Rate Gains in Block-Coded Modulation Systems with Interblock Memory

Chia-Ning Peng, *Member, IEEE*, Houshou Chen, *Member, IEEE*, John T. Coffey, *Member, IEEE*, and Richard G. C. Williams, *Member, IEEE*

***Abstract*—This paper examines the performance gains achievable by adding interblock memory to, and altering the mapping of coded bits to symbols in, block-coded modulation systems. The channel noise considered is additive Gaussian, and the twin design goals are to maximize the asymptotic coding gain and to minimize the number of apparent nearest neighbors. In the case of the additive white Gaussian noise channel, these goals translate into the design of block codes of a given weighted or 'normalized' distance whose rate is as high as possible, and whose number of codewords at minimum normalized distance is low.**

The effect of designing codes for normalized distance rather than Hamming distance is to ease the problem of determining the best codes for given parameters in the cases of greatest interest, and many such best codes are given.

***Index Terms*—Block-coded modulation, code constructions, gluing, multilevel codes, normalized distance, split weight enumerators.**

I. Introduction

BLOCK-coded modulation (BCM) systems derived from the multilevel construction method form a class of high-performance bandwidth-efficient communication schemes. This method provides a way to generate systematic constructions of block modulation codes with arbitrarily high asymptotic minimum Euclidean distance, by linking the design to the theory of binary block codes. The resulting codes can be decoded easily using decoders for the individual component binary block codes. These modulation codes were originally constructed by Imai and Hirakawa in 1977 [14], and thus appear at the very beginning of the coded modulation field. The codes can also be seen as a special case of more general constructions, such as the generalized concatenated code and related constructions, as in, e.g., Blokh and Zyablov [1], Zinov'ev [37], and Ginzburg [11]; these more general constructions are extensively discussed in the review chapter of Dumer [8]. The codes have been very widely studied since then [2]–[4], [6], [16]–[23], [25], [27], [30], [32]–[36]. We refer the reader interested in a review of the extensive literature on this subject to the articles of Kasami *et al.* [17], Huber *et al.* [13], and Williams [34].

We will treat the basic concepts of block-coded modulation schemes as standard, apart from a brief review in Section II. In summary, in the standard block-coded modulation scheme, a coded sequence of n symbols from a 2^b-ary modulation scheme is obtained by using b binary codes, each of length n, with an assignment of bit labels to signal constellation points obtained via a set-partitioning scheme. Each binary code involves bits of the same significance from the bit labels of the n transmitted symbols. Each transmitted symbol, conversely, has a bit label containing one bit from each of the b component codes of length n. The resulting minimum squared Euclidean distance can be shown to be $\min_i d_i \cdot a_i \cdot E^2$, where d_i is the minimum distance of the ith binary block code and a_i is obtained from the set partitioning labeling scheme. Thus the minimum squared Euclidean distance is determined by the *minimum* of the distances of the component binary block codes weighted by the set partitioning constants.

Two simple modifications to these systems proposed by Lin [19] (see also [20], [21], [23], and [35]), namely, the introduction of "interblock memory" combined with a staggered mapping of code bits to transmitted symbols, have the effect of transforming the code design problem into one in which the criterion of interest is a weighted or "normalized" Hamming distance of a single binary block code of length bn. Thus a codeword that has Hamming weight w_i in bits $(i-1)n+1$ to in for each i has normalized weight $\sum_i w_i \cdot a_i$; the minimum Euclidean distance of the overall coded modulation system will be shown to be $\min_i \sum_i w_i \cdot a_i \cdot E^2$. Compared to regular BCM systems, the systems with interblock memory support substantially higher transmission rates with given probability of error. The main cost is a more complicated structure, and hence more expensive decoding.

This paper will derive the relationship between the motivating coded modulation problem and the normalized distance coding problem, and develop constructions, bounds, and many optimal codes in this framework. It is an interesting fact that two extremely simple bounds turn out to be exact in very many of the cases of interest.

It should be acknowledged here that the relevance of the normalized distance criterion is directly tied to the goal of maximizing the minimum Euclidean distance of the system. Sev-

Manuscript received October 30, 1998; revised September 7, 1999. This work was supported in part by the U.S. Army Research Office under Grant DAAH04-96-1-0377. The material in this paper was presented in part at the IEEE International Symposium on Information Theory, Ulm, Germany, June–July 1997, and at the Workshop on Coding and Cryptography, Paris, France, January 1999.

C.-N. Peng is with Texas Instruments Broadband Access Group, San Jose, CA 95124 USA (e-mail: cnpeng@ti.com).

H. Chen is with the Department of Electrical Engineering, National Chi-Nan University, Nantou, Taiwan 545, R.O.C. (e-mail: houshou@ncnu.edu.tw).

J. T. Coffey is with the Electrical Engineering and Computer Science Department, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: scoffey@eecs.umich.edu).

R. G. C. Williams is with 3-COM, San Diego, CA 92128 USA (e-mail: Richard_Williams@3com.com).

Communicated by E. Soljanin, Associate Editor for Coding Techniques.

Publisher Item Identifier S 0018-9448(00)02891-1.

Bits $1, \ldots, n$	(Previous codeword)		
(Next codeword)	Bits $n+1, \ldots, 2n$	(Previous codeword)	
	(Next codeword)	$\cdots$	
			Bits $bn-n+1, \ldots, bn$

eral alternative approaches have been proposed recently to the code design problem for regular BCM: for example, the capacity rule [13] in which rates are assigned to the component codes according to random coding principles, and the error probability rule [2], [3], [32] in which the component codes are chosen to balance the individual error probabilities at all coding levels. (Indeed, Wachsmann and Huber [32] go so far as to suggest that virtually any technique *except* the traditional Euclidean distance approach will work well!)

Kasami *et al.* [17] discuss block-coded modulation schemes in which there is interblock memory between the component row codes, but in which the coded bits are assigned to symbols in the usual way. The resulting schemes do not have higher overall Euclidean distance in general, but do have far fewer apparent nearest neighbors, no smaller a Euclidean distance, and no more complex a decoder.

The interblock memory systems considered in this paper will all use the "staggered" assignment of coded bits to symbols, in which each coded bit affects exactly one transmitted symbol. The abbreviation BCMIM will refer to interblock memory systems of this type, rather than the kind considered by Kasami *et al.*Previous work on such block-coded modulation with interblock memory systems include the original paper of Lin [19] and subsequent papers by Yamaguchi and Imai [35] and by Lin *et al.* [20], [21], [23] that give codes, with associated trellis decoding structures and performance simulations, though not general bounds and constructions. This paper generalizes the performance metric and explores the limits of such schemes.

II. Definitions of BCM and BCMIM Systems

We assume throughout that the modulation scheme is 2^b-ary, with bit labels assigned by a set partitioning scheme in which the intrasubset squared Euclidean distances rise in the ratio $1 : a_2 : \cdots : a_b$.

A baseline BCM "codeword array" is constructed by taking b binary linear codes $C_1, \cdots, C_b$, each of length n. A sequence of transmitted symbols is obtained by filling the rows of a $b \times n$ array with codewords from these respective codes, then reading labels upwards, column by column.

The codeword array thus has the form:

1st row:	LSLB's	a codeword of C_1
2nd row:	2nd LSLB's	a codeword of C_2
	$\cdots$	
$(b-1)$-th row:	$(b-1)$-th LSLB's	a codeword of C_{b-1}
b-th row:	MSLB's	a codeword of C_b

where LSLB denotes the least significant label bits, and MSLB denotes the most significant label bits.

The minimum squared Euclidean distance between two sequences of n symbols in a baseline BCM system is well known to be

$$D_E^2 = \min_i d_i \cdot a_i \cdot E^2$$

where d_i is the minimum Hamming distance of code C_i. (See, for example, Sayegh [27].) The quantity $\min_i d_i a_i$ is the distance gain of the BCM system over the uncoded system.

For a baseline BCM system using a length bn code and a 2^b-ary modulation scheme to have a distance gain D, we simply choose the codeword array such that each row code C_i has minimum Hamming distance $d_{\min} \geq D/a_i$. The maximum rate of a baseline BCM system with distance gain D is then

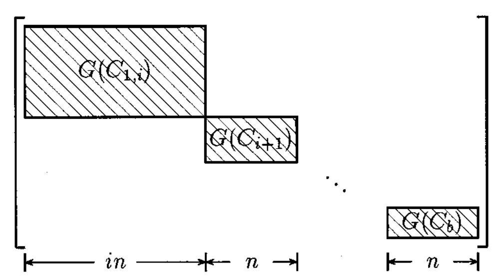
$$\frac{1}{n}\sum_{i=1}^{b} K\left(n, \frac{D}{a_i}\right)$$

bits per symbol, where $K(n,d)$ is the maximum dimension of a length n binary linear code with minimum Hamming distance $d_{\min} \geq d$.

A. Block-Coded Modulation with Interblock Memory

BCMIM systems differ from BCM systems in two ways: the generator matrix is allowed a more general structure, and the codewords are mapped to symbols in a different way.

A BCMIM system with interblock memory between the first i blocks is one with generator matrix of the form



in which $G(C_{1,i})$ is the generator matrix of a binary linear code of length in.

The resulting codewords are mapped to symbols in the following staggered way (see the top of this page), where the symbols will be read and transmitted going from right to left. (Thus bit bn of the codeword affects the transmitted stream first.)

Thus the difference in generator matrix is that we allow interdependencies between the first i rows of the codeword array; and the difference in mapping is that we allow a codeword of length in to affect in transmitted symbols, rather than a block of n symbols as in regular BCM.

The system is decoded one codeword at a time, and it is assumed that we do not use the results of "later" codewords to redecode "earlier" codewords. This is similar to the standard "staged decoder" assumption in BCM systems [27]. The purpose of transmitting blocks from right to left is that each codeword can be decoded as soon as it is received.

B. Asymptotic Performance and Normalized Weight

Definition. Normalized Weight: Given the basic block size n and the sequence $1 : a_2 : \cdots : a_b$ of increasing intrasubset squared Euclidean distances, the *normalized weight* of a word c of length bn is defined as

$$W_n(c) = w_1(c) + a_2 \cdot w_2(c) + \cdots + a_b \cdot w_b(c) \qquad (1)$$

where $w_i(c)$ is the Hamming weight of the ith length n basic block of c. The normalized distance between two codewords and the minimum normalized distance of a code are defined analogously.

The minimum normalized distance derives its significance from the fact that it represents the asymptotic (in signal-to-noise ratio) improvement in squared Euclidean distance of the BCMIM system over the uncoded system. Assuming that all past codewords have been decoded correctly, i.e., assuming that all blocks above the one to be decoded in the codeword array above are known and correct, the squared Euclidean distance between two codewords $c_1 = [c_{1,1}| \cdots |c_{1,b}]$ and $c_2 = [c_{2,1}| \cdots |c_{2,b}]$ is

$$\begin{aligned} E^2 \cdot a_1 \cdot w(c_{1,1} - c_{2,1}) + \cdots + E^2 \cdot a_b \cdot w(c_{1,b} - c_{2,b}) \\ = \sum_{1 \le k \le b} E^2 \cdot a_k \cdot w_k(c_1 - c_2) = E^2 W_n(c_1 - c_2) \end{aligned}$$

i.e., the minimum squared Euclidean distance is higher than in the uncoded case by exactly the minimum normalized weight of the code, as claimed.

The probability of error is not exactly predicted by this minimum squared Euclidean distance due to the conditioning on previous decoding results. The conditional distribution of the additive noise given that previous blocks were correct is not Gaussian; more seriously, the conditional noise distribution given that errors did occur in the blocks above the current one in the codeword array above will be both non-Gaussian and more severe than the unconditional distribution. (Kofman *et al.* [18] develop rigorous analytical bounds on bit-error probability for a bit-interleaved coded modulation system using convolutional codes with 8-PSK, taking these complicating factors into account.) Thus there will be an error propagation effect in the system. This effect may be ameliorated by the use of decoding algorithms that include soft output [12], or by occasional insertion of known symbols. At higher signal-to-noise ratios, however, the previous few symbols will be correct with high probability and the conditional noise distribution will be close to the unconditional one. Thus normalized distance has operational significance in representing the exact gain of a BCMIM system over a baseline BCM system at asymptotically high signal-to-noise ratios.

Note that if we take a baseline BCM system and adopt the staggered assignment of the codeword array to symbols as above, then the minimum Euclidean distance of the resulting system is $\min_{\boldsymbol{c} \neq \boldsymbol{0}} \sum_i a_i w_i(c)$ as for BCMIM; but since the codeword array consists of independently coded rows, this reduces to $\min_i a_i d_i$ as for BCM systems without the staggered assignment. This gives an interpretation of the advantage achieved by BCMIM systems: if the minimum normalized distance is fixed, a BCMIM system allows us to augment the code by codewords that span several of the basic blocks, and that each have at least the target normalized distance. This gives us a system with the same asymptotic distance gain, but higher rate. Thus on a graph of R_d/W versus E_s/N_0, at low error probability, the BCMIM operating point would be directly above the BCM operating point. On a graph of R_d/W versus E_b/N_0, the BCMIM operating point would be asymptotically above and to the left of the BCM operating point, with the horizontal distance being $10 \log_{10} R_{\rm BCMIM}/R_{\rm BCM}$ decibels.

The net coding gain of a block-coded modulation system at a given block error probability can be defined for the purposes of this discussion to be the distance on an R_d/W versus E_b/N_0 graph between the operating point of the coded modulation scheme at that error probability and an operating point at the same rate on a curve interpolated between uncoded systems. Thus in Fig. 1, the uncoded QAM curve is given by $E_b/N_0 = A + 10 \log_{10}(2^R - 1)/R$ decibels, where A is chosen to fit the uncoded quadrature phase-shift keying (QPSK) error probability to the target error probability.

C. Nearest Neighbors

It is widely known that the most serious single disadvantage of block-coded modulation schemes is the "nearest neighbor" problem [9], [10], [17], [34]. The standard staged decoding procedure, in which each row of the codeword array is decoded in sequence, beginning with the top row, has the effect of producing a large number of "apparent nearest neighbors," i.e., a large multiplicity of the most likely error events. These cause the coding gain at finite signal-to-noise ratios to be less than the asymptotic coding gain. The exact difference varies according to target probability of error; the usual rule of thumb, proposed by Forney [9], is that the signal-to-noise ratio increases by 0.2 dB for each doubling in the number of apparent nearest neighbors for error probabilities in the range 10^{-5}–10^{-6}.

We will take rough account of the effect of number of nearest neighbors by seeking codes that have as few codewords of minimum normalized weight as possible. The number of apparent nearest neighbors depends on both the transmitted sequence and the overall coded modulation system into which the code is embedded, as noted in the Appendix. Thus the number of minimum normalized weight codewords bears on the number of apparent nearest neighbors only indirectly. It is, however, a property of the code rather than associated modulation schemes.

D. Complexity Comparisons

In adding extra codewords to a baseline BCM system to obtain a BCMIM system, we almost always increase the state complexity of a trellis decoding algorithm. Thus the extra performance of BCMIM over BCM comes at some price. However, it is, of course, possible to decode the resulting codes via other methods, maximum-likelihood or suboptimal, thus producing a very wide variety of performance/complexity tradeoffs. We do

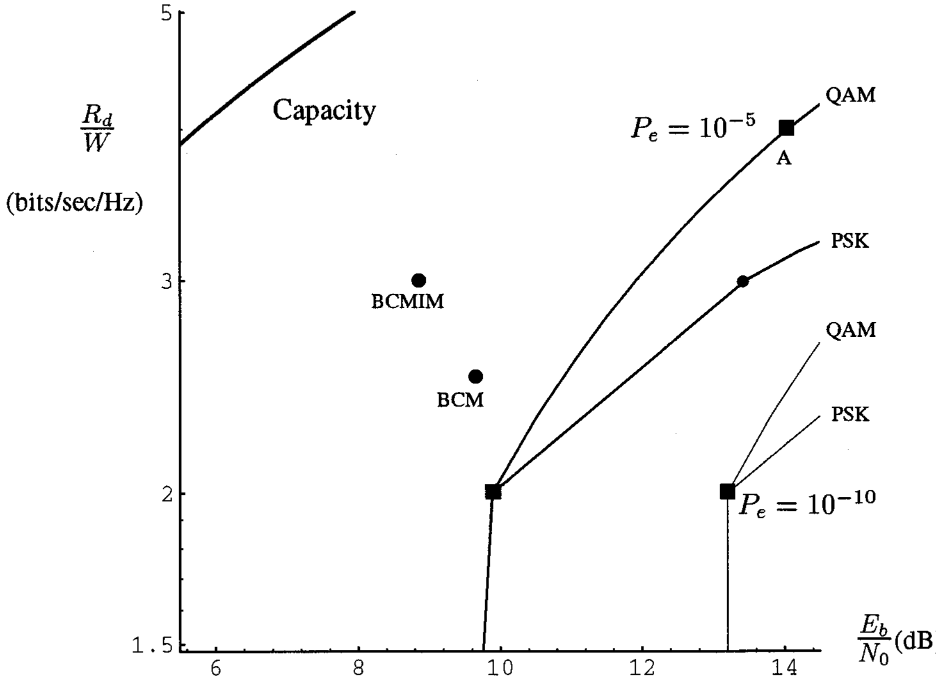


Fig. 1. Performance of BCMIM and BCM systems.

not examine these in this paper, and instead focus on finding codes and evaluating the extra rate achievable by adding interblock memory for a fixed basic block length.

One more comparison is particularly relevant. Codes of basic block length n and interblock memory between the first two basic blocks are effectively length $2n$ codes. Instead of comparing them in performance to the best baseline systems of basic block length n as above, we can compare the resulting BCMIM system to the best baseline BCM system with basic block length $2n$. The reasoning here is that if we take the block length to be a very rough guide to decoding complexity, the BCMIM system will involve decoding a length $2n$ code plus some length n codes, while the baseline system will involve a collection of length $2n$ codes. (Thus the BCMIM system is in some way intermediate in decoding complexity between baseline BCM systems of basic block lengths n and $2n$.)

We discuss this point in the Appendix and simply state the conclusion here: the BCMIM systems in most cases have higher rate, and in the cases where the information is available have fewer apparent nearest neighbors than the longer baseline counterpart. Thus BCMIM can be seen either as adding rate to a baseline system of length n, or as reducing the number of apparent nearest neighbors of a baseline system of length $2n$.

E. Statement of Problem

The above discussion of coding gains motivates the normalized distance coding problem. (We do not translate from normalized distance back to coding gains in the remainder of the paper.) The discussion will be confined to linear codes.

We thus consider the following problem in “classical” coding theory.

Given a nondecreasing sequence $1, a_2, \cdots, a_b$, find binary linear codes of length in and basic block length n, of highest dimension for given minimum normalized distance d_n, where the normalized weight of a codeword is defined as $\sum_{j=1}^{i} a_j w_j$, with w_j the ordinary Hamming weight of the codeword in bits $(j-1)n+1$ to jn.

Among those codes of highest dimension, find the ones with smallest number of codewords of normalized weightd_n.

When discussing codes it is more convenient to speak of dimension, while in comparing overall systems it is convenient to speak of rate; since it is trivial to convert from one to the other we use both terms throughout.

F. Notation

The notation for codes will be of the form

$$[n_1|n_2|\cdots|n_b, k, d_n]_{\boldsymbol{a}}$$

to indicate a code of length $n_1+n_2+\cdots+n_b$, dimension k, and normalized distance, i.e., $w_1 + a_2 w_2 + \cdots + a_b w_b$, at least d_n. The default values of the a_i's are $a_i = 2^{i-1}$ as for quadrature amplitude-modulated (QAM) systems. Thus if the subscript $\boldsymbol{a}$ is omitted, it is implied that $a_i = 2^{i-1}$. Then, for example, $[n|n, k, d_n]$ will indicate a code of length $2n$, dimension k, in which for every nonzero codeword the weight on the left plus twice the weight on the right is at least d_n. The notation $[n, k, d]$ will always refer to a binary linear code of length n, dimension k, and Hamming distance d.

In 8-PSK, we have $a_2 = 3.414$ and $a_3 = 6.828$. We denote this set of a_i's by the subscript $\circ$, so that, for example, $[n|n, k, d_n]_\circ$ denotes a length $2n$ code of dimension k, in which for every nonzero codeword the weight in the first n bits plus 3.4 times the weight in the last n bits is at least d_n.

For an $[n_1|\cdots|n_i, k, d_n]$ code, the *punctured future subcode* is the $[n_i, k_{ip}, d_{ip}]$ code obtained by deleting all nonzero codewords that have weight 0 in the last n_i bits, then deleting the first $n_1+\cdots+n_{i-1}$ bits in all remaining codewords. The punctured past code is defined similarly. The *shortened future subcode* is the $[n_i, k_{is}, d_{is}]$ code obtained by deleting all codewords of the $[n_1|\cdots|n_i, k, d_n]$ code that have nonzero weight in the first $n_1+\cdots+n_{i-1}$ bits. The shortened past subcode is defined similarly.

The notation p_i will denote the shortened past subcode of length i, and similarly f_j will denote the shortened future code of length j. Any linear code of overall length nb may be represented uniquely as $p_i \oplus f_{bn-i} \oplus g_i$, i.e., as the direct sum of the past and future shortened codes and some gluing code g_i. The unsubscripted codes p and f will denote the shortened past subcode of length n and the shortened future subcode of length n, respectively.

III. Bounds and Constructions for Normalized Weight Codes

This section will focus on using the concept of normalized weight to design the code in a BCMIM system with a given distance gain so that the rate of the designed code is maximum. We will consider a general BCMIM system employing a 2^b-ary signal constellation and using a code of length nb with the ith n-bit basic block protecting the ith least significant label bits. We assume the minimum squared intrasubset Euclidean distance within the signal set increases as $1 : a_2 : a_3 : \cdots : a_b$ (a_1 is normalized to 1).

The problem is obviously related to the regular Hamming distance problem, particularly when the interblock memory is between just two basic blocks. However, techniques that are best for Hamming distance are not necessarily best for normalized distance, and *vice versa*. We identify the related Hamming distance construction where applicable.

When we have interblock memory between three or more basic blocks, on the other hand, a simple result (Lemma 1) turns out to be very useful.

A. Upper Bounds on the Maximum Rate Gain in Introducing Interblock Memory to a Baseline BCM System

Lemma 1 (Full Dimension Lemma): The maximum dimension gain obtainable in extending interblock memory from between the first $i-1$ basic blocks to between the first i blocks is upper-bounded by

$$n - K\left(n, \frac{d_n}{a_i}\right)$$

bits.

Proof: More generally, we compare a code with parameters $[n_1|\cdots|n_i, k_i, d_n]$ (with interblock memory between the first i blocks) to the best system with interblock memory extending only to block $i-1$: a direct sum of an $[n_1|\cdots|n_{i-1}, k^*_{i-1}, d_n]$ code of highest dimension and an $[n_i, K(n_i, d_n/a_i), d_n/a_i]$ code. Since shortening the last n_i positions in the $[n_1|\cdots|n_i, k_i, d_n]$ code gives a code with interblock memory between the first $i-1$ basic blocks and normalized distance at least d_n, we must have $k_i - n_i \leq k^*_{i-1}$. The gain obtained by extending the interblock memory to include the ith basic block, $k_i - (k^*_{i-1} + K(n_i, d_n/a_i))$, is then $\leq n_i - K(n_i, d_n/a_i)$ bits. Making all the n_i's equal reduces to the assertion of the lemma. □

Note that the normalizing constants $a_2, \cdots, a_i$ and the normalized distance d_n have not been used, beyond the implicit assumption that they are all positive. Note also that we have not assumed that the code C has been constructed by starting with the best code for the memory-$(i-1)$ case, and then adding a gluing code: this is not guaranteed in general to produce the best memory-i code. We get equality in the above expression, however, only if (but not necessarily if) the past length-$(i-1)n$ subcode obtained by shortening the positions in the last basic block is optimal in dimension for the memory-$(i-1)$ case. A second necessary condition for equality is that the bits in the ith basic block are linearly independent, i.e., the punctured future code of length n is the $[n, n, 1]$ code. Finally, note that since the a_i's form an increasing sequence, the upper bounds on dimension gain follow a "law of diminishing returns" with increasing i.

1) Plotkin Bound: Suppose we have a code with n_1 bits in the first basic block, n_2 in the second, and so on. A straightforward application of the classical Plotkin bound argument, in which we evaluate

$$\sum_{u \in C} \sum_{v \in C} d_n(u, v)$$

via both rows and columns [24, pp. 41–42] gives the result

$$M \leq \frac{d_n}{d_n - \frac{1}{2}(n_1 + a_2 n_2 + \cdots + a_b n_b)}$$

with equality if and only if every nonzero codeword has the minimum normalized weight d_n.

In the case where all the a_i's are integers, we can transform a code with basic block lengths $n_1, n_2, \cdots$ and normalized distance d_n into a code of length $n_1 + a_2 n_2 + \cdots$ and Hamming distance d_n, by replicating each block a_i times. The Plotkin bound above is then simply the regular Plotkin bound applied to the corresponding lengthened code.

We will see later (Section IV) that this gives tight results in many cases when applied to shortened codes.

2) Griesmer-Type Bound: Suppose a codeword of minimum normalized weight has weight distribution $(w_1, w_2, \cdots, w_b)$, i.e., has weight w_i in basic block i for $1 \leq i \leq b$. Then we can obtain a code with basic block lengths $(n_1 - w_1, n_2 - w_2, \cdots, n_b - w_b)$, minimum normalized weight $\geq \lceil d_n/2 \rceil$, and dimension one less. This follows by taking a generator matrix for the original code in which the given minimum-weight codeword is one row; then deleting the given codeword and all the columns in its support.

(Equivalently, we can transfer bits in higher basic blocks to the first basic block, replicating a_i times, to obtain a code with basic block lengths

$$(n_1 + a_2 w_2 + \cdots + a_b w_b, n_2 - w_2, \cdots, n_b - w_b)$$

which contains a codeword of weight d_n whose support is in the first basic block. Since at least half of the other codewords have $\leq \lfloor d_n/2 \rfloor$ ones in these positions, we have the claimed normalized distance in the residual code (cf. [31, p. 46]).)

In the usual situation, we have $n = d_n$ and interblock memory between the first two basic blocks. If we have the codeword $(1_n|0_n)$, then the procedure gives us a $[0|n, k-1, \lceil d_n/2 \rceil]$ code, i.e., an $[n, k-1, \lceil \lceil d_n/2 \rceil / a_2 \rceil]$ binary linear code. A necessary condition then for an $[n|n, k, d_n]$ code to exist, and to contain the codeword $(1_n|0_n)$, is that

$$k \leq 1 + K(n, \lceil \lceil d_n/2 \rceil / a_2 \rceil).$$

The dimension gain is therefore upper-bounded by

$$K(n, \lceil \lceil d_n/2 \rceil / a_2 \rceil) - K(n, \lceil d_n/a_2 \rceil).$$

It may happen that $\lceil \lceil d_n/2 \rceil / a_2 \rceil > \lceil d_n/(2a_2) \rceil$. It is easy to verify that this cannot happen for integer a_2, however. Thus for QAM constellations, the dimension gain by adding interblock memory is upper-bounded by $K(n, d_n/4) - K(n, d_n/2)$ when the code contains the codeword $(1_n|0_n)$.

The most natural way to construct a BCMIM system is based on the corresponding baseline BCM system with highest rate, utilizing the trellis structures of the row codes in the baseline BCM system. In this case, with $n = d_n$, the codeword $(1_n|0_n)$ is always present.

3) Shortening–Puncturing: With interblock memory between the first i blocks, the best code with basic block length n and normalized distance d_n has dimension at most $i-1$ higher than the best code with basic block length $n-1$ and normalized distance $d_n - 1$, since we can puncture the longer code in one position of the first basic block and shorten in one position of each of the other basic blocks. Thus in the case $n = d_n$ with interblock memory between the first two basic blocks, the dimension cannot jump by 2 when we move to basic block length $n+1$.

Note that this relation does not depend on the normalizing constants $a_2, \cdots, a_b$.

4) Split Linear Programming: The most powerful general upper bound for codes with ordinary Hamming distance is the linear programming bound, and a suitable modification provides good results for normalized distance also. Following Jaffe [15], we fix a partition $(p_1, \cdots, p_r)$ of n, and let $C \subset \mathbb{F}_2^n$ be a linear code. Then if $c_{w_1, w_2, \cdots, w_r}$ denotes the number of codewords of C that have weight w_i in partition element p_i for each i, the *split weight enumerator* of C is

$$\sum_{(w_1, \cdots, w_r)} c_{w_1, w_2, \cdots, w_r} t_1^{w_1} t_2^{w_2} \cdots t_r^{w_r}$$

where the t_i's are indeterminates.

Then the split weight enumerator of $C^{\perp}$ is

$$\frac{1}{|C|} \sum_{(w_1, \cdots, w_r)} c_{w_1, w_2, \cdots, w_r} \prod_{j=1}^{r} (1+t_j)^{p_j - w_j} (1-t_j)^{w_j}.$$

Since the coefficients of this polynomial must be nonnegative, we have constraints that can be used with linear programming.

In our situation, the p_i's all have the same size (the basic block length), and we seek the maximum of $\sum c(w_1, \cdots, w_b)$ subject to the linear constraints $c(0, \cdots, 0) = 1$, $c(w_1, \cdots, w_b) = 0$ for all $w_1, \cdots, w_b$ such that $1 \leq w_1 + a_2 w_2 + \cdots + a_b w_b < d_n$, and

$$\sum_{(w_1, \cdots, w_b)} c(w_1, \cdots, w_b) \prod_{j=1}^{b} P_{e_j}(w_j; p_j) \geq 0$$

for all $w_1, \cdots, w_b$ and all $e_1, \cdots, e_b$ with $0 \leq w_j, e_j \leq p_j$, where

$$P_k(w, n) = \sum_{i=0}^{k} (-1)^i \binom{w}{i} \binom{n-w}{k-i}$$

is the usual Krawtchouk coefficient.

B. Lower Bounds on the Maximum Rate Gain in Introducing Interblock Memory to a Baseline BCM System

The extension bound below relates the normalized and Hamming distance problems directly, and is particularly useful when the interblock memory is between the first two basic blocks. We can also apply the standard constructions for producing new codes from old ones from the Hamming distance problem, though the normalized weight criterion alters the relative usefulness of these. We list below the constructions that were found most useful in the main case. The distance bounds follow by elementary adaptation of the reasoning for the ordinary Hamming distance case (see, for example, [24]).

1) Extension Bound: This bound relates the normalized and Hamming distance problems directly. The bound is normally useful only in the case of interblock memory between the first two basic blocks, though it can be extended in obvious ways. Assume that a_2 is an integer dividing both n and d_n. Then if an $[n + n/a_2, k, d/a_2]$ code exists, we can take any n/a_2 bits, replicate them a_2 times, and take the resulting n bits to form the first basic block, with the remaining n bits forming the second block. The normalized distance of this code is clearly d, so we have constructed an $[n|n, k, d]$ code. Note also that the normalized weight enumerator of this code is the same as the Hamming weight enumerator for the original code.

This bound gives the highest dimension possible for $n = 12, 14$, and 16 in the QAM problem (Table I). The bound falls one short in the cases $n = 6, 8$, and 10, and at least one short in the cases $n = 18$ and 20.

2) $|u|u+v|$ Constructions: If C_1 and C_2 are codes of the same length, with parameters $[n, k_1, d_1]$ and $[n, k_2, d_2]$, respectively, then the code consisting of all words of the form $|u+v|u|$, with $u \in C_1, v \in C_2$, has parameters

$$[n|n, k_1 + k_2, d_n \geq \min\{d_2, (1+a_2)d_1, (a_2 - 1)d_1 + d_2\}].$$

A regular $|u|u+v|$ construction does not by itself produce many good codes, since the distance guarantee becomes $d_n \geq \min\{(a_2+1)d_1, a_2 d_2 - (a_2 - 1)d_1\}$, and the negative term tends to rule out good codes.

Another construction is to take codes with parameters $[n_1|n_2, k_1, d_1]$ and $[n_1|n_2, k_2, d_2]$ and to apply a $|u_1, u_1 + v_1|u_2, u_2 + v_2]$ construction, i.e., a regular $|u|u+v|$ construction on each of the halves. This results in a code with parameters $[2n_1|2n_2, k_1 + k_2, d_n \geq \min\{2d_1, d_2\}]$.

TABLE I
CODES FOR QAM-BASED SYSTEMS

n, d_n	i	Max. rate gain, bits/n symbols					Opt. rate gain	k	N_{min} (N_{min+1})
		Upper bounds			Lower bound				
		Full rate	Griesmer[a]	Split LP	$X, rX, X4$	Expansion			
6	2	3	2	2	1	1	2	6	
	3	1	-	-	-		1	12	
	4	0	-	-	-		0	18	
7	2	4	3	3	3	2	3	7	22
	3	1	-	-	-		1	14	22-29
	4	0	-	-	-		0	21	22-29
8	2	4	3	3	3	2	3	8	71
	3	1	-	-	1		1	16	87
	4	0	-	-	0		0	24	87
9	2	7	3	4	3	3	4	7	0 (31-33)
	3	4	-	-	-		4	16	0 (31-56)
	4	1	-	-	-		1	25	0 (31-56)
	5	0	-	-	-		0	34	0 (31-56)
10	2	7	3	4	3	3	4	8	6-12
	3	4	-	-	-		4	18	6-68
	4	1	-	-	-		1	28	6-68
	5	0	-	-	-		0	38	6-68
11	2	8	4	5	3	4	4	8	0 (24–55)
	3	4	-	-	-		4	19	0 (24-198)
	4	1	-	-	-		1	30	0 (24-198)
	5	0	-	-	-		0	41	0 (24-198)
12	2	8	4	5	3	4	4	9	6-60
	3	4	-	-	-		4	21	6-214
	4	1	-	-	-		1	33	6-214
	5	0	-	-	-		0	45	6-214
13	2	10	5	6	3	5	5	9	0 (26–96)
	3	5	-	-	-		5	22	
	4	1	-	-	-		1	35	
	5	0	-	-	-		0	48	
14	2	10	5	7	5	5	5	10	9-34
	3	5	-	-	-		5	24	9-113
	4	1	-	-	-		1	38	9-113
	5	0	-	-	-		0	52	9-113
15	2	11	6	6	4	6	6	11	39-136
	3	5	-	-	-		5	26	39-310
	4	1	-	-	-		1	41	39-310
	5	0	-	-	-		0	56	39-310
16	2	11	6	6	4	6	6	12	271-503
	3	5	-	-	4		5	28	271-1671
	4	1	-	-	1		1	44	271-1671
	5	0	-	-	-		0	60	271-1671

[a]If the code contains $(1_n 0_{xn-n})$.

3) X and Reverse-X Constructions: The X construction [24, p. 581] takes codes $C_1 \sim [n_1, k_1, d_1]$, $C_2 \sim [n_1, k_2, d_2]$, with $C_1 \subset C_2$ and $C_3 \sim [n_3, k_2 - k_1, d_3]$. Then the new code consists of the words $|x_i + u|y_i|$, where the x_i's are coset representatives of C_2 in C_1, $u \in C_2$, and $y_i \in C_3$.

This code has normalized parameters $[n_1|n_3, k_2, \geq \min\{d_1, d_2 + a_2 d_3\}]$.

In a reverse-X construction, we interchange the blocks in the generator matrix above to get a code with normalized parameters $[n_3|n_1, k_2, \geq \min\{a_2 d_1, a_2 d_2 + d_3\}]$.

4) $X4$ Construction: This construction takes four codes $C_i \sim [n_i, k_i, d_i]$ with C_2 a union of b distinct cosets of C_1, with coset representatives $x_1, \cdots, x_b$, and C_4 a union of b distinct cosets of C_3, with coset representatives $y_1, \cdots, y_b$. The new code consists of all words of the form $|x_i + u|y_i + v|$, where $u \in C_1$, $v \in C_3$. The x_i's and y_i's can be paired to make the resulting code linear. When this is done, the resulting generator matrix can be taken to have the form

$$\left[\begin{array}{c|c} G_1 & 0 \\ \hline G_{2/1} & G_{4/3} \\ \hline 0 & G_3 \end{array}\right].$$

where $C_2 = \langle G_1, G_{2/1} \rangle$ and $C_4 = \langle G_3, G_{4/3} \rangle$ [29].

The normalized parameters are

$$[n_1|n_3, k_2+k_3, \geq \min\{d_1, a_2d_3, d_2+a_2d_4\}].$$

IV. Best BCMIM Codes

In this section we find a large number of best codes for normalized distance. We concentrate on the natural case in which the normalized distance is equal to the basic block length. The results are summarized in Table I.

A. Best Codes for QAM-Based BCMIM Systems with Basic Block Length 8

The codes in the baseline BCM system will be $[8, 1, 8]$, $[8, 4, 4]$, $[8, 7, 2]$, and $[8, 8, 1]$ codes, as given by Cusack [6]. The case $n = d_n = 8$ is a main focus of the papers of Lin *et al.*Lin and Ma give an $[8|8, 8, 8]$ code [20] and Lin, Wang, and Ma follow by extending this to an $[8|8|8, 16, 8]$ code [21]. These papers give efficient trellis representations and performance simulations, but do not discuss general constructions, bounds, or optimality.

1) Best $[8|8, k, 8]$ *Codes:* If we allow interblock memory between the first two basic blocks, we are seeking the best $[8|8, k, 8]$ code. The baseline BCM case with no interblock memory is obtained by taking the direct sum of the $[8, 1, 8]$ code and the $[8, 4, 4]$ code, and so has dimension 5.

From the full-dimension lemma, the gain in dimension when interblock memory is introduced is upper-bounded by

$$n - K(n, d_n/a_2) = 8 - 4 = 4.$$

The first necessary condition for equality in this bound is that the shortened past subcode p has highest possible dimension, i.e., that $k_p = 1$. This implies that the codeword $(1_8|0_8)$ would be in the code, so that we can apply the Griesmer argument, from which the maximum rate gain is upper-bounded by

$$K(n, d_n/(2a_2)) - K(n, d_n/a_2) = K(8, 2) - K(8, 4) = 3$$

i.e., we have $k \leq 8$ for an $[8|8, k, 8]$ code.

This upper bound can be achieved using the code discussed by Lin and Ma [20] with generator matrix $G_{8/2}$ (see Table II).

This code can be obtained from an $X4$ construction, with C_1 an $[8, 1, 8]$ code, C_2 and C_3 $[8, 4, 4]$ codes, and C_4 an $[8, 7, 2]$ code. (Lin and Ma [20] cite [29], which introduces the $X4$ construction.)

(None of the other constructions of Section III give $k = 8$. The expansion construction gives $k = 7$, the three constructions based on the $|u|u+v|$ method give dimensions $5, 6$, and 7, respectively, and the X and reverse-X constructions give dimensions 5 and 7, respectively.)

This code has split weight enumerator

$$1+x^8+56x^4y^2+14y^4+112x^4y^4+14x^8y^4+56x^4y^8+y^8+x^8y^8$$

and thus has 71 codewords of minimum normalized weight (one of type 8-0, 56 of type 4-2, and 14 of type 0-4). This is the minimum number possible. In fact, it is also the maximum number possible: this code is (up to permutations) the unique $[16, 8]$ code with normalized distance 8.

TABLE II
Generator Matrix for $[8|8, 8, 8]$ Code

$$G_{8/2} = \begin{bmatrix} G_c \\ G_b \;\; 0 \\ 0 \;\; G_a \end{bmatrix} = \left[\begin{array}{cccccccc|cccccccc}
1&0&0&0&1&0&1&1&1&0&0&0&0&0&0&1\\
1&0&0&1&0&1&1&0&1&0&0&0&0&0&1&0\\
1&0&1&0&1&1&0&0&1&0&0&0&0&1&0&0\\
\hline
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0\\
\hline
0&0&0&0&0&0&0&0&1&0&0&0&1&0&1&1\\
0&0&0&0&0&0&0&0&1&0&0&1&0&1&1&0\\
0&0&0&0&0&0&0&0&1&0&1&0&1&1&0&0\\
0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1
\end{array}\right]$$

The uniqueness follows from the following reasoning. We have $C = p \oplus f \oplus g$, where p and f are the shortened past and future subcodes of length n, and g is an appropriate gluing code. We must have a p of dimension 1, for if not we would have $k(f)+k(g) = 8$, could take a generator matrix for the length 16 code that had an identity matrix in the right half, and would need eight rows of weight $\geq$6 and each distance $\geq$4 apart in the left half of the generator matrix. But $A(8, 4, 6) = 4$ [24, p. 684] so this is impossible. Hence $k(p) = 1$, and $k(f)+k(g) = 7$. Since the shortened past subcode has dimension 1 and length equal to the target normalized distance, the overall code must contain the codeword $(1_8|0_8)$. Then for every nonzero sequence on the right, there corresponds at least one sequence on the left of weight $\leq$4. Thus we must have at least two nonzero bits on the right in every nonzero codeword, so the punctured future code $f \oplus g$ must be an $[8, 7, 2]$ even-weight code. Taking a systematic generator matrix for this, together with the codeword $(1_8|0_8)$, we need seven more rows of length 8, weight 4, and distance exactly 4 apart for the left of the generator matrix. These can only be obtained by taking the 14 codewords of weight 4 from an $[8, 4, 4]$ Hamming code and selecting one word out of every complementary pair. All such choices lead to the same code up to permutation.

We note that the best $[16, 8]$ code in terms of ordinary Hamming distance, a $[16, 8, 5]$ shortened quadratic residue code, also unique [28], cannot by the above discussion be partitioned into two halves to get normalized distance 8 (though 7 is possible). This shows that we cannot solve the normalized distance problem in general by simply taking the best (linear) code for ordinary Hamming distance and finding the best partition for this code. (It is possible to partition the nonlinear $(16, 256, 6)$ code to get an $(8|8, 256, 8)$ code with 57 minimum normalized weight codewords, by taking the first 8 bits to be any eight bits that form the support of a codeword.)

2) Best $[8|8|8, k, 8]$ *Codes:* The baseline $[8|8|8, k, 8]$ code is obtained by a direct sum of an $[8|8, k_2, 8]$ code for the first two basic blocks and an $[8, 7, 2]$ code for the third basic block. Thus using the above code, we can achieve dimension 15 without extending the interblock memory to include the third basic block.

The full-dimension lemma indicates that the dimension of any $[8|8|8, k_3, 8]$ code will be at most 8 more than the dimension of the best $[8|8, k_2, 8]$ code. Thus with interblock memory extended to the third basic block, we can potentially increase the dimension by at most 1.

This can, in fact, be achieved by the code considered by Lin *et al.* [21] in which the $[8|8, 8, 8]$ code for the first two basic blocks and the $[8, 7, 2]$ baseline code for the third basic block are glued by the word

$$[0\,0\,0\,0\,0\,0\,1\,1\,|\,0\,0\,0\,0\,0\,0\,0\,1\,|\,0\,0\,0\,0\,0\,0\,0\,1].$$

This code has the disadvantage that it has 131 codewords of minimum normalized weight. This can be reduced to a minimum of 87 with no loss in rate, though at the cost of increasing the dimension of the gluing code. (Other things being equal, a smaller dimension gluing code is to be preferred, as the decoder is then usually less complex.) We seek the minimum number of codewords of normalized weight 8 in any linear code with basic block length 8 and length 24 with dimension 16. The past shortened subcode p_{16} must be the unique $[8|8, 8, 8]$ linear code given above, and the future f_8 subcode must have the 8×8 identity matrix as generator matrix. Suitable row operations can reduce any candidate generator matrix to the form

$$\left[\begin{array}{c|c|c} \multicolumn{2}{c|}{G_{8/2}} & \mathbf{0} \\ \hline X & \mathbf{0} \quad \begin{matrix}1\\0\end{matrix} & I_8 \end{array}\right]$$

where $\left[\frac{1}{0}\right]$ denotes a single column of 1's and 0's. The rows of X have weight ≤ 4 without loss of generality, since $G_{8/2}$ contains the codeword $(1_8|0_8)$. The rows of X that are paired with rows of weight 0 in the middle must have weight exactly 4, and the others have weight 2-4. If one row of X has weight 3, then adding to a suitably chosen codeword of $G_{8/2}$ we would get a codeword with weights $(1|1|1)$, i.e., normalized weight 7, thus ruling out this possibility. Also any row of weight 4 of X must be at distance 2 from the $[8, 4]$ Hamming code at the left of $G_{8/2}$. We conclude that without loss of generality the rows of X have weight either 2 or 4, and the last eight rows of the overall matrix have weights of the form $(2|1|1)$, $(4|1|1)$, or $(4|0|1)$.

The rows of type $(2|1|1)$ each contribute four codewords of minimum normalized weight if they are distinct in the left eight components, and more otherwise. (Each word of weight 2 is distance 2 from exactly three codewords of H_8, providing three minimum normalized weight codewords, plus the row itself.) Similarly, the rows of type $(4|1|1)$ each provide four such codewords if distinct in the left eight components, and more otherwise. The rows of type $(4|0|1)$ each contribute two such codewords if distinct from each other: the row itself, and the row plus the word $(1_8|0_{16})$. We therefore fill all rows of X with words of weight 4 that are distinct from each other and their converses and the words of the $[8, 4]$ Hamming code above (there are more than enough to make this possible) and get a code with 87 minimum normalized weight codewords (71 from $G_{8/2}$ and 16 more from the rows involving X).

A generator matrix that results from this procedure is given in Table III.

Together with eight uncoded bits for the fourth basic block, we can build a BCMIM system with the parameters $[8|8|8|8, 24, 8]$. This is therefore a 16-QAM system that transmits 24 information bits per eight symbols.

TABLE III
GENERATOR MATRIX FOR $\mathbb{C}_{8/3}^{\min}$

$$\left[\begin{array}{c|c|c}
\multicolumn{2}{c|}{1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0} & 0\,0\,0\,0\,0\,0\,0\,0 \\
\multicolumn{2}{c|}{1\,0\,0\,0\,1\,0\,1\,1\,1\,0\,0\,0\,0\,0\,0\,1} & 0\,0\,0\,0\,0\,0\,0\,0 \\
\multicolumn{2}{c|}{1\,0\,1\,0\,1\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,1} & 0\,0\,0\,0\,0\,0\,0\,0 \\
\multicolumn{2}{c|}{0\,0\,1\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,1} & 0\,0\,0\,0\,0\,0\,0\,0 \\
\multicolumn{2}{c|}{1\,0\,1\,1\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,0\,1} & 0\,0\,0\,0\,0\,0\,0\,0 \\
\multicolumn{2}{c|}{1\,0\,0\,1\,0\,1\,1\,0\,0\,0\,0\,0\,1\,0\,0\,1} & 0\,0\,0\,0\,0\,0\,0\,0 \\
\multicolumn{2}{c|}{0\,0\,1\,0\,0\,1\,1\,1\,0\,0\,0\,0\,0\,1\,0\,1} & 0\,0\,0\,0\,0\,0\,0\,0 \\
\multicolumn{2}{c|}{0\,0\,0\,1\,1\,1\,0\,1\,0\,0\,0\,0\,0\,0\,1\,1} & 0\,0\,0\,0\,0\,0\,0\,0 \\
\hline
0\,1\,0\,0\,1\,0\,1\,1 & 0\,0\,0\,0\,0\,0\,0\,0 & 1\,0\,0\,0\,0\,0\,0\,0 \\
0\,0\,1\,0\,1\,0\,1\,1 & 0\,0\,0\,0\,0\,0\,0\,0 & 0\,1\,0\,0\,0\,0\,0\,0 \\
0\,0\,0\,1\,1\,0\,1\,1 & 0\,0\,0\,0\,0\,0\,0\,0 & 0\,0\,1\,0\,0\,0\,0\,0 \\
0\,0\,0\,0\,1\,1\,1\,1 & 0\,0\,0\,0\,0\,0\,0\,0 & 0\,0\,0\,1\,0\,0\,0\,0 \\
1\,1\,0\,0\,0\,0\,1\,1 & 0\,0\,0\,0\,0\,0\,0\,0 & 0\,0\,0\,0\,1\,0\,0\,0 \\
1\,0\,1\,0\,0\,0\,1\,1 & 0\,0\,0\,0\,0\,0\,0\,0 & 0\,0\,0\,0\,0\,1\,0\,0 \\
1\,0\,0\,1\,0\,0\,1\,1 & 0\,0\,0\,0\,0\,0\,0\,0 & 0\,0\,0\,0\,0\,0\,1\,0 \\
1\,0\,0\,0\,0\,1\,1\,1 & 0\,0\,0\,0\,0\,0\,0\,0 & 0\,0\,0\,0\,0\,0\,0\,1
\end{array}\right]$$

Making these most significant label bits uncoded adds eight minimum normalized weight codewords, whereas adding the most significant label bits to the least significant label bits adds none. Thus with interblock memory between the first four basic blocks and a distance gain of 8, we have a maximum of 24 information bits and a minimum of 87 codewords of normalized weight 8. (The full-dimension lemma indicates that we cannot hope to gain dimension by extending the interblock memory to include the fourth basic block; we can still improve the code in the sense of reducing the number of minimum normalized weight codewords.)

B. Best Codes for QAM-Based BCMIM Systems with Basic Block Length 9

The codes in a baseline BCM system are $[9, 1, 9]$, $[9, 2, 6]$, $[9, 5, 3]$, and $[9, 8, 2]$.

1) Best $[9|9, k, 9]$ *Codes:* With interblock memory between the first two blocks, we seek the best $[9|9, k, 9]$ code. A direct sum of the $[9, 1, 9]$ and $[9, 2, 6]$ codes gives $k = 3$.

Applying split weight linear programming to this case indicates that $k \leq 7$. On fixing the number of codewords accordingly at 128 and using split weight linear programming to minimize the number of codewords with normalized weight 9, we find that the minimum of the objective is 0. This suggests that we might be able to find a $[9|9, 7, 10]$ code. This is in fact possible, and we will give a couple of different constructions.

This case is also interesting in that it is the first case in which the Griesmer-type bound of Section III-A2 is not tight (see Table I).

Since the Griesmer argument indicates that $k \leq 6$ if $k_p = 1$, we must have $k_p = 0$, and so $k_f + k_g = 7$. The punctured future code C_{fg} is then a $[9, 7]$ code, and its dual $C_{fg}^{\perp}$ is a $[9, 2, \leq 6]$ code. Shortening the overall code in any set of m positions holding a codeword of $C_{fg}^{\perp}$ gives a code with $n_1 = 9$, $n_2 = 9 - m$, and $k_s \geq 7 - m + 1$ (this is construction Y1 from MacWilliams and Sloane [24, p. 592]). Suppose that $C_{fg}^{\perp}$ had minimum distance ≤ 5. Then applying Y1 with $m = 5$ gives a code with $n_1 = 9$, $n_2 = 4$, and $k \geq 3$. But the Plotkin bound for this case gives $M \leq 10/(10 - (9/2 + 4)) < 7$. Thus we conclude that $C_{fg}^{\perp}$ is the unique $[9, 2, 6]$ linear code.

We can take the generator matrix for C_{fg} to be

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

Now applying Y1 again, with $m = 6$, we get a code with $n_1 = 9$, $n_2 = 3$, and $k \geq 2$. Applying the Plotkin bound to this case, we find that $M \leq 10/2.5 = 4$; thus the shortened code would meet the Plotkin bound with equality. This in turn would imply that every nonzero codeword has the minimum nonzero normalized weight, as noted earlier. This means that in each of the rows of weight 2 on the right, we would have to have a row of weight exactly 6 on the left. This further narrows the possibilities to generator matrices where, with the above matrix on the right, rows 2 and 3 on the left generate a $[9, 2, 6]$ linear code, as do rows 4 and 5, and rows 6 and 7.

There are at least two inequivalent codes that satisfy these constraints. The first is obtained by a version of Piret's construction [24, pp. 588–589] applied to the $[9, 6, 2]$ irreducible cyclic code. We first construct the code consisting of 0 and all words of the form

$$u_j = |\gamma(x)^j \theta_1(x)|\gamma(x)^{j+a}\theta_1(x)|$$

for $0 \leq j \leq 62$, where $\theta_1(x) = x^6 + x^3$ is the idempotent of the given irreducible code, and $\gamma(x) = x^8 + x^6 + x^4 + x^3$. A generator matrix for this code can be taken as having rows $u_j, 0 \leq j \leq 5$. The key property is that all nonzero codewords are either rows or permutations of rows of the generator matrix, and so we need only select the remaining parameter a to maximize the minimum distance of the rows of the generator matrix

$$d' = \min_{0 \leq j \leq 5} (w_j + 2w_{j+a})$$

where $w_j = \mathrm{wt}\,(\gamma(x)^j \theta_1(x))$. We have $(w_0, w_1, \cdots, w_6) = (2, 6, 6, 4, 6, 4, 4)$, and $w_{j+7} = w_j$ for all j, so the (unique) best choice for a in this case is 3, giving $d' = 10$. (The factor 2 on the right of the expression for d' and the corresponding optimal choice of a are the only differences with the development in [24].) This results in a $[9|9, 6, 10]$ code, and adding the row $(0_9|1_9)$ to the generator matrix does not affect the minimum normalized distance, thus giving a $[9|9, 7, 10]$ code. (A brute-force search showed that adding the row $(r_l|1_9)$ for any $r_l \neq 0_9$ gives a lower minimum normalized distance.) From the construction, taking the same cyclic shift of the left and right sides of a codeword produces a codeword. The codewords of minimum normalized weight come in sets of 9, and there are four such sets, with weights 2-4, 6-2, 4-3, and 4-3, thus producing 36 codewords of minimum normalized weight. A generator matrix for this code, $G_{9/2}^{(1)}$, is shown at the bottom of this page.

A slightly smaller number of minimum normalized weight codewords can be obtained from an optimal $[18, 7, 7]$ code partitioned in an appropriate way. The generator matrix, $G_{9/2}^{(2)}$, that results is also shown below, and has 33 codewords of minimum normalized weight. (This code has more codewords of the next higher weight than the code based on Piret's construction, however; 36 codewords of normalized weight 12, versus 27.) The even (Hamming) weight subcode obtained by deleting the first row is an $[18, 6, 8]$ code. It is known [7] that there are exactly two such codes up to equivalence, both obtainable by shortening the $[24, 12]$ Golay code. The corresponding Hamming weight enumerators are $1 + 46x^8 + 16x^{12} + x^{16}$ and $1 + 45x^8 + 18x^{12}$. The code generated by the last six rows of the matrix above has the second of these weight enumerators. There is no way to partition and augment the first type to get a $[9|9, 7, 10]$ code. (For ex-

$$G_{9/2}^{(1)} = \left[\begin{array}{ccccccccc|ccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{array}\right]$$

$$G_{9/2}^{(2)} = \left[\begin{array}{ccccccccc|ccccccccc} 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{array}\right]$$

ample, split linear programming shows that no $[9|9,7,10]$ code can contain a codeword with Hamming weight 16.)

Split linear programming, constraining the code to have 128 codewords, normalized distance 10, and to have a $[9,2,6]$ code as dual of the punctured future code gives a lower bound of 31 codewords of minimum normalized weight. We remark that in the case of a $[9|9,7,10]$ code, there is a particularly large number of solutions to the linear programming constraints that are feasible candidates for codes, in the sense of having nonnegative integer weights in both code and dual code. Many are ruled out by the extra constraint that the punctured future code is the dual of the $[9,2,6]$ code. We do not know of any such code with 31 codewords of minimum normalized weight, though an integer weight enumerator, with integer dual weight enumerator, exists.

The Golay-based code can be extended to form $[9|9|9,16,10]$ and $[9|9|9|9,25,10]$ codes with 56 minimum normalized weight codewords, as indicated in Table I. We omit the details.

C. Best Codes for QAM-Based BCMIM Systems with Basic Block Length 10

The codes in a baseline BCM system are $[10,1,10]$, $[10,3,5]$, $[10,6,3]$, and $[10,9,2]$.

1) Best $[10|10,k,10]$ *Codes:* With no interblock memory between the first two blocks, the baseline BCM system gives $k=4$. Split linear programming gives $k\leq 8$, and this bound can easily be achieved by modifying the $[24,12]$ Golay code. Partitioning the Golay code so that each half holds a codeword of weight 12, and then shortening two positions on each side gives a $[10|10,8,10]$ code, where the normalized distance follows because no codeword of weight 8 in the Golay code can split 8-0 and every codeword has even weight on each side. This code has 20 codewords of minimum normalized weight.

A slightly more complicated construction allows us to reduce the number of normalized weight 10 codewords. We partition the Golay code in such a way that each side forms an X_{12}, in the notation of [5]. Each side contains exactly three codewords of weight 8: $(1_8 0_4)$, $(1_4 0_4 1_4)$, and $(0_4 1_8)$ up to permutation. Shortening in two appropriately chosen bits (e.g., bits 1 and 5) eliminates these three codewords and gives a $[10|10,8,10]$ code with $N_{\min}=12$. Split integer programming gives a lower bound of 6 on the number of minimum normalized weight codewords.

2) Best $[10|10|10,k,10]$ *Codes:* The full-dimension lemma indicates that the dimension is at most 10 greater than in the case above, i.e., $k\leq 18$. This upper bound is achievable. We form a $[10|10,8,10]$ code by partitioning the Golay code into two U_{12}'s, then shortening in two positions on either side. The punctured past and future codes are both $[10,8,2]$ codes in which one of the cosets has weight enumerator $25x^2+100x^4+110x^6+20x^8+x^{10}$. We choose any ten distinct words $x_1,\cdots,x_{10}$ of weight 8 from this coset, and form the generator matrix

$$\left[\begin{array}{c|c|c} G_1\ [10,8,2] & G_2\ [10,8,2] & 0 \\ \hline \begin{matrix}x_1\\ x_2\\ \vdots\\ x_{10}\end{matrix} & 0 & I_{10} \end{array}\right].$$

This code has minimum normalized weight 10 and 68 codewords of this normalized weight: 18 with weight distribution $2|0|2$, 30 with weight distribution $2|2|1$, and 20 from the $[10|10,8,10]$ shortened past subcode.

For four basic blocks, we can again achieve the bound of the full-dimension lemma, obtaining a code with parameters $[10|10|10|10,28,10]$. This is achieved by the code with generator matrix

$$\left[\begin{array}{c|c|c|c} G_1\ [10,8,2] & G_2\ [10,8,2] & 0 & 0 \\ \hline \begin{matrix}x_1\\ x_2\\ \vdots\\ x_{10}\end{matrix} & 0 & I_{10} & 0 \\ \hline 0 & \begin{matrix}y_1\\ y_2\\ \vdots\\ y_{10}\end{matrix} & 0 & I_{10} \end{array}\right]$$

where $y_1,\cdots,y_{10}$ are distinct words of weight 2 from the same coset as the x_i's. No codeword with positive weight in the fourth block has normalized weight lower than 12. The code therefore has minimum normalized weight 10 with 68 codewords of this normalized weight.

D. Best Codes for QAM-Based BCMIM Systems with Basic Block Lengths 11 *and* 12

The codes in the baseline systems are $[11,1,11]$, $[11,3,6]$, $[11,6,4]$, and $[11,10,2]$; and $[12,1,12]$, $[12,4,6]$, $[12,8,3]$, and $[12,11,2]$, respectively. The systems without interblock memory therefore give $[11|11,4,11]$, $[11|11|11,10,11]$, and $[11|11|11|11,20,11]$ codes for basic block length 11, and $[12|12,5,12]$, $[12|12|12,13,12]$, and $[12|12|12|12,24,12]$ codes for basic block length 12.

The case $n=d_n=11$ is interesting as the first case in which the split linear programming bound on dimension is not tight when $i=2$. Split linear programming gives a maximum of 512 codewords even when the c's are constrained to be integers, suggesting that a dimension gain of five might be possible. However, if such a (linear) code existed, it would have a past shortened subcode p of dimension 0 by the Griesmer argument, so the punctured future code would be an $[11,9]$ code. The dual would be an $[11,2,\leq 7]$ code, and a Y1 shortening would give an $[11|4,3,11]$ code, which the Plotkin bound shows cannot exist.

This also rules out a basic block length 12, normalized distance 12, code with interblock memory between the first two blocks and dimension 10 (again, split linear programming gives maximum objective exactly 1024), since the optimum k cannot jump by two by the shortening/puncturing argument.

We can however construct codes with the next higher dimensions, i.e., $[12|12,9,12]$ and (by shortening/puncturing) $[11|11,8,11]$ codes. We use a reverse-X construction. Taking a $[15,4,8]$ simplex code and puncturing in three positions that hold a codeword in the dual code gives a $[12,4,6]$ code with weight enumerator $1+12x^6+3x^8$, which we use as both C_1 and C_3. Taking the dual of this simplex code and shortening in the same three positions gives a $[12, 8, 3]$ shortened Hamming code C_2 that contains C_1 as a subcode. Applying the reverse-X construction gives a $[12|12, 8, 12]$ code; we need to augment this to get a $k = 9$ code.

If we augment by adding the codeword $(1_{12}|0_{12})$, the punctured past code has weight enumerator

$$1 + 3x^4 + 24x^6 + 3x^8 + x^{12}.$$

The only potential problem is if codewords of minimum weight in each half are paired, i.e., if we get a 4-3 break in some codeword. Since there are only three codewords of minimum weight in the punctured past code, we can avoid this, choosing a generator matrix that pairs the three words of weight 4 on the left with words of weight ≥ 4 on the right.

One generator matrix that results from this procedure is shown at the bottom of this page, and has 60 codewords of minimum normalized weight ($a_{12,0} = 1$; $a_{0,6} = 12$; $a_{4,4} = 15$; and $a_{6,3} = 32$). Split integer programming gives a minimum of 6. (Constraining k_p to be 1, as in the code above, gives a minimum of 29 codewords of minimum normalized weight.)

One extra property of this code will be used below. Codewords of odd weight in the right basic block must involve the last row of the generator matrix. Since the last row on the left is distance exactly 6 from all linear combinations of the rows above it, we see that codewords with odd weight in the right basic block have weight exactly 6 in the left basic block.

1) An $[11|11, 8, 12]$ *Code:* We can obtain an $[11|11, 8, 11]$ code from the $[12|12, 9, 12]$ code via shortening/puncturing. However, it is possible to increase the minimum normalized distance to 12 without sacrificing rate. One such code is obtained by taking the $[24, 12, 8]$ Golay code and partitioning to get an $[8|16, 12, 8]$ code with the codeword $(1_8|0_{16})$. All other codewords of weight 8 from the original Golay code have distribution 4-4, 2-6, or 0-8 across the basic blocks and thus have normalized weight at least 12. Shortening in one position on the left therefore gives a $[7|16, 11, 12]$ code. Shortening in three positions on the right gives a $[7|13, 8, 12]$ code, and then transferring two bits from right to left and doubling gives an $[11|11, 8, 12]$ code, with 55 minimum normalized weight codewords.

One can verify that although this procedure involves the choice of six positions, the normalized weight enumerator of the resulting code does not depend on the choices.

There are at least two more inequivalent $[11|11, 8, 12]$ codes, though none are known that have as few minimum normalized weight codewords. Split integer programming gives a lower bound of 24 on the number of minimum normalized weight codewords.

The code described above can be extended to form $[11|11|11, 19, 12]$ and $[11|11|11|11, 30, 12]$ codes, meeting the upper bound of the full dimension lemma in each case, with 198 codewords of minimum normalized weight. We omit the details.

2) Best $[12|12|12, k, 12]$ *and* $[12|12|12|12, k, 12]$ *Codes:* We will use the same approach to construct a $[12|12|12, 21, 12]$ code as in the basic block length 10 case. The $[12, 4, 6]$ punctured simplex code above has 48 cosets with weight enumerator

$$5x^4 + 7x^6 + 3x^8 + x^{10}.$$

We choose $x_1, \cdots, x_{12}$ to be words of weight 4 from distinct cosets among these. We form the generator matrix

$$\left[\begin{array}{c|c|c}
G_1\ [12, 1, 12] & 0 & 0 \\ \hline
G_2\ [12, 4, 6] & G_4\ [12, 4, 3] & 0 \\ \hline
0 & G_3\ [12, 4, 6] & 0 \\ \hline
 & x_1 & \\
0 & x_2 & I_{12} \\
 & \vdots & \\
 & x_{12} &
\end{array}\right].$$

The x_i's are distance 2 from each other, and distance 4 from C_3. They are distance at least 1 from $\langle G_3, G_4\rangle$. Also codewords of odd weight in $\langle G_3, G_4\rangle$ are paired with words of weight exactly 6 in the first basic block. Then the weight distributions between basic blocks are of the form $(0|{\geq}4|1)$, $(0|{\geq}2|2)$, $({\geq}6|1|1)$, $({\geq}4|{\geq}2|1)$, and $(|\cdot|{\geq}3)$. The normalized weight is thus at least 12. There are 214 codewords of minimum normalized weight.

$$\left[\begin{array}{cccccccccccc|cccccccccccc}
1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&1&1&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&1&0&1&0&0&1&1&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&0&1&0&0&0&0&0&0&0&1&1\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&0&0&0&1&0&0&0&0&0&1
\end{array}\right]$$

To form a $[12|12|12|12,k,12]$ code, we take a generator matrix of the form

$$\left[\begin{array}{c|c|c|c} G_1\ [12,1,12] & 0 & 0 & 0 \\ \hline G_2\ [12,4,6] & G_4\ [12,4,3] & 0 & 0 \\ \hline 0 & G_3\ [12,4,6] & 0 & 0 \\ \hline 0 & \begin{matrix} x_1 \\ x_2 \\ \vdots \\ x_{12} \end{matrix} & I_{12} & 0 \\ \hline \begin{matrix} y_1 \\ y_2 \\ \vdots \\ y_{12} \end{matrix} & 0 & 0 & I_{12} \end{array}\right].$$

The words $y_1, \cdots, y_{12}$ are words of weight 6, each from distinct cosets with weight enumerator

$$x^2 + 8x^4 + 14x^6 + 8x^8 + x^{10}$$

of the $[12,5,4]$ code $\langle C_2, \underline{1} \rangle$ above. The only case that needs to be checked is the case $(y_i|0|0|e_i) + (v|u|0|0)$, which has weight distribution $(\geq 2|\geq 3|0|1)$. Thus none of the codewords with ones in the fourth basic block have normalized weight as low as 12. The code is thus $[12|12|12|12,33,12]$ with 214 codewords of normalized weight 12.

E. Best Codes for QAM-Based BCMIM Systems with Basic Block Lengths 13–15

The codes of highest dimension with basic block lengths in this range can all be obtained from the $n = 16$ case below via shortening and puncturing. There is one case that is not covered: the optimum dimension for a $[13|13,k,13]$ code is 9. However, the normalized distance can be increased to 14 without sacrificing rate.

To construct such a code, we begin with the (unique) $[19,8,7]$ code. We apply the expansion construction, taking five positions and doubling them for the left block, and leaving the other 14 for the right block. This produces a $[10|14,8,14]$ code. We then augment by adding to the generator matrix a word with a 1 in an eleventh position on the left, a 10 in each pair of doubled positions on the left, and a word y in the rightmost 14 positions that is a coset leader in the $[14,8]$ punctured future code of the $[10|14,8,14]$ code above. The augmented word is normalized distance at least $6 + 2\,\mathrm{wt}\,y$ from codewords of the $[10|14,8,14]$ code. We therefore choose the five positions from the $[19,8,7]$ code to double so that the $[14,8,4]$ punctured future code has covering radius 4. Equivalently we seek to puncture the $[19,8,7]$ code in five positions, in such a way that the resulting $[14,8]$ code has covering radius at least 4. Some trial and error shows that this is possible. Thus we obtain an $[11|14,9,14]$ code, and transferring one bit from right to left and doubling gives a $[13|13,9,14]$ code. One such generator matrix is shown at the bottom of this page.

We obtain a code with 96 codewords of minimum normalized weight. Split linear programming gives a lower bound of 26 such codewords in any $[13|13,9,14]$ code.

F. Best Codes for QAM-Based BCMIM Systems with Basic Block Length 16

The codes in the baseline BCM system will be $[16,1,16]$, $[16,5,8]$, $[16,11,4]$, and $[16,15,2]$.

1) Best $[16|16,k,16]$ *Codes:* Either split linear programming or the Y1/Plotkin argument show that $k \leq 12$. This can be achieved using the duplication construction, starting with the $[24,12]$ Golay code, and forming the first block by duplicating an arbitrary subset of eight bits. This gives 759 codewords of normalized weight 16.

This number can be reduced to 503 as follows. We use an $X4$ construction with C_1 the $[16,1,16]$ code, C_3 the $[16,5,8]$ first-order Reed–Muller code, and C_4 the extended Hamming $[16,11,4]$ code. This still leaves C_2 to be chosen. We choose C_2 to be the $[16,7,4]$ doubly even code formed by taking the $[16,5,8]$ first-order Reed–Muller code, plus two linearly independent cosets; we choose the cosets so that they both have weight enumerators

$$4x^4 + 24x^8 + 4x^{12}$$

and their "sum" (i.e., the translate of one coset by any element of the other coset) has the same weight enumerator. This is possible if we choose one coset to be the one represented by the

$$\left[\begin{array}{ccccccccccccc|ccccccccccccc}
1&1&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&1&1&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&1&1&0&1&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&1&1&0&1&0&1\\
0&0&0&1&1&0&0&0&0&0&0&0&0&1&1&0&0&0&1&1&1&0&1&0&0&0\\
0&0&0&0&0&1&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&1&0&1&0&0\\
0&0&0&0&0&0&0&1&1&0&0&0&0&1&1&0&1&1&0&0&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&1&1&0&0&1&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&1&0&1&1&0&0&1&1&0&0\\
0&0&1&1&0&1&0&1&0&1&0&1&0&1&1&1&1&1&1&1&1&1&0&1&1&1
\end{array}\right].$$

Boolean function v_1v_2 and the other to be the one represented by the Boolean function v_1v_3 [24, p. 418]. The "sum" coset is represented by the Boolean function $v_1v_2 + v_1v_3$. Letting $f(\boldsymbol{v}) = v_1v_3$, we see that $v_1v_2 + v_1v_3 = f(A\boldsymbol{v})$, with

$$A = (1\ 1\ 0\ 0/0\ 1\ 0\ 0/0\ 0\ 1\ 0/0\ 0\ 0\ 1)$$

from which we conclude (see [24, Theorem 14.4, p. 417]) that the cosets represented by v_1v_3 and $v_1v_2 + v_1v_3$ have the same weight distribution.

The code C_2 thus has weight enumerator

$$1 + 12x^4 + 102x^8 + 12x^{12} + x^{16}.$$

A straightforward application of the $X4$ bound gives the lower bound 12 on normalized distance with these codes; however, only codewords with weight distribution 4|4 can have normalized weight this low, and all others have normalized weight at least 16. Since the number of codewords of weight 4 in C_2 is low, we can match each to words of weight greater than 4 on the right, with some trial and error.

A resulting generator matrix is shown at the bottom of this page. This matrix has normalized weight enumerator

$$1 + 503x^{16} + 512x^{20} + 2064x^{24} + 512x^{28} + 503x^{32} + x^{48}.$$

(Once we have chosen C_2, this number of minimum normalized weight codewords is minimum: split linear programming constraining the left code to have the weight enumerator of C_2 gives a minimum of exactly 503 for the number of codewords of normalized weight 16. Furthermore, this number is the only feasible solution to the split linear programming problem when the past subcode is constrained to have the weight enumerator of C_2.)

Split linear programming gives an overall lower bound of 271 on the number of minimum normalized weight codewords.

2) Best $[16|16|16, k, 16]$ *Codes:* The full-dimension lemma combined with the result above indicates that $k \le 28$ in this case. We can achieve this upper bound using the following construction. We begin by forming a $[16|16, 12, 16]$ code C from the Golay code by partitioning the 24 positions of the Golay code into a U_8 and U_{16}, again in the notation of [5], then bringing the U_8 to the left and doubling it. The resulting code has generator matrix

$$\left[\begin{array}{c|c} 0 & G_1 \\ \hline G_3 & G_2 \end{array}\right]$$

where G_3 generates the $[16, 8, 2]$ code $[8, 8, 1] \otimes [2, 1, 2]$. We find from [5, Table III.B] that a U_{16} intersects every codeword of weight 8 from the Golay code in at least two places, and there are 16 such codewords that intersect in exactly two places. Thus the punctured future code $\langle G_1, G_2 \rangle$ is a $[16, 12, 2]$ (even) code, and, furthermore, has covering radius 2 since there are too many codewords of weight 2 for the covering radius to be 1.

We now select a coset leader y of weight 2 of the punctured future code and select a generator matrix for the $[16|16|16, 28, 16]$ code of the form

$$\left[\begin{array}{c|c|c} 0 & G_1 & 0 \\ \hline G_3 & G_2 & 0 \\ \hline x_1 & y & \\ x_2 & y & \\ \vdots & \vdots & I_{16} \\ x_{16} & y & \end{array}\right].$$

Our requirements for the x_i's are that i) they all have weight at least 8 (for codewords of the form $(x_i|y|1)$); ii) they are distance at least 8 from C_3 (for codewords of the form $(x_i|y|1) + C$); iii) they are distance at least 8 apart (for codewords of the form $(x_i + x_j|0|2)$); iv) for codewords of the form $(x_i + x_j|0|2) + C$, we must avoid weights $(\le 3|2|2)$. Note that other cases give normalized weight ≥ 16 regardless of the choice of the x_i's.

We can find such x_i's by concatenating the $[8, 4, 4]$ Hamming code with the $(2, 2, 2)$ nonlinear code $\{01, 10\}$. The resulting x_i's all have weight exactly 8, are distance exactly 8 or 16 apart, and distance exactly 8 from C_3. Furthermore, codewords of the

$$\left[\begin{array}{cccccccccccccccc|cccccccccccccccc}
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&0&1&0&1&1&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&0&1&0&0&0&0&0&1&1&0\\
0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&1&0&1&1&0&0&1&0\\
0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&1&1&0&1&1&0&0&1&0&1\\
0&0&0&0&0&0&0&0&1&1&0&0&1&1&0&0&1&0&1&0&0&0&0&0&1&0&0&1&0&0&1&1\\
0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{array}\right]$$

TABLE IV
CODES FOR 8-PSK-BASED SYSTEMS

n, d_n^*	i	Max. rate gain, bits/n symbols, upper bounds: Full rate lemma	Griesmer[a]	Integer Split LP	Int. Split LP $k_p = 0$	Opt. rate gain	k	Actual d_n	N_{min}
6	2	1	1	1	0	1	7	6	1
	3	0	-	-	-	0	13	6	1
7	2	3	2	2	2	2	7	8.8	21
	3	1	-	-	-	1	14	8.8	21
8	2	4	3	3	3	3	8	8.8	7–8
	3	1	-	-	-	1	16	8.8	7–8
9	2	4	3	3	3	3	9	9.4	3–8
	3	1	-	-	-	1	18	9.4	3–8
10	2	4	3	3	3	3	10	10.4	6–10
	3	1	-	-	-	1	20	10.4	6–10
11	2	5	4	4	3	4	11	11	1
	3	1	-	-	-	1	22	11	1
12	2	5	4	4	3	4	12	12	1
	3	1	-	-	-	1	24	12	1

[a] If the code contains $(1_n 0_{xn-n})$.

form $(x_i + x_j|0|2) + C$ that have weight 2 in the middle basic block must come from codewords of the original Golay code with weight distribution $(6|2)$ across the $(U_8|U_{16})$ partition. The word $x_i + x_j$ is a word of weight 4 or 8 concatenated with the $[2,1,2]$ repetition code, and is thus distance at least 4 from the word of weight 6 concatenated with the $[2,1,2]$ repetition code.

We conclude that this is a $[16|16|16, 28, 16]$ code. There are 1671 codewords of minimum normalized weight.

This code can easily be extended to more basic blocks, forming $[16|16|16|16, 44, 16]$ and $[16|16|16|16|16, 60, 16]$ codes, i.e., the full dimension lemma bound can be achieved in each case.

V. BEST CODES FOR 8-PSK-BASED BCMIM SYSTEM

In an 8-PSK modulation system, the minimum squared intra-subset Euclidean distance increases with the ratio $1 : 3.414 : 6.828$. The results for basic blocks lengths 6 to 12 are shown in Table IV. Note that in this case the codes of highest rate *and* minimum number of codewords at the target minimum normalized distance $d_n^* = n$ are known in each case in this list.

The results in the columns listed as split linear programming were obtained as usual by rounding a noninteger optimum number of codewords down to the nearest power of two. It is interesting to note that in every case in the table, integer split linear programming gave a maximum that was a power of two (the same one as the noninteger split linear programming optimum rounded down).

The column titled "integer split linear programming, $k_p = 0$" indicates the case in which the past subcode is constrained to have dimension 0, i.e., the word $(1_n 0_{xn-n})$ is not in the code. This helps establish whether a minimum normalized distance greater than n is possible without sacrificing rate. In three cases ($n = 6, 11$, and 12) it is not. This is tight for the cases considered: we can achieve $d_n > n$ if it is not ruled out by this split linear programming result, and in the cases where it is ruled out, we can find codes with exactly one codeword of minimum normalized weight.

The case $n = 6$ above is straightforward. For $n = 12$, the baseline codes are $[12, 1, 12]$, $[12, 7, 4]$, and $[12, 11, 2]$. To construct a $[12|12, 12, 12]_\circ$ code, we take the $[24, 12]$ Golay code and partition into two U_{12}'s. Since these are codewords in the $[24, 12]$ code, the possible weight distributions of words of Hamming weight 8 are $6|2$, $4|4$, and $2|6$, all of which have normalized weight greater than 12. This leaves one codeword of minimum normalized weight, the word $(1_{12}|0_{12})$, but the discussion above shows that this cannot be avoided.

Shortening and puncturing produces codes of highest dimension for all basic block lengths in the range 7 to 11. For basic block lengths 7 to 10, however, it produces one codeword at normalized distance n, rather than the zero indicated in the table.

Codes of highest dimension in the range 7 to 10 are produced by shortening and puncturing a $[10|10, 10, > 10]_\circ$ code. To construct this code, we note that since the past subcode must have dimension zero, the punctured future subcode must be the $[10, 10, 1]$ code. Taking an identity matrix on the right of the generator matrix, the rows on the left must have Hamming weight at least 7, and be distance at least 4 apart. Then all nonzero codewords have weight distribution $[\geq 7|1]$, $[\geq 4|2]$, and $[\geq 0|3]$, and thus have normalized distance at least 10.4. Since $A(10, 4, 7) = 13$ [24, p. 684], we can find ten such words. We obtain a code with normalized weight enumerator

$$1 + 10x^{10.4} + 33x^{10.8} + 12x^{12.8} + \cdots.$$

The nonexistence of $[12|12, 12, > 12]_\circ$ and $[11|11, 11, > 11]_\circ$ codes can be demonstrated in a similar way, without using split linear programming. A $[12|12, 12, > 12]_\circ$ code would have 12 codewords with weight distribution $[\geq 9|1]$ and 66 codewords with weight distribution $[\geq 6|2]$. The rows of type $[\geq 9|1]$ would have to have weight distribution exactly $[9|1]$, but $A(12, 6, 9) =$

TABLE V
COMPARISON TO LONGER BASELINE BCM FOR QAM-BASED SYSTEMS

n	d_n	Rate, bits/$2n$ symbols		Min. (norm.) wt. codewords		Component codes for BCM
		BCMIM	BCM	BCMIM	BCM	
8	8	16	16	71	30 & 140	$[16,5,8]^*$ $[16,11,4]^*$
9	10	14	12	31-33	6 & 0	$[18,3,10]$ $[18,9,6]^*$
10	10	16	15	6-12	10 & (1-16)	$[20,4,10]^3$ $[20,11,5]$
11	12	16	15	24-55	(3-6) & 77	$[22,3,12]$ $[22,12,6]$
12	12	18	19	6-60	28 & 336	$[24,5,12]^*$ $[24,14,6]^*$
13	14	18	16	26-96	(2-4) & (1-116)	$[26,3,14]$ $[26,13,7]$
14	14	20	19	9-34	24 & 0	$[28,5,14]^*$ $[28,14,8]^*$
15	15	22	20	39-136	16 & 0	$[30,5,15]^*$ $[30,15,8]$
16	16	24	23	271-503	62 & (263-908)	$[32,6,16]^*$ $[32,17,8]$

4, so we cannot find a set of twelve such words. (Similarly for $n=11$, we have $A(11,6,8)=3$, whereas we require 11.)

For basic block lengths 7 and 8, the minimum normalized distance can be extended to 8.8 without sacrificing dimension. For the $n=7$ case, we first construct a $[7|7,7,8.8]_\circ$ code by taking a generator matrix with identity matrix at the right and seven distinct rows of weight 6 at the left. This gives 21 codewords of weight 8.8, which integer split linear programming shows is the minimum possible. For $n=8$, we take an identity matrix at the right of the generator matrix, and eight distinct rows of weight 6 at the left. This gives eight codewords of weight 8.8, whereas integer split linear programming gives a lower bound of 7.

A. Extension to Third Basic Block

The normalizing constant $a_3 = 6.8$ is so high that for each of the basic block lengths considered, the full-dimension lemma gives an upper bound of one on the dimension gain achievable by extending interblock memory to the third basic block. This can be achieved in each case, without increasing the number of minimum normalized weight codewords, by a gluing code of dimension 1. Thus in each case, we take the best $[n|n,k,d_n]_\circ$ code for the first two basic blocks, an $[n,n-1,2]$ even-weight code as shortened future code, and glue these together by adding a word of length $3n$ to the generator matrix.

For $n=7$, we glue the $[7|7,7,8.8]_\circ$ code for the first two basic blocks and the $[7,6,2]$ code for the third basic block using the gluing code with generator matrix shown at the bottom of this page, and similarly for $n=8$.

For $n=10$, the gluing code again has weight 0 in the middle block, and weight 1 in the third block. The first block can be chosen as any word of weight 7 that is distance at least 4 from each of the left hand rows in the $[10|10,10,10.4]_\circ$ code above. Since $A(10,4,7)=13$, and only ten of these words have been used, y can be chosen to be any of the remaining three words. This gives a $[10|10|10,20,10.4]_\circ$ code with ten codewords of normalized weight 10.4. Shortening and puncturing gives a $[9|9|9,18,9.4]_\circ$ code with eight codewords of normalized weight 9.4.

For $n=12$, the gluing code generator matrix can be taken to be any row with weight distribution $[6|0|1]$ between the three basic blocks. This produces codewords with weight distributions $[6|0|1]$, $[\geq 0| \geq 2|1]$, and $[\geq 0| \geq 0| \geq 2]$, all of which give normalized weight greater than 12. We thus obtain a $[12|12|12,24,12]_\circ$ code with one codeword of normalized weight 12; this is the best possible. Shortening and puncturing gives an $[11|11|11,22,11]_\circ$ code that has one codeword of normalized weight 11.

VI. CONCLUSION

We have considered in this paper the addition of interblock memory to block-coded modulation systems, combined with the staggered assignment of code bits to code symbols; this is known to allow significantly higher rates in block-coded modulation systems at the cost of some increase in decoding complexity and potential error propagation.

For the analysis of the performance of such systems at high signal-to-noise ratios, we have introduced the notion of normalized distance. We have developed bounds and constructions for codes designed with the normalized distance criterion, and have

$$[1\ 1\ 1\ 1\ 1\ 1\ 1\quad 0\ 0\ 0\ 0\ 0\ 0\ 0\quad 1\ 0\ 0\ 0\ 0\ 0\ 0]$$

used these to find many optimal codes for the situations of most interest, for both QAM- and PSK-based systems.

APPENDIX
COMPARISONS WITH BASELINE SYSTEMS WITH LONGER BLOCKS

We noted in Section II-D that an $[n|n,k,d_n]$ code will involve decoding a length $2n$ code. Thus one comparison that might be made is to the rates achievable with baseline BCM systems with basic block length $2n$, which also involve decoding codes of length $2n$ (albeit two distinct codes rather than one as in the BCMIM system).

The results obtained for BCMIM codes in Table I are compared on this basis to longer baseline BCM systems in Table V.

The number of apparent nearest neighbors in a BCM system depends on the transmitted sequence and the overall modulation scheme into which the codes are embedded. For a 16-QAM system, for example, the maximum number of nearest neighbors is known to be

$$a_1(d)4^d + a_2(d/2)4^{d/2} + a_3(d/4)2^{d/4} + a_4(d/8)$$

(where $a_i(w)$ denotes the number of codewords of weight w in the ith code) and the average number over all transmitted sequences is

$$a_1(d)3^d + a_2(d)2.25^{d/2} + a_3(d)2^{d/4} + a_4(d)$$

assuming all codewords are equally likely [22], [34]. We thus give the number of minimum-weight codewords for each of the component codes of the BCM system in Table V: the notation $n_1 \& n_2$ indicates that there are n_1 codewords of Hamming weight d_n in the first code listed, and n_2 in the second.

The number of apparent nearest neighbors for a code with interblock memory may be derived similarly as, for the 16-QAM case, a maximum of

$$\sum_{W(c)=d_n} 4^{w_1(c)} 4^{w_2(c)} 2^{w_3(c)} 1^{w_4(c)}$$

and an average of

$$\sum_{W(c)=d_n} 3^{w_1(c)} 2.25^{w_2(c)} 2^{w_3(c)} 1^{w_4(c)}$$

where $w_i(c)$ denotes the Hamming weight of the word c in the ith basic block [26, pp. 152–155].

The star on some BCM component codes indicates that the code is known to be unique. A number as superscript indicates that there are exactly that many inequivalent codes with the given parameters. This information is collected in Jaffe's table [15].

From the table, the BCMIM system has higher rate in all cases except basic block lengths 8 and 12. The rates are the same for basic block length 8, and basic block length 12 provides the only example in this list in which a higher rate is achievable using the longer baseline BCM system. For basic block length 12, the maximum number of nearest neighbors of the baseline BCM system is

$$28 \cdot 4^{12} + 336 \cdot 4^6 \approx 28.1 \cdot 4^{12}$$

whereas for the BCMIM system, using the weight distribution of minimum normalized weight codewords given in Section IV-D, we have a maximum of

$$4^{12} + 4^6 + 15 \cdot 4^8 + 32 \cdot 4^9 \approx 1.56 \cdot 4^{12}.$$

The effective coding gain is thus 0.78 dB better for the BCMIM system. Similarly, the basic block length 8 case gives the same rates, but due to the smaller number of nearest neighbors the BCMIM system has an effective coding gain of 0.55 dB over the longer baseline BCM system.

Codes in which the interblock memory is extended to the third and higher basic blocks usually have a relatively simple structure in higher basic blocks, as shown earlier. Thus we would expect that an $[n|n|n,k,d_n]$ code would in general be much less complex to decode than most length $3n$ codes, and a comparison to baseline BCM codes with basic block length $3n$ would not be meaningful.

Similar results hold for 8-PSK systems: the best BCMIM systems of Table IV all have higher rates than the corresponding BCM systems, except for the case $n = 6$.

REFERENCES

[1] E. L. Blokh and V. V. Zyablov, "Coding of generalized concatenated codes," *Probl. Pered. Inform*, vol. 10, no. 3, pp. 45–50, 1974. (In Russian. English translation in *Probl. Inform. Transm.,* vol. 10, no. 3, pp. 218–222, July–Sept. 1974.).

[2] A. G. Burr, "Maximum likelihood and multistage decoding for optimized multilevel coded modulation," in *Proc. 1998 Information Theory Workshop*, Killarney, Ireland, June 1998, pp. 81–82.

[3] A. G. Burr and T. J. Lunn, "Block-coded modulation optimized for finite error rate on the white Gaussian noise channel," *IEEE Trans. Inform. Theory*, vol. 43, pp. 373–385, Jan. 1997.

[4] A. R. Calderbank, "Multi-level codes and multi-stage decoding," *IEEE Trans. Commun.*, vol. 37, pp. 222–229, Mar. 1990.

[5] J. H. Conway and N. J. A. Sloane, "Orbit and coset analysis of the Golay and related codes," *IEEE Trans. Inform. Theory*, vol. 36, pp. 1038–1050, Sept. 1990.

[6] E. L. Cusack, "Error control codes for QAM signaling," *Electron. Lett.*, vol. 20, no. 2, pp. 62–63, Jan. 1984.

[7] S. M. Dodunekov and S. B. Encheva, "Uniqueness of some linear subcodes of the extended binary Golay code," *Probl. Pered. Inform.*, vol. 29, no. 1, pp. 45–51, Jan.–Mar. 1993. In Russian. English translation in *Probl. Inform. Transm.,* vol. 29, no. 1, pp. 38–43, Jan.–Mar. 1993.

[8] I. Dumer, "Concatenated codes and their multilevel generalizations," in *Handbook of Coding Theory*, V. S. Pless and W. C. Huffman, Eds. Amsterdam, The Netherlands: North-Holland, 1998, vol. I.

[9] G. D. Forney Jr., "Coset codes—Part I: Introduction and geometrical classification," *IEEE Trans. Inform. Theory*, vol. 34, pp. 1123–1151, Sept. 1988.

[10] G. D. Forney Jr. and G. Ungerboeck, "Modulation and coding for linear Gaussian channels," *IEEE Trans. Inform. Theory*, vol. 44, pp. 2384–2415, Oct. 1998.

[11] V. V. Ginzburg, "Multidimensional signals for a continuous channel," *Probl. Pered. Inform.*, vol. 20, no. 1, pp. 28–46, 1984. In Russian. English translation in *Probl. Inform. Transm.*, vol. 20, no. 1, pp. 20–34, Jan.–Mar. 1984.

[12] J. Hagenauer, "Source-controlled channel coding," *IEEE Trans. Commun.*, vol. 43, pp. 2449–2457, Sept. 1995.

[13] J. Huber, U. Wachsmann, and R. Fischer, "Coded modulation by multilevel codes: Overview and state of the art," in *ITG-Fachbericht: Codierung für Quelle, Kanal und Übertragung*, Aachen, Germany, Mar. 1998, pp. 255–266.

[14] H. Imai and S. Hirakawa, "A new multilevel coding method using error-correcting codes," *IEEE Trans. Inform. Theory*, vol. IT-23, pp. 371–377, May 1977.

[15] D. B. Jaffe, "Binary Linear Codes: New Results on Nonexistence," Dept. Math. Stat., Univ. Nebraska, Lincoln, Tech. Rep., Oct. 1996. [Online] Available from http://math.unl.edu/~djaffe/codes/codes.ps.gz.

[16] T. Kasami, T. Takata, T. Fujiwara, and S. Lin, "A concatenated coded modulation scheme for error control," *IEEE Trans. Commun.*, vol. 38, no. 6, pp. 752–763, June 1990.

[17] T. Kasami, T. Takata, T. Fujiwara, and S. Lin, "On multilevel block modulation codes," *IEEE Trans. Inform. Theory*, vol. 37, pp. 965–975, July 1991.

[18] Y. Kofman, E. Zehavi, and S. Shamai, "Performance analysis of a multilevel coded modulation system," *IEEE Trans. Commun.*, vol. 42, pp. 299–312, Feb./Mar./Apr. 1994.

[19] M.-C. Lin, "A coded modulation scheme with inter block memory," in *Proc. 1990 IEEE Intl. Symp. Information Theory*, San Diego, CA, Jan. 1990, p. 51.

[20] M.-C. Lin and S.-H. Ma, "A coded modulation scheme with inter block memory," *IEEE Trans. Commun.*, vol. 42, pp. 911–916, Feb./Mar./Apr. 1994.

[21] M.-C. Lin, J.-Y. Wang, and S.-C. Ma, "On block-coded modulation with interblock memory," *IEEE Trans. Commun.*, vol. 45, pp. 1401–1411, Nov. 1997.

[22] T. J. Lunn and A. G. Burr, "Number of neighbors for staged decoding of block-coded modulation," *Electron. Lett.*, vol. 29, no. 20, pp. 1830–1831, Oct. 1993.

[23] S.-C. Ma and M.-C. Lin, "A trellis coded modulation scheme constructed from block-coded modulation with interblock memory," *IEEE Trans. Inform. Theory*, vol. 40, pp. 1348–1363, Sept. 1994.

[24] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam, The Netherlands: North-Holland, 1977.

[25] R. H. Morelos-Zaragoza, T. Kasami, S. Lin, and H. Imai, "On block-coded modulation using unequal error protection codes over Rayleigh fading channels," *IEEE Trans. Commun.*, vol. 46, pp. 1–4, Jan. 1998.

[26] C.-N. Peng, "Block coded modulation schemes with interblock memory," Ph.D. dissertation, Univ. Michigan, Ann Arbor, 1998.

[27] S. I. Sayegh, "A class of optimum block codes in signal space," *IEEE Trans. Commun.*, vol. COM-34, pp. 1043–1045, Oct. 1986.

[28] J. Simonis, "The $[18, 9, 6]$ code is unique," *Discr. Math.*, vol. 106/107, pp. 439–448, Sept. 1992.

[29] N. J. A. Sloane, S. M. Reddy, and C.-L. Chen, "New binary codes," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 503–510, July 1972.

[30] T. Takata, S. Ujita, T. Kasami, and S. Lin, "Multistage decoding of multilevel block M-PSK modulation codes and its performance analysis," *IEEE Trans. Inform. Theory*, vol. 39, pp. 1204–1218, July 1993.

[31] J. H. van Lint, *Introduction to Coding Theory*. New York: Springer-Verlag, 1982.

[32] U. Wachsmann and J. Huber, "Distance profile of multilevel coded transmission and rate design," in *Proc. 1998 Information Theory Workshop*, Killarney, Ireland, June 1998, pp. 10–11.

[33] R. G. C. Williams, "Low complexity block coded modulation," Ph.D. dissertation, Univ. Manchester, Manchester, U.K., 1988.

[34] R. G. C. Williams, "Block coding for voiceband modems," *British Telecom Tech. J.*, vol. 10, no. 1, pp. 101–111, Jan. 1992.

[35] K. Yamaguchi and H. Imai, "A new block-coded modulation scheme and its soft decision decoding," in *Proc. 1993 IEEE Intl. Symp. Information Theory*, San Antonio, TX, Jan. 1993, p. 64.

[36] L. Zhang and B. Vucetic, "Multilevel block codes for Rayleigh fading channels," *IEEE Trans. Commun.*, vol. 43, pp. 24–31, Jan. 1995.

[37] V. A. Zinov'ev, "Generalized concatenated codes," *Probl. Pered. Inform.*, vol. 12, no. 1, pp. 5–15, 1976. (In Russian. English translation in *Probl. Inform. Transm.*, vol. 12, no. 1, pp. 2–9, Jan.–Mar. 1976.).