with transfer function $(1 \mp D^m)$, $m = 1, 2, \cdots$. Lower bounds to the cardinality of sets with prescribed minimum Euclidean distance have been provided. Matched spectral null codes of zeroth order are constructions offering large code sets, for small word length even optimal sets. Spectral null codes of higher order, $K \geq 1$, are far from optimal for the small values of the codeword lengths that have been investigated. Upper and lower bounds have been furnished to the size of codes with minimum squared Euclidean distance greater than unity.

ACKNOWLEDGMENT

The authors wish to thank A. J. E. M. Janssen for finding the asymptotical behavior of S_n described in Section IV-C.

REFERENCES

[1] J. K. Wolf and G. Ungerboeck, "Trellis coding for partial-response channels," *IEEE Trans. Commun.*, vol. COM-34, pp. 765–773, Aug. 1986.

[2] R. Karabed and P. H. Siegel, "Matched spectral-null codes for partial-response channels," *IEEE Trans. Inform. Theory*, vol. 37, no. 3, pp. 818–855, May 1991.

[3] K. Hole and Ø. Ytrehus, "Improved coding techniques for precoded partial-response channels," *IEEE Trans. Inform. Theory*, vol. 40, no. 2, pp. 482–493, Mar. 1994.

[4] K. A. S. Immink, "Coding techniques for partial-response channels," *IEEE Trans. Commun.*, vol. 36, pp. 1163–1165, Oct. 1988.

[5] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam, New York: North-Holland, 1981.

[6] K. A. S. Immink and G. F. M. Beenker, "Binary transmission codes with higher order spectral zeros at zero frequency," *IEEE Trans. Inform. Theory*, vol. IT-33, no. 3, pp. 452–454, May 1987.

[7] R. M. Roth, P. H. Siegel, and A. Vardy, "Higer-order spectral-null codes—Constructions and bounds," *IEEE Trans. Inform. Theory*, vol. 40, no. 6, pp. 1826–1840, Nov. 1994.

[8] K. A. S. Immink, *Coding Techniques for Digital Recorders*. Englewood Cliffs, NJ: Prentice-Hall, 1991.

[9] B. Bose and T. R. N. Rao, "Theory of unidirectional error correcting/detecting codes," *IEEE Trans. Comput.*, vol. C-31, no. 6, pp. 521–530, June 1982.

[10] B. Bose, "Burst unidirectional error detecting codes," *IEEE Trans. Comput.*, vol. C-35, no. 4, pp. 350–353, Apr. 1986.

[11] M. Blaum, "Systematic unidirectional burst detecting codes," *IEEE Trans. Comput.*, vol. 37, no. 4, pp. 453–457, Apr. 1988.

An Efficient Block-Based Addressing Scheme for the Nearly Optimum Shaping of Multidimensional Signal Spaces

A. K. Khandani, *Associate Member, IEEE*, and P. Kabal, *Member, IEEE*

***Abstract*—We introduce an efficient addressing scheme for the nearly optimum shaping of a multidimensional signal constellation. The 2-D (two-dimensional) subspaces are partitioned into K energy shells of equal cardinality. The average energy of a 2-D shell can be closely approximated by a linear function of its index. In an $N = 2n$-D space, we obtain K^n shaping clusters of equal cardinality. Shaping is achieved by selecting $T \leq K^n$ of the N-D clusters with the least sum of the 2-D indices. This results in a set of T integer n-tuples with the components in the range $[0, K-1]$ and the sum of the components being at most a given number L. The problem of addressing is to find a one-to-one mapping between the set of such n-tuples and the set of integers $[0, T-1]$ such that the mapping and its inverse can be easily implemented. In the proposed scheme, the N-D clusters are grouped into blocks of identical binary weight vectors. This results in a simple rule for the addressing of points within the blocks. The addressing of the blocks is based on some recursive relationship which allows us to decompose the problem into simpler parts. The overall scheme requires a modest amount of memory and has a small computational complexity.**

***Index Terms*—Optimum shaping, addressing decomposition, recursive addressing, binary weight vectors, shell mapping.**

I. INTRODUCTION

A digital communication system is usually modeled as a discrete-time system. In the discrete model, the channel provides us with a given number of dimensions, say N, per signaling interval. In each signaling interval, the input data are encoded such that one of M equiprobable symbols is produced. To transmit these symbols, we select M points over the channel space. Each of the source symbols is represented by one of these points. This collection of points is called a signal constellation.

A signal constellation is usually selected as a finite subset of a regular array of points (packing) bounded within a shaping region. The main objective in selecting a shaping region is to minimize the average energy of the constellation for a given number of points from the given packing. The reduction in the average energy per two dimensions due to using a region $\mathcal{C}$ as the boundary instead of a hypercube is called the shaping gain of $\mathcal{C}$ and is denoted as $\gamma_s(\mathcal{C})$.

The price to be paid for a shaping gain $\gamma_s > 1$ involves: i) an increase in the factor CER_s (Constellation-Expansion Ratio) which is defined as the ratio of the employed number of points per two dimensions to the minimum necessary number of points per two dimensions [1], ii) an increase in the factor PAR_s (Peak-to-Average-power Ratio) which is defined as the ratio of the peak of energy per two dimensions to the average energy per two dimensions [1], and iii) an increase in the addressing complexity where addressing is the assignment of the input data to the constellation points.

Manuscript received November 2, 1992; revised December 10, 1994. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC). This work was presented in part at the IEEE International Conference on Communications, ICC'93.

A. K. Khandani is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, N2L 3G1.

P. Kabal is with the Department of Electrical Engineering, McGill University, Montreal, PQ, Canada, H3A 2A7.

IEEE Log Number 9414736.

The major problem associated with shaping in a high-dimensional space is the addressing complexity. As the cardinality of a multidimensional signal constellation is usually a huge number, one cannot use a lookup table for this purpose. This means that one needs an algorithm to implement the addressing and its inverse. The present work concentrates on finding an addressing scheme with low implementation complexity and small performance degradation.

II. Basic Structure

The scheme to be presented here, like that in [2] and other prior work [3], is based on partitioning of a 2-D signal constellation into K energy shells. Each of these 2-D shells contains the same number of points. The first shell contains a subset of the least energy points, the second shell contains the next-least energy points, and so forth. The 2-D shells are indexed in the radial direction by $J \in [0, K-1]$. It can be shown that the average energy of the 2-D shells is well approximated by a linear function of the shell index [2]. Keeping this point in mind, the cost associated with a 2-D shell is taken to be its index.

In an $N=2n$-D space, we obtain K^n clusters of equal cardinality. The N-D clusters are described by the index vectors $(J_0, \cdots, J_{n-1}) \in [0, K-1]^n$. The cost of such an N-D cluster is

$$\sum_{0 \le i \le n-1} J_i.$$

The final constellation is selected as the collection of the N-D clusters of cost not greater than L. The corresponding shaping set is [2]

$$TC_n(K, L) \equiv \bigcup_{l=0}^{L} F_n(K, l) \tag{1}$$

where

$$F_n(K, L) \equiv \left\{ (J_0, \cdots, J_{n-1}) \in [0, K-1]^n : J_i \in \{0, 1, \cdots, K-1\}, \text{ and } \sum_{i=0}^{n-1} J_i = L \right\}. \tag{2}$$

Note that $F_n(K, L)$ denotes the set of the N-D clusters with a total cost (sum of the 2-D indices) of L. Later, a recursion on K will be developed for the cardinality of the shaping set which is the basis for the addressing scheme introduced in this work. The cardinality of the shaping set is denoted as T.

Addressing involves a one-to-one mapping between the input data, which may be represented by integers I in the range $0 \le I \le T-1$, and the elements of the shaping set such that the mapping and its inverse can be easily implemented. The basic idea is to use the binary expansion of the index vector to partition the shaping set into the union of subsets (blocks) B_n. By the binary expansion of the index vector $(J_0, \cdots, J_{n-1})$, we mean a $k \times n$-dimensional matrix $\boldsymbol{G}$, whose ith column is the binary expansion of J_i, $i = 0, \cdots, n-1$. These blocks are arranged in a preselected order and are indexed by $0 \le b \le B_{\max}$. We assume that for $b_2 > b_1$, the points of $B_n(b_2)$ correspond to larger data values compared to the points of $B_n(b_1)$. For a given block b, the cardinality is denoted by $\Delta T_b = |B_n(b)|$ and the total number of points in the preceding blocks is denoted by T_b, i.e.

$$T_b = \sum_{i=0}^{b-1} \Delta T_i.$$

For a given data value $0 \le I \le T-1$, we first find the index b such that $T_b \le I < T_{b+1}$. Then, the residue $R_{es} = I - T_b$ is used to address a point within the block $B_n(b)$. As we will see later, this scheme allows us to decompose the addressing of a $TC_n(2^k, L)$ set into the addressing of k independent $TC_n(2, L)$ sets.

A. Blocks of Identical Binary Weight Vectors

For a given $K = 2^k$, $N = 2n$, and an integer $L \in [0, n(K-1)]$, consider k binary n-D vectors $\boldsymbol{g}_i$, $i = 0, \cdots, k-1$, where the weight of $\boldsymbol{g}_i$ is equal to $w^{(i)}$ and

$$L = \sum_{i=0}^{k-1} w^{(i)} 2^i. \tag{3}$$

Obviously,

$$0 \le w^{(i)} \le \min(n, L/2^i).$$

The k-tuple $\boldsymbol{w} = (w^{(k-1)}, \cdots, w^{(0)})$ is denoted as a binary weight vector of L. For $n = 1$, this is the normal binary representation with k bits. For $n > 1$, the binary weight vector of L for

$$L \notin \{0, 1, n(K-1)-1, n(K-1)\}$$

is not unique. We use the set of binary weight vectors of L to partition the $F_n(K, L)$ into blocks.

For a given k-tuple $\boldsymbol{w}$ of the form mentioned, consider the set of the binary $k \times n$ matrices such that the weight of the ith row is equal to $w^{(i)}$. Assume that for a given matrix of this set, say $\boldsymbol{G}$, the (i, j)th element is the i th digit (coefficient of 2^i) in the binary representation of the shell index along the jth 2-D subspace. By permuting the elements of the rows we obtain different matrices corresponding to a subset of points of $F_n(K, L)$. Note that such permutation does not change the weight of the rows, and consequently, the sum of the shell indices, as given in (3), remains constant. Applying such permutations results in

$$\prod_{i=0}^{k-1} C_n^{w^{(i)}}$$

points, where $C_n^{w^{(i)}}$ is the combinatorial coefficient. The union of these points result in one block. In other words, a block of $F_n(K, L)$ is the set of elements $\boldsymbol{p}$ satisfying

$$\boldsymbol{p} = \sum_{i=0}^{k-1} \boldsymbol{P}[p_i \odot 2^i \boldsymbol{g}_i] \tag{4}$$

where p_i's, $0 \le p_i < C_n^{w^{(i)}}$, are a set of integer numbers which in $\boldsymbol{P}[p_i \odot 2^i \boldsymbol{g}_i]$ determine the permutation applied to the elements of $2^i \boldsymbol{g}_i$. The procedure for permuting elements will be explained later. One can also look at the problem as if the points $(p_0, \cdots, p_{k-1})$ belong to a k-D cubic constellation.

This is similar to the labeling structure of [3] where the rows of matrix $\boldsymbol{G}$ correspond to the shaping codes of [3]. The major difference is that in [3] the weights of the shaping codes are selected independently. This results in a simple addressing scheme. This simplicity is obtained at the price of some performance loss. However, in our case, the weight of the shaping codes are jointly selected according to the range of the input data. This selection is achieved such that union of the corresponding blocks results in the optimum shaping set. Concerning the implementation of shaping codes, Calderbank and Ozarow [3] briefly address this problem and propose using a lookup table when the number of codewords is small and using a Voronoi constellation, otherwise. Our implementation method is more general and also more efficient.

We assume that the blocks have a lexicographic ordering according to the elements of the binary weight vector. For a given data value I, the block addressing is the determination of the label b (and $\boldsymbol{w}_b$) such that $T_b \le I < T_{b+1}$. This is discussed in the following.

B. Block Addressing

Assume that the shaping set is equal to $TC_n(2^k, L_{\max})$. If the most significant component of $\boldsymbol{w}$ is known to be equal to $w^{(k-1)}$, the shaping set reduces to the union of $C_n^{w^{(k-1)}}$ of its subsets each of cardinality

$$|TC_n(2^{k-1}, L_{\max} - w^{(k-1)}2^{k-1})|.$$

Each of these subsets is the collection of integer n-tuples with $n - w^{(k-1)}$ components in the range $[0, 2^{k-1} - 1]$ and $w^{(k-1)}$ components in the range $[2^{k-1}, 2^k - 1]$, where the sum of the components is less than or equal to $L_{\max}$. Considering the lexicographic ordering of the blocks, $w^{(k-1)}$ is selected as the largest integer satisfying

$$R_{k-1} = I_{k-1} - \sum_{i=0}^{w^{(k-1)}-1} C_n^i \, |TC_n(2^{k-1}, L_{\max} - i2^{k-1})| \geq 0 \quad (5)$$

where $I_{k-1} = I$ is the data value. The residue R_{k-1} is expanded as

$$R_{k-1} = P_{k-1} \, |TC_n(2^{k-1}, L_{\max} - w^{(k-1)}2^{k-1})| + I_{k-2} \quad (6)$$

where

$$0 \leq I_{k-2} < |TC_n(2^{k-1}, L_{\max} - w^{(k-1)}2^{k-1})|$$
$$0 \leq P_{k-1} < C_n^{w^{(k-1)}}. \quad (7)$$

The P_{k-1} is used in addressing within the blocks to permute $\boldsymbol{g}_{k-1}$.

After $\boldsymbol{g}_{k-1}$ is known, the shaping set reduces to the set

$$TC_n(2^{k-1}, L_{\max} - w^{(k-1)}2^{k-1})$$

shifted by the offset vector $2^{k-1}\boldsymbol{g}_{k-1}$. The I_{k-2} is used to address a point of this set. By replacing k by $k-1$ and $L_{\max}$ by $L_{\max} - w^{(k-1)}2^{k-1}$ in the original problem, the same procedure is used to find $w^{(k-2)}$. The procedure is repeated for k steps until all the elements of the coefficient vector are computed. The formulation for the Jth step, $J = k-2, \cdots, 1$, is as follows:

$$R_J = I_J - \sum_{i=0}^{w^{(J)}-1} C_n^i \, |TC_n(2^J, L_{\max} - w^{(k-1)}2^{k-1} - \cdots - w^{(J+1)}2^{J+1} - i2^J)| \geq 0 \quad (8)$$

$$R_J = P_J \, |TC_n(2^J, L_{\max} - w^{(k-1)}2^{k-1} - \cdots - w^{(J)}2^J)| + I_{J-1} \quad (9)$$

where

$$0 \leq I_{J-1} < |TC_n(2^J, L_{\max} - w^{(k-1)}2^{k-1} - \cdots - w^{(J)}2^J)|$$
$$0 \leq P_J < C_n^{w^{(J)}}. \quad (10)$$

Considering that $|TC_n(1, \beta)| = 1$, $\forall \beta$, the final step of the recursion reduces to

$$P_0 = I_0 - \sum_{i=0}^{w^{(0)}-1} C_n^i, \qquad 0 \leq P_0 < C_n^{w^{(0)}}. \quad (11)$$

1) Storage Requirement: We need a set of $k-1$ memory blocks to store the cardinality of TC_n sets where the ith block, $i = 1, \cdots, k-1$, contains the values of

$$|TC_n(2^i, L_{\max} - j2^i)|, \qquad j = 0, \cdots, \lfloor L_{\max}/2^i \rfloor.$$

Considering that

$$|TC_n(2^\alpha, \beta)| \leq 2^{n\alpha}, \ \forall \beta$$

the

$$|TC_n(2^\alpha, \beta)|, \ \forall \beta$$

can be stored with $n\alpha$ bits. This results in a total memory of size

$$M_{TC_n} = \sum_{i=1}^{k-1} \left(1 + \left\lfloor \frac{L_{\max}}{2^i} \right\rfloor\right) \left\lceil \frac{\min[ni, \lceil \log_2 T \rceil]}{8} \right\rceil. \quad (12)$$

Examples of the value of $L_{\max}$ are given in Tables I and II.

We need another block of memory to store the combinatorial coefficients. This is used in addressing within the blocks and will be computed later. Examples of the total memory size M_{tC} are given in Table I.

2) Computation Requirement: The calculation of the summation in (5), (8) requires $w^{(i)}$, $i = 1, \cdots, k-1$, multiply-adds. It is easy to verify that the total number of the multiply-adds

$$N_{op} = \sum_{i=1}^{k-1} w^{(i)}$$

is maximized when $(w^{(k-1)}, \cdots, w^{(1)})$ is the binary weight vector of $\lfloor L_{\max}/2 \rfloor$ of the lowest lexicographical order, i.e.,

$$w^{(k-1)} \leq \cdots \leq w^{(1)} \leq n.$$

For the cases shown in Table I, this maximum value is equal to $\lfloor L_{\max}/2 \rfloor$ corresponding to $w^{(i)} = 0$, $i = 2, \cdots, k-1$. The expansion of the residue in (6), (9) requires one multiplication, one division, and two additions per recursion step, times $k-1$ steps.

For $K = 8$ $(k = 3)$ and $\text{CER}_s = 1.5$, the maximum number of multiplications is equal to $2 + \lfloor L_{\max}/2 \rfloor$ and the number of divisions is equal to 2. Referring to Table I, we have $L_{\max} \simeq N$ resulting in about one multiplication per two dimensions.

3) Tradeoff Between the Storage and the Computational Complexities: As an alternative to computing (5) and (8), one can use a lookup table to store the required values. The corresponding total memory size is shown in columns M_{tL} of Tables I and II. In this case, the number of multiplications, divisions per block is equal to $k-1$. Columns N_{add}, N_{mul} in Tables I and II show the corresponding computational complexities.

Example: Consider an $N = 8$-D space with $K = 4$ $(k = 2)$ shells per 2-D subspaces and $\text{CER}_2 = \sqrt{2}$. To have $\text{CER}_2 = \sqrt{2}$, we need a

TABLE I
PARAMETERS OF THE ACHIEVED POINTS USING $K = 8$, $\text{CER}_s = 1.5$
(Columns M_{tC}, M_{tL} denote the memory size in bytes (8-bits) per N-D for a computational-based, lookup table-based addressing scheme. Columns N_{add}, N_{mul} denote the number of additions, multiplications (including divisions) per N-D for the lookup table-based addressing scheme. The computational complexity for the computational-based addressing scheme is about one multiplication per 2-D. Values inside parenthesis are the optimum γ_s.)

N	$L_{\max}$	$B_{\max}$	$A_{\max}$	M_{tC}	N_{add}	N_{mul}	M_{tL}	γ_s dB	PAR_s
16	17	139	12	0.04 k	50	4	0.11 k	0.90 (0.92)	3.52
32	29	612	22	0.17 k	85	4	0.45 k	1.07 (1.09)	3.66
64	54	3564	43	1.0 k	160	4	2.6 k	1.18 (1.21)	3.76
128	104	23966	84	6.6 k	300	4	17 k	1.27 (1.30)	3.84
256	202	170313	165	40 k	575	4	116 k	1.32 (1.35)	3.88

TABLE II
PARAMETERS OF THE ACHIEVED POINTS USING A LOOKUP-TABLE-BASED ADDRESSING SCHEME FOR $K = 4$, $\text{CER}_s = 1.25$
(Column M_{tL} Denotes the memory size in bytes (8-bits) per N-D. Columns N_{add}, N_{mul} denote the number of additions, multiplications (including divisions) per N-D. Values inside parenthesis are the optimum γ_s.)

N	$L_{\max}$	$B_{\max}$	$A_{\max}$	N_{add}	N_{mul}	M_{tL}	γ_s dB	PAR_s
16	9	25	8	35	2	0.04 k	0.76 (0.81)	2.84
32	16	72	16	60	2	0.18 k	0.90 (0.96)	2.94
64	28	210	28	100	2	1.1 k	1.00 (1.06)	3.00
128	53	729	53	195	2	6.6 k	1.07 (1.15)	3.05
256	103	2704	103	375	2	40 k	1.12 (1.20)	3.09

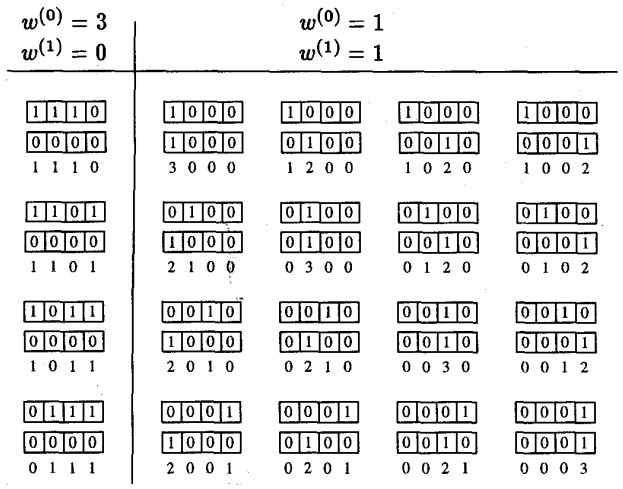

Fig. 1. The points in the two blocks of $F_4(4,3)$.

shaping set of cardinality $T=64$. Using the results of [2], we have

$$\{|F_4(4,\ell)|\}=\{1,4,10,20,31,40,44,40,31,20,10,4,1,0,0,\cdots\}. \tag{13}$$

It is seen that

$$|TC_4(4,4)|=\sum_{l=0}^{4}|F_4(4,l)|=66.$$

This means that $L_{\max}=4$ and only 29 points of the 31 points in $F_4(4,4)$ are included in the shaping set.

The blocks of $F_4(4,L)$, $0\le L\le 4$, correspond to the 2-tuple of integers $(w^{(1)},w^{(0)})$ such that $0\le w^{(1)}\le 2$, $0\le w^{(0)}\le 4$, and $L=2w^{(1)}+w^{(0)}$. This results in a total of 9 blocks. As an example, Fig. 1 shows the individual points in the two blocks of $F_4(4,3)$.

The $w^{(1)}$ is selected as the largest integer satisfying

$$R_1=I-\sum_{i=0}^{w^{(1)}-1}C_4^i\,|TC_4(2,4-2i)|\ge 0. \tag{14}$$

The rest of the problem is formulated as

$$R_1=P_1\,|TC_4(2,4-2w^{(1)})|+I_0$$
$$0\le I_0<|TC_4(2,4-2w^{(1)})|, 0\le P_1<C_4^{w^{(1)}} \tag{15}$$

$$P_0=I_0-\sum_{i=0}^{w^{(0)}-1}C_4^i, 0\le P_0<C_4^{w^{(0)}}. \tag{16}$$

Using the results of [2], we have

$$\{|TC_4(2,\ell)|\}=\{1,5,11,15,16,16,16,\cdots\}.$$

Fig. 2 shows the final lookup table containing the precomputed values. Column "I" contains the values of

$$\sum_{i=0}^{J}C_4^i\,|TC_4(2,4-2i)|,\quad J=0,1.$$

Column "II" contains the values of

$$|TC_4(2,4-2J)|,\quad J=0,1,2.$$

Column "III" contains the values of

$$\sum_{i=0}^{J}C_4^i,\quad J=0,1,2,3,4.$$

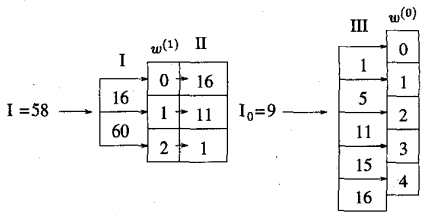

Fig. 2. Example of the lookup table used for the block addressing.

For a data value $I=58$, searching in the first lookup table, we obtain, $w^{(1)}=1$ and

$$|TC_4(2,4-2w^{(1)})|=11$$

(column "II"). The residue is $R_1=58-16=42$. Using (15), we obtain, $P_1=3$ and $I_0=9$. Using the second table with $I_0=9$, we obtain, $w^{(0)}=2$ and $P_0=9-5=4$.

C. Addressing of Points within the Blocks

This is a mapping between the integer numbers P_i, $0\le P_i<C_n^{w^{(i)}}$, and the set of binary n-tuples of weight $w^{(i)}$. Such a mapping is discussed in detail in [4]. The basic theorem is as follows [4]:

Theorem: Consider the set $G(n,w)$ of binary sequences $\boldsymbol{g}=(g_0,\cdots,g_{n-1})$ of length n and weight w. Define the partial weights

$$w_k=\sum_{i=k}^{n-1}g_i.$$

The binary sequences $\boldsymbol{g}\in G(n,w)$ can be labeled according to

$$P(\boldsymbol{g})=\sum_{k=0}^{n-1}g_kC_{n-k-1}^{w^{(k)}} \tag{17}$$

where $C_n^w=0$ for $w>n$ and $0\le P(\boldsymbol{g})<C_n^w$.

We assume that the C_{n-i}^w's, $1\le i\le n-1$, are precomputed and stored. As we have $C_m^w=C_m^{m-w}$, just the values of C_m^w for $w\le 1+\lfloor m/2\rfloor$ are stored. As

$$\sum_w C_m^{2w}=\sum_w C_m^{2w+1}=2^{m-1}$$

we obtain $C_m^w<2^{m-1}$, $\forall w$. This means that the C_m^w's, $\forall w$ can be represented with $m-1$ bits. Representing C_m^w with $m-1$ bits substantially increases the memory size. In spite of this, as for $N\le 128$ the memory size is still quite small, we use this type of the representation. This results in the memory size

$$M_c=\sum_{i=1}^{n-1}\left(1+\left\lfloor\frac{n-i}{2}\right\rfloor\right)\left\lceil\frac{\min[n-i-1,\lceil\log_2 T\rceil]}{8}\right\rceil. \tag{18}$$

For $N=256$, we use $\lceil(\log_2 C_m^w)/8\rceil$ bytes (8 bits) to represent the C_m^w. In this case, an additional byte is assigned to each C_m^w which stores the corresponding word length. This results in the memory size

$$M_c=\sum_{i=1}^{n-1}\sum_{w=1}^{1+\lfloor(n-i)/2\rfloor}1+\left\lceil\frac{\log_2 C_{n-i}^w}{8}\right\rceil. \tag{19}$$

The whole mapping requires at most $k(n-1)$ comparisons and $A=\sum_i w_i$ additions. Column $A_{\max}$ in Tables I and II shows the maximum value of $\sum_i w_i$ over $0\le L\le L_{\max}$.

Example (continued): In the example given earlier, we obtained $\boldsymbol{w}=(1,2)$ and $(P_1,P_0)=(3,4)$. Using (17), the rows of the final matrix are equal to $\boldsymbol{g}_1=(1,0,0,0)$ and $\boldsymbol{g}_0=(1,0,1,0)$. This results in the 2-D shells indexed by $2\boldsymbol{g}_1+\boldsymbol{g}_0=(3,0,1,0)$ along the first to the fourth 2-D subspaces.

D. Decoding

Decoding is the recovery of data value I from the sequence of 2-D shells. For decoding we first compute the matrix G and the k-tuple $\boldsymbol{w}$. Next, the rows of matrix G are used in (17) to calculate the P's. This involves at most $A_{\max}$ additions. Examples of the values of $A_{\max}$ are given in Tables I and II. Then, (9) and (8) are used recursively (in the order mentioned) up to computing I (no division is required).

III. Comparison with Other Methods

A. Previous Relevant Works

In the work of Wei [5] shaping is a side effect of the method employed to transmit a nonintegral number of bits per two dimensions. Addressing of this method is achieved by a lookup table. Forney and Wei generalize this method in [1].

Conway and Sloane in [6] introduced the idea of the Voronoi constellation based on using the Voronoi region of a lattice Λ_s as the shaping region. The Voronoi constellations are further considered by Forney in [7]. The idea of trellis shaping is introduced by Forney in [8]. This is, in fact, an infinite-dimensional Voronoi constellation obtained from a convolutional code.

In [3], Calderbank and Ozarow introduced a shaping method which is directly achieved on the 2-D subspaces. In this method, the 2-D subspaces are partitioned into equal-sized shells of increasing average energy. A shaping code is then used to specify the sequence of the 2-D shells. The shaping code is designed so that the lower energy shells are used more frequently.

Lang and Longstaff in [9] use an addressing scheme for channel coding which first divides the constellation into energy shells. Then a point in a shell is found by successively decomposing the space into lower dimensional subspaces. Prior to [9], a similar addressing scheme was used by Fischer in [10] to label the points of a vector quantizer with a pyramid quantization region. The addressing scheme of Lang and Longstaff is further discussed by Khandani and Kabal in [2], by Kschischang in [11] (see also Kschischang and Pasupathy [12]) and by Laroia, Farvardin, and Tretter in [13].

In [14], Kschischang and Pasupathy discuss a shaping method which is based on using the 2-D points with nonequal probability (see also [11]). In [15], Livingston discusses a shaping method in which the 2-D subspaces are partitioned into energy shells of increasing size. In this method, the 2-D shells are used with equal probability inducing a nonuniform distribution on the 2-D points. Kschischang in [16] discusses the structure of a prefix code which closely approximates the optimum nonuniform probabilities induced on the 2-D points.

In our previous works [2], [17] some addressing schemes are given which achieve (or closely approximate) points on the optimum tradeoff curves. Laroia, Farvardin, and Tretter in [13] suggest methods to reduce the addressing complexity of the method discussed in [9]. A comparison between the performance and the complexity of most of these schemes is available in Section III-C of this manuscript.

The technique of shell mapping is suggested by the Motorola Information Systems Group for inclusion in the forthcoming V.fast modem standard [18]. This proposal discusses the method in conjunction with a practical coding scheme and provides numerical comparison with other relevant proposals.

Finally, the technique of shell mapping is addressed by Forney in a paper discussing a variety of recent advances in modem technology [19].

B. Recent Developments in the Technique of Shell Mapping

The addressing method discussed in this correspondence can be categorized under the general technique of shell mapping. This technique has been under active developments in the recent years.

TABLE III
Parameters of the Point Achieved Using the Method of [17] for $N = 32$ (Values inside parenthesis are the optimum γ_s. Column M_t is the memory size in bytes (8-bits)/32-D (no computation).)

N	CER_s	M_t	γ_s dB	PAR
32	1.2	0.88 k	0.88 (0.91)	2.80
32	1.3	0.72 k	0.95 (1.00)	3.09
32	1.4	0.84 k	0.99 (1.04)	3.36

The major points which have resulted in these developments are as follows:

1) Shaping using a circular 2-D subconstellation [1].
2) Observing that the energy shells of the integer lattice can be represented by integer numbers and using this property, in conjunction with the additivity property of energy, to recursively decompose the addressing of a constellation into its subspaces of half dimensionality [9], [2], [11] (see also [12]), [13]. In a space of dimensionality $n = N/2$, this results in a hierarchy composed of $\log_2 n$ addressing stages. Complexity of the addressing at each stage is determined by the number of energy shells which in the case of the integer lattice has a linear growth with dimensionality (the overall complexity in this case grows as $n \log_2 n$, [12]).
3) Shaping using a finite number of 2-D energy shells [3].
4) Observing that assuming continuous approximation, the average energy of a set of K equi-volume 2-D energy shells can be represented by the integer numbers $[0, K-1]$, [2] (this fact is used for the first time in [3] to compute the shaping performance.). The addressing method of [13] makes use of a similar representation method without mentioning a clear justification for it.
5) Using the concentration property of the higher dimensional spaces to merge the energy shells while keeping the degradation in performance negligible [2] (see also [20]), [13].
6) Reducing the computational complexity of the method introduced in [9] by subdividing the shells into subsets with a cardinality which is an integral power of two [2], [17].
7) Observing that the recursive decomposition of addressing can be based on the 2-D energy shells instead of on the dimensionality. In this way, the number of the recursion steps becomes independent of the space dimensionality (present work).

C. Numerical Comparisons

In the following, we compare our addressing scheme with that of [2], [8], [9], [12], [13], [17], [18].

A four-state trellis diagram of [8] achieves $\gamma_s = 0.97$ dB, $\text{CER}_s = 1.5$, $\text{PAR}_s = 3.75$.

The direct shell mapping method of [9], [12] is based on the first two points mentioned in Section III-B. This method achieves optimum tradeoff but has a substantially higher complexity than the methods discussed in the present work or in [2], [13], [17], [18].

The proposal in [18] is based on the first four points mentioned in Section III-B. The overall complexity in a 16-D space is about 42 multiply-adds/2-D together with a few divisions and a memory of about 0.5 kbytes to achieve nearly optimum tradeoff with a CER_s up to 1.5.

The method of [13] is based on the first five points mentioned in Section III-B. In [13], an example for $N = 64$ is given which needs 1440 multiply-adds (assuming a 16-bit processor) and a memory of 1.5 kbytes to achieve a tradeoff point with $\gamma_s = 1.15$ dB, $\text{CER}_2 = 1.5$.

The methods of [2], [17] is based on the first six points mentioned in Section III-B. Table III shows some examples of the performance of the scheme discussed in [17].

For $N = 64$, the present scheme can achieve $\gamma_s = 1.18$ dB, $\text{CER}_s = 1.5$, $\text{PAR}_s = 3.76$ using 160 additions, 4 multiplications, and 2.6 kbytes of memory. As an alternative, we can achieve the same tradeoff point using about one multiply-add per two dimensions and a memory of 1.0 kbyte. As another example for $N = 64$, we can achieve $\gamma_s = 1.0$ dB, $\text{CER}_s = 1.25$, $\text{PAR}_s = 3.0$ using 100 additions, 2 multiplications, and 1.1 kbytes of memory.

For $N = 128$, we need 2 multiplications, 2 divisions, 300 additions, and a total memory of 17 kbytes to achieve $\text{CER}_s = 1.5$, $\gamma_s = 1.27$ dB. As an alternative, we can achieve the same tradeoff point with about one multiplication per two dimensions and a total memory of 6.6 kbytes. As another example for $N = 128$, we need one multiplication, one division, 195 additions, and a total memory of 6.6 kbytes to achieve $\text{CER}_s = 1.25$, $\gamma_s = 1.07$ dB.

From the numerical comparisons give above we can conclude that the present scheme is best suited for realizing near-optimum gains in spaces of high dimensionality, i.e., $N \geq 64$ while the scheme of [17] is best suited for $N \leq 32$.

IV. Summary

We have introduced an efficient addressing scheme for nearly optimum shaping of multidimensional signal spaces. This is based on partitioning the space into union of blocks such that the addressing of points within the blocks has a low complexity. Addressing of blocks is based on a recursive relationship which allows us to decompose the corresponding problem into smaller parts of a lower complexity.

References

[1] G. D. Forney, Jr. and L. F. Wei, "Multidimensional constellations–Part I: Introduction, figures of merit, and generalized cross constellations," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 877–892, Aug. 1989.

[2] A. K. Khandani and P. Kabal, "Shaping multi-dimensional signal spaces—Part II: Shell-addressed constellations," *IEEE Trans. Inform. Theory*, vol. 39, pp. 1809–1819, Nov. 1993.

[3] A. R. Calderbank and L. H. Ozarow, "Non-equiprobable signaling on the gaussian channel," *IEEE Trans. Inform. Theory*, vol. 36, pp. 726–740, July 1990.

[4] J. P. M. Schalkwijk, "An algorithm for source coding," *IEEE Trans. Inform. Theory*, vol. IT-18, no. 3, pp. 395–399, May 1972.

[5] L. F. Wei, "Trellis coded modulation with multidimensional constellations," *IEEE Trans. Inform. Theory*, vol. IT–33, pp. 483–501, July 1987.

[6] J. H. Conway and N. J. A. Sloane, "A fast encoding method for lattice codes and quantizers," *IEEE Trans. Inform. Theory*, vol. IT-31, pp. 106–109, Jan. 1985.

[7] G. D. Forney, Jr., "Multidimensional constellations—Part II: Voronoi constellations," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 941–958, Aug. 1989.

[8] ——, "Trellis shaping," *IEEE Trans. Inform. Theory*, vol. 38, pp. 281–300, Mar. 1992.

[9] G. R. Lang and F. M. Longstaff, "A Leech lattice modem," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 968–973, Aug. 1989.

[10] T. R. Fischer, "A pyramid vector quantizer," *IEEE Trans. Inform. Theory*, vol. IT-32, pp. 568–583, July 1986.

[11] F. R. Kschischang, "Shaping and coding gain criteria in signal constellation design," Ph.D. dissertation, Toronto Univ., Toronto, Ont., Canada, June 1991.

[12] F. R. Kschischang and S. Pasupathy, "Optimal shaping properties of the truncated polydisc," *IEEE Trans. Inform. Theory*, vol. 40, pp. 892–903, May 1994.

[13] R. Laroia, N. Farvardin, and S. A. Tretter, "On optimal shaping of multi-dimensional constellations," *IEEE Trans. Inform. Theory*, vol. 40, pp. 1044–1056, July 1994.

[14] F. R. Kschischang and S. Pasupathy, "Optimal nonuniform signaling for gaussian channels," *IEEE Trans. Inform. Theory*, vol. 39, pp. 913–929, May 1993.

[15] J. R. Livingston, "Shaping using variable-size regions," *IEEE Trans. Inform. Theory*, vol. 38, pp. 1347–1353, July 1992.

[16] F. R. Kschischang, "Huffman codes for shaping gain," in *16th Biennial Symp. on Communications* (Kingston, Ont., Canada, May 27–29, 1992.), pp. 79–82.

[17] A. K. Khandani and P. Kabal, "Shaping of multi-dimensional signal constellations using a lookup table," *IEEE Trans. Inform. Theory*, vol. 40, pp. 2058–2062, Nov 1994.

[18] Motorola Information Systems Group, "Signal mapping and shaping for V. fast," Contribution D196, CCITT Study Group XVII, June 1992.

[19] G. D. Forney, Jr., "Advances in modem technology since V.32/V .32bis," in *Int. Conf. on Data Transmission—Advances in Modem and ISDN Technology and Applications*, (London, U.K., IEE, Sept. 1992), pp. 1–6.

[20] A. K. Khandani and P. Kabal, "Shaping multidimensional signal constellation," in *IEEE Int. Symp. Inform. Theory.* (Budapest, Hungary, June 1991), p. 4.

Optimal Binary Index Assignments for a Class of Equiprobable Scalar and Vector Quantizers

Steven W. McLaughlin, *Member, IEEE*,
David L. Neuhoff, *Fellow, IEEE*,
and Jonathan J. Ashley, *Member, IEEE*

***Abstract*—The problem of scalar and vector quantization in conjunction with a noisy binary symmetric channel is considered. The issue is the assignment of the shortest possible distinct binary sequences to quantization levels or vectors so as to minimize the mean-squared error caused by channel errors. By formulating the assignment as a matrix (or vector in the scalar case) and showing that the mean-squared error due to channel errors is determined by the projections of its columns onto the eigenspaces of the multidimensional channel transition matrix, a class of source/quantizer pairs is identified for which the optimal index assignment has a simple and natural form. Among other things, this provides a simpler and more accessible proof of the result of Crimmins *et al.*, that the natural binary code is an optimal index assignment for the uniform scalar quantizer and uniform source. It also provides a potentially useful approach to further developments in source-channel coding.**

***Index Terms*— Source-channel coding, natural binary code, vector quantization.**

I. Introduction

We consider the problem of scalar and vector quantization in conjunction with a binary symmetric channel. We seek the nonredundant assignment of distinct binary sequences or *indices* to quantization levels or vectors that minimizes the mean-squared error caused by channel errors. The index assignment problem considered has previously been addressed in [1]–[14]. See [14] for a discussion of other work prior to 1984. Some contemporary work includes [15], [16].

Manuscript received September 13, 1993; revised May 30, 1995. This work was partially supported by a grant from The IBM Almaden Research Center. The material in this correspondence was presented in part at the Joint Dimacs/IEEE Workshop on Coding and Quantization, October 1992.

S. W. McLaughlin was with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA. He is now with the Department of Electrical Engineering, Rochester Institute of Technology, Rochester, NY 14623 USA.

D. L. Neuhoff is with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA.

J. J. Ashley is with the IBM Almaden Research Center, San Jose, CA 95120 USA.

IEEE Log Number 9414823.