}
$$  \(10\)

Now, either $[u_i^L, u_{i-1}^R] = 0$ for all $i \geq 1$, or not. It will be shown that weight $(z(D)) \geq d^*$ in the first case, and that weight $(z(D)) \geq d_0 + d_1$ in the second case.

If $[u_i^L, u_{i-1}^R] = 0$ for all $i \geq 1$, then by Eq. (6) and Eq. (9), one has

$$z(D) = u_0 G_0 = u_0^L K^*$$  \(11\)

which means that $z(D)$ is a nonzero vector in the rowspace of $K^*$, i.e., a nonzero codeword in $C^*$, and so weight $(z(D)) \geq d^*$ in this case.

If, on the other hand, $[u_i^L, u_{i-1}^R] \neq 0$ for some $i \geq 1$, let $M$ denote the largest such index. Then, $u_M = 0$, and Eq. (9) implies

$$z(D) = u_0 G_0 + \cdots + [u_M^L, u_{M-1}^R] G_1 D^M$$  \(12\)

But $u_0 G_0$ is a nonzero word from $C_0$, and so has weight $\geq d_0$, and $[u_M^L, u_{M-1}^R] G_1$ is a nonzero word from $C_1$, and so has weight $\geq d_1$; thus, weight $(z(D)) \geq d_0 + d_1$ in this case, which proves the assertion about $d_{\text{free}}$.

It remains to be shown that $G(D)$ can be chosen noncatastrophically. Lemma 1, which follows, tells, in principle, whether a given $G(D)$ is catastrophic or not. Lemma 2 then tells that it is always possible to choose the matrices $G_0$ and $G_1$ so that $G(D)$ is noncatastrophic.

**Lemma 1.** Let the linear transformation $T : C_0 \to C_1$ be defined by $u G_0 \to u G_1$. Then $G(D)$ is noncatastrophic if and only if every subspace of $C_0$ fixed by $T$ is a subspace of $C^*$.

**Proof:** Denote the rows of $K^*$ by $(x_1, x_1, \ldots, x_{k^*})$, the rows of $K_0$ by $(y_1, y_1, \ldots, y_m)$, and the rows of $K_1$ by $(z_1, z_2, \ldots, z_m)$. Then $T$ is completely characterized by the $k$ values

$$
\begin{align*}
    Tx_i &= x_i, & \text{for } i = 1, 2, \ldots, k^* \\
    Ty_i &= z_i, & \text{for } i = 1, 2, \ldots, m
\end{align*}
$$  \(13\) \(14\)

Note that Eq. (13) says that $T$ not only fixes $C^*$, it fixes $C^*$ pointwise.

It is first assumed that every $T$-fixed subspace of $C_0$ is a subspace of $C^*$, and then shown that $G(D)$ is noncatastrophic. Let $u(D)$ be a nonzero input such that the corresponding output $z(D)$ is finite, i.e., $z_i = 0$ for $i > i_0$.

If one defines

$$
\begin{align*}
    a_i &= u_i^L K^* \in C^* \\
    b_i &= u_i^R K_0 \in C_0^*
\end{align*}
$$  \(15\) \(16\)

one has, by Eq. (9),
\[ x_i = a_i + b_i + T b_{i-1}, \quad \text{for } i \geq 1 \quad (17) \]

Thus, since it is assumed that \( x_i = 0 \) for \( i > i_0 \), it follows that (recall that the codes are binary)

\[ T b_{i-1} = b_i + a_i, \quad \text{for } i > i_0 \quad (18) \]

so that \( (b_{i_0}, b_{i_0+1}, \ldots) + C^* \) is a \( T \)-fixed subspace of \( C_0 \). However, it is assumed that all \( T \)-fixed subspaces of \( C_0 \) are subspaces of \( C^* \), and so \( b_i \in C^* \) for all \( i \geq i_0 \). But then, by Eq. (17), \( a_i = 0 \), and so \( u_i^R = 0 \), for all \( i > i_0 \). Thus, the input \( u(D) \) is necessarily finite, and so \( G(D) \) is noncatastrophic.

Conversely, suppose that \( B \) is a nonzero \( T \)-fixed subspace of \( C_0 \) that properly contains \( C^* \). Choose \( a_0 \in C^* \) and \( b_0 \in B - C_0^* \) arbitrarily. Now, since \( B \) is \( T \)-fixed, \( T b_0 \) is also in \( B \), and so it can be decomposed uniquely into the sum of an element of \( C^* \), which is called \( a_1 \), and an element of \( C_0^* \), which is called \( b_1 \). Note that since \( b_0 \neq 0 \), then \( b_1 \neq 0 \) also, for otherwise \( T \) would map the \( k^* + 1 \) dimensional space \( C^* + (b_0) \) into the \( k^* \)-dimensional space \( C^* \). This process is continued inductively by constructing an infinite sequence of pairs \( (a_i, b_i) \), with \( a_i \in C^* \), \( b_i \in C_0^* \), \( b_i \neq 0 \), such that

\[ T b_i = a_{i+1} + b_{i+1}, \quad \text{for } i \geq 0 \quad (19) \]

Now for each \( i \geq 1 \), define the vector \( u_i \) as follows:

\[ u_i^L K^* = a_i \quad (20) \]

\[ u_i^R K_0 = b_i \quad (21) \]

Since \( b_i \neq 0 \), then by Eq. (20), \( u_i \neq 0 \), and so the sequence \( (u_i) \) is an infinite sequence of nonzero elements. It will now be shown that, if \( (u_i) \) is the input, then the corresponding output is finite. One has

\[ T b_i = T(u_i^R K_0) \]

\[ = T[0, u_i^R] K_0 \]

\[ = (0, u_i^R) K_1, \quad \text{by definition of } T \]

\[ = u_i^R K_1 \]

and so by Eq. (9), for \( i \geq 1, \)

\[ x_i = u_i^R K^* + u_i^R K_0 + u_{i-1}^R K_1, \quad \text{by Eq. (9)} \]

\[ = a_i + b_i + T b_{i-1}, \quad \text{by Eqs. (19) and (20)} \]

\[ = 0, \quad \text{by Eq. (18)} \]

Thus, the infinite input sequence \( (u_i) \) produces a finite output sequence \( (x_i) \), and so \( G(D) \) is catastrophic, as was asserted.

Corollary 1. If \( C_0 \cap C_1 = C^* \), then any generator matrix of the form Eq. (6) is noncatastrophic.

Proof: If \( B \) is a \( T \)-fixed subspace of \( T \), then since \( T : C_0 \rightarrow C_1 \) and \( B = T(B) \), \( B \) is a subspace of both \( C_0 \) and \( C_1 \). Thus, \( B \subseteq C_0 \cap C_1 = C^* \), and so by Lemma 1, \( G(D) \) is noncatastrophic.

Lemma 1 allows one to tell, in principle, whether or not a given \( G(D) \) is catastrophic. Corollary 1 assures that if \( C_0 \cap C_1 = C^* \), then nothing can go wrong. However, if \( C_0 \cap C_1 \supset C^* \), more work is necessary to find a noncatastrophic generator matrix. Lemma 2, which follows, gives an explicit construction for a noncatastrophic \( G(D) \) in the general case.

Lemma 2. Suppose that \( C_0 \) and \( C_1 \) are subspaces of \( V_n(F) \), the \( n \)-dimensional vector space over \( F \), with \( C_0 \neq C_1 \) but \( \dim(C_0) = \dim(C_1) = k \), and that \( C^* \) is a subspace of both \( C_0 \) and \( C_1 \), with \( \dim(C^*) = k^* \). Then, if \( (u_1, u_2, \ldots, u_{k^*}) \) is a basis for \( C^* \), there exist bases for \( C_0 \) and \( C_1 \) of the form

\[ \langle C_0 \rangle = \langle u_1, \ldots, u_{k^*}, \alpha_1, \ldots, \alpha_m \rangle \]

\[ \langle C_1 \rangle = \langle u_1, \ldots, u_{k^*}, \beta_1, \ldots, \beta_m \rangle \quad (22) \]

such that the linear transformation \( T : C_0 \rightarrow C_1 \) defined by

\[ T u_i = u_i, \quad \text{for } i = 1, \ldots, k^* \]

\[ T \alpha_i = \beta_i, \quad \text{for } i = 1, \ldots, m \quad (23) \]

fixes no subspace of \( C_0 \) larger than \( C^* \).

Proof: Begin by constructing two descending sequences of subspaces \( (A_i) \) and \( (B_i) \):
\[ C_0 = A_0 \cup A_1 \cup \cdots \cup A_{N+1} = C^* \]
\[ C_1 = B_0 \cup B_1 \cup \cdots \cup B_{N+1} = C^* \]

such that

\[ \dim(A_i) = \dim(B_i), \quad i = 0, 1, \ldots, N+1 \]
\[ A_{i+1} = A_i \cap B_i, \quad i = 0, 1, \ldots, N \]

Figure 1 illustrates the construction of Lemma 2. This construction can be done inductively as follows. Assume that \( A_0, A_1, \ldots, A_i \) and \( B_0, B_1, \ldots, B_i \) have already been constructed. (For \( i = 0 \), this simply requires setting \( A_0 = C_0 \) and \( B_0 = C_1 \).) Let \( A_{i+1} = A_i \cap B_i \). If \( A_{i+1} = C^* \), define \( B_{i+1} = C^* \), \( N = i \), and stop. Otherwise, one has \( C^* \subseteq A_{i+1} \subseteq B_i \), and so by Lemma A2 in Appendix A, there exists a subspace \( B_{i+1} \neq A_{i+1} \) such that \( C^* \subseteq B_{i+1} \subseteq B_i \), with \( \dim B_{i+1} = \dim A_{i+1} \).

Now define integers \( k_0, \ldots, k_N \) by \( k_i = \dim(A_i) - k^* \), so that

\[ k_N < k_{N-1} < \cdots < k_0 = m \]

Next, using Lemma A1 in Appendix A, choose bases \( \{u_1, \ldots, u_{k^*}, \alpha_1, \ldots, \alpha_m\} \) for \( C_0 \) and \( \{u_1, \ldots, u_{k^*}, \beta_1, \ldots, \beta_m\} \) for \( C_1 \) such that

\begin{align*}
\{u_1, \ldots, u_{k^*}, \alpha_1, \ldots, \alpha_m\} &= A_i \\
\{u_1, \ldots, u_{k^*}, \beta_1, \ldots, \beta_m\} &= B_i
\end{align*} \quad (24)

for \( i = 1, 2, \ldots, N \). Now define the transformation \( T \) as in Eq. (22). Plainly, \( T \) fixes \( C^* \) pointwise, and also, from Eq. (23),

\[ T : A_i \rightarrow B_i, \quad \text{for } i = 0, 1, \ldots, N \] \quad (25)

If now \( D \) is a subspace of \( C_0 \) fixed by \( T \), then \( D \subseteq C_0 = A_0 \), and \( D = T(D) \subseteq T(C_0) = C_1 = B_0 \), so that \( D \subseteq A_0 \cap B_0 = A_1 \). Also, \( D = T(D) \subseteq T(A_1) = B_1 \), using Eq. (24), so that \( D \subseteq A_1 \cap B_1 = A_2 \). Continuing inductively, one finds that in fact \( D \subseteq A_3, \ldots, D \subseteq A_{N+1} = C^* \). Thus, a linear transformation \( T \) is constructed such that any \( T \)-fixed subspace of \( C_0 \) is a subspace of \( C^* \).

Lemma 2 tells how to construct noncatastrophic generator matrices \( G_0 \) and \( G_1 \): Just let the rows of \( G_0 \) be the vectors \( (u_1, \ldots, u_{k^*}, \alpha_1, \ldots, \alpha_m) \), and let the rows of \( G_1 \) be the vectors \( (u_1, \ldots, u_{k^*}, \beta_1, \ldots, \beta_m) \) in Eq. (21). Then, the mapping \( T : C_0 \rightarrow C_1 \) defined by \( uG_0 \rightarrow uG_1 \) is the same as the mapping described in Lemma 2, and so the resulting \( G(D) \) is noncatastrophic. This completes the proof of Theorem 1.

**Corollary 2.** Suppose that \( C_0 \) is an \([n, k, d_0]\) linear block code, and that \( C^* \) is an \([n, k^*, d^*]\) code, which is a subcode of \( C_0 \). If the automorphism group of \( C^* \) contains a permutation that does not fix \( C_0 \), then there exists an \([n, k, m, d]\) PUM convolutional code, with \( m = k - k^* \) and \( d \geq \min(d^*, 2d_0) \).

**Proof:** Let \( \pi \) be an automorphism of \( C^* \) that does not fix \( C_0 \), and let \( C_1 = C_{0}^{\pi} \). Then, \( C_1 \) is an \([n, k, d_0]\) code not equal to \( C_0 \). Now apply Theorem 1.

**Corollary 3.** If \( C_0 \) is an \([n, k, d_0]\) linear block code that contains the all-ones vector, and if \( k \neq 1, n - 1, n \), then there exists an \((n, k, k - 1)\) PUM code with \( d_{\text{free}} \geq 2d_0 \).

**Proof:** Here, Corollary 2 is applied, with \( C^* \) being the \([n, 1, n]\) code consisting of the two vectors \([00\ldots1]\) and \([11\ldots1]\). Clearly \( C^* \) is fixed by all permutations of \( \{1, 2, \ldots, n\} \). Furthermore, the only binary linear codes that are fixed by all permutations of \( \{1, 2, \ldots, n\} \) have dimensions \( 0, 1, n - 1, n \), or \( n \), so there must be an automorphism of \( C^* \) that doesn’t fix \( C_0 \). Thus, by Corollary 2, there exists an \((n, k, k - 1)\) PUM code with \( d_{\text{free}} \geq \min(2d_0, n) \). However, since \( k \geq 2 \), the minimum distance \( d_0 \) of \( C^* \) must be \( < n \); and since \( C_0 \) contains the all-ones vector, there must be a minimum weight \( n - d_0 \). Hence, \( n - d_0 \leq d_0 \), and so \( d_0 \leq n/2 \). Hence, \( \min(2d_0, n) = 2d_0 \), so that in fact \( d_{\text{free}} \geq 2d_0 \).

**Corollary 4.** (Same as Theorem 5 in [7]). Suppose that \( C_0 \) is an \([n, k, d_0]\) linear block code, \( C^* \) is an \([n, k^*, d^*]\) code that is a subcode of \( C_0 \), and \( k - k^* \). Then, for every integer \( i \) in the range \( 1 \leq i \leq k - k^* - 1 \), there is a noncatastrophic \([n, k - i, k^* - i - k^* + d^*]\) PUM code with \( d \geq \min(d^*, 2d_0) \).

**Proof:** Let \( C_i^0 \) be any \((n, k - i)\) subcode of \( C_0 \) that contains \( C^* \). (The conditions on \( i \) guarantee that \( \dim C_{i+1}^* \leq \dim C_i^* \leq \dim C_0 \), so this is possible.) By Lemma A2, there exists a subcode \( C_i^1 \) not equal to \( C_i^0 \) but having the same dimension, and also lying between \( C_0 \) and \( C^* \). Thus, \( C_i^0 \) and \( C_i^1 \) are both \([n, k - i, d^*]\) block codes, with \( d^* \geq d_0 \). By applying Theorem 1 to the codes \( C_i^0, C_i^1, \) and \( C^* \), one obtains a noncatastrophic \([n, k - i, k - i - k^* + d^*]\) PUM code with \( d \geq \min(d^*, 2d^*) \geq \min(d^*, 2d_0) \).
III. Examples

In this section, four examples of PUM codes are presented, that were constructed with the help of the results in Section II. Example 1 describes a Hamming [8,4,3,8] PUM code, which was originally discovered by Lauer [4]. Example 2 gives a generalization of Example 1 to a class of Reed-Muller \([\mu, \nu, \mu + 1, \nu, \nu + 1]\) PUM codes, one code for each pair of positive integers \((\mu, \nu)\) except \((1,1)\) and \((2,1)\). (The code of Example 1 corresponds to the pair \((3,1)\).) The codes in Example 2 were also found, using different methods, by Lauer. Finally, in Examples 3 and 4, two new PUM Golay codes, with parameters \([24, 12, 7, 12]\) and \([24, 12, 10, 16]\), are presented.

Example 1 (a Hamming PUM Code). Let \(C_0\) be the \([7, 4, 3]\) binary cyclic code with generator polynomial \(g_0(x) = 1 + x + x^3\), and let \(C_1\) be the \([7, 4, 3]\) binary cyclic code with generator polynomial \(g_1(x) = 1 + x^2 + x^3\). Take as a generator matrix for \(C_0\) the 4 x 7 binary matrix \(G_0\), whose rows are \(g_0(x)g_0(x), g_0(x), xg_0(x), \) and \(x^2g_0(x),\) and for \(C_1\) the 4 x 7 binary matrix \(G_1\), whose rows are \(g_0(x)g_1(x), g_1(x), xg_1(x), \) and \(x^2g_1(x),\) i.e.,

\[
G_0 = \begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 0
\end{pmatrix}
\]

Now, if each code is extended to length 8 by appending an overall parity-check, one obtains codes \(C_0\) and \(C_1\), both of which are binary \([8,4,4]\) codes with generator matrices

\[
G_0 = \begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 1 & 0
\end{pmatrix}
\]

\[
G_1 = \begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 0
\end{pmatrix}
\]

Since \(C_0 \cap C_1\) is the binary \([7,1,7]\) repetition code, it follows that \(C_0 \cap C_1\) is the binary \([8,1,8]\) repetition code. Thus, in Theorem 1, \(C^*\) can be taken to be the \([8,1,8]\) repetition code with the 1 x 8 generator matrix

\[
K^* = (1 1 1 1 1 1 1 1)
\]

It follows from Corollary 1 that the matrix

\[
G(D) = \begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\]

generates a noncatastrophic \([8,4,3,8]\) PUM code. Furthermore, from [4] (Formula (3) with \(L = 0\)) or [7] (Corollary 1 to Theorem 1, with \(L = 1\)), any \([8,4,3]\) convolutional code must have \(d_{\text{free}} \leq 8\), so this code is optimal.\(^4\)

Example 2 (Some Reed-Muller PUM Codes). Let \(\mu\) and \(\nu \geq 1\) be positive integers. Let \(A_\mu\) be the \([2^\mu, \mu + 1, 2^{\mu-1}]\) first-order Reed-Muller code, and let \(B_\mu\)

\[^4\text{This code first appeared in the literature in [4], Table 1. It is apparently used by the Soviets in their } Regatta\text{ space communication system.}\]

\[^5\text{MacWilliams and Sloane [6], Chapter 13, is a good reference for Reed-Muller codes.}\]
codes. A similar family of $[2^\mu, \mu, \mu - 1, 2^\mu]$ codes was constructed in [7, Example 4].

**Example 3 (the [24, 12, 7, 12] Golay PUM Code).**

It is well known that there exists a [24, 12, 8] binary linear code, viz., the famous Golay code. It turns out that there are two isomorphic copies of the Golay code that contain a common [24, 5, 12] subcode, so that by Theorem 1, there exists a noncatastrophic [24, 12, 7, 12] PUM convolutional code. In this example, the construction of this code is detailed.

Define, for $A, B, C, D, E, F \in GF(8)$, the following two functions:

\[ f_{A,B,C}(x, y) = \text{Tr}(Axy) + \text{Tr}((B + Cy)x^6) \]  
\[ g_{D,E,F}(x, y) = \text{Tr}((Dy + E)x) + \text{Tr}((Fy)x^6) \]

where $\text{Tr}(x) = x + x^2 + x^4$ is the trace mapping from $GF(8)$ to $GF(2)$. Then, if $\beta$ is a fixed nonzero element of trace 0 in $GF(8)$, the following set of 212 length-24 vectors is a [24, 12, 8] Golay code, which is called $A_0$:

\[ A_0 = \{ f_{A,B,C}(x, \beta) + e_0 \mid f_{A,B,C}(x, \beta^2) + e_1 \} \times f_{A,B,C}(x, \beta^4) + e_2 \}_{x \in GF(8)} \]  

In Eq. (28), the parameters $A$, $B$, and $C$ assume all values in $GF(8)$, and the parameters $e_0$, $e_1$, and $e_2$ assume all values in $GF(2)$. The proof that $A_0$ is indeed a [24, 12, 8] code appears in Appendix B as Lemma B5.

Similarly, the following set of 212 length-24 vectors is another [24, 12, 8] Golay code, which is called $B_0$:

\[ B_0 = \{ g_{D,E,F}(x, \beta) + \delta_0 \mid g_{D,E,F}(x, \beta^2) + \delta_1 \} \times g_{D,E,F}(x, \beta^4) + \delta_2 \}_{x \in GF(8)} \]  

In Eq. (29), the parameters $D$, $E$, and $F$ assume all values in $GF(8)$, and the parameters $\delta_0$, $\delta_1$, and $\delta_2$ assume all values in $GF(2)$. The proof that $B_0$ is indeed a [24, 12, 8] code appears in Appendix B as Lemma B6.

In Appendix B (Lemma B10), it is shown that $A_0 \cap B_0$ is a [24, 9, 8] code consisting of the following set of vectors:

\[ A_1 = \{ f_{A,0,C}(x, \beta) + e_0 \mid f_{A,0,C}(x, \beta^2) + e_1 \} \times f_{A,0,C}(x, \beta^4) + e_2 \}_{x \in GF(8)} \]  

The code $A_1$, in turn, contains a [24, 5, 12] subcode consisting of the following set of vectors:

\[ A_3 = \{ f_{0,0,C}(x, \beta) + e_0 \mid f_{0,0,C}(x, \beta^2) + e_1 \} \times f_{0,0,C}(x, \beta^4) + e_0 + e_1 \}_{x \in GF(8)} \]

(see Lemma B8 in Appendix B). Finally, $A_3$ contains a [24, 2, 16] subcode $A_4$:

\[ A_4 = \{ e_0 \mid e_0 + e_1 \}_{x \in GF(8)} \]

(see Lemma B9). It follows from Theorem 1 that there exists both a [24, 12, 7, 12] code and a [24, 12, 10, 16] code, and by the bounds in [4] and [7], they are optimal. To actually construct noncatastrophic generator matrices for these two codes, however, more work is necessary. Here are the needed intermediate subspaces (see Fig. 2):

\[ B_1 = \{ g_{0,E,F}(x, \beta) + \delta_0 \mid g_{0,E,F}(x, \beta^2) + \delta_1 \} \times g_{0,E,F}(x, \beta^4) + \delta_2 \}_{x \in GF(8)} \]
\[ A_2 = \{ f_{0,0,C}(x, \beta) + e_0 \mid f_{0,0,C}(x, \beta^2) + e_1 \} \times f_{0,0,C}(x, \beta^4) + e_2 \}_{x \in GF(8)} \]
\[ B_2 = \{ g_{0,0,0}(x, \beta) + \delta_0 \mid g_{0,0,0}(x, \beta^2) + \delta_1 \} \times g_{0,0,0}(x, \beta^4) + \delta_2 \}_{x \in GF(8)} \]

In Eq. (35), $e$ assumes only the two values 0 and 1.

---

6 In [8], [24, 5, 12] and [24, 2, 16] subcodes of a [24, 12, 8] Golay code were found, which led, via Corollary 4, to the construction of both [24, 11, 6, 12] and [24, 11, 9, 16] PUM codes.
In order to show that the subspaces in Fig. 2 behave as depicted, the following must be proved:

\[ A_0 \cap B_0 = A_1 \quad (36) \]
\[ A_1 \cap B_1 = A_2 \quad (37) \]
\[ A_2 \cap B_2 = A_3 \quad (38) \]

These relationships are proved in Appendix B in Lemmas B10, B11, and B12.

Example 4 (the \([24,12,10,16]\) Golay PUM Code). The \([24,5,12]\) binary linear code of Example 3 contains a \([24,2,16]\) subcode, so that by Theorem 1, there exists a noncatastrophic \([24,12,10,16]\) PUM convolutional code. In this example, the construction of this code is detailed. The subspaces \(A_0, B_0, A_1, B_1, A_2, \) and \(A_4\) defined in Example 3 are used. Additionally, subspaces \(B_0, A'_1,\) and \(B'_1\) are defined as follows:

\[ B_0 = \begin{bmatrix} g_{00,0,0}(x,\beta) + g_{01,0,0}(x,\beta^2) + g_{10,0,0}(x,\beta^3) + g_{11,0,0}(x,\beta^4) \end{bmatrix} + \delta_0 \in \mathbb{GF}(s) \]
\[ A'_1 = \begin{bmatrix} c_0(1-x^2) \end{bmatrix} \]
\[ B'_1 = \begin{bmatrix} g_{0,0,0}(x,\beta) + g_{0,0,0}(x,\beta^2) + g_{1,0,0}(x,\beta^3) + g_{1,0,0}(x,\beta^4) \end{bmatrix} + \delta_0 + \delta_1 \in \mathbb{GF}(s) \]

In Eq. (41), \(e\) assumes only the two values 0 and 1.

The proof that the subspaces behave as depicted in Fig. 3, i.e., that \(A_2 \cap B'_2 = A'_3\) and \(A'_3 \cap B'_3 = A_4\), is given in Appendix B, Lemmas B13 and B14.
In binary, these matrices are

\[
\begin{bmatrix}
011 & 011 \\
101 & 101 \\
100 & 100 \\
001 & 001 \\
00\beta & 00\beta \\
00\beta^2 & 00\beta^2 \\
100 & 100 \\
0\beta & 0\beta \\
0\beta^2 & 0\beta^2 \\
0\beta^3 & 0\beta^3 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
0000000 & 0111111 & 1111111 \\
1111111 & 0000000 & 1111111 \\
1111111 & 0000000 & 0000000 \\
0111010 & 0011101 & 0100111 \\
0111010 & 1101101 & 1010111 \\
0111010 & 0111010 & 1110101 \\
1111000 & 1110100 & 1110100 \\
0111010 & 0111010 & 0111010 \\
0011101 & 0011101 & 0011101 \\
\end{bmatrix}
\]

The codes in Examples 1, 3, and 4 are quite interesting as combinatorial objects, but they have potential for applications. To illustrate, Fig. 4 shows a plot of the performance of these three codes and the NASA standard [2, 1, 6, 10] (non-PUM) code on an additive white Gaussian channel. Figure 4 shows that the low-complexity Hamming [8, 4, 3, 8] code is only a bit weaker than the NASA code, while the two Golay codes are both a bit stronger. Since the state complexity of the [24, 12, 7, 12] Golay code is only 1 greater than that of the NASA code, it may be that there is a relatively low-complexity decoding algorithm for this code, whose performance will significantly exceed that of the NASA code. In any case, these performance curves certainly justify a serious study of efficient decoding algorithms for these and other PUM codes.

**Acknowledgment**

The authors thank Ivan Onyszchuk of the Communications Systems Research Section for providing the curves used in Fig. 4.
Fig. 1. The construction of Lemma 2.

Fig. 2. The subspaces needed for the construction of a noncatastrophic encoder for the [24, 12, 7, 12] PUM code.

Fig. 3. The subspaces needed for the construction of a noncatastrophic encoder for the [24, 12, 10, 16] code.

Fig. 4. Performance curves for three PUM codes, compared with the NASA standard [2, 1, 6, 10] code, on an additive white Gaussian channel.
Appendix A

Two Results From Linear Algebra

In this Appendix, two simple results from linear algebra are provided that are needed in the proof of Lemma 2.

Lemma A1. If \( 0 = A_0 \subseteq A_1 \subseteq \cdots \subseteq A_m = V \) is an ascending chain of subspaces of an \( n \)-dimensional vector space \( V \), with \( \dim A_i = k_i \), then there exists a basis \( \langle \alpha_1, \ldots, \alpha_n \rangle \) for \( V \) such that
\[
\langle \alpha_1, \ldots, \alpha_{k_i} \rangle = A_i, \quad \text{for } i = 1, \ldots, m \tag{A-1}
\]

Proof: One proceeds recursively, as follows. Choose a basis \( \langle \alpha_1, \ldots, \alpha_{k_1} \rangle \) for \( A_1 \). If \( m = 1 \), one is done. Otherwise, by using a standard result in linear algebra [2, Lemma 4.2.5], the basis \( \langle \alpha_1, \ldots, \alpha_{k_1} \rangle \) for \( A_1 \) can be extended to a basis \( \langle \alpha_1, \ldots, \alpha_{k_1}, \ldots, \alpha_{k_2} \rangle \) of \( A_2 \), etc.

Lemma A2. Suppose that \( V \) is an \( n \)-dimensional vector space over \( F \), and \( S \) and \( T \) are subspaces of \( V \) with \( S \subseteq T \subseteq V \). Then there exists a subspace \( T' \neq T \) such that \( \dim T' = \dim T \) and \( S \subseteq T' \subseteq V \).

Proof: Suppose that \( \dim S = k \) and \( \dim T = k + j \), where \( j > 0 \). By Lemma A1, it is possible to find a basis for \( V \) of the form
\[
\langle V \rangle = \langle \alpha_1, \ldots, \alpha_k, \beta_1, \ldots, \beta_j, \gamma_1, \ldots, \gamma_h \rangle
\]
where \( k + j + h = n \), and
\[
\langle \alpha_1, \ldots, \alpha_k \rangle = S
\]
\[
\langle \alpha_1, \ldots, \alpha_k, \beta_1, \ldots, \beta_j \rangle = T
\]
If \( h \geq j \), define \( T' \) as follows:
\[
T' = \langle \alpha_1, \ldots, \alpha_k, \gamma_1, \ldots, \gamma_j \rangle
\]
If, on the other hand, \( h < j \), define \( T' \) as follows:
\[
T' = \langle \alpha_1, \ldots, \alpha_k, \gamma_1, \ldots, \gamma_h, \beta_1, \ldots, \beta_{j-h} \rangle
\]
Appendix B
Proofs Needed in Examples 3 and 4

In this Appendix, the assertions made in Section III about the subspaces $A_0, A_1, A_2, A_3, A_4, B_0, B_1, B_2, B'_1,$ and $B'_2$ of $V_{24}(2)$ are proved.

**Lemma B1.** Let $A$ and $B$ be elements of $GF(8)$ such that $Tr(Ax + Bx^6) = 0$ for all $x \in GF(8)$. Then $A = B = 0$.

**Proof:** Since $Tr(y) = y + y^2 + y^4$ and $y^8 = y$ for all $y \in GF(8)$, it follows that

$$Tr(Ax + Bx^6) = Ax + A^2x^2 + A^4x^4 + Bx^6 + B^2x^5 + B^4x^3$$

(B-1)

for all $x \in GF(8)$. Thus, the equation $Tr(Ax + Bx^6) = 0$ is a polynomial equation of sixth degree with 8 roots in $GF(8)$, and hence, the coefficients of the polynomial must be zero, i.e., $A = B = 0$, as asserted.

**Lemma B2.** Let $f_{A,B,C}(x,y)$ be defined as in Eq. (26), and suppose there exists a nonzero element $y^*$ in $GF(8)$ such that $f_{A,B,C}(y^*,y) = 0$, for all $y \in GF(8)$.

Then, $A = 0$ and $B = Cy^*$. Further, if Eq. (B-2) holds and if $f_{A,B,C}(x,y)$ is not identically zero, then for any $y \neq y^*$, the number of solutions $x \in GF(8)$ to $f_{A,B,C}(x,y) = 0$ is exactly four.

**Proof:** If Eq. (B-2) holds, then by Eq. (26), one has

$$Tr((Ay^*)x + (B + Cy^*)x^6) = 0,$$  \hspace{1cm} \text{for all } x \in GF(8) \hspace{1cm} \text{(B-3)}$$

Then, $A = 0$ and $B = Cy^*$. Further, if Eq. (B-2) holds and if $f_{A,B,C}(x,y)$ is not identically zero, then for any $y \neq y^*$, the number of solutions $x \in GF(8)$ to $f_{A,B,C}(x,y) = 0$ is exactly four.

**Proof:** In view of the definitions in Eq. (26) and Eq. (27) of $f$ and $g$, the given conditions are equivalent to

$$Tr((Ay + Dy + E)x$$

$$+ (B + Cy + Fy)x^6) = 0,$$  \hspace{1cm} \text{for all } x \in GF(8) \hspace{1cm} \text{(B-7)}$$

for $y = y_1, y_2$. Thus, according to Lemma B1, $Ay + Dy + E = 0$ and $B + Cy + Fy = 0$ for $y = y_1, y_2$. The two equations $Ay_1 + Dy_1 + E = 0$ imply that $A = D$ and $E = 0$, and the two equations $Cy_1 + Fy_1 + B = 0$ imply that $C = F$ and $B = 0$.

**Lemma B5.** The code $A_0$ defined in Eq. (28) is a $[24, 12, 8]$ code.
Proof: The mapping from 6-tuples \([A, B, C, \epsilon_0, \epsilon_1, \epsilon_2]\) to codewords in \(A_0\) is linear. The kernel of this mapping is the set of 6-tuples such that the corresponding codeword is 0, i.e.,

\[
f_{A, B, C}(x, \beta^{2i}) + \epsilon_i = 0, \quad \text{for } i = 0, 1, 2 \quad (B-8)
\]

for all \(x \in GF(8)\). By substituting \(x = 0\) into these equations, one finds that \(\epsilon_i = 0\) for \(i = 0, 1, 2\), so that in fact

\[
f_{A, B, C}(x, \beta^{2i}) = 0, \quad \text{for } i = 0, 1, 2 \quad (B-9)
\]

for all \(x \in GF(8)\). It then follows from Lemma B2 that \(A = B = C = 0\). Thus, the kernel of the mapping contains only the 6-tuple \([0, 0, 0, 0, 0, 0]\), and so the mapping is one-to-one. But since the set of 6-tuples is 12-dimensional, it follows that the code is also 12-dimensional. Thus, \(A_0\) is a \((24, 12)\) code. It remains to prove that its minimum distance is 8.

To show that the minimum distance is 8, first consider the \((24, 9)\) code \(A_0'\) defined by Eq. (28) with \(e_0 = \epsilon_1 = \epsilon_2 = 0\). Each word in \(A_0'\) has three 8-bit segments, viz., the 8 bits corresponding to the function \(f_{A, B, C}(x, y)\) for \(y = \beta, \beta^2, \text{ and } \beta^4\). Since in each segment the bit corresponding to \(z = 0\) is 0, each segment may in fact be viewed as a 7-bit codeword with components indexed by the consecutive powers of a primitive root of \(GF(8)\). Thus, for the “\(A, B, C\)” codeword in \(A_0'\), the \(y\)-segment’s \(i\)th component is given by \(f_{A, B, C}(\beta^i, y)\), where

\[
f_{A, B, C}(x, y) = Tr((Ay)(B+C)y + (A^2y^2)(B^2+C^2y^2))
\]

\[= (AB + A^4C^4)y + (A^2B^2 + AC)y^2
\]

\[+ (A^4B^4 + A^2C^2)y^4
\]

\[= \text{Tr}((AB + A^4C^4)y) \quad (B-12)
\]

It follows that each 7-bit segment is a codeword in the \((7, 6)\) binary cyclic code with generator polynomial \(g(x) = x - 1\). In particular, each segment has even weight. The value of the weights modulo 4 can be computed by a theorem of McEliece-Solomon [9, Theorem 1] or [1, Theorem 16.33], which says that if an even-weight binary vector \(a = (a_0, \ldots, a_{n-1})\) is described by its Mattson-Solonon (MS) polynomial (discrete Fourier transform) \(A(x) = A_0 + \cdots + A_{n-1}x^{n-1}\), i.e., if

\[
a_i = \sum_{j=0}^{n-1} A_j \beta^{-ij} \quad (B-11)
\]

where \(\beta\) is a primitive \(n\)th root of unity, and if \(\Gamma_2(a) = \sum_{i=0}^{(n-1)/2} A_j A_{n-j}\), then \(w(a) \equiv 2\Gamma_2(a) \pmod{4}\). The MS polynomials for the 7-bit segments are given by Eq. (B-10), and so the value of \(\Gamma_2\) for the \(y\)-segment is

\[
\Gamma_2(y) = (Ay)(B + Cy) + (A^2y^2)(B^2 + C^2y^2)
\]

\[+ (B^4 + C^4y^4)(A^4y^4)
\]

\[= (AB + A^4C^4)y + (A^2B^2 + AC)y^2
\]

\[+ (A^4B^4 + A^2C^2)y^4
\]

\[= \text{Tr}((AB + A^4C^4)y) \quad (B-12)
\]

If the three segments are combined into one 21-bit word, the overall weight is still even, and the overall weight \(w(a) \equiv 0 \pmod{4}\). The MS polynomials for the 7-bit segments are given by Eq. (B-10), and so the value of \(\Gamma_2\) for the \(y\)-segment is

\[
\Gamma_2 = \Gamma_2(\beta) + \Gamma_2(\beta^2) + \Gamma_2(\beta^4)
\]

\[= \text{Tr}((AB + A^4C^4)(\beta + \beta^2 + \beta^4))
\]

\[= \text{Tr}(0) = 0 \quad (B-13)
\]

Thus, each 7-bit segment has weight 0, 2, 4, or 6, and the overall weight is divisible by four. Furthermore, if one of the segments has weight zero, then by Lemma B2 either the other two segments are both zero, or else the other two segments have weight 4. It follows that the weights in the \((24, 9)\) code are 0, 8, 12, and 16. Now the original code \(A_0\) is obtained from \(A_0'\) by complementing some or all of the segments, i.e., by replacing a segment of weight \(w\) with one of weight \(8 - w\). Thus, in \(A_0\), the segments have weight 0, 2, 4, 6, or 8. But since \(8 - w \equiv w \pmod{4}\), the weights in \(A_0\) must also be divisible by four, and so in \(A_0\) the only weights that can occur are 0, 8, 12, 16, 20, and 24. The weight 4 can only occur as 0 + 4 + 0 and 0 + 2 + 2. Both of these cases can be eliminated by observing that since a zero-weight segment can only occur in an uncomplemented segment, and Lemma B2 says that if a codeword in \(A_0\) has a zero-weight segment, then either the other two segments both have weight 8, or both have weight 4. Weight 20 is
ruled out by observing that the complement of a word of weight 20 is a word of weight 4.

Lemma B5 says that the code $A_0$ is a $[24, 12, 8]$ binary linear code. According to MacWilliams and Sloane [6, Section 20.6], such a code must be equivalent to the Golay code. The next Lemma indicates that the code $B_0$ defined in Eq. (29) is also equivalent to the Golay code.

Lemma B6. The code $B_0$ defined in Eq. (29) is a $[24, 12, 8]$ code.

Proof: The proof is virtually the same as the proof of Lemma B5. The key difference is that in place of Eq. (B-10), one has for the code $B_0$
\begin{align*}
g_{D,E,F}(x, y) &= \text{Tr}((Dy + E)x + (Fy)x^5) \\
&= (Dy + E)x + (D^2y^2 + E^2)x^2 \\
&\quad + (F^4y^4)x^3 + (D^4y^4 + E^4)x^4 \\
&\quad + (F^2y^2)x^5 + (Fy)x^6 
\end{align*}
so that in place of Eq. (B-12) one has
\[ \Gamma_2(y) = (Dy + E)(Fy) + (D^2y^2 + E^2)(F^2y^2) \\
\quad + (F^4y^4)(D^4y^4 + E^4) \\
\quad = (EF + D^4F^4)y + (E^2F^2 + DF)y^3 \\
\quad\quad + (F^2F^2 + D^2F^2)y^4 \\
\quad\quad = \text{Tr}((EF + D^4F^4)y) 
\]
and in place of Eq. (B-13) one has
\[ \Gamma_2 = \Gamma_2(\beta) + \Gamma_2(\beta^2) + \Gamma_2(\beta^4) \\
\quad = \text{Tr}((EF + D^4F^4)(\beta + \beta^2 + \beta^4)) \\
\quad = \text{Tr}(0) = 0 
\]
Further details are omitted here.

Lemma B7. The code $A_1$ defined in Eq. (30) is a $[24, 9, 8]$ code.

Proof: Just as in the proof of Lemma B5, the mapping from 6-tuples $[A, 0, C; \epsilon_0, \epsilon_1, \epsilon_2]$ to codewords in $A_1$ is a linear, one-to-one mapping, which implies that $A_1$ is a $(24, 9)$ code. Since $A_1$ is a subcode of $A_0$, its minimum distance must be $\geq 8$. There are, however, many codewords in $A_1$ of weight 8, e.g., that obtained by taking $A = C = 0$, $\epsilon_0 = 1$, and $\epsilon_1 = \epsilon_2 = 0$.

Lemma B8. The code $A_3$ defined in Eq. (31) is a $[24, 5, 12]$ code.

Proof: Using the formula Eq. (26) for $f_{A,B,C}(x, y)$, one finds that any codeword in $A_3$ can be represented as follows:
\[ [\text{Tr}(C\beta x^5) + \epsilon_0]\text{Tr}(C\beta^2 x^6) + \epsilon_1]\text{Tr}(C\beta^4 x^5) \\
\quad + (\epsilon_0 + \epsilon_1)x \in GF(8) \] (B-17)

Two cases are considered: $C = 0$ and $C \neq 0$. If $C = 0$, then the codeword in Eq. (B-17) becomes $[\epsilon_0] \epsilon_1 \epsilon_0 + \epsilon_1]$, which is either identically zero or has weight 16. If $C \neq 0$, then since there are exactly 4 elements in $GF(8)$ with trace 0, the codeword in Eq. (B-17) has weight 12. Thus, the only weights that occur in $A_3$ are 0, 12, and 16, and so $A_3$ is a $[24, 5, 12]$ code, as asserted.

Lemma B9. The code $A_4$ defined in Eq. (32) is a $[24, 2, 16]$ code.

Proof: According to the definition in Eq. (32), each codeword in $A_4$ has three 8-bit segments. Either all three segments are identically zero, or else one segment is zero and the other two have weight 8. Thus, in $A_4$ the only weights that occur are 0 and 16, so that $A_4$ is a $[24, 2, 16]$ code, as asserted.

Lemma B10. $A_0 \cap B_0 = A_1$.

Proof: Note that from Eq. (26) and Eq. (27), $f_{A,0,C}(x, y) = g_{A,0,C}(x, y)$. Thus, the code $A_0 \cap B_0$ contains the code $A_1$ as defined in Eq. (30). To prove the opposite inclusion, note that by the definitions Eq. (28) and Eq. (29) of $A_0$ and $B_0$, any word in the intersection will produce an equation of the form $f_{A,B,C}(x, \beta^2) + \epsilon_i = g_{D,E,F}(x, \beta^2) + \delta_i$ for all $x \in GF(8)$, for $i = 0, 1, 2$. By substituting $x = 0$ on both sides of this equation, one gets $\epsilon_i = \delta_i$, so that in fact, $f_{A,B,C}(x, \beta^2) = g_{D,E,F}(x, \beta^2)$ for
all \( x \in GF(8) \), for \( i = 0, 1, 2 \). By Lemma B4, \( A = D, C = F, E = 0 \), and \( B = 0 \), so that a word in the intersection must be of the form Eq. (30), i.e., it must lie in \( A_1 \).

**Lemma B11.** \( A_1 \cap B_1 = A_2 \).

**Proof:** Given the definitions in Eq. (30) and Eq. (33) of \( A_1 \) and \( B_1 \), any word in the intersection \( A_1 \cap B_1 \) will produce an equation of the form \( f_{A_0, C}(x, \beta^2) + \epsilon_i = g_{0, E, F}(x, \beta^2) + \delta_i \), for all \( x \in GF(8) \) and \( i = 0, 1, 2 \). By substituting \( x = 0 \) on both sides of these equations, one gets \( \epsilon_i = \delta_i \), so that in fact one has \( f_{A_0, C}(x, \beta^2) = g_{0, E, F}(x, \beta^2) \). By Lemma B4, this implies \( A = E = 0 \) and \( C = F \). Thus, the intersection \( A_1 \cap B_1 \) is exactly the same as \( A_2 \), as defined in Eq. (34).

**Lemma B12.** \( A_2 \cap B_2 = A_3 \).

**Proof:** Given Lemma B4 and the definitions Eq. (34) and Eq. (35) of \( A_2 \) and \( B_2 \), this result is immediate.

**Lemma B13.** \( A_2 \cap B_2' = A_2' \).

**Proof:** If a word in \( A_2 \), as defined in Eq. (34), is the same as a word in \( B_2' \), as defined in Eq. (39), then by setting \( x = 0 \), one finds that \( \epsilon_0 = \delta_0, \epsilon_1 = \delta_1, \) and \( \epsilon_2 = \delta_2 \). Thus, also \( f_{0, 0, C}(x, y) = g_{0, E, 0}(x, y) \) for all \( x \in GF(8) \) and \( y = \beta, \beta^2, \beta^4 \). It then follows from Lemma B4 that \( C = E = 0 \), and so a word in the intersection \( A_2 \cap B_2' \) must be of the form described in Eq. (40).

**Lemma B14.** \( A_3' \cap B_3' = A_4 \).

**Proof:** If a word in \( A_3' \), as defined in Eq. (40), is the same as a word in \( B_3' \), as defined in Eq. (41), then by setting \( x = 0 \), one finds that \( \epsilon_0 = \delta_0, \epsilon_1 = \delta_1, \) and \( \epsilon_2 = \delta_0 + \delta_1 \). Thus, also \( f_{0, 0, C}(x, y) = g_{0, E, 0}(x, y) \) for all \( x \in GF(8) \) and \( y = \beta, \beta^2, \beta^4 \). It then follows from Lemma B4 that \( e = 0 \), and so a word in the intersection \( A_3' \cap B_3' \) must be of the form described in Eq. (32).
References


