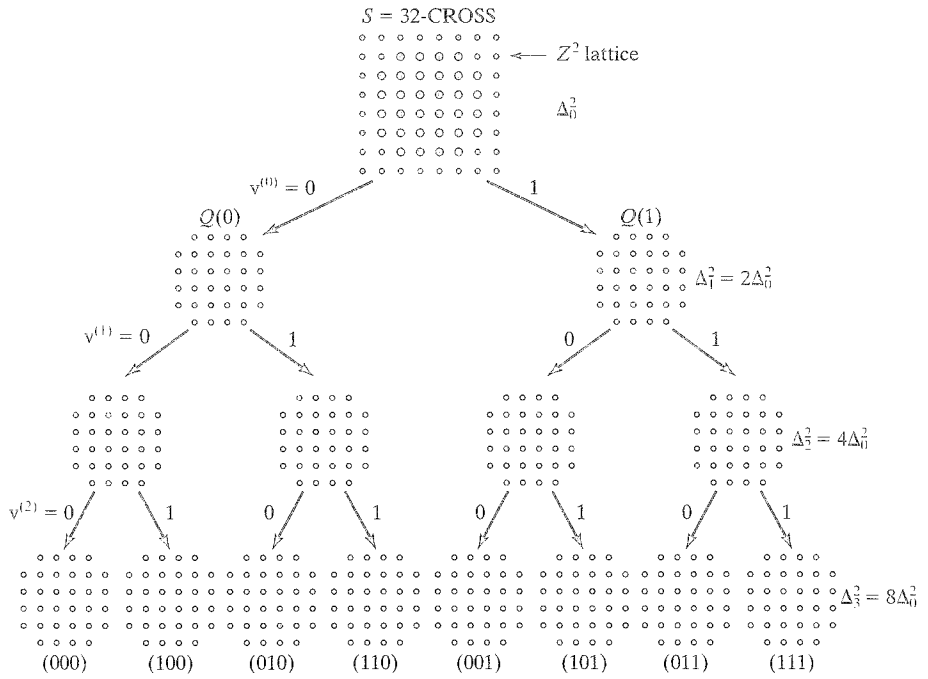


FIGURE 18.28: Three-level partitioning of the naturally mapped 32-CROSS signal constellation.

shown in Figure 18.27, has free distance $d^2_{free} = 5$ and average number of nearest neighbors $A_{d_{free}} = 6.716$, and achieves a real coding gain of 3.6 dB at a BER of 10^{-5} compared with uncoded 16-QAM ($\eta = 4.0$) and 64-QAM ($\eta = 6.0$), respectively, without bandwidth expansion. (The fractional value of $A_{d_{free}}$ is due to the nonlinearity of the code and the boundary effects of the constellation.)

An equivalent encoder, described in Example 18.12 and sketched in Figure 18.23, was designed using a systematic code search. When this encoder is used with the naturally mapped 32-CROSS constellation (see Figure 18.28), it requires only one AND gate and one differentially encoded information bit, and differential encoding can be embedded within the encoder. (Note, as mentioned in Section 18.2, that level 3 in the partition tree is an example of a case in which not all subsets at the same level are isomorphic.)

For two-dimensional signal constellations, since it is impossible to achieve 90° invariance with linear codes (the best that can be done is 180° invariance), nonlinear codes are needed for full rotational invariance. This was the crucial insight made by Wei [28] in the design of the V.32 code.

18.5 MULTIDIMENSIONAL TCM

Up to this point in our discussion of TCM we have considered only the case in which the $(k + 1)$ convolutional encoder output bits at each time unit are mapped into one

signal point drawn from a 2^{k+1}-ary constellation described in one-dimensional (1-D) or two-dimensional (2-D) Euclidean space. In this section we consider the more general case of designing TCM systems for use with multidimensional (multi-D) signal sets, pioneered by Wei [29].

Although it is possible to construct multi-D signal sets directly in multidimensional Euclidean space, we consider here only the more practical case of constructing L-dimensional or $2L$-dimensional signal sets as the L-fold Cartesian product of a 1-D or 2-D signal set. First, let $S = \{s_0, s_1, \cdots, s_{M-1}\}$ be a 1-D or 2-D signal set with $M = 2^I$ signal points. Then, the L-fold Cartesian product

$$S^L = S \times S \times \cdots \times S \quad (L \text{ times}) \tag{18.66}$$

represents the L-dimensional or $2L$-dimensional signal set consisting of all possible combinations of L signal points drawn from S, and the multi-D signal set S^L contains a total of 2^{IL} signal points. Transmitting a signal from the multi-D signal set S^L is then equivalent to transmitting L signals from the constituent 1-D or 2-D signal set S, and the multi-D signal $\mathbb{Y}_l$ transmitted at time unit l is represented as the L-tuple $\mathbb{Y}_l = (y_{l1}, y_{l2}, \cdots, y_{lL})$, where y_{li}, $1 \le i \le L$, represents a signal from the constituent signal set S. For example, 3×8-PSK denotes the six-dimensional (6-D) signal set consisting of all $8^3 = 512$ possible combinations of three 8-PSK signals, and the 3-tuple $\mathbb{Y}_l = (y_{l1}, y_{l2}, y_{l3})$ represents the 3×8-PSK signal transmitted at time unit l.

In a TCM system design using a multi-D signal set, the $(k + 1)$ convolutional encoder output bits at each time unit are mapped into a signal point drawn from the multi-D 2^{IL}-ary signal constellation S^L, so that $k + 1 = IL$. Because each time unit now corresponds to the transmission of L signals from the elementary signal set S, each branch of the trellis representing the encoder/modulator is labeled with L signals from S. In decoding, the soft Euclidean metrics for each of the L signal labels on a branch are added to form the branch metrics.

Multi-D TCM systems have several advantages compared with 1-D and 2-D TCM systems:

1. Fractional values of spectral efficiency can be realized.
2. Full rotational invariance can be achieved with linear codes.
3. Additional power savings can be realized using a technique called *shaping*.
4. The signal sets have a smaller peak-to-average power ratio.
5. Higher-speed decoding can be achieved.

The first two advantages are explained in detail later in this section, and advantages 3 and 4 are illustrated in an example discussing the V.34 high-speed modem standard at the end of the section. The higher decoding speed stems from the fact that the metric calculations on a trellis branch are normally done with quantized lookup tables. Thus, for multi-D TCM with L signals on each branch, as opposed to 1-D or 2-D TCM with only one signal per branch, performing the branch metric calculations in parallel results in a decoding speed advantage equal to a factor of L.

To better explain the basics of multi-D TCM design, we now introduce an example.

EXAMPLE 18.15 8-State, Rate $R = 2/3$ Trellis-Coded 2×8-PSK

In this example we consider an 8-state, rate $R = 2/3$, linear convolutional encoder along with 3 uncoded information bits, that is, a rate $R = k/(k+1) = 5/6$ encoder with $\tilde{k} = 2$, mapped into a four-dimensional (4-D) 2×8-PSK signal set. The multi-D TCM system encodes 5 information bits per unit time. Because two 8-PSK signals are transmitted per unit time, the spectral efficiency is

$$\eta = 5/2 = 2.5 \text{ bits/symbol.} \tag{18.67}$$

We begin by considering the partitioning of the 2×8-PSK signal set $S^2 = S \times S$, where S denotes the constituent 2-D 8-PSK signal set. Because this signal set has $8^2 = 64$ signal points, a binary partitioning tree will contain $k+1 = 6$ levels, and the partitioning should maximize the MSSDs Δ_p^2 at each level p in the tree. The best binary partitioning of the 2×8-PSK signal set is illustrated in Figure 18.29, where $Q^2\left(v_1^{(0)} \times v_2^{(0)} \cup v_1^{(0)} \times v_2^{(0)}\right)$, $Q^2\left(v_1^{(0)} \times v_2^{(0)}\right)$, $Q^2\left(v_1^{(1)}v_1^{(0)} \times v_2^{(1)}v_2^{(0)} \cup v_1^{(1)}v_1^{(0)} \times v_2^{(1)}v_2^{(0)}\right)$, $Q^2\left(v_1^{(1)}v_1^{(0)} \times v_2^{(1)}v_2^{(0)}\right)$, $Q^2\Big(v_1^{(2)}v_1^{(1)}v_1^{(0)} \times v_2^{(2)}v_2^{(1)}v_2^{(0)} \cup v_1^{(2)}v_1^{(1)}v_1^{(0)} \times v_2^{(2)}v_2^{(1)}v_2^{(0)}\Big)$, and $Q^2\left(v_1^{(2)}v_1^{(1)}v_1^{(0)} \times v_2^{(2)}v_2^{(1)}v_2^{(0)}\right)$ refer to 4-D subsets in the 2×8-PSK signal set. (The subscripts 1 and 2 on the label bits refer to the first and second signal, respectively, in the 4-D signal set.) For example, $Q^2(0 \times 0 \cup 1 \times 1)$ represents the 4-D subset $Q(0) \times Q(0) \cup Q(1) \times Q(1)$, which contains all pairs of points from the 2-D subset $Q(0)$ and all pairs of points from the 2-D subset $Q(1)$, and $Q^2(00 \times 00)$ represents the 4-D subset $Q(00) \times Q(00)$ containing all pairs of points from the 2-D subset $Q(00)$ (see Figure 18.12).

The MSSDs Δ_p^2 at each level p in the partitioning tree are also indicated in Figure 18.29. Note that the distances Δ_p^2 increase as we proceed down the partitioning tree, a property characteristic of any good signal set partitioning. To see how the values of Δ_p^2 are determined, we represent each 4-D signal point $\mathbb{Y} = (y_1, y_2)$ as a pair of integer labels of the 8-PSK signal set; for example, $\mathbb{Y} = (0, 0)$ represents the 4-D signal point corresponding to two uses of the 8-PSK signal labeled 0. At partition level $p = 0$ we see that there are signal pairs such as $\mathbb{Y} = (0, 0)$ and $\mathbb{Y}' = (0, 1)$ for which $d_E^2(\mathbb{Y}, \mathbb{Y}') = 0 + 0.586 = 0.586$, so that $\Delta_0^2 = 0.586$. At partition level $p = 1$ we see that there are signal pairs such as $\mathbb{Y} = (0, 0)$ and $\mathbb{Y}' = (1, 7)$, both belonging to subset $Q^2(0 \times 0 \cup 1 \times 1)$, for which $d_E^2(\mathbb{Y}, \mathbb{Y}') = 0.586 + 0.586 = 1.172$, so that $\Delta_1^2 = 1.172$. At partition level $p = 2$ we see that there are signal pairs such as $\mathbb{Y} = (0, 0)$ and $\mathbb{Y}' = (0, 2)$, both belonging to subset $Q^2(0 \times 0)$, for which $d_E^2(\mathbb{Y}, \mathbb{Y}') = 0 + 2.0 = 2.0$, so that $\Delta_2^2 = 2.0$. At partition level $p = 3$ we see that there are signal pairs such as $\mathbb{Y} = (0, 0)$ and $\mathbb{Y}' = (2, 6)$, both belonging to subset $Q^2(00 \times 00 \cup 10 \times 10)$, for which $d_E^2(\mathbb{Y}, \mathbb{Y}') = 2.0 + 2.0 = 4.0$, so that $\Delta_3^2 = 4.0$, and so on.

From the set-partitioning encoder diagram shown in Figure 18.14 we see that at each time unit, the three output bits of a rate $R = 2/3$ encoder select one of the subsets at partition level $p = 3$, and then the three uncoded bits select one of the 4-D signal points. Thus, these 6 bits determine the two 8-PSK signals to be transmitted in a given time unit. The mapping from these 6 bits to a particular pair of 8-PSK signals

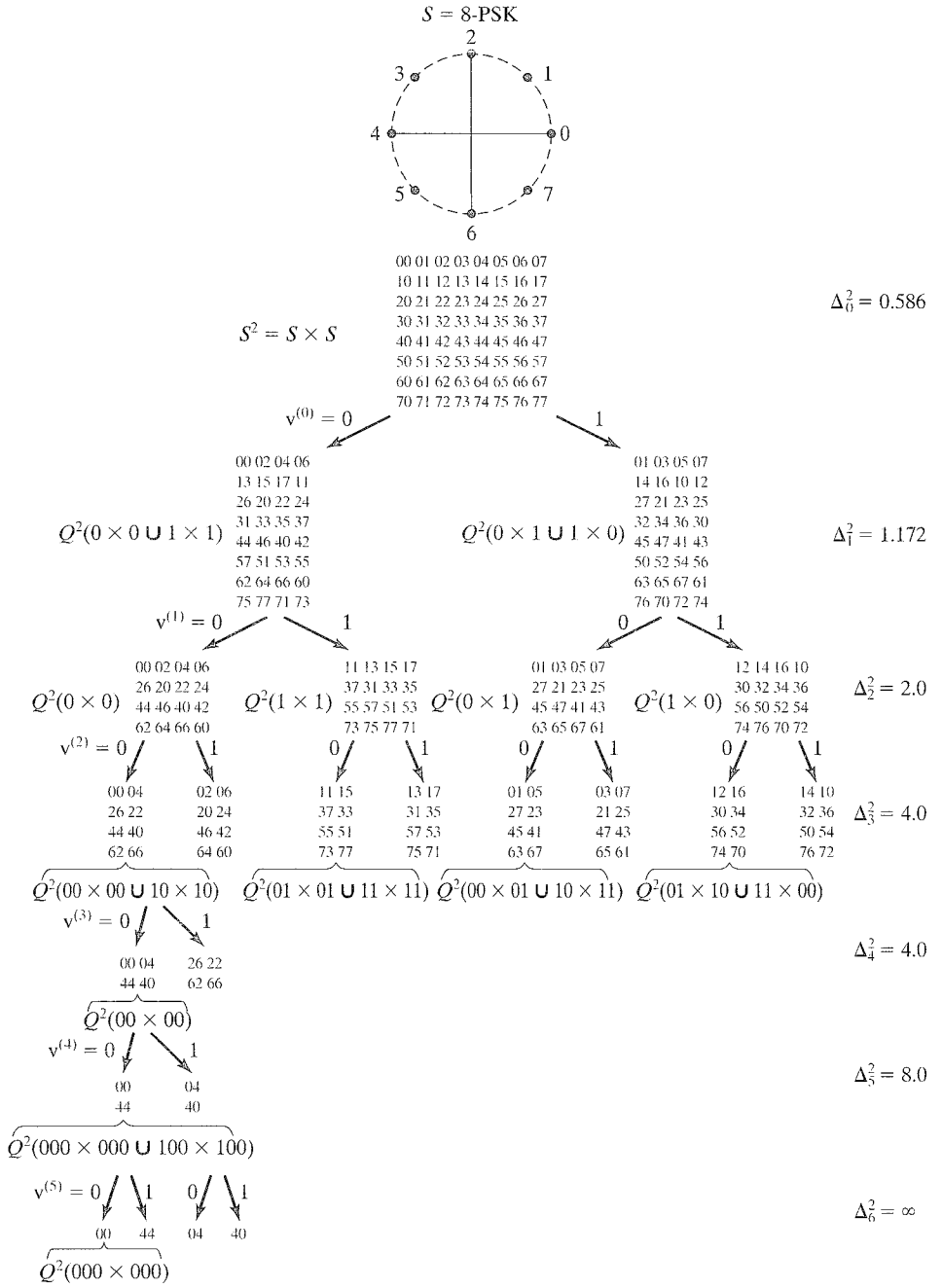


FIGURE 18.29: Partitioning tree for 2×8-PSK.

depends on the way the subsets in the partitioning tree are formed. For example, the subset $Q^2(0 \times 1 \cup 1 \times 0)$ at partition level $p = 1$ is generated as the coset of $Q^2(0 \times 0 \cup 1 \times 1)$ by adding a signal pair $\mathbb{g}_0$ ($\mathbb{g}_0 \in S^2, \mathbb{g}_0 \notin Q^2(0 \times 0 \cup 1 \times 1)$), called a *coset generator*, to each member of the subset $Q^2(0 \times 0 \cup 1 \times 1)$. Choosing the coset generator as $\mathbb{g}_0 = (0, 1)$ and using integer notation for the signal points within each subset (see Figure 18.29), we can represent the coset generation as

$$\begin{aligned} Q^2(0 \times 1 \cup 1 \times 0) &= Q^2(0 \times 0 \cup 1 \times 1) + \mathbb{g}_0 \\ &= Q^2(0 \times 0 \cup 1 \times 1) + (0, 1) \pmod 8, \end{aligned} \tag{18.68}$$

where the addition is modulo-8. At partition level $p = 2$, we form the subsets $Q^2(1 \times 1)$, $Q^2(0 \times 1)$, and $Q^2(1 \times 0)$ as cosets of the subset $Q^2(0 \times 0)$ by choosing a second coset generator $\mathbb{g}_1$ ($\mathbb{g}_1 \in Q^2(0 \times 0 \cup 1 \times 1), \mathbb{g}_1 \notin Q^2(0 \times 0)$). For the choice $\mathbb{g}_1 = (1, 1)$, we form the subsets at level 2 as

$$Q^2(1 \times 1) = Q^2(0 \times 0) + \mathbb{g}_1 = Q^2(0 \times 0) + (1, 1) \pmod 8 \tag{18.69a}$$

$$Q^2(0 \times 1) = Q^2(0 \times 0) + \mathbb{g}_0 = Q^2(0 \times 0) + (0, 1) \pmod 8 \tag{18.69b}$$

$$Q^2(1 \times 0) = Q^2(0 \times 0) + \mathbb{g}_1 + \mathbb{g}_0 = Q^2(0 \times 0) + (1, 2) \pmod 8. \tag{18.69c}$$

Similarly, at partition level $p = 3$, we choose a coset generator $\mathbb{g}_2$ ($\mathbb{g}_2 \in Q^2(0 \times 0), \mathbb{g}_2 \notin Q^2(00 \times 00 \cup 10 \times 10)$) to generate the subsets. For the choice $\mathbb{g}_2 = (0, 2)$, the subsets at level 3 are formed as

$$\begin{aligned} Q^2(00 \times 10 \cup 10 \times 00) &= Q^2(00 \times 00 \cup 10 \times 10) + \mathbb{g}_2 \\ &= Q^2(00 \times 00 \cup 10 \times 10) + (0, 2) \pmod 8 \end{aligned} \tag{18.70a}$$

$$\begin{aligned} Q^2(01 \times 01 \cup 11 \times 11) &= Q^2(00 \times 00 \cup 10 \times 10) + \mathbb{g}_1 \\ &= Q^2(00 \times 00 \cup 10 \times 10) + (1, 1) \pmod 8 \end{aligned} \tag{18.70b}$$

$$\begin{aligned} Q^2(01 \times 11 \cup 11 \times 01) &= Q^2(00 \times 00 \cup 10 \times 10) + \mathbb{g}_2 + \mathbb{g}_1 \\ &= Q^2(00 \times 00 \cup 10 \times 10) + (1, 3) \pmod 8 \end{aligned} \tag{18.70c}$$

$$\begin{aligned} Q^2(00 \times 01 \cup 10 \times 11) &= Q^2(00 \times 00 \cup 10 \times 10) + \mathbb{g}_0 \\ &= Q^2(00 \times 00 \cup 10 \times 10) + (0, 1) \pmod 8 \end{aligned} \tag{18.70d}$$

$$\begin{aligned} Q^2(00 \times 11 \cup 10 \times 01) &= Q^2(00 \times 00 \cup 10 \times 10) + \mathbb{g}_2 + \mathbb{g}_0 \\ &= Q^2(00 \times 00 \cup 10 \times 10) + (0, 3) \pmod 8 \end{aligned} \tag{18.70e}$$

$$\begin{aligned} Q^2(01 \times 10 \cup 11 \times 00) &= Q^2(00 \times 00 \cup 10 \times 10) + \mathbb{g}_1 + \mathbb{g}_0 \\ &= Q^2(00 \times 00 \cup 10 \times 10) + (1, 2) \pmod 8 \end{aligned} \tag{18.70f}$$

$$\begin{aligned} Q^2(01 \times 00 \cup 11 \times 10) &= Q^2(00 \times 00 \cup 10 \times 10) + \mathbb{g}_2 + \mathbb{g}_1 + \mathbb{g}_0 \\ &= Q^2(00 \times 00 \cup 10 \times 10) + (1, 4) \pmod 8. \end{aligned} \tag{18.70g}$$

The subsets at the remaining three levels of the partition tree can be formed in a similar fashion. For the choices of coset generators $\mathbb{g}_3 = (2, 2)$, $\mathbb{g}_4 = (0, 4)$, and

$\mathbb{g}_5 = (4, 4)$, the mapping from the binary code vector $\mathbb{v} = (v^{(5)}v^{(4)}v^{(3)}v^{(2)}v^{(1)}v^{(0)})$ to the 8-ary signal vector $\mathbb{Y} = (y_1, y_2)$ can be given as follows:

$$\begin{aligned}\mathbb{Y} = (y_1, y_2) &= \sum_{0 \le p \le 5} v^{(p)}\mathbb{g}_p \pmod 8 \\ &= v^{(5)}(4, 4) + v^{(4)}(0, 4) + v^{(3)}(2, 2) + v^{(2)}(0, 2) \\ &\quad + v^{(1)}(1, 1) + v^{(0)}(0, 1) \pmod 8, \end{aligned} \tag{18.71}$$

where $v^{(5)}$, $v^{(4)}$, and $v^{(3)}$ represent the three uncoded bits, and $v^{(2)}$, $v^{(1)}$, and $v^{(0)}$ represent the three encoder output bits. For example, the code vector $\mathbb{v}$ =(111010) generates the signal vector $\mathbb{Y} = (4, 4) + (0, 4) + (2, 2) + (1, 1) = (7, 3)$. If the coset generators are given in binary form as 3-bit column vectors

$$\begin{aligned} \mathbb{g}_0 &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \quad \mathbb{g}_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix} \quad \mathbb{g}_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \\ \mathbb{g}_3 &= \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 0 \end{bmatrix} \quad \mathbb{g}_4 = \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \mathbb{g}_5 = \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \end{aligned}, \tag{18.72}$$

then the mapping from the binary code vector $\mathbb{v}$ to the binary signal vector $\mathbb{Y} = (\mathbb{y}_1, \mathbb{y}_2) = \left[\left(y_1^{(2)} y_1^{(1)} y_1^{(0)}\right)^T, \left(y_2^{(2)} y_2^{(1)} y_2^{(0)}\right)^T \right]$ is given as

$$\begin{aligned}\mathbb{Y} = (\mathbb{y}_1, \mathbb{y}_2) &= \sum_{0 \le p \le 5} v^{(p)}\mathbb{g}_p \\ &= v^{(5)} \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \oplus v^{(4)} \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \oplus v^{(3)} \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 0 \end{bmatrix} \oplus \\ &\quad v^{(2)} \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \oplus v^{(1)} \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix} \oplus v^{(0)} \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \end{aligned} \tag{18.73}$$

and for $\mathbb{v} = (111010)$, we see that $\mathbb{Y} = \left[(111)^T, (011)^T\right]$, the binary equivalent of $\mathbb{Y} = (7, 3)$. The mapping function represented by (18.73) is shown in Figure 18.30.

Next we must choose the 8-state, rate $R = 2/3$ encoder to maximize the overall MFSE distance d^2_{free}. From (18.18) we see that d^2_{free} is the minimum of δ^2_{free}, the minimum distance between trellis paths, and δ^2_{min}, the minimum distance between parallel transitions. The value of δ^2_{min} can be determined directly from the partitioning tree as the MSSD at partition level $p = 3$, so that $\delta^2_{min} = \Delta^2_3 = 4.0$. Thus, the encoder must be chosen to make δ^2_{free} as large as possible. The best 8-state, rate $R = 2/3$ encoder has parity-check polynomials (in octal form) $\mathbb{h}^{(2)} = (0\,4)$, $\mathbb{h}^{(1)} = (0\,6)$, and $\mathbb{h}^{(0)} = (1\,1)$, and $\delta^2_{free} = 2.929$, so that $d^2_{free} = \min\{2.929, 4.0\} = 2.929$. To determine the asymptotic coding gain γ in this case, we must compare

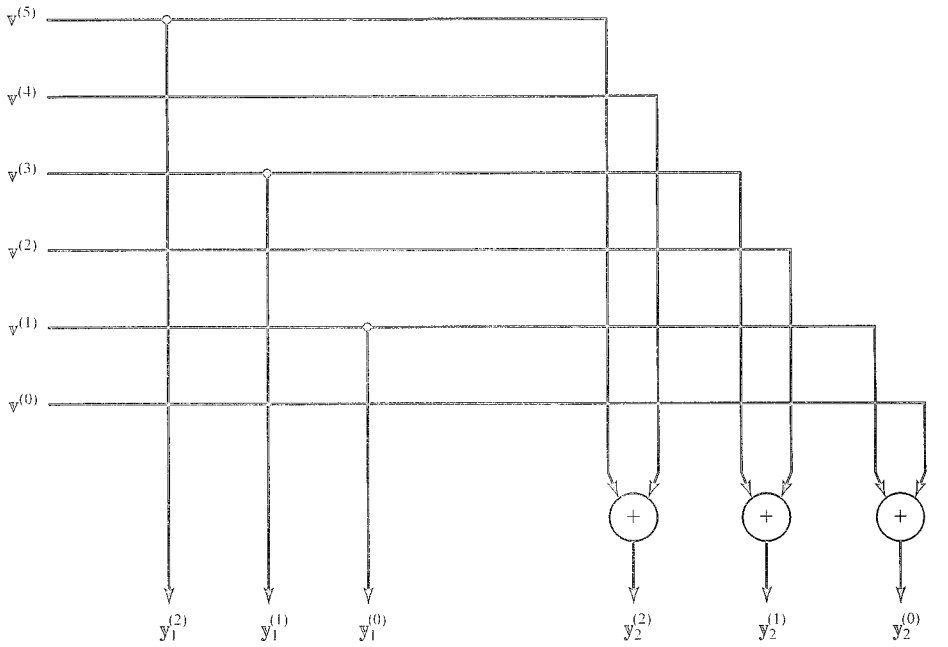


FIGURE 18.30: A 2×8-PSK signal set mapper.

with an uncoded system with the same spectral efficiency as the coded system, that is, $\eta = 2.5$ bits/symbol. For this purpose we consider the 2×8-PSK subset $Q^2(0 \times 0 \cup 1 \times 1)$ consisting of 32 signal points, which has $\eta = 2.5$ bits/symbol and $d^2_{min} = 1.172$. Thus, $\gamma = 10\log_{10}(d^2_{free}/d^2_{min}) = 10\log_{10}(2.929/1.172) = 3.98$ dB. The overall encoder block diagram for this 4-D TCM system is shown in Figure 18.31. Note that the encoder delays in this case are for a $2T$-second time interval, that is, the time required to transmit two 8-PSK signals.

The following remarks apply to Example 18.15.

- Unlike 2-D signal sets, multi-D signal sets are capable of achieving fractional spectral efficiencies.
- The particular choices for coset generators determine the mapping from encoder output bits to signal points, but they do not affect the partitioning itself.
- The overall MFSE distance d^2_{free} can be increased in two ways, either by using more encoder states, thereby increasing δ^2_{free}, and/or by encoding more bits, that is, using a larger $\tilde{k}$, thereby increasing δ^2_{min} (see Problem 18.28).

We now generalize the preceding example into a design procedure for multi-D TCM. The major steps in the design procedure are as follows:

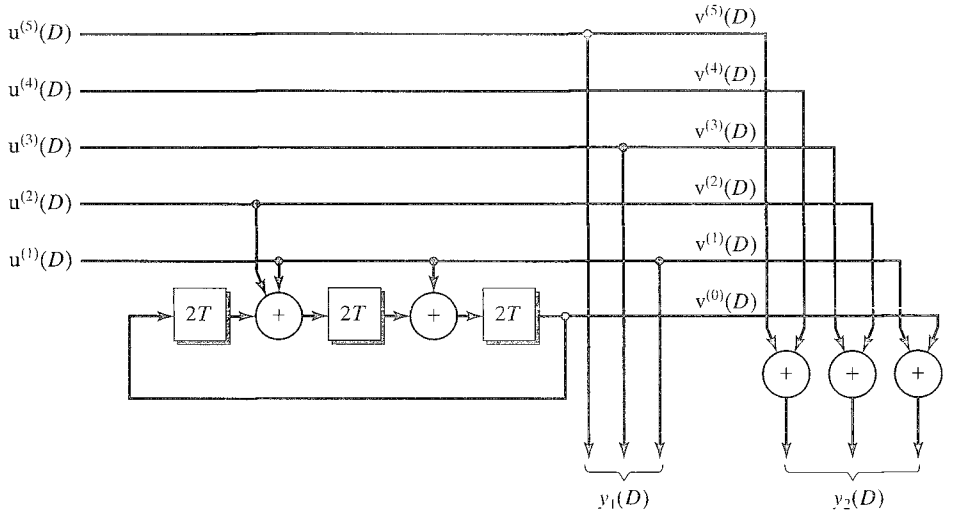


FIGURE 18.31: Encoder block diagram for 8-state, rate $R = 2/3$ trellis-coded 2×8-PSK.

1. Partition the multi-D signal space S^L such that the MSSD Δ_p^2 is maximized at each partition level $p = 1, 2, \cdots, k$.
2. Select a rate $\tilde{k}/(\tilde{k}+1)$, $1 \le \tilde{k} \le k$, convolutional encoder to maximize δ_{free}^2, the minimum distance between trellis paths, and minimize $A_{d_{free}}$, the number of nearest-neighbor codewords, for some given number of states.

The partitioning of the multi-D signal set S^L uses the multilevel coding technique first introduced by Imai and Hirakawa [30] and developed further in [31]. We begin by representing each of the 2^I signal points in the constituent 2-D signal set S by a binary vector $\mathbf{y} = (y^{(I-1)}, \cdots, y^{(1)}, y^{(0)})$, where $y^{(j)}$ determines the subset selected at level $j+1$ in the complete binary partitioning tree for S, and by letting the MSSDs in the partitioning tree be represented by $\delta_0^2 \le \delta_1^2 \le \cdots \le \delta_I^2 = \infty$. (At level I in the complete tree, there is only one signal point in each subset, so $\delta_I^2 = \infty$.) Then, we can represent the 2^{IL} multi-D signal points $\mathbb{Y}$ in S^L as an $I \times L$ binary matrix

$$\begin{aligned} \mathbb{Y} &= (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_L) \\ &= \begin{bmatrix} y_1^{(I-1)} & y_2^{(I-1)} & \cdots & y_L^{(I-1)} \\ \vdots & \vdots & \cdots & \vdots \\ y_1^{(1)} & y_2^{(1)} & \cdots & y_L^{(1)} \\ y_1^{(0)} & y_2^{(0)} & \cdots & y_L^{(0)} \end{bmatrix}, \end{aligned} \tag{18.74}$$

where each column $\mathbf{y}_i = \left(y_i^{(I-1)}, \cdots, y_i^{(1)}, y_i^{(0)}\right)^T$, $1 \le i \le L$, of $\mathbb{Y}$ is the binary representation of one of the 2^I signal points in S. (Alternatively, we can represent

$\mathbb{Y}$ in integer form as the vector $(y_1, y_2, \cdots, y_L)$ by expressing each binary column $\mathbf{y}_i$ as the integer $y_i = \sum_{0 \le j \le I-1} y_i^{(j)} 2^j$, $1 \le i \le L$.)

Each row of $\mathbb{Y}$ is a binary L-tuple and can be regarded as a codeword in the $(L, L, 1)$ binary block code $\mathbb{C}_0$. (An (n, k, d) binary block code has block length n, dimension k, and minimum Hamming distance d.) Then, we can represent the multi-D signal set S^L as the *I-level block code*

$$\Lambda_0 = \mathbb{C}_0 * \mathbb{C}_0 * \cdots * \mathbb{C}_0, \quad (I \text{ times}) \tag{18.75}$$

where Λ_0 represents the set of all 2^{IL} possible combinations of I codewords drawn from $\mathbb{C}_0$, and $\mathbb{Y}$ represents a particular member of the set. The first code in the chain of I codes that constitute Λ_0 (corresponding to the bottom row of $\mathbb{Y}$) determines the L bits that select the L level-1 subsets in the L partitioning trees for S, whereas the last code in the chain (corresponding to the top row of $\mathbb{Y}$) determines the L bits that select the L signal points at level I.

To partition Λ_0 we begin by forming a sequence of linear subcodes of $\mathbb{C}_0$, denoted by $\mathbb{C}_0, \mathbb{C}_1, \cdots, \mathbb{C}_L$, such that

$$\mathbb{C}_L \subset \mathbb{C}_{L-1} \subset \cdots \subset \mathbb{C}_1 \subset \mathbb{C}_0, \tag{18.76}$$

and the dimension of subcode $\mathbb{C}_i$ is given by

$$\dim(\mathbb{C}_i) = L - i, \quad 0 \le i \le L. \tag{18.77}$$

(Note that the $(L, 0, \infty)$ code $\mathbb{C}_L$ contains only the all-zero codeword.) If d_i represents the minimum distance of subcode $\mathbb{C}_i$, then

$$d_0 \le d_1 \le \cdots \le d_{L-1} < d_L, \tag{18.78}$$

where $d_0 = 1$, and $d_L = \infty$. We will see shortly that to maximize the MSSDs Δ_p^2 at each partition level p in the multi-D partitioning tree, we should choose each subcode $\mathbb{C}_i$ to maximize its minimum Hamming distance d_i.

To divide the multilevel code Λ_0 (multi-D signal set S^L) into 2^{IL} subsets, each containing a single point, requires a binary partition tree with IL levels. Thus, we use the code sequence $\mathbb{C}_0, \mathbb{C}_1, \cdots, \mathbb{C}_L$ to form a sequence of multilevel subcodes of Λ_0, denoted by $\Lambda_0(0), \Lambda_1(0), \cdots, \Lambda_{IL}(0)$, such that

$$\Lambda_{IL}(0) \subset \Lambda_{IL-1}(0) \subset \cdots \subset \Lambda_1(0) \subset \Lambda_0(0), \tag{18.79}$$

and the dimension of multilevel subcode $\Lambda_i(0)$ is given by

$$\dim[\Lambda_i(0)] = IL - i, \quad 0 \le i \le IL. \tag{18.80}$$

where $\Lambda_0(0) \equiv \Lambda_0$ and $\Lambda_{IL}(0)$ contains I copies of the all-zero codeword. Our goal in designing this IL-level partition tree is to maximize the MSSDs Δ_p^2 at each level.

Before proceeding with the design of a multi-D partitioning tree, we note that the SE distance between two multi-D signal points $\mathbb{Y}$ and $\mathbb{Y}'$ is simply the sum of the SE distances between the corresponding 2-D signal points; that is,

$$d_E^2(\mathbb{Y}, \mathbb{Y}') = \sum_{1 \le i \le L} d_E^2(\mathbf{y}_i, \mathbf{y}_i'). \tag{18.81}$$

For the entire multi-D signal set S^L, the MSE distance is given by $\Delta_0^2 = \delta_0^2$, the MSE distance of the constituent 2-D signal set S. This follows from the fact that two distinct multi-D signals, each represented by a vector of L signal points, may differ in only one component, in which case (18.81) contains only one nonzero term. Because the MSE distance in the constituent 2-D signal set is δ_0^2, we obtain $\Delta_0^2 = \delta_0^2$. In terms of the multilevel code representation $\Lambda_0(0)$, we note that two distinct multilevel codewords, each represented by a set of I binary codewords of length L from the code $\mathbb{C}_0$, may differ in only one codeword. If the two multilevel codewords differ only in the jth codeword, then the MSE distance is $d_0\delta_j^2$. Minimizing over all j, $0 \le j \le I-1$, we can represent the minimum distance between two multi-D signal points as

$$\Delta_0^2 = \min\left(d_0\delta_0^2, d_0\delta_1^2, \cdots, d_0\delta_{I-1}^2\right) = \delta_0^2, \tag{18.82}$$

since $d_0 = 1$, and $\delta_0^2 \le \delta_1^2 \le \cdots \le \delta_{I-1}^2$.

Now, to partition $\Lambda_0(0)$, of size 2^{IL}, into two subsets of size 2^{IL-1}, namely, the subcode $\Lambda_1(0)$ and its coset $\Lambda_1(1)$, we must replace one of the codes $\mathbb{C}_0$ of size 2^L in the I-level code $\mathbb{C}_0 * \mathbb{C}_0 * \cdots * \mathbb{C}_0$ with the code $\mathbb{C}_1$ of size 2^{L-1}. Further, we should do this in such a way that the MSSD Δ_1^2 at level 1 in the multi-D partitioning tree is maximized. If the code $\mathbb{C}_1$ is inserted at the jth level, the multilevel subcode $\Lambda_1(0)$ is given by $\mathbb{C}_0 * \cdots * \mathbb{C}_0 * \mathbb{C}_1 * \mathbb{C}_0 * \cdots * \mathbb{C}_0$, and an argument similar to that used to obtain (18.82) gives

$$\Delta_1^2 = \min\left(d_0\delta_0^2, \cdots, d_0\delta_{j-1}^2, d_1\delta_j^2, d_0\delta_{j+1}^2, \cdots, d_0\delta_{I-1}^2\right); \tag{18.83}$$

that is, if two multilevel codewords in $\Lambda_1(0)$ differ only in the jth codeword, then the MSSD is $d_1\delta_j^2$. Assuming that $d_1 > d_0 = 1$, and $\delta_1^2 > \delta_0^2$, we see from (18.83) that $\Delta_1^2 = \delta_0^2$, unless $\mathbb{C}_1$ is inserted at level $j = 0$, in which case $\Delta_1^2 = \min(d_1\delta_0^2, \delta_1^2) > \delta_0^2$. In other words, $\mathbb{C}_1$ must partition the initial level of the multilevel code $\Lambda_0(0)$, corresponding to the initial level of partitioning of the constituent 2-D signal sets, to maximize Δ_1^2. The two subsets at level 1 of the partition tree are now given by the subcode $\Lambda_1(0) = \mathbb{C}_1 * \mathbb{C}_0 * \cdots * \mathbb{C}_0$ and its coset

$$\Lambda_1(1) = \Lambda_1(0) + \mathbb{g}_0, \tag{18.84}$$

where the coset generator $\mathbb{g}_0$ is a multilevel codeword not belonging to the subcode $\Lambda_1(0)$; that is, $\mathbb{g}_0 \in \Lambda_0(0)$, $\mathbb{g}_0 \notin \Lambda_1(0)$. Here we note that (18.84) corresponds to (18.68) in Example 18.15, with the subcode $\Lambda_1(0)$ corresponding to the subset $Q^2(0 \times 0 \cup 1 \times 1)$ and the coset $\Lambda_1(1)$ corresponding to the coset $Q^2(0 \times 1 \cup 1 \times 0)$. (The particular choice of $\mathbb{g}_0$ does not affect the formation of the coset $\Lambda_1(1)$.) Further, since the lowest-order bit $v^{(0)}$ in the binary code vector $\mathbf{v}$ selects the first branch in the partition tree, we can represent the two subsets at level 1 as

$$\Lambda_1(v^{(0)}) = \Lambda_1(0) + v^{(0)}\mathbb{g}_0. \tag{18.85}$$

Next, to partition the subcode $\Lambda_1(0)$ and its coset $\Lambda_1(1)$ each of size 2^{IL-1}, into four subsets of size 2^{IL-2}, namely, the subcode $\Lambda_2(0)$ and its cosets $\Lambda_2(1)$, $\Lambda_2(2)$, and $\Lambda_2(3)$, we must replace one of the codes $\mathbb{C}_i$ of size 2^{L-i} in the multilevel subcode $\Lambda_1(0) = \mathbb{C}_1 * \mathbb{C}_0 * \cdots * \mathbb{C}_0$ with the code $\mathbb{C}_{i+1}$ of size 2^{L-i-1}. Again, we should do

this in such a way that the MSSD Δ_2^2 at level 2 in the multi-D partitioning tree is maximized. In this case, two choices for the subcode $\Lambda_2(0)$ are possible:

$$\text{(a)}\ \ \Lambda_2(0) = \mathbb{C}_2 * \mathbb{C}_0 * \cdots * \mathbb{C}_0, \tag{18.86a}$$

and

$$\text{(b)}\ \ \Lambda_2(0) = \mathbb{C}_1 * \mathbb{C}_0 * \cdots * \mathbb{C}_0 * \mathbb{C}_1 * \mathbb{C}_0 * \cdots * \mathbb{C}_0. \tag{18.86b}$$

To examine the effect of these two choices for the subcode $\Lambda_2(0)$ on Δ_2^2, we state a general lower bound on the MSSDs Δ_p^2 at partition level p.

> THEOREM 18.1 [32] Assume that the multilevel subcode $\Lambda_0(0)$ at partition level p is given by $\Lambda_0(0) = \mathbb{C}_{p_0} * \mathbb{C}_{p_1} * \cdots * \mathbb{C}_{p_{l-1}}$ and that the associated minimum Hamming distances of each component code are given by $d_{p_0}, d_{p_1}, \cdots, d_{p_{l-1}}$, respectively. Then, the MSSD Δ_p^2 at partition level p is given by
>
> $$\Delta_p^2 \geq \min\left(d_{p_0}\delta_0^2, d_{p_1}\delta_1^2, \cdots, d_{p_{l-1}}\delta_{l-1}^2\right). \tag{18.87}$$

The following comments apply to Theorem 18.1:

- The proof of (18.87) is based on a generalization of the argument used to obtain (18.82) and (18.83).
- In general, (18.87) gives a lower bound on Δ_p^2, although the bound holds with equality for most signal sets of interest.
- It can be shown (see Problem 18.29) that the partition level p equals the sum of the redundancies of the l linear block codes that define the subcode $\Lambda_p(0)$; that is,

$$p = \sum_{0 \leq j \leq l-1} p_j. \tag{18.88}$$

- From (18.87) we see that to maximize the MSSDs Δ_p^2 at each partition level p, each subcode $\mathbb{C}_i$ should be chosen to maximize its minimum Hamming distance d_i.

We can now use (18.87) to determine Δ_2^2 for the two choices for the subcode $\Lambda_2(0)$ given in (18.86). For choice (a) we see that $\Delta_2^2 = \min(d_2\delta_0^2, \delta_1^2)$. For choice (b), assuming $\mathbb{C}_1$ is inserted at level j, we obtain $\Delta_2^2 = \min(d_1\delta_0^2, \delta_1^2, d_1\delta_j^2) = \min(d_1\delta_0^2, \delta_1^2) = \Delta_1^2$, unless $\mathbb{C}_1$ is inserted at level $j = 1$, in which case $\Delta_2^2 = \min(d_1\delta_0^2, d_1\delta_1^2, \delta_2^2) = \min(d_1\delta_0^2, \delta_2^2) \geq \Delta_1^2$. Thus, for choice (b) it is always best to insert $\mathbb{C}_1$ at level $j = 1$, that is, to choose $\Lambda_2(0) = \mathbb{C}_1 * \mathbb{C}_1 * \mathbb{C}_0 * \cdots * \mathbb{C}_0$, to maximize Δ_2^2. Whether choice (a) or choice (b) gives the largest value of Δ_2^2 depends on the particular values of $d_1, d_2, \delta_0^2, \delta_1^2$, and δ_2^2. Choice (a) implies that the first two partitions in the multi-D signal set involve partitions at level 1 of the constituent 1-D or 2-D signal set, whereas choice (b) implies that the second partition in the multi-D signal set involves a partition at level 2 of the constituent signal set.

Once the subcode $\Lambda_2(0)$ has been chosen, the four subsets at level 2 of the partition chain are given by the subcode $\Lambda_2(0)$ and its cosets

$$\Lambda_2(2) = \Lambda_2(0) + \mathbb{g}_1, \tag{18.89a}$$

$$\Lambda_2(1) = \Lambda_2(0) + \mathbb{g}_0, \tag{18.89b}$$

$$\Lambda_2(3) = \Lambda_2(0) + \mathbb{g}_1 + \mathbb{g}_0, \tag{18.89c}$$

where the coset generator $\mathbb{g}_1$ is a multilevel codeword belonging to the subcode $\Lambda_1(0)$ but not belonging to the subcode $\Lambda_2(0)$; that is, $\mathbb{g}_0 \in \Lambda_1(0), \mathbb{g}_0 \notin \Lambda_2(0)$. Here, as before, we note the analogy between the general level-2 coset generation of (18.89) and the specific, for 2×8-PSK, level-2 coset generation of (18.69) in Example 18.15. (Again, the particular choice of $\mathbb{g}_1$ does not affect the formation of the cosets of $\Lambda_2(0)$.) Further, since the two lowest-order bits $v^{(1)}$ and $v^{(0)}$ in the binary code vector $\mathbb{v}$ select the first two branches in the partition tree, we can represent the four subsets at level 2 as

$$\Lambda_2(2v^{(1)} + v^{(0)}) = \Lambda_2(0) + v^{(1)}\mathbb{g}_1 + v^{(0)}\mathbb{g}_0. \tag{18.90}$$

The higher levels are partitioned in a similar manner. First, (18.87) is used to determine the multi-D subcode $\Lambda_p(0)$ that maximizes Λ_p^2 at partition level p, depending on the minimum Hamming distances d_i, $0 \le i \le L$, of the subcodes $\mathbb{C}_i$ and the MSSDs δ_j^2, $0 \le j \le I-1$, of the constituent 2-D signal set S. Then, a new coset generator $\mathbb{g}_{p-1}$ is chosen to generate the 2^p subsets at level p as follows:

$$\Lambda_p(2^{(p-1)}v^{(p-1)} + \cdots + 2v^{(1)} + v^{(0)}) = \Lambda_p(0) + v^{(p-1)}\mathbb{g}_{p-1} + \cdots + v^{(1)}\mathbb{g}_1 + v^{(0)}\mathbb{g}_0, \tag{18.91}$$

where $(v^{(p-1)}, \cdots, v^{(1)}, v^{(0)})$, representing the p lowest-order bits in the binary vector $\mathbb{v}$, select the first p branches in the partition tree. The process ends at partition level IL, where the multi-D subcode $\Lambda_{IL}(0) = \mathbb{C}_L * \mathbb{C}_L * \cdots * \mathbb{C}_L$ contains the single multi-D codeword consisting of I copies of the all-zero codeword. The $2^{IL} = 2^{k+1}$ subsets at level IL contain only one multi-D signal each, which can be expressed as

$$\begin{aligned}&\Lambda_{IL}\left(2^{IL-1}v^{(IL-1)} + \cdots + 2v^{(1)} + v^{(0)}\right)\\ &\quad = \Lambda_{IL}(0) + v^{(IL-1)}\mathbb{g}_{IL-1} + \cdots + v^{(1)}\mathbb{g}_1 + v^{(0)}\mathbb{g}_0,\end{aligned} \tag{18.92}$$

where the encoder output vector $\mathbb{v} = (v^{(IL-1)}, \cdots, v^{(1)}, v^{(0)}) = (v^{(k)}, \cdots, v^{(1)}, v^{(0)})$ selects the path through the partition tree. Finally, since the selected subset at level IL contains the multi-D signal $\mathbb{Y}$ to be transmitted, and noting that $\Lambda_{IL}(0)$ is the (multilevel) all-zero codeword, we can write

$$\mathbb{Y} = (\mathbb{y}_1, \mathbb{y}_2, \cdots, \mathbb{y}_L) = \sum_{0 \le p \le IL-1} v^{(p)}\mathbb{g}_p. \tag{18.93}$$

(If the coset generators $\mathbb{g}_p = (\mathbb{g}_{p1}, \mathbb{g}_{p2}, \cdots, \mathbb{g}_{pL})$, $0 \le p \le IL-1$, where $\mathbb{g}_{pi} = (g_{pi}^{(I-1)}, \cdots, g_{pi}^{(1)}, g_{pi}^{(0)})$, $1 \le i \le L$, are expressed in integer form, that is, $\mathbb{g}_p =$

$(g_{p1}, g_{p2}, \cdots, g_{pL})$, $g_{pi} = \sum_{0 \le j \le I-1} g_{pi}^{(j)} 2^j$, then (18.93) using modulo-2^I addition gives $\mathbb{Y} = (y_1, y_2, \cdots, y_L)$ in integer form.) Again, we see that (18.93), and its integer form noted previously, represents the general form for generating multi-D signals $\mathbb{Y}$, corresponding to (18.71) and (18.73) in Example 18.15.

The following example illustrates the general procedure.

EXAMPLE 18.16 2 × 8-PSK Set Partitioning

For the 2 × 8-PSK signal set, the MSSDs of the constituent 8-PSK signal set are $\delta_0^2 = 0.586 \le \delta_1^2 = 2.0 \le \delta_2^2 = 4.0$. Now, we choose the sequence of linear codes $\mathbb{C}_2 \subset \mathbb{C}_1 \subset \mathbb{C}_0$ as follows:

- $\mathbb{C}_0$ is the $(2, 2, 1)$ code that consists of the codewords (00), (01), (10), and (11).
- $\mathbb{C}_1$ is the $(2, 1, 2)$ code that consists of the codewords (00) and (11).
- $\mathbb{C}_2$ is the $(2, 0, \infty)$ code that consists only of the codeword (00).

The choices of multilevel subcodes that result in the best Δ_p^2 at each partition level p, along with the coset generators (in integer form) and the values of Δ_p^2 calculated using (18.87), are listed in Table 18.10. Note that in this case Δ_p^2 is maximized at each partition level p if the first two levels of the multi-D partition involve partitions at level 1 of the constituent 8-PSK signal set, the next two levels of the multi-D partition involve partitions at level 2 of the 8-PSK signal set, and the last two levels of the multi-D partition involve partitions at level 3 of the 8-PSK signal set.

The mapping from the coset generators to the multi-D binary signal vector

$$\begin{aligned} \mathbb{Y} &= (\mathbb{y}_1, \mathbb{y}_2) \\ &= \begin{bmatrix} y_1^{(2)} & y_2^{(2)} \\ y_1^{(1)} & y_2^{(1)} \\ y_1^{(0)} & y_2^{(0)} \end{bmatrix} \end{aligned} \tag{18.94}$$

TABLE 18.10: The 2 × 8-PSK signal set partition.

Partition level p	Subcode $\mathbb{A}_p(\mathbb{0})$	MSSD Δ_p^2	Coset generator $\mathbb{g}_p$
0	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_0$	$\min(4, 2, 0.586) = 0.586$	$(0, 1)$
	↓		
1	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_1$	$\min(4, 2, 1.172) = 1.172$	$(1, 1)$
	↓		
2	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_2$	$\min(4, 2, \infty) = 2.0$	$(0, 2)$
	↓		
3	$\mathbb{C}_0 * \mathbb{C}_1 * \mathbb{C}_2$	$\min(4, 4, \infty) = 4.0$	$(2, 2)$
	↓		
4	$\mathbb{C}_0 * \mathbb{C}_2 * \mathbb{C}_2$	$\min(4, \infty, \infty) = 4.0$	$(0, 4)$
	↓		
5	$\mathbb{C}_1 * \mathbb{C}_2 * \mathbb{C}_2$	$\min(8, \infty, \infty) = 8.0$	$(4, 4)$
	↓		
6	$\mathbb{C}_2 * \mathbb{C}_2 * \mathbb{C}_2$	$\min(\infty, \infty, \infty) = \infty$	—

Adapted from [33].

or $\mathbb{Y} = (y_1, y_2)$ in integer form, corresponding to a particular binary encoder output vector $\mathbf{v} = (v^{(5)}v^{(4)}v^{(3)}v^{(2)}v^{(1)}v^{(0)})$, was illustrated in the previous example.

For larger multi-D signal constellations, there may be several sequences of L subcodes of $\mathbb{C}_0$ that can be used to partition the multilevel code Λ_0, none of which maximizes Δ_p^2 at every partition level. Which partition is best in a particular case will depend on the rate $R = \tilde{k}/(\tilde{k}+1)$ of the convolutional encoder that determines the partition level and thus δ_{min}^2. An example for the 3×8-PSK multi-D signal set illustrates this point.

EXAMPLE 18.17 3×8-PSK Set Partitioning

As for the 2×8-PSK signal set, for 3×8-PSK we again have $\delta_0^2 = 0.586 \leq \delta_1^2 = 2.0 \leq \delta_2^2 = 4.0$. In this case we consider two options for the sequence of linear codes, $\mathbb{C}_3 \subset \mathbb{C}_2^a \subset \mathbb{C}_1^a \subset \mathbb{C}_0$ or $\mathbb{C}_3 \subset \mathbb{C}_2^b \subset \mathbb{C}_1^b \subset \mathbb{C}_0$, defined as follows:

- $\mathbb{C}_0$ is the $(3, 3, 1)$ code consisting of all eight binary 3-tuples.
- $\mathbb{C}_1^a$ is the $(3, 2, 2)$ code consisting of the codewords (000), (011), (101), and (110).
- $\mathbb{C}_1^b$ is the $(3, 2, 1)$ code consisting of the codewords (000), (011), (111), and (100).
- $\mathbb{C}_2^a$ is the $(3, 1, 2)$ code consisting of the codewords (000) and (011).
- $\mathbb{C}_2^b$ is the $(3, 1, 3)$ code consisting of the codewords (000) and (111).
- $\mathbb{C}_3$ is the $(3, 0, \infty)$ code consisting only of the codeword (000).

We see that code sequence $\mathbb{C}_3 \subset \mathbb{C}_2^a \subset \mathbb{C}_1^a \subset \mathbb{C}_0$ achieves a better minimum distance at the first partition level but that the sequence $\mathbb{C}_3 \subset \mathbb{C}_2^b \subset \mathbb{C}_1^b \subset \mathbb{C}_0$ achieves a better distance at the second level. Either of these sequences alone, or combinations of the two sequences, can be used to partition the 3×8-PSK signal set. Three partitions of 3×8-PSK, corresponding to three distinct combinations of the foregoing two code sequences, along with the corresponding coset generators (in integer form) and values of Δ_p^2 calculated using (18.87), are listed in Table 18.11. Partition I uses only the code sequence $\mathbb{C}_3 \subset \mathbb{C}_2^a \subset \mathbb{C}_1^a \subset \mathbb{C}_0$ for partitioning, whereas partitions II and III use combinations of the two code sequences. Note that partition I is best at partition level $p = 1$, II and III are best at $p = 2$, I and II are best at $p = 4$, and III is best at $p = 6$. Thus, for example, if the code rate is $R = 3/4$, either partition I or partition II should be selected, whereas partition III is the best choice for a code rate of $R = 5/6$. (It should be noted that although different code sequences can be used to partition different levels of a constituent signal set, it is not allowable to "jump back and forth" between two code sequences in partitioning a given level of the constituent signal set; that is, only a proper subcode sequence can be used to partition a given level.) Finally, using the coset generators from Table 18.11 in (18.93), we can determine the signal set mappings corresponding to the three partitions of 3×8-PSK (see Problem 18.30).

TABLE 18.11: Three 3 × 8-PSK signal set partitions.

Partition I			
Partition level p	Subcode $\Lambda_p(\mathbb{0})$	MSSD Δ_p^2	Coset generator $\mathbf{g}_p$
0	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_0$	$\min(4, 2, 0.586) = 0.586$	(1,1,1)
1	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_1^1$	$\min(4, 2, 1.172) = 1.172$	(1,1,0)
2	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_2^1$	$\min(4, 2, 1.172) = 1.172$	(0,1,1)
3	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_3$	$\min(4, 2, \infty) = 2.0$	(2,2,2)
4	$\mathbb{C}_0 * \mathbb{C}_1^1 * \mathbb{C}_3$	$\min(4, 4, \infty) = 4.0$	(2,2,0)
5	$\mathbb{C}_0 * \mathbb{C}_2^1 * \mathbb{C}_3$	$\min(4, 4, \infty) = 4.0$	(0,2,2)
6	$\mathbb{C}_0 * \mathbb{C}_3 * \mathbb{C}_3$	$\min(4, \infty, \infty) = 4.0$	(4,4,4)
7	$\mathbb{C}_1^1 * \mathbb{C}_3 * \mathbb{C}_3$	$\min(8, \infty, \infty) = 8.0$	(4,4,0)
8	$\mathbb{C}_2^1 * \mathbb{C}_3 * \mathbb{C}_3$	$\min(8, \infty, \infty) = 8.0$	(0,4,4)
9	$\mathbb{C}_3 * \mathbb{C}_3 * \mathbb{C}_3$	$\min(\infty, \infty, \infty) = \infty$	—
Partition II			
Partition level p	Subcode $\Lambda_p(\mathbb{0})$	MSSD Δ_p^2	Coset generator $\mathbf{g}_p$
0	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_0$	$\min(4, 2, 0.586) = 0.586$	(0,0,1)
1	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_1^2$	$\min(4, 2, 0.586) = 0.586$	(0,1,1)
2	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_2^2$	$\min(4, 2, 1.757) = 1.757$	(1,1,1)
3	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_3$	$\min(4, 2, \infty) = 2.0$	(2,2,2)
4	$\mathbb{C}_0 * \mathbb{C}_1^1 * \mathbb{C}_3$	$\min(4, 4, \infty) = 4.0$	(2,2,0)
5	$\mathbb{C}_0 * \mathbb{C}_2^1 * \mathbb{C}_3$	$\min(4, 4, \infty) = 4.0$	(0,2,2)
6	$\mathbb{C}_0 * \mathbb{C}_3 * \mathbb{C}_3$	$\min(4, \infty, \infty) = 4.0$	(4,4,4)
7	$\mathbb{C}_1^1 * \mathbb{C}_3 * \mathbb{C}_3$	$\min(8, \infty, \infty) = 8.0$	(4,4,0)
8	$\mathbb{C}_2^1 * \mathbb{C}_3 * \mathbb{C}_3$	$\min(8, \infty, \infty) = 8.0$	(0,4,4)
9	$\mathbb{C}_3 * \mathbb{C}_3 * \mathbb{C}_3$	$\min(\infty, \infty, \infty) = \infty$	—

(*continued overleaf*)

TABLE 18.11: (*continued*)

Partition III			
Partition level p	**Subcode** $\Lambda_p(\mathbb{0})$	**MSSD** Δ_p^2	**Coset generator** $\mathbf{g}_p$
0	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_0$	$\min(4, 2, 0.586) = 0.586$	(0,0,1)
	$\downarrow$		
1	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_1^2$	$\min(4, 2, 0.586) = 0.586$	(0,1,1)
	$\downarrow$		
2	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_2^2$	$\min(4, 2, 1.757) = 1.757$	(1,1,1)
	$\downarrow$		
3	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_3$	$\min(4, 2, \infty) = 2.0$	(0,0,2)
	$\downarrow$		
4	$\mathbb{C}_0 * \mathbb{C}_1^2 * \mathbb{C}_3$	$\min(4, 2, \infty) = 2.0$	(0,2,2)
	$\downarrow$		
5	$\mathbb{C}_0 * \mathbb{C}_2^2 * \mathbb{C}_3$	$\min(4, 6, \infty) = 4.0$	(4,4,4)
	$\downarrow$		
6	$\mathbb{C}_1^1 * \mathbb{C}^2 * \mathbb{C}_3$	$\min(8, 6, \infty) = 6.0$	(2,2,2)
	$\downarrow 2$		
7	$\mathbb{C}_1^1 * \mathbb{C}_3 * \mathbb{C}_3$	$\min(8, \infty, \infty) = 8.0$	(4,4,0)
	$\downarrow$		
8	$\mathbb{C}_2^1 * \mathbb{C}_3 * \mathbb{C}_3$	$\min(8, \infty, \infty) = 8.0$	(0,4,4)
	$\downarrow$		
9	$\mathbb{C}_3 * \mathbb{C}_3 * \mathbb{C}_3$	$\min(\infty, \infty, \infty) = \infty$	—

Adapted from [33].

We now consider the partitioning of the 4×8-PSK signal set.

EXAMPLE 18.18 4×8-PSK Set Partitioning

For the 4×8-PSK signal set we still have $\delta_0^2 = 0.586 \leq \delta_1^2 = 2.0 \leq \delta_2^2 = 4.0$. In this case only one sequence of linear codes results in a good partition. We choose $\mathbb{C}_4 \subset \mathbb{C}_3 \subset \mathbb{C}_2 \subset \mathbb{C}_1 \subset \mathbb{C}_0$ as follows:

- $\mathbb{C}_0$ is the $(4, 4, 1)$ code consisting of all 16 binary 4-tuples.
- $\mathbb{C}_1$ is the $(4, 3, 2)$ code consisting of the 8 binary 4-tuples of even weight.
- $\mathbb{C}_2$ is the $(4, 2, 2)$ code consisting of the codewords (0000), (1010), (0101), and (1111).
- $\mathbb{C}_3$ is the $(4, 1, 4)$ code consisting of the codewords (0000) and (1111).
- $\mathbb{C}_4$ is the $(4, 0, \infty)$ code consisting only of the codeword (0000).

The choices of multilevel subcodes that result in the best Δ_p^2 at each partition level p, along with the coset generators (in integer form) and the values of Δ_p^2 calculated using (18.87), are listed in Table 18.12. Using the coset generators from Table 18.12 in (18.93), we can determine the signal set mapping corresponding to the partition of 4×8-PSK.

TABLE 18.12: The 4×8-PSK signal set partition.

Partition level p	Subcode $\mathbb{A}_p(\mathbb{0})$	MSSD Δ_p^2	Coset generator $\mathbb{g}_p$
0	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_0$	$\min(4, 2, 0.586) = 0.586$	(0,0,0,1)
1	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_1$	$\min(4, 2, 1.172) = 1.172$	(0,0,1,1)
2	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_2$	$\min(4, 2, 1.172) = 1.172$	(0,1,0,1)
3	$\mathbb{C}_0 * \mathbb{C}_0 * \mathbb{C}_3$	$\min(4, 2, 2.343) = 2.0$	(0,0,0,2)
4	$\mathbb{C}_0 * \mathbb{C}_1 * \mathbb{C}_3$	$\min(4, 4, 2.343) = 2.343$	(1,1,1,1)
5	$\mathbb{C}_0 * \mathbb{C}_1 * \mathbb{C}_4$	$\min(4, 4, \infty) = 4.0$	(0,0,2,2)
6	$\mathbb{C}_0 * \mathbb{C}_2 * \mathbb{C}_4$	$\min(4, 4, \infty) = 4.0$	(0,2,0,2)
7	$\mathbb{C}_0 * \mathbb{C}_3 * \mathbb{C}_4$	$\min(4, 8, \infty) = 4.0$	(0,0,0,4)
8	$\mathbb{C}_1 * \mathbb{C}_3 * \mathbb{C}_4$	$\min(8, 8, \infty) = 8.0$	(2,2,2,2)
9	$\mathbb{C}_1 * \mathbb{C}_4 * \mathbb{C}_4$	$\min(8, \infty, \infty) = 8.0$	(0,0,4,4)
10	$\mathbb{C}_2 * \mathbb{C}_4 * \mathbb{C}_4$	$\min(8, \infty, \infty) = 8.0$	(0,4,0,4)
11	$\mathbb{C}_3 * \mathbb{C}_4 * \mathbb{C}_4$	$\min(16, \infty, \infty) = 16.0$	(4,4,4,4)
12	$\mathbb{C}_4 * \mathbb{C}_4 * \mathbb{C}_4$	$\min(\infty, \infty, \infty) = \infty$	—

Adapted from [33].

The following remarks relate to Examples 18.16, 18.17, and 18.18:

- Example 18.17 illustrates that, in some cases, there is not a single choice for the sequence of binary codes that partitions a multilevel code that maximizes Δ_p^2 at every partition level p.
- A summary of $L \times 8$-PSK partitions developed in these three examples appears in Table 18.13 for $L = 2, 3$, and 4.
- Exactly the same sets of code sequences used to partition multi-D 8-PSK signal sets can be used to partition multi-D 16-PSK and QAM signal sets. For example, a summary of the $L \times 16$-PSK and $L \times 16$-QAM partitions appears in Tables 18.14(a) and (b), respectively, for $L = 2$, 3, and 4, where $\delta_0^2 = 0.152 \le \delta_1^2 = 0.586 \le \delta_2^2 = 2.0 \le \delta_3^2 = 4.0$ for the constituent 16-PSK signal set, and $\delta_0^2 = 1.0 \le \delta_1^2 = 2.0 \le \delta_2^2 = 4.0 \le \delta_3^2 = 8.0$ for the constituent 16-QAM signal set. In both cases the mapping from coset generators to multi-D signals in integer form using (18.93) requires addition modulo-16.

Given a partitioning of the multi-D signal space S^L such that the MSSD Δ_p^2 is maximized at each partition level p, a number of coded information bits $\tilde{k}$, $1 \le \tilde{k} \le k$, must be chosen to give the desired MSSD $\Delta_{\tilde{k}+1}^2$. Then, a rate $R = \tilde{k}/(\tilde{k}+1)$

TABLE 18.13: Summary of $L \times 8$-PSK partitions, $L = 2, 3, 4$.

Partition level	$L = 2$		$L = 3$(I)		$L = 3$(II)		$L = 3$(III)		$L = 4$	
	MSSD	Generator	MSSD	Generator	MSSD	Generator	MSSD	Generator	MSSD	Generator
p	Δ_p^2	g_p	Δ_p^2	g_p	Δ_p^2	g_p	Δ_p^2	g_p	Δ_p^2	g_p
0	0.586	(0,1)	0.586	(1,1,1)	0.586	(0,0,1)	0.586	(0,0,1)	0.586	(0,0,0,1)
1	1.172	(1,1)	1.172	(1,1,0)	0.586	(0,1,1)	0.586	(0,1,1)	1.172	(0,0,1,1)
2	2.0	(0,2)	1.172	(0,1,1)	1.757	(1,1,1)	1.757	(1,1,1)	1.172	(0,1,0,1)
3	4.0	(2,2)	2.0	(2,2,2)	2.0	(2,2,2)	2.0	(0,0,2)	2.0	(0,0,0,2)
4	4.0	(0,4)	4.0	(2,2,0)	4.0	(2,2,0)	2.0	(0,2,2)	2.343	(1,1,1,1)
5	8.0	(4,4)	4.0	(0,2,2)	4.0	(0,2,2)	4.0	(4,4,4)	4.0	(0,0,2,2)
6	—	—	4.0	(4,4,4)	4.0	(4,4,4)	6.0	(2,2,2)	4.0	(0,2,0,2)
7	—	—	8.0	(4,4,0)	8.0	(4,4,0)	8.0	(4,4,0)	4.0	(0,0,0,4)
8	—	—	8.0	(0,4,4)	8.0	(0,4,4)	8.0	(0,4,4)	8.0	(2,2,2,2)
9									8.0	(0,0,4,4)
10	—	—	—	—	—	—	—	—	8.0	(0,4,0,4)
11	—	—	—	—	—	—	—	—	16.0	(4,4,4,4)
π_0 π_1 π_2	1 3 5		0 3 6		2 3 6		2 6 5		4 8 11	

Adapted from [33].

TABLE 18.14: Summary of $L \times 16$-PSK and $L \times 16$-QAM partitions, $L = 2, 3, 4$.

(a) $L \times 16$-PSK

Partition level	$L = 2$		$L = 3$(I)		$L = 3$(II)		$L = 3$(III)		$L = 4$	
	MSSD	Generator	MSSD	Generator	MSSD	Generator	MSSD	Generator	MSSD	Generator
p	Δ_p^2	$\mathbb{g}_p$	Δ_p^2	$\mathbb{g}_p$	Δ_p^2	$\mathbb{g}_p$	Δ_p^2	$\mathbb{g}_p$	Δ_p^2	$\mathbb{g}_p$
0	0.152	(0,1)	0.152	(1,1,1)	0.152	(0,0,1)	0.152	(0,0,1)	0.152	(0,0,0,1)
1	0.304	(1,1)	0.304	(1,1,0)	0.152	(0,1,1)	0.152	(0,1,1)	0.304	(0,0,1,1)
2	0.586	(0,2)	0.304	(0,1,1)	0.457	(1,1,1)	0.457	(1,1,1)	0.304	(0,1,0,1)
3	1.172	(2,2)	0.586	(2,2,2)	0.586	(2,2,2)	0.586	(0,0,2)	0.586	(0,0,0,2)
4	2.0	(0,4)	1.172	(2,2,0)	1.172	(2,2,0)	0.586	(0,2,2)	0.609	(1,1,1,1)
5	4.0	(4,4)	1.172	(0,2,2)	1.172	(0,2,2)	1.757	(2,2,2)	1.172	(0,0,2,2)
6	4.0	(0,8)	2.0	(4,4,4)	2.0	(4,4,4)	2.0	(4,4,4)	1.172	(0,2,0,2)
7	8.0	(8,8)	4.0	(4,4,0)	4.0	(4,4,0)	4.0	(4,4,0)	2.0	(0,0,0,4)
8	—	—	4.0	(0,4,4)	4.0	(0,4,4)	4.0	(0,4,4)	2.343	(2,2,2,2)
9	—	—	4.0	(8,8,8)	4.0	(8,8,8)	4.0	(8,8,8)	4.0	(0,0,4,4)
10	—	—	8.0	(8,8,0)	8.0	(8,8,0)	8.0	(8,8,0)	4.0	(0,4,0,4)
11	—	—	8.0	(0,8,8)	8.0	(0,8,8)	8.0	(0,8,8)	4.0	(0,0,0,8)
12	—	—	—	—	—	—	—	—	8.0	(4,4,4,4)
13	—	—	—	—	—	—	—	—	8.0	(0,0,8,8)
14	—	—	—	—	—	—	—	—	8.0	(0,8,0,8)
15	—	—	—	—	—	—	—	—	16.0	(8,8,8,8)
π_0 π_1 π_2 π_3	1 3 5 7		0 3 6 9		2 3 6 9		2 5 6 9		4 8 12 15	

(continued overleaf)

TABLE 18.14: (*continued*)

(b) $L \times 16$-QAM

Partition level	$L = 2$		$L = 3$(I)		$L = 3$(II)		$L = 3$(III)		$L = 4$	
	MSSD	Generator	MSSD	Generator	MSSD	Generator	MSSD	Generator	MSSD	Generator
p	Δ_p^2	$\mathbf{g}_p$	Δ_p^2	$\mathbf{g}_p$	Δ_p^2	$\mathbf{g}_p$	Δ_p^2	$\mathbf{g}_p$	Δ_p^2	$\mathbf{g}_p$
0	1	(0,1)	1	(1,1,1)	1	(0,0,1)	1	(0,0,1)	1	(0,0,0,1)
1	2	(1,1)	2	(1,1,0)	1	(0,1,1)	1	(0,1,1)	2	(0,0,1,1)
2	2	(0,2)	2	(0,1,1)	2	(2,2,2)	2	(0,0,2)	2	(0,1,0,1)
3	4	(0,4)	2	(2,2,2)	3	(1,1,1)	2	(0,2,2)	2	(0,0,0,2)
4	4	(2,2)	4	(2,2,0)	4	(2,2,0)	3	(1,1,1)	4	(1,1,1,1)
5	8	(4,4)	4	(0,2,2)	4	(0,2,2)	4	(4,4,4)	4	(0,0,2,2)
6	8	(0,8)	4	(4,4,4)	4	(4,4,4)	6	(2,2,2)	4	(0,2,0,2)
7	16	(8,8)	8	(4,4,0)	8	(4,4,0)	8	(4,4,0)	4	(0,0,0,4)
8	—	—	8	(0,4,4)	8	(0,4,4)	8	(0,4,4)	8	(2,2,2,2)
9	—	—	8	(8,8,8)	8	(8,8,8)	8	(8,8,8)	8	(0,0,4,4)
10	—	—	16	(8,8,0)	16	(8,8,0)	16	(8,8,0)	8	(0,4,0,4)
11	—	—	16	(0,8,8)	16	(0,8,8)	16	(0,8,8)	8	(0,0,0,8)
12	—	—	—	—	—	—	—	—	16	(4,4,4,4)
13	—	—	—	—	—	—	—	—	16	(0,0,8,8)
14	—	—	—	—	—	—	—	—	16	(0,8,0,8)
15	—	—	—	—	—	—	—	—	32	(8,8,8,8)
π_0 π_1	1	3	0	3	3	2	4	6	4	8

Adapted from [33, 34].

convolutional encoder must be selected to maximize δ_{free}^2, the minimum distance between trellis paths, and to minimize $A_{d_{free}}$, the number of nearest-neighbor codewords, for the desired number of states. Finally, the $k+1 = IL$-bit encoder output vector $\mathbf{v} = (v^{(k)}, v^{(k-1)}, \cdots, v^{(1)}, v^{(0)})$, which includes $k - \tilde{k}$ uncoded information bits, along with the coset generators $\mathbb{g}_p$, $0 \le p \le k$, determines the L-dimensional signal point to be transmitted according to (18.93), resulting in a special efficiency of $\eta = k/L = (IL-1)/L$ bits/symbol.

It is possible to achieve other spectral efficiencies by starting the partitioning with a subcode of the multilevel code (multi-D signal set) $\Lambda_0(0)$ instead of with $\Lambda_0(0)$ itself. In general, we can start the partitioning with any of the multilevel subcodes (multi-D subsets) $\Lambda_q(0)$, $q = 0, 1, \cdots, L-1$, by letting $k+1 = IL-q$, resulting in a spectral efficiency of $\eta = k/L = (IL-q-1)/L$ bits/symbol. For example, if the encoder output bit $v^{(0)}$ is used to select a subset of $\Lambda_1(0)$ instead of a subset of $\Lambda_0(0)$, a spectral efficiency of $\eta = (IL-2)/L$ results, and the range of spectral efficiencies achievable with an L-dimensional signal set is $\eta = (IL-1)/L, (IL-2)/L, \cdots, (I-1)L/L = I-1$ bits/symbol. The values of δ_{free}^2 and $A_{d_{free}}$ that can be achieved depend on the level at which partitioning begins, and thus, in general, we expect that different codes will be optimum for different values of q.

When $q > 0$, the mapping function from the binary code vector $\mathbf{v} = (v^{(IL-q-1)}, \cdots, v^{(1)}, v^{(0)})$ to the binary signal vector $\mathbb{Y} = (\mathbb{y}_1, \mathbb{y}_2, \cdots, \mathbb{y}_L)$ is given by

$$\mathbb{Y} = (\mathbb{y}_1, \mathbb{y}_2, \cdots, \mathbb{y}_L) = \sum_{q \le p \le IL-1} v^{(p-q)} \mathbb{g}_p, \tag{18.95a}$$

or when the coset generators $\mathbb{g}_p$ are expressed in integer form, $\mathbb{Y}$ is given by

$$\mathbb{Y} = (y_1, y_2, \cdots, y_L) = \sum_{q \le p \le IL-1} v^{(p-q)} \mathbb{g}_p \pmod{2^I}. \tag{18.95b}$$

For example, if a rate $R = 2/3$ convolutional encoder with 2 uncoded information bits is used in Example 18.15 to partition the subset $\Lambda_1(0)(\mathcal{Q}^2(0 \times 0 \cup 1 \times 1))$ of 2×8-PSK instead of the entire signal set $\Lambda_0(0)$, the mapping function (18.71) becomes

$$\begin{aligned} \mathbb{Y} = (y_1, y_2) &= \sum_{1 \le p \le 5} v^{(p-1)} \mathbb{g}_p \pmod 8 \\ &= v^{(4)}(4,4) + v^{(3)}(0,4) + v^{(2)}(2,2) + v^{(1)}(0,2) + v^{(0)}(1,1) \pmod 8, \end{aligned} \tag{18.96}$$

where $v^{(4)}$ and $v^{(3)}$ represent the 2 uncoded bits and $v^{(2)}$, $v^{(1)}$, and $v^{(0)}$ represent the 3 encoder output bits. In this case the spectral efficiency is $\eta = 4/2 = 2.0$ bits/symbol, and the MSSDs are $\Delta_0^2 = 1.172$, $\Delta_1^2 = 2.0$, $\Delta_2^2 = 4.0$, $\Delta_3^2 = 4.0$, $\Delta_4^2 = 8.0$, and $\Delta_5^2 = \infty$ (see Table 18.10).

The best codes for multi-D 8-PSK, 16-PSK, and QAM signal sets with $L = 2, 3$, and 4 are listed in Tables 18.15(a)–(c). For the QAM signal sets the codes are based on the 4-D, 6-D, and 8-D integer lattices $\mathbb{Z}^4$, $\mathbb{Z}^6$, and $\mathbb{Z}^8$, respectively. The codes were found by computer search [33, 34]. Each table gives the following information:

TABLE 18.15: Optimum codes for multi-D 8-PSK, 16-PSK, and QAM signal constellations.

(a) Codes for 8-PSK

TRELLIS-CODED 2 × 8-PSK

$\eta = 2.5$ bits/symbol, $q = 0$, $d^2_{min} = 1.172$, $A_{min} = 4$ (2×8-PSK)

ν	$\tilde{k}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	1	3	90°	1.757	8	2.0	4	1.76
2	1	—	—	2	5	90°	2.0	4	2.929	32	2.32
3	2	—	04	06	11	45°	2.929	16	—	—	3.98
4	2	—	16	12	23	45°	3.515	56	—	—	4.77
5	2	—	10	06	41	45°	3.515	16	—	—	4.77
6	2	—	004	030	113	45°	4.0	6	4.101	80	5.33
	2	—	044	016	107	90°	4.0	6	4.101	48	5.33
7	3	110	044	016	317	90°	4.0	2	4.101	25	5.33

TRELLIS-CODED 2 × 8-PSK

$\eta = 2.0$ bits/symbol, $q = 1$, $d^2_{min} = 2.0$, $A_{min} = 2$ (1×QPSK)

ν	$\tilde{k}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	1	3	45°	3.172	8.0	4.0	6	2.00
2	1	—	—	2	5	45°	4.0	6.0	5.172	32	3.01
3	2	—	04	02	11	180°	4.0	2.0	5.172	16	3.01
4	3	04	14	02	21	90°	5.172	8.0	—	—	4.13
5	3	24	14	06	43	90°	6.0	6.0	—	—	4.77
6	3	012	050	004	125	90°	6.343	5.5	—	—	5.01
7	3	110	044	016	317	90°	7.515	25.0	—	—	5.75

TRELLIS-CODED 3 × 8-PSK

$\eta = 2.67$ bits/symbol, $q = 0$, $d^2_{min} = 1.172$, $A_{min} = 12$ (3×8-PSK, I)

ν	$\tilde{k}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)	Signal set
1	1	—	—	1	3	45°	1.172	4	—	—	0.00	II
2	1	—	—	2	5	45°	1.757	16	—	—	1.76	II
3	2	—	04	02	11	45°	2.0	6	2.343	16	2.32	I
4	3	14	04	02	21	90°	2.343	12	—	—	3.01	I
	3	10	04	02	21	180°	2.343	8	—	—	3.01	I
5	3	30	14	02	53	90°	2.929	48	—	—	3.98	I
6	3	050	022	006	103	90°	3.172	12	—	—	4.33	I
7	3	056	112	004	225	90°	3.515	84	—	—	4.77	I
	3	100	050	022	255	180°	3.515	76	—	—	4.77	I

TABLE 18.15: (*continued*)

TRELLIS-CODED 3 × 8-PSK

$\eta = 2.33$ bits/symbol, $q = 1$, $d^2_{min} = 1.757$, $A_{min} = 8$ (3×8-PSK, II)

ν	$\tilde{k}$	$h^{(4)}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)	Signal set
1	1	—	—	—	1	3	90°	2.0	6	2.343	16	0.56	II
2	2	—	—	3	1	7	90°	2.586	6	—	—	1.68	II
3	2	—	—	06	02	11	90°	3.515	16	—	—	3.01	II
	2	—	—	04	02	11	180°	3.757	24	—	—	3.30	II
4	3	—	10	04	06	21	45°	3.157	12	—	—	3.30	III
	2	—	—	14	02	27	90°	4.0	15	4.343	24	3.57	II
5	3	—	22	16	06	41	45°	4.0	7	—	—	3.57	III
6	3	—	010	046	060	105	45°	4.0	3	4.686	8	3.57	III
	4	060	024	014	002	101	180°	4.0	2	—	—	3.57	III

TRELLIS-CODED 3 × 8-PSK

$\eta = 2.00$ bits/symbol, $q = 2$, $d^2_{min} = 2.0$, $A_{min} = 2$ (1 × QPSK)

ν	$\tilde{k}$	$h^{(4)}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)	Signal set
1	1	—	—	—	1	3	180°	3.757	24.0	—	—	2.74	II
2	1	—	—	—	2	5	180°	4.0	15.0	5.757	144	3.01	II
3	2	—	—	04	02	11	45°	4.0	7.0	—	—	3.01	III
4	2	—	—	12	04	27	45°	4.0	3.0	5.757	32	3.01	III
5	3	—	14	24	02	41	180°	5.757	17.5	—	—	4.59	III
	3	—	16	22	06	53	360°	5.757	17.0	—	—	4.59	III
6	3	—	030	042	014	103	180°	6.0	11.0	—	—	4.77	III
	4	014	044	024	006	103	180°	6.0	4.0	—	—	4.77	II

TRELLIS-CODED 4 × 8-PSK

$\eta = 2.75$ bits/symbol, $q = 0$, $d^2_{min} = 1.172$, $A_{min} = 24$ (4×8-PSK)

ν	$\tilde{k}$	$h^{(4)}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	—	1	3	45°	1.172	8	1.757	64	0.00
2	2	—	—	2	1	5	45°	1.757	48	—	—	1.76
3	2	—	—	04	02	11	45°	2.0	8	2.343	64	2.32
4	3	—	10	04	02	21	45°	2.343	40	—	—	3.01
5	3	—	30	14	02	41	45°	2.343	8	2.929	288	3.01
6	4	030	020	052	014	101	45°	2.929	136	—	—	3.98

TRELLIS-CODED 4 × 8-PSK

$\eta = 2.50$ bits/symbol, $q = 1$, $d^2_{min} = 1.172$, $A_{min} = 4$ (2×8-PSK)

ν	$\tilde{k}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	1	3	45°	2.0	8	2.343	64	2.32
2	2	—	2	1	5	45°	2.343	40	—	—	3.01
3	2	—	04	02	11	45°	2.343	8	3.172	32	3.01
4	3	14	04	02	21	45°	3.172	16	—	—	4.33
5	3	24	14	02	41	45°	3.515	64	—	—	4.77
6	3	014	024	042	103	45°	4.0	28	4.686	1088	5.33

(*continued overleaf*)

TABLE 18.15: *(continued)*

TRELLIS-CODED 4 × 8-PSK

$\eta = 2.25$ bits/symbol, $q = 2$, $d^2_{min} = 2.0$, $A_{min} = 8$ (4 × 8-PSK)

ν	$\tilde{k}$	$h^{(4)}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	—	1	3	45°	2.343	8	3.172	32	0.69
2	2	—	—	3	1	5	45°	3.172	16	—	—	2.00
3	2	—	—	06	02	11	45°	4.0	28	4.343	64	3.01
	2	—	—	02	06	11	90°	4.0	28	4.686	64	3.01
4	3	—	04	06	12	21	45°	4.0	12	4.686	32	3.01
5	4	10	04	06	22	41	45°	4.0	4	4.686	16	3.01

TRELLIS-CODED 4 × 8-PSK

$\eta = 2.00$ bits/symbol, $q = 3$, $d^2_{min} = 2.0$, $A_{min} = 2$ (1 × QPSK)

ν	$\tilde{k}$	$h^{(4)}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	—	1	3	90°	4.0	28	4.686	64	3.01
2	2	—	—	2	3	5	45°	4.0	12	4.686	32	3.01
3	3	—	02	04	03	11	45°	4.0	4	4.686	16	3.01
4	4	10	04	02	03	21	45°	4.686	8	—	—	3.70
5	4	02	10	04	22	41	45°	6.343	16	—	—	5.01
6	4	034	044	016	036	107	45°	6.686	6	—	—	5.24
	4	044	024	014	016	103	90°	7.029	24	—	—	5.46

(b) Codes for 16-PSK

TRELLIS-CODED 2 × 16-PSK

$\eta = 3.5$ bits/symbol, $q = 0$, $d^2_{min} = 0.304$, $A_{min} = 4$ (2 × 16-PSK)

ν	$\tilde{k}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	1	3	45°	0.457	8	—	—	1.76
2	1	—	2	5	45°	0.586	4	0.761	32	2.84
3	2	04	06	11	22.5°	0.761	16	—	—	3.98
4	2	16	12	23	22.5°	0.913	56	—	—	4.77
5	2	10	06	41	22.5°	0.913	16	—	—	4.77
6	2	004	030	113	22.5°	1.066	80	—	—	5.44
	2	044	016	107	45°	1.066	48	—	—	5.44
7	2	074	132	217	22.5°	1.172	4	1.218	228	5.85

TRELLIS-CODED 2 × 16-PSK

$\eta = 3.0$ bits/symbol, $q = 1$, $d^2_{min} = 0.586$, $A_{min} = 2$ (1 × 8-PSK)

ν	$\tilde{k}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	1	3	22.5°	0.89	8	—	—	1.82
2	1	—	—	2	5	22.5°	1.172	4	1.476	32	3.01
3	2	—	04	02	11	90°	1.476	16	—	—	4.01
4	2	—	14	06	23	45°	1.757	8	—	—	4.77
5	2	—	30	16	41	45°	1.781	16	—	—	4.83
6	2	—	044	016	107	45°	2.0	4	2.085	48	5.33
7	3	110	044	016	317	45°	2.085	25	—	—	5.51

TABLE 18.15: (*continued*)

TRELLIS-CODED 3 × 16-PSK

$\eta = 3.67$ bits/symbol, $q = 0$, $d^2_{min} = 0.304$, $A_{min} = 12$ (3 × 16-PSK, I)

ν	$\tilde{k}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)	Signal set
1	1	—	—	1	3	22.5°	0.304	4	—	—	0.00	II
2	1	—	—	2	5	22.5°	0.457	16	—	—	1.76	II
3	2	—	04	02	11	22.5°	0.586	6	0.609	16	2.84	I
4	3	14	04	02	21	45°	0.609	12	—	—	3.01	I
	3	10	04	02	21	90°	0.609	8	—	—	3.01	I
5	3	30	14	02	53	45°	0.761	48	—	—	3.98	I
6	3	050	022	006	103	45°	0.890	12	—	—	4.66	I
7	3	056	112	004	225	45°	0.913	84	—	—	4.77	I
	3	100	050	022	255	90°	0.913	76	—	—	4.77	I

TRELLIS-CODED 3 × 16-PSK

$\eta = 3.33$ bits/symbol, $q = 1$, $d^2_{min} = 0.457$, $A_{min} = 8$ (3 × 16-PSK, II)

ν	$\tilde{k}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)	Signal set
1	1	—	—	1	3	45°	0.586	6	0.609	16	1.08	II
2	2	—	3	1	7	45°	0.738	6	—	—	2.08	II
3	2	—	06	02	11	45°	0.913	16	—	—	3.01	II
	2	—	04	02	11	90°	1.043	24	—	—	3.58	II
4	3	10	04	06	21	22.5°	1.043	12	—	—	3.58	III
	2	—	14	02	27	45°	1.172	12	1.195	24	4.09	II
5	3	34	16	06	41	22.5°	1.172	4	—	—	4.09	III
6	3	032	046	006	103	22.5°	1.218	8	—	—	4.26	III
7	3	014	102	044	203	22.5°	1.370	32	—	—	4.77	III
	3	006	072	062	223	45°	1.476	8	—	—	5.09	III

TRELLIS-CODED 3 × 16-PSK

$\eta = 3.00$ bits/symbol, $q = 2$, $d^2_{min} = 0.586$, $A_{min} = 2$ (1 × 8-PSK)

ν	$\tilde{k}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)	Signal set
1	1	—	—	1	3	90°	1.043	24	—	—	2.50	II
2	1	—	—	2	5	90°	1.172	12	1.628	144	3.01	II
3	2	—	04	02	11	22.5°	1.172	4	—	—	3.01	III
4	2	—	12	04	27	22.5°	1.628	32	—	—	4.44	III
5	2	—	14	02	41	22.5°	1.628	16	—	—	4.44	III
	2	—	22	14	43	45°	1.757	16	—	—	4.77	III
6	2	—	054	020	115	22.5°	1.757	8	2.085	48	4.77	III
	3	020	004	012	101	45°	2.0	6	2.085	72	5.33	II
	3	050	030	026	101	90°	2.0	6	2.085	60	5.33	II
7	3	060	106	050	213	45°	2.0	6	2.214	56	5.33	III
	3	016	110	052	203	90°	2.0	6	2.343	64	5.33	III

(*continued overleaf*)

TABLE 18.15: (*continued*)

TRELLIS-CODED 4 × 16-PSK

$\eta = 3.75$ bits/symbol, $q = 0$, $d^2_{min} = 0.304$, $A_{min} = 24$ (4 × 16-PSK)

ν	$\tilde{k}$	$\mathbf{h}^{(4)}$	$\mathbf{h}^{(3)}$	$\mathbf{h}^{(2)}$	$\mathbf{h}^{(1)}$	$\mathbf{h}^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	—	1	3	22.5°	0.304	8	0.457	64	0
2	2	—	—	2	1	5	22.5°	0.457	48	—	—	1.76
3	2	—	—	04	02	11	22.5°	0.586	8	0.609	64	2.84
4	3	—	10	04	02	21	22.5°	0.609	40	—	—	3.01
5	3	—	30	14	02	41	22.5°	0.609	8	0.761	288	3.01
6	4	030	020	052	014	101	22.5°	0.761	136	—	—	3.98

TRELLIS-CODED 4 × 16-PSK

$\eta = 3.50$ bits/symbol, $q = 1$, $d^2_{min} = 0.304$, $A_{min} = 4$ (2 × 16-PSK)

ν	$\tilde{k}$	$\mathbf{h}^{(3)}$	$\mathbf{h}^{(2)}$	$\mathbf{h}^{(1)}$	$\mathbf{h}^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	1	3	22.5°	0.586	8	0.609	64	2.84
2	2	—	2	1	5	22.5°	0.609	40	—	—	3.01
3	2	—	04	02	11	22.5°	0.609	8	0.890	32	3.01
4	3	14	04	02	21	22.5°	0.890	16	—	—	4.66
5	3	24	14	02	41	22.5°	0.913	64	—	—	4.77
6	3	014	024	042	103	22.5°	1.172	24	1.218	1088	5.85

TRELLIS-CODED 4 × 16-PSK

$\eta = 3.25$ bits/symbol, $q = 2$, $d^2_{min} = 0.586$, $A_{min} = 8$ (4 × 16-PSK)

ν	$\tilde{k}$	$\mathbf{h}^{(4)}$	$\mathbf{h}^{(3)}$	$\mathbf{h}^{(2)}$	$\mathbf{h}^{(1)}$	$\mathbf{h}^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	—	1	3	22.5°	0.609	8	0.890	32	0.17
2	2	—	—	3	1	5	22.5°	0.890	16	—	—	1.82
3	2	—	—	06	02	11	22.5°	1.172	24	1.195	64	3.01
	2	—	—	02	06	11	45°	1.172	24	1.218	64	3.01
4	3	—	04	06	12	21	22.5°	1.172	8	1.218	32	3.01
5	4	10	04	06	22	41	22.5°	1.218	16	—	—	3.18
6	4	050	030	024	016	101	22.5°	1.499	72	—	—	4.08

TRELLIS-CODED 4 × 16-PSK

$\eta = 3.00$ bits/symbol, $q = 3$, $d^2_{min} = 0.586$, $A_{min} = 2$ (1 × 8-PSK)

ν	$\tilde{k}$	$\mathbf{h}^{(3)}$	$\mathbf{h}^{(2)}$	$\mathbf{h}^{(1)}$	$\mathbf{h}^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	1	3	45°	1.172	24	1.218	64	3.01
2	2	—	2	3	5	22.5°	1.172	8	1.218	32	3.01
3	3	02	04	03	11	22.5°	1.218	16	—	—	3.18
4	3	04	10	06	21	22.5°	1.781	48	—	—	4.83
5	3	22	16	06	41	22.5°	1.804	24	—	—	4.88
	3	24	14	02	43	45°	1.827	64	—	—	4.94
6	3	050	024	006	103	22.5°	2.0	8	2.343	64	5.33

TABLE 18.15: (*continued*)

(c) Codes for QAM

TRELLIS-CODED $\mathbb{Z}^4$

$\eta = J + 1/2$ bits/symbol, $q = 0$, $d^2_{min} = 2$, $A_{min} = 24\ (\mathbb{Z}^4/2)$

ν	$\tilde{k}$	$h^{(3)}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	—	1	3	180°	2	8	3	64	0.00
2	2	—	—	1	3	5	90°	3	48	—	—	1.76
3	2	—	—	02	06	11	90°	4	88	—	—	3.01
4	2	—	—	10	06	23	90°	4	24	5	512	3.01
5	3	—	34	10	06	41	90°	4	8	5	288	3.01
6	4	030	020	052	004	101	180°	5	136	—	—	3.98

TRELLIS-CODED $\mathbb{Z}^4/2$

$\eta = J$ bits/symbol, $q = 1$, $d^2_{min} = 2$, $A_{min} = 4\ (\mathbb{Z}^2/2)$

ν	$\tilde{k}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	1	3	90°	4	88	—	—	3.01
2	1	—	—	2	5	90°	4	24	6	512	3.01
3	2	—	04	02	11	90°	4	8	6	320	3.01
4	3	14	04	02	21	180°	6	176	—	—	4.77
5	3	24	14	06	43	180°	6	48	—	—	4.77
6	3	024	014	042	103	180°	8	1112	—	—	6.02
7	3	044	034	106	203	180°	8	216	—	—	6.02

TRELLIS-CODED $\mathbb{Z}^6$

$\eta = J + 2/3$ bits/symbol, $q = 0$, $d^2_{min} = 2$, $A_{min} = 60\ (\mathbb{Z}^6/2, \mathrm{I})$

ν	$\tilde{k}$	$h^{(4)}$	$h^{(3)}$	$h^{(2)}$	$h^{(1)}$	$h^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)	Signal set
1	1	—	—	—	1	3	90°	2	28	3	256	0.00	I
2	1	—	—	—	2	5	90°	2	12	3	64	0.00	II
3	2	—	—	02	06	11	90°	2	4	3	32	0.00	III
4	2	—	—	14	02	21	180°	3	48	—	—	1.76	II
	2	—	—	16	04	23	360°	3	40	—	—	1.76	II
	3	—	12	04	02	21	180°	4	316	—	—	3.01	I
5	3	—	24	14	02	41	180°	4	124	—	—	3.01	I
6	3	—	024	042	010	105	180°	4	60	5	512	3.01	I
	4	044	024	072	022	103	90°	4	28	—	—	3.01	III

(*continued overleaf*)

TABLE 18.15: (*continued*)

TRELLIS-CODED $\mathbb{Z}^6/2$

$\eta = J + 1/3$ bits/symbol, $q = 1$, $d^2_{min} = 2$, $A_{min} = 12$ ($\mathbb{Z}^6/4$, II)

ν	$\tilde{k}$	$\mathbf{h}^{(4)}$	$\mathbf{h}^{(3)}$	$\mathbf{h}^{(2)}$	$\mathbf{h}^{(1)}$	$\mathbf{h}^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)	Signal set
1	1	—	—	—	1	3	90°	2	4	4	64	0.00	III
	1	—	—	—	1	3	360°	3	56	—	—	1.76	II
2	1	—	—	—	2	5	360°	3	32	5	288	1.76	II
	2	—	—	3	1	5	180°	4	348	—	—	3.01	I
3	2	—	—	02	06	11	180°	4	60	6	6400	3.01	II
	3	—	06	04	03	11	90°	4	28	—	—	3.01	III
4	3	—	14	04	12	23	90°	4	12	5	160	3.01	III
5	4	10	12	22	04	41	90°	5	96	—	—	3.98	III
6	4	044	006	022	010	111	90°	6	1012	—	—	4.77	III
	4	030	046	014	042	101	180°	6	856	—	—	4.77	III

TRELLIS-CODED $\mathbb{Z}^6/4$

$\eta = J$ bits/symbol, $q = 2$, $d^2_{min} = 2$, $A_{min} = 4$ ($\mathbb{Z}^2/2$)

ν	$\tilde{k}$	$\mathbf{h}^{(4)}$	$\mathbf{h}^{(3)}$	$\mathbf{h}^{(2)}$	$\mathbf{h}^{(1)}$	$\mathbf{h}^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)	Signal set
1	1	—	—	—	1	3	180°	4	60	6	1024	3.01	II
2	2	—	—	3	2	5	90°	4	12	5	128	3.01	III
3	3	—	04	03	02	11	90°	5	64	—	—	3.98	III
4	3	—	04	06	12	21	90°	6	176	—	—	4.77	III
5	3	—	30	14	16	41	90°	6	32	7	288	4.77	III
	4	20	10	04	02	41	180°	7	240	—	—	5.44	II
6	4	024	010	004	042	101	180°	8	700	—	—	6.02	II
	4	044	024	014	002	103	360°	8	316	—	—	6.02	II

TRELLIS-CODED $\mathbb{Z}^8$

$\eta = J + 3/4$ bits/symbol, $q = 0$, $d^2_{min} = 2$, $A_{min} = 112$ ($\mathbb{Z}^8/2$)

ν	$\tilde{k}$	$\mathbf{h}^{(3)}$	$\mathbf{h}^{(2)}$	$\mathbf{h}^{(1)}$	$\mathbf{h}^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	1	3	90°	2	48	3	512	0.00
2	2	—	2	1	5	90°	2	16	3	384	0.00
3	3	04	02	01	11	90°	3	224	—	—	1.76
4	3	10	04	02	21	90°	4	1264	—	—	3.01
5	3	24	14	02	41	90°	4	496	—	—	3.01
6	3	050	032	004	103	90°	4	240	5	5120	3.01

TRELLIS-CODED $\mathbb{Z}^8/2$

$\eta = J + 1/2$ bits/symbol, $q = 1$, $d^2_{min} = 2$, $A_{min} = 24$ ($\mathbb{Z}^4/2$)

ν	$\tilde{k}$	$\mathbf{h}^{(4)}$	$\mathbf{h}^{(3)}$	$\mathbf{h}^{(2)}$	$\mathbf{h}^{(1)}$	$\mathbf{h}^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	—	1	3	90°	2	16	4	1024	0.00
2	2	—	—	2	1	5	90°	4	1264	—	—	3.01
3	2	—	—	04	02	11	90°	4	496	—	—	3.01
4	2	—	—	12	04	23	90°	4	240	6	20480	3.01
5	3	—	14	34	06	41	90°	4	112	6	11264	3.01
	3	—	04	14	22	43	180°	4	112	6	10240	3.01
6	4	014	006	056	022	103	90°	4	48	6	3584	3.01

TABLE 18.15: (*continued*)

TRELLIS-CODED $\mathbb{Z}^8/4$

$\eta = J + 1/4$ bits/symbol, $q = 2$, $d^2_{min} = 2$, $A_{min} = 16$ ($\mathbb{Z}^8/8$)

ν	$\tilde{k}$	$\mathbb{h}^{(4)}$	$\mathbb{h}^{(3)}$	$\mathbb{h}^{(2)}$	$\mathbb{h}^{(1)}$	$\mathbb{h}^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	—	1	3	90°	4	496	—	—	3.01
2	1	—	—	—	2	5	90°	4	240	6	4096	3.01
3	2	—	—	06	02	11	90°	4	112	6	4096	3.01
	2	—	—	02	06	11	180°	4	112	6	2048	3.01
4	3	—	10	14	06	21	90°	4	48	6	512	3.01
5	4	10	04	06	22	41	90°	4	16	6	512	3.01

TRELLIS-CODED $\mathbb{Z}^8/8$

$\eta = J$ bits/symbol, $q = 3$, $d^2_{min} = 2$, $A_{min} = 4$ ($\mathbb{Z}^2/2$)

ν	$\tilde{k}$	$\mathbb{h}^{(4)}$	$\mathbb{h}^{(3)}$	$\mathbb{h}^{(2)}$	$\mathbb{h}^{(1)}$	$\mathbb{h}^{(0)}$	Invariance	d^2_{free}	$A_{d_{free}}$	d^2_{next}	$A_{d_{next}}$	γ (dB)
1	1	—	—	—	1	3	180°	4	112	8	16384	3.01
2	2	—	—	2	3	5	90°	4	48	8	16384	3.01
3	3	—	04	02	03	11	90°	4	16	8	11264	3.01
4	4	10	04	02	03	21	90°	8	6896	—	—	6.02
5	4	20	10	02	06	41	90°	8	3056	—	—	6.02
6	4	042	020	010	006	101	90°	8	1264	—	—	6.02

Adapted from [33, 34].

- The spectral efficiency η in bits/symbol. For the QAM signal sets, η is given as an integer J plus some fraction, where the constituent 2-D signal set S has 2^{J+1} signal points; that is, $J = I - 1$.
- The value of q that determines the multilevel subcode at which partitioning begins.
- The MSE distance d^2_{min} and number of nearest neighbors $A_{\min}$ of the uncoded signal set with the same spectral efficiency chosen for comparison.
- The number of coded bits $\tilde{k}$, parity-check polynomials $\mathbb{h}^{(\tilde{k})}, \cdots, \mathbb{h}^{(1)}, \mathbb{h}^{(0)}$ in octal form, minimum rotational invariance, MFSE distance d^2_{free}, the average number of nearest neighbors $A_{d_{free}}$, and the asymptotic coding gain γ for the best code of each constraint length ν.
- When d^2_{free} occurs along parallel transitions, the next largest SE distance d^2_{next} and the average number of next-nearest neighbors $A_{d_{next}}$ are also given.
- In the $L = 3$ cases, the partition that results in the best code is also given.
- Sometimes more than one code is listed for a given constraint length to indicate the best code for different degrees of rotational invariance. In these cases, the code with the worst rotational invariance always has a larger free distance or a smaller average number of nearest or next-nearest neighbors.

The following comments apply to Table 18.15:

- For a given multi-D signal constellation, each value of q results in a different spectral efficiency. For example, for 4×8-PSK, the values of $q = 0, 1, 2$, and 3 result in spectral efficiencies $\eta = 2.75, 2.5, 2.25$, and 2.0 bits/symbol, respectively.
- In general, different values of q result in different optimum codes for a given multi-D signal constellation.
- The uncoded signal sets that are chosen for comparison are uncoded subsets of a multi-D or 2-D signal set with the same spectral efficiency. For the QAM signal sets, these are uncoded subsets of an integer lattice. For example, $\mathbb{Z}^4/2$ denotes the best binary partitioning of the integer lattice $\mathbb{Z}^4$. In the 6-D case, the partition (I, II, or III from Example 18.17) that determines the uncoded subset is indicated.
- Unlike the case with 2-D signal sets, multi-D signal sets are capable of achieving full rotational invariance with linear PCEs, as first noted by Wei [35]. In fact, many of the best codes listed in Table 18.15 have full rotational invariance.
- Tables of optimum codes for multi-D QPSK signal sets are also given in [33]. Again, many of the best codes achieve full rotational invariance with linear PCEs.
- Tables of optimum geometrically uniform codes for $L \times M$-PSK signal sets are given in [36].
- An estimate of the real coding gain at a BER of 10^{-5} for the codes in Tables 18.6–18.9 and Table 18.15 can be obtained by adding the adjustment factor $-0.2\log_2(A_{d_{free}}/A_{min})$ dB to the asymptotic coding gain γ given in the table. For example, from Table 18.15(a), we see that the estimated real coding gain of the 8-state, 2×8-PSK code of Example 18.15 with spectral efficiency $\eta = 2.5$ bits/symbol is given by $\gamma_{adj} \approx \gamma - 0.2\log_2(A_{d_{free}}/A_{min}) = 3.98 - 0.40 = 3.58$ dB.

As noted previously, multi-D signal sets are capable of achieving full rotational invariance with linear PCEs. We now illustrate this fact with an example.

EXAMPLE 18.19 Rotational Invariance for Multi-D TCM

Consider the 8-state, rate $R = 2/3$, 2×8-PSK TCM system with spectral efficiency $\eta = 2.5$ bits/symbol of Example 18.15. The parity-check matrix is given by

$$\mathbb{H}(D) = \left[D^2/(D^3+1) \;\; (D^2+D)/(D^3+1) \;\; 1\right], \tag{18.97}$$

and the binary parity-check equation can be written as

$$\begin{aligned}\mathbb{V}(D)\mathbb{H}^T(D) &= \mathbb{v}^{(2)}(D)\left[D^2/(D^3 \oplus 1)\right] \oplus \mathbb{v}^{(1)}(D)\left[(D^2 \oplus D)/(D^3 \oplus 1)\right] \oplus \mathbb{v}^{(0)}(D) \\ &= \mathbb{0}(D)\end{aligned} \tag{18.98a}$$

or

$$D^2\mathbb{v}^{(2)}(D) \oplus (D^2 \oplus D)\mathbb{v}^{(1)}(D) \oplus (D^3 \oplus 1)\mathbb{v}^{(0)}(D) = \mathbb{0}(D). \tag{18.98b}$$

For the naturally mapped constituent 8-PSK signal set shown in Figure 18.6, a 45° phase rotation of a signal point y results in the rotated signal point $y_r = y + 1 \pmod 8$, or in binary notation (see also Problem 18.25),

$$v_r^{(0)} = v^{(0)} \oplus 1, \tag{18.99a}$$

$$v_r^{(1)} = v^{(1)} \oplus v^{(0)}, \tag{18.99b}$$

$$v_r^{(2)} = v^{(2)} \oplus v^{(0)} \circ v^{(1)}. \tag{18.99c}$$

To determine the effect of a 45° phase rotation of the 2×8-PSK signal vector $\mathbb{Y} = (y_1, y_2)$ on the binary code vector $\mathbb{v} = (v^{(5)}v^{(4)}v^{(3)}v^{(2)}v^{(1)}v^{(0)})$, we express the mapping function of (18.71) as

$$\mathbb{Y} = (y_1, y_2) = (4v^{(5)} + 2v^{(3)} + v^{(1)})(1, 1) + (4v^{(4)} + 2v^{(2)} + v^{(0)})(0, 1) \pmod 8. \tag{18.100}$$

Now, after a 45° phase rotation, the integer representations of the rotated signal points are given by

$$y_{i,r} = y_i + 1 \pmod 8, \qquad i = 1, 2, \tag{18.101}$$

and we can write (18.100) as

$$\begin{aligned}\mathbb{Y}_r = (y_{1,r}, y_{2,r}) &= (4v^{(5)} + 2v^{(3)} + v^{(1)})(1, 1) \\ &\quad + (4v^{(4)} + 2v^{(2)} + v^{(0)})(0, 1) + (1, 1) \pmod 8 \\ &= (4v^{(5)} + 2v^{(3)} + v^{(1)} + 1)(1, 1) + (4v^{(4)} + 2v^{(2)} + v^{(0)})(0, 1) \pmod 8.\end{aligned} \tag{18.102}$$

By comparing (18.100) and (18.102), we see that bits $v^{(5)}$, $v^{(3)}$, and $v^{(1)}$ are affected by the multi-D 2×8-PSK 45° phase rotation in the same way as bits $v^{(2)}$, $v^{(1)}$, and $v^{(0)}$ are affected when the constituent 8-PSK signal set is rotated by 45°, whereas bits $v^{(4)}$, $v^{(2)}$, and $v^{(0)}$ are not affected at all. Thus, we can write

$$\begin{aligned} &v_r^{(0)} = v^{(0)} \qquad v_r^{(2)} = v^{(2)} \qquad v_r^{(4)} = v^{(4)} \\ &v_r^{(1)} = v^{(1)} \oplus 1 \quad v_r^{(3)} = v^{(3)} \oplus v^{(1)} \quad v_r^{(5)} = v^{(5)} \oplus v^{(1)} \circ v^{(3)}.\end{aligned} \tag{18.103}$$

To check for rotational invariance of the linear PCE of (18.98a), we use (18.103) to form the rotated encoder output sequences $\mathbb{v}_r^{(0)}(D) = \mathbb{v}^{(0)}(D)$, $\mathbb{v}_r^{(1)}(D) = \mathbb{v}^{(1)}(D) \oplus \mathbb{1}(D)$, and $\mathbb{v}_r^{(2)}(D) = \mathbb{v}^{(2)}(D)$. Substituting these rotated sequences into (18.98b), we obtain

$$\begin{aligned}\mathbb{V}_r(D)\mathbb{H}^T(D) &= D^2\mathbb{v}_r^{(2)}(D) \oplus (D^2 \oplus D)\mathbb{v}_r^{(1)}(D) \oplus (D^3 \oplus 1)\mathbb{v}_r^{(0)}(D) \\ &= D^2\mathbb{v}^{(2)}(D) \oplus (D^2 \oplus D)[\mathbb{v}^{(1)}(D) \oplus \mathbb{1}(D)] \oplus (D^3 \oplus 1)\mathbb{v}^{(0)}(D) \\ &= (D^2 \oplus D)\mathbb{1}(D) = \mathbb{0}(D);\end{aligned} \tag{18.104}$$

that is, the rotated sequences satisfy the PCE. Thus, this 2×8-PSK TCM system with a linear PCE is invariant to 45° phase rotations. Because only $v^{(5)}$, $v^{(3)}$, and $v^{(1)}$ are affected by the phase rotation, these are the only bits that must be differentially encoded. (In this example, only $v^{(1)}$ is an encoder input bit, whereas $v^{(3)}$ and $v^{(5)}$ are uncoded input bits.)

We now conclude our discussion of rotational invariance for multi-D TCM with a few remarks.

- In general, for full $(360/M)°$ rotational invariance of M-PSK signal sets, $I = \log_2 M$ bits are affected by the phase rotation and must be differentially encoded. (For QAM signal sets, full 90° rotational invariance requires that 2 bits be differentially encoded.) In the case $q = 0$, that is, when partitioning begins with the entire multi-D signal set, the levels in the partition chain corresponding to the I affected bits are denoted by $\pi_0, \pi_1, \cdots, \pi_{I-1}$. These affected bits are listed in Tables 18.13 and 18.14 summarizing multi-D partitions. (Note that the bits to be differentially encoded depend only on the signal set and the partition, and not on the particular code.)

- In the case $q > 0$, that is, when partitioning begins with a subset of the entire multi-D signal set, the I affected bits correspond to levels $\pi_0 - q$, $\pi_1 - q$, $\ldots$, $\pi_{I-1} - q$ in the partition chain. (If $\pi_j - q$ is less than zero for some j, full rotational invariance cannot be achieved. See [33] for more details on determining the bits to be differentially encoded when full rotational invariance is not possible.)

- If $v^{(0)}$, the parity bit, is one of the bits affected by a phase rotation, it must be fed back to the differential encoder from the encoder output, since it is not an encoder input bit. The reader is again referred to [33] for details.

- The possibility of achieving full rotational invariance with linear PCEs and multi-D signal sets depends on the multi-D mapping functions of (18.93) and (18.95). (As shown in Section 18.4, the simple mapping function used for 2-D signal sets does not allow full rotational invariance to be achieved with linear PCEs.) For a given multi-D signal set, partition, and linear PCE, the degree of rotational invariance can be determined by checking whether the rotated encoder output sequences satisfy the PCE (see Problem 18.32). In general, it is possible to achieve full rotational invariance with linear encoders of rates $R = 1/2, 2/3$, and $3/4$ and multi-D signal sets, but higher encoder rates still require nonlinear encoders similar to those employed in Section 18.4.

- Uncoded bits affected by a phase rotation do not influence the rotational invariance of the code; however, they must still be differentially encoded.

- More general classes of both 2-D and multi-D TCM systems can be obtained by defining the code alphabet over a ring or a group rather than using conventional finite-field algebra. These more general formulations often make

it easier to obtain rotationally invariant systems and may lead to codes with better distance properties, but code searches, performance analysis, and encoder implementation are more difficult. Some approaches to constructing rotationally invariant codes based on group designs are summarized in [37, 38], and tables of optimum geometrically uniform codes over groups for $L \times M$-PSK signal sets are given in [39].

We now use an example to show how the performance analysis techniques of Section 18.3 can be applied to multi-D TCM.

EXAMPLE 18.20 Performance Analysis for Multi-D TCM

Consider the 4-state, rate $R = 1/2$ binary feedback encoder used in Example 18.9. The encoder diagram and binary error trellis (with one uncoded information bit) of this encoder are shown in Figures 18.9(a) and 18.9(b), respectively. In this example, we use the same encoder along with three uncoded information bits to partition the 32-point, 4-D subset $Q^2\,(0 \times 0 \cup 1 \times 1)$ (corresponding to subcode $\Lambda_1(0)$) of the 2×8-PSK, 64-point, 4-D signal set (see Figure 18.29). The first two levels of partitioning are shown in Figure 18.32.

In this case, $q = 1$, the spectral efficiency $\eta = 2.0$ bits/symbol, the MSSDs are $\Delta_0^2 = 1.172$, $\Delta_1^2 = 2.0$, and $\Delta_2^2 = 4.0$, there are eight parallel transitions on each branch of the binary error trellis, and the mapping function (see (18.95a)) from the

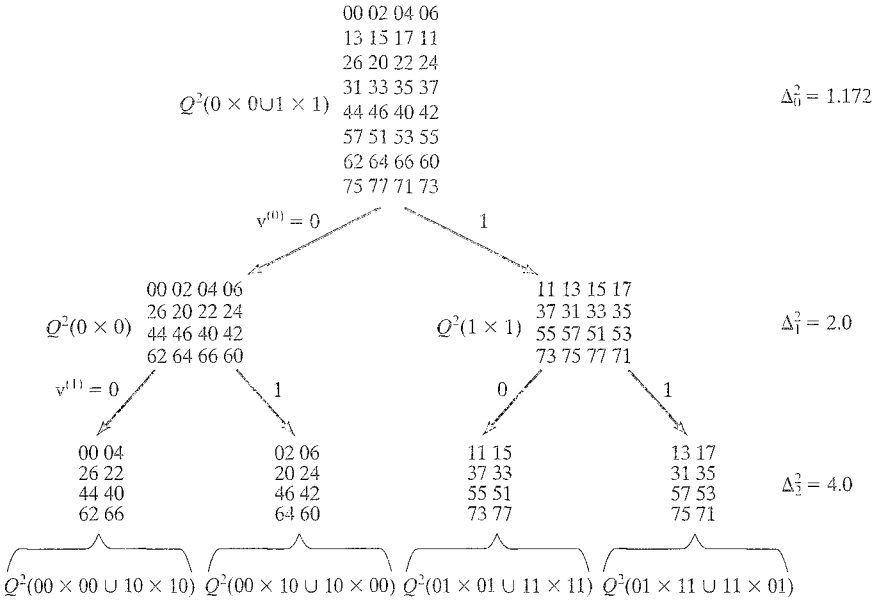


FIGURE 18.32: Partitioning of the 32-point, 4-D signal set $Q^2\,(0 \times 0 \cup 1 \times 1)$.

binary code vector $\mathbf{v} = (v^{(4)}v^{(3)}v^{(2)}v^{(1)}v^{(0)})$ to the binary signal vector $\mathbb{Y}$ is given by

$$\begin{aligned}\mathbb{Y} = (\mathbb{y}_1, \mathbb{y}_2) &= v^{(4)}\mathbb{g}_5 \oplus v^{(3)}\mathbb{g}_4 \oplus v^{(2)}\mathbb{g}_3 \oplus v^{(1)}\mathbb{g}_2 \oplus v^{(0)}\mathbb{g}_1 \\ &= v^{(4)}\begin{bmatrix} 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \oplus v^{(3)}\begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \oplus v^{(2)}\begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 0 \end{bmatrix} \oplus \\ &\quad v^{(1)}\begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \oplus v^{(0)}\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}. \end{aligned} \tag{18.105}$$

(If the coset generators $\mathbb{g}_i$ are expressed in integer form, then (18.95b) using modulo-8 addition gives $\mathbb{Y} = (y_1, y_2)$ in integer form.) This TCM system is listed in Table 18.15(a), where it is seen that $d^2_{free} = 4.0$, $A_{d_{free}} = 6$, $d^2_{next} = 5.172$, and $A_{d_{next}} = 32$.

Before we can calculate the AWEFs for the trellis paths $A^t_{av}(X)$ and the parallel transitions $A^p_{av}(X)$, we must first determine the AEWEs of the 32 possible binary error vectors $\mathbf{e} = (e^{(4)}e^{(3)}e^{(2)}e^{(1)}e^{(0)})$. We do this by using the AEWEs for naturally mapped 8-PSK listed in Table 18.5. First, we use a binary error vector $\mathbf{e}$ in (18.105) to compute the 4-D error signal $\mathbb{E} = (\mathbb{e}_1, \mathbb{e}_2)$. Next, we use $\mathbb{e}_1$ and $\mathbb{e}_2$ to determine the AEWEs of each component of $\mathbb{E}$ from Table 18.5. We then multiply these two AEWEs together to form the AEWE corresponding to the error vector $\mathbf{e}$. For example, the error vector $\mathbf{e} = (01011)$ results in the error signal $\mathbb{E} = [(001)^T, (111)^T]$, or $\mathbb{E} = (1, 7)$ in integer form. From Table 18.5 we see that the corresponding AEWEs are given by $\Delta^2_{(001)}(X) = X^{0.586}$, and $\Delta^2_{(111)}(X) = 0.5X^{0.586} + 0.5X^{3.414}$. Thus, the AEWE for the error vector $\mathbf{e} = (01011)$ is given by

$$\begin{aligned}\Delta^2_{(01011)}(X) &= \Delta^2_{(001)}(X)\Delta^2_{(111)}(X) \\ &= (X^{0.586})(0.5X^{0.586} + 0.5X^{3.414}) \\ &= 0.5X^{1.172} + 0.5X^{4.0};\end{aligned} \tag{18.106}$$

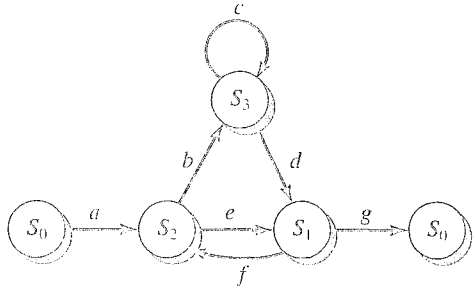
that is, for branches in the binary error trellis labeled with the error vector $\mathbf{e} = (01011)$, half the branches have squared Euclidean weight 1.172, and the other half have weight 4.0. The complete list of AEWEs corresponding to the 32 possible error vectors in this example is given in Table 18.16.

The modified state diagram, similar to the modified state diagram of Figure 18.19(a) for Example 18.9, is shown in Figure 18.33, where in this case each branch is labeled with the sum of the eight AEWEs corresponding to the eight parallel transitions on that branch. For example, the branch labeled c represents the self-loop around the state S_3, caused by the input bit 0 and resulting in the output bits 01, for the rate $R = 1/2$ encoder. Thus, the eight possible parallel transition error vectors corresponding to this branch are $\mathbf{e} = (00001)$, (00101), (01001), (01101), (10001), (10101), (11001), and (11101), and from Table 18.16 the branch label is given by

$$\begin{aligned}c &= X^{1.172} + 2X^{4.0} + 4(0.25X^{1.172} + 0.5X^{4.0} + 0.25X^{6.828}) + X^{6.828} \\ &= 2X^{1.172} + 4X^{4.0} + 2X^{6.828}.\end{aligned} \tag{18.107a}$$

TABLE 18.16: AEWEs for the 32-point, 4-D signal set $Q^2(0 \times 0 \cup 1 \times 1)$ with the mapping of (18.105).

$\mathbf{e}$	$\Delta_{\mathbf{e}}^2(X)$	$\mathbf{e}$	$\Delta_{\mathbf{e}}^2(X)$
00000	X^0	10000	X^8
00001	$X^{1.172}$	10001	$X^{6.828}$
00010	X^2	10010	X^6
00011	$\frac{1}{2}X^{1.172} + \frac{1}{2}X^4$	10011	$\frac{1}{2}X^4 + \frac{1}{2}X^{6.828}$
00100	X^4	10100	X^4
00101	$\frac{1}{4}X^{1.172} + \frac{1}{2}X^4 + \frac{1}{4}X^{6.828}$	10101	$\frac{1}{4}X^{1.172} + \frac{1}{2}X^4 + \frac{1}{4}X^{6.828}$
00110	X^6	10110	X^2
00111	$\frac{1}{2}X^4 + \frac{1}{2}X^{6.828}$	10111	$\frac{1}{2}X^{1.172} + \frac{1}{2}X^4$
01000	X^4	11000	X^4
01001	X^4	11001	X^4
01010	X^2	11010	X^6
01011	$\frac{1}{2}X^{1.172} + \frac{1}{2}X^4$	11011	$\frac{1}{2}X^4 + \frac{1}{2}X^{6.828}$
01100	X^4	11100	X^4
01101	$\frac{1}{4}X^{1.172} + \frac{1}{2}X^4 + \frac{1}{4}X^{6.828}$	11101	$\frac{1}{4}X^{1.172} + \frac{1}{2}X^4 + \frac{1}{4}X^{6.828}$
01110	X^2	11110	X^6
01111	$\frac{1}{2}X^{1.172} + \frac{1}{2}X^4$	11111	$\frac{1}{2}X^4 + \frac{1}{2}X^{6.828}$



$a = 4X^2 + 4X^6$
$b = 2X^{1.172} + 4X^4 + 2X^{6.828}$
$c = 2X^{1.172} + 4X^4 + 2X^{6.828}$
$d = 2X^{1.172} + 4X^4 + 2X^{6.828}$
$e = 2X^{1.172} + 4X^4 + 2X^{6.828}$
$f = X^0 + 6X^4 + X^8 = 1 + 6X^4 + X^8$
$g = 4X^2 + 4X^6$

FIGURE 18.33: Modified state diagram for Example 18.20.

We determine the remaining branch labels in a similar way, yielding

$$\begin{aligned} a &= 4X^{2.0} + 4X^{6.0} \\ b &= 2X^{1.172} + 4X^{4.0} + 2X^{6.828} \\ d &= 2X^{1.172} + 4X^{4.0} + 2X^{6.828} \\ e &= 2X^{1.172} + 4X^{4.0} + 2X^{6.828} \\ f &= X^0 + 6X^{4.0} + X^{8.0} = 1 + 6X^{4.0} + X^{8.0} \\ g &= 4X^{2.0} + 4X^{6.0}. \end{aligned} \tag{18.107b}$$

In this case, we note from (18.107) that $a = g$, and $b = c = d = e$. Now, we can compute the AWEF for the trellis paths as follows:

$$A_{av}^{t}(X) = \frac{aeg(1-c) + abdg}{(1-c)(1-ef) - bdf} = \frac{a^2 b}{1 - b(1+f)}, \tag{18.108}$$

where the simplified expression follows from the foregoing identities. Carrying through the algebra associated with (18.108), we obtain the following result:

$$\begin{aligned} A_{av}^{t}(X) &= \frac{\begin{array}{c} 32X^{5.172} + 64X^{8.0} + 64X^{9.172} + 32X^{10.828} + 128X^{12.0} \\ +32X^{13.172} + 64X^{14.828} + 64X^{16.0} + 32X^{18.828} \end{array}}{\begin{array}{c} 1 - 4X^{1.172} - 8X^{4.0} - 12X^{5.172} - 4X^{6.828} - 24X^{8.0} \\ -2X^{9.172} - 12X^{10.828} - 4X^{12.0} - 2X^{14.828} \end{array}} \\ &= 32X^{5.172} + 128X^{6.344} + 64X^{8.0} + 576X^{9.172} + \cdots. \end{aligned} \tag{18.109}$$

Thus, for a given path through the trellis, there are an average of 32 error paths with a free SE distance of 5.172, an average of 128 error paths with a distance of 6.344, and so on.

The AWEF for the parallel transitions is given by the set of intrasubset distances within the subsets at the final partition level $p = \tilde{k} + 1 = 2$. From the subset Q^2 ($00 \times 00 \cup 10 \times 10$) in Figure 18.32 we see that

$$A_{av}^{p}(X) = 6X^{4.0} + X^{8.0}. \tag{18.110}$$

(Note that $A_{av}^{p}(X)$ is just the set of distances corresponding to the parallel transitions in the self-loop around the all-zero state S_0. This set of distances is represented by the branch label f in the modified state diagram (with the weight-0 term, which corresponds to comparing a branch to itself, removed), since the branch label f gives the AEWE for the self-loop around the state S_0.) Equation (18.110) implies that there are six parallel transition error events with a distance of 4.0 and one with a distance of 8.0.

Summarizing, we see from (18.109) and (18.110) that the minimum distance between parallel transitions is $\delta_{min}^2 = 4.0$, the minimum distance between trellis paths is $\delta_{free}^2 = 5.172$, the overall MFSE distance is

$$d_{free}^2 = \min\left\{\delta_{free}^2, \delta_{min}^2\right\} = 4.0, \tag{18.111}$$

the number of nearest neighbors is $A_{d_{free}} = 6$, and the number of next-nearest neighbors is $A_{d_{next}} = 32$ ($d^2_{next} = 5.172$), as noted in Table 18.15(a). Finally, we can use the preceding expressions for $A^I_{av}(X)$ and $A^P_{av}(X)$ in (18.29a) to evaluate the standard transfer function bound on event-error probability $P(E)$.

We now conclude our discussion of performance analysis for multi-D TCM with a few comments.

- In the multi-D case, the AEWEs depend on the mapping function given by (18.93) or (18.95).
- Branches in the binary error trellis with the same set of binary labels must have the same labels in the modified state diagram; that is, $a = g$, $b = d$, and $c = e$ in Example 18.20.
- The parallel transition AWEF can be found from the AEWE of the self-loop around the all-zero state S_0 by dropping the weight-0 term, since this gives the set of distances in the subsets at the final level of partitioning.
- Following the procedure of Example 18.9, we can use a modified state diagram augmented by the input weight enumerators to determine the AIOWEFs $A'_{av}(W, X)$ and $A^P_{av}(W, X)$ (see Problem 18.33). We can then use these in (18.29b) to evaluate the standard transfer function bound on bit-error probability $P_b(E)$.
- The foregoing method of performance analysis applies to multi-D TCM systems with linear encoders. Nonlinear encoders, similar to those employed in Section 18.4, also can be used to produce rotationally invariant multi-D TCM systems, but the same analysis method does not apply.

In applications requiring large values of spectral efficiency, such as high-speed modems, large multi-D signal constellations are typically used. For example, consider the 192-point 2-D generalized CROSS constellation shown in Figure 18.34. A 4-D version of this signal set, combined with a power-saving technique called *shaping*, can be used to achieve a spectral efficiency of $\eta = 7.0$ bits/symbol. The 4-D signal set contains a total of $192^2 = 36864$ signal points, enough to support a spectral efficiency of slightly more than 7.0 bits/symbol (since $\log_2 36864 = 15.17$, allowing for 1 parity bit and slightly more than 14 information bits spread over two symbols); however, if only a subset of $2^{15} = 32768$ signal points are actually used, the spectral efficiency is exactly 7.0 bits/symbol. This can be accomplished by identifying two categories of signal points, called *inner points* and *outer poinis*, in the constituent 2-D signal set, as illustrated in Figure 18.34. In this case, the set of inner points comprises the 128 lowest-energy signal points, and the set of outer points comprises the 64 highest-energy signal points. If all possible combinations of points from the two constituent constellations are allowed, *except that a pair of outer points is not a valid multi-D signal point*, $64^2 = 4096$ signal points are excluded, leaving exactly $2^{15} = 32768$ signal points. By eliminating all combinations of two outer points, we are excluding the multi-D signal points with the highest energies, thus achieving a power savings of about 0.3 dB compared with using the full signal

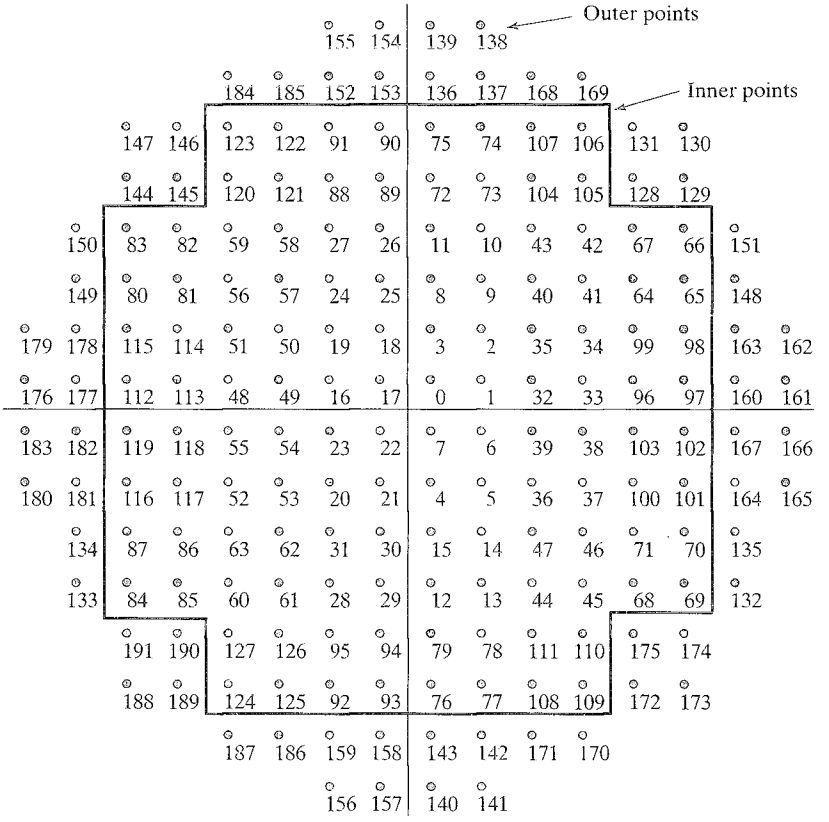


FIGURE 18.34: A 192-point generalized CROSS constellation.

set (see Problem 18.35). This power-saving approach to multi-D constellation design is called shaping. Ultimately, for very large signal dimensionalities and very large constituent 2-D signal sets, an approximately spherical constellation (known to be optimal in high-dimensional signal spaces) is approached, and a *shaping gain* of as much as 1.53 dB can be achieved.

Now, we illustrate how a rate $R = 3/4$ encoder, combined with 11 uncoded bits, can be used to select a signal point from this multi-D shaped constellation. First, consider that all combinations of two inner signal points form a 4-D subset of $128^2 = 2^{14}$ points. This 4-D subset can be partitioned into $2^3 = 8$ subsets of $2^{11} = 2048$ points each. For a particular value of the parity bit, say $v^{(0)} = 0$, the three coded information bits $v^{(1)}$, $v^{(2)}$, and $v^{(3)}$ can be used to select one of the 8 subsets, and then the 11 uncoded bits can be used to select a signal point from the shaped constellation. Similarly, there are $64 \times 128 = 2^{13}$ mixed combinations of an inner point followed by an outer point, and the same number of mixed combinations of an outer point followed by an inner point, for a total of 2^{14} mixed combinations. Thus, for the other value of the parity bit, say $v^{(0)} = 1$, the three coded bits $v^{(1)}$, $v^{(2)}$, and $v^{(3)}$ along with the 11 uncoded bits can be used to select a mixed signal point from the shaped constellation.

We also note that the preceding example results in a *constellation expansion ratio* (CER) of 192/128 = 1.5; that is, the 4-D TCM design achieves a spectral efficiency of $\eta = 7.0$ bits/symbol by using a 2-D signal set with 50% more signal points than would be required to achieve the same spectral efficiency without coding. This is less than the 100% CER (to 256 signal points) that would be needed in a 2-D design and results in a lower peak-to-average power ratio for the multi-D design, an important practical consideration. (The *peak-to-average power ratio* (PAR) of a rectangular signal set is simply the ratio of the maximum energy of any point in the signal set to the average energy of the signal set. For two signal sets with the same distance between signal points, the signal set with the smaller CER will also have the smaller PAR (see Problem 18.36).)

We close this section with an example of the rate $R = 2/3$, 16-state, linear, 4-D, 90° rotationally invariant code designed by Wei [29] and used in the V.34 high-speed modem standard.

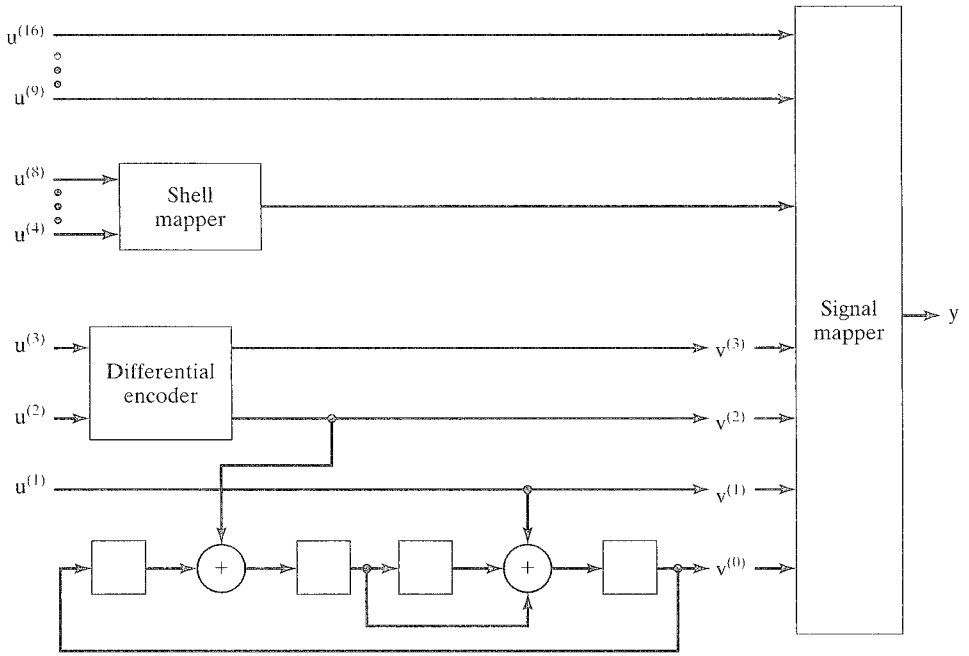
EXAMPLE 18.21 The V.34 4-D TCM System Design

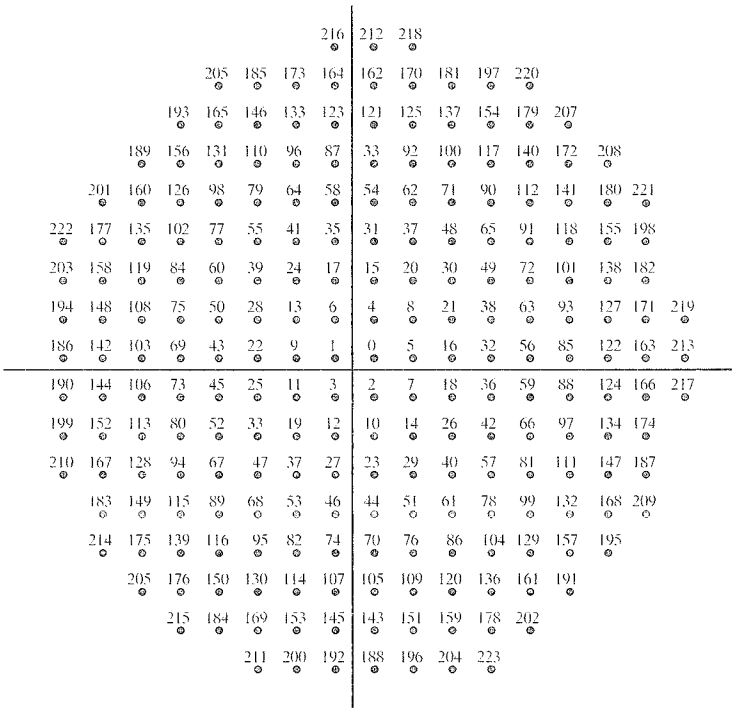
The V.34 high-speed modem standard uses a rate $R = 2/3$, 16-state linear encoder along with a multi-D signal constellation consisting of two (or more) uses of a 192-point (or larger) 2-D generalized CROSS constellation and as many as 8 uncoded information bits (plus the 2 coded information bits for a total of 10 information bits) per symbol to achieve spectral efficiencies as large as $\eta = 10.0$ bits/symbol and data rates as high as 33600 bps. Advanced line probing, adaptive equalization, and precoding techniques are used to increase the signaling rates to as high as 3429 symbols/second, compared with the previous standard of 2400 symbols/second. (In V.34, the data rates are not necessarily integer multiples of the symbol rates, since the signal mapping technique, called *shell mapping*, allows the mapping of a fractional number of bits per symbol.) The V.34 standard also includes two other codes that offer slightly better performance at a cost of increased complexity. One is a rate $R = 3/4$, 32-state, linear code that gains 0.3 dB compared with the 16-state code but is four times more complex. The other is a rate $R = 4/5$, 64-state, nonlinear code that gains an additional 0.15 dB but is another factor of 4 times more complex. In both of these cases more bits are encoded by the convolutional code, so fewer uncoded bits are needed to achieve a particular data rate.

A block diagram of the rate $R = 2/3$, 16-state, linear code is shown in Figure 18.35, along with a sketch of a 2-D (224-point) generalized CROSS constellation used in the V.34 standard. The rate $R = 2/3$ convolutional encoder has three input information bits: two coded ($u^{(1)}$ and $u^{(2)}$) and one uncoded ($u^{(3)}$). In addition, bits $u^{(2)}$ and $u^{(3)}$ are differentially encoded, since they are affected by phase rotations. Full 90° rotational invariance can be achieved with a linear code in this case, since the signal constellation is multidimensional. The rate $R = 2/3$ systematic feedback linear convolutional encoder has parity-check matrix

$$\mathbb{H}(D) = [(D+1)/(D^4+1) \;\; D^3/(D^4+1) \;\; 1] \tag{18.112}$$

and produces three output bits: $v^{(1)}$ and $v^{(2)}$ (information bits) and $v^{(0)}$ (a parity bit). (Note that the encoder diagram shown in Figure 18.35 is slightly different from the encoder diagram implied by (18.112). Problem 18.37 illustrates the equivalence





Signal constellation

FIGURE 18.35: The V.34 TCM system encoder and signal constellation.

of these two encoder diagrams.) Along with the differentially encoded bit $v^{(3)}$, the encoder output bits $v^{(0)}$, $v^{(1)}$, and $v^{(2)}$ enter the multi-D signal mapper (modulator). The constituent 2-D signal constellation is divided into 4 subsets, and thus there are 16 subsets in each pair of 2-D signals. The four bits $v^{(0)}$, $v^{(1)}$, $v^{(2)}$, and $v^{(3)}$ are mapped into one of the 16 subsets in each signaling pair. In the basic 4-D scheme, the additional uncoded information bits needed to achieve the desired spectral efficiency are then used to select the particular 4-D signal point to be transmitted. For example, for the shaped 4-D constellation based on the 192-point constituent signal set of Figure 18.34, if a spectral efficiency of $\eta = 7.0$ bits/symbol is desired, an additional 11 uncoded information bits are needed in each 4-D signaling interval.

This concept can be extended by further expanding the constituent 2-D signal set beyond 192 points and four dimensions in conjunction with the shell-mapping approach. In this approach, the signal mapping extends over many symbol intervals so that enough dimensions are included to produce an approximately spherical constellation. In the V.34 standard, the constituent 2-D signal set contains 448 points, the 224-point constellation of Figure 18.35 and its mirror image. Shell mapping is applied to a 16-D constellation, that is, four 4-D symbol intervals or eight 2-D symbol intervals, to achieve 0.8 dB of shaping gain. A total of 20 information bits over four 4-D symbol intervals are devoted to shell mapping, and 8 information bits per 4-D symbol are left uncoded. This arrangement gives a total spectral efficiency of

$$\eta = \frac{3}{2}(\text{coding}) + \frac{20}{8}(\text{shaping}) + \frac{8}{2}(\text{uncoded}) = 8.0 \text{ bits/symbol}. \qquad (18.113)$$

The size of the constituent 2-D signal set can vary depending on the desired shaping gain and CER. Usually, an addition to the CER of about 25% owing to shaping is enough to approach a 1.0-dB shaping gain. In this example the constellation size of 448 yields a CER $= 448/256 = 1.75$ compared with an uncoded system with $\eta = 8.0$ bits/symbol, giving a total of 75% constellation expansion, 50% due to coding and 25% due to shaping. Shaping was included in the standard because it was ultimately a simpler way of picking up an additional 0.8 dB than by using a more complex code.

The rate $R = 2/3$, 16-state, linear 4-D code used in the V.34 standard has free distance $d^2_{free} = 4.0$, number of nearest neighbors $A_{d_{free}} = 12$, and achieves a real coding gain (without bandwidth expansion) of 4.2 dB at a BER of 10^{-5} compared with uncoded 256-QAM ($\eta = 8.0$). (Note that the smaller free distance here compared with the 8-state V.32 code does not translate into less coding gain, since the uncoded systems used to compute the coding gain are different.)

The notion of shaping gain was investigated in [40–43], and its application to modem design was presented in [44]. A more thorough discussion of the considerations that went into the design of the V.34 modem can be found in [45, 46].

The performance of Ungerboeck's codes quickly dispelled the belief that power reduction is attainable only with a corresponding decrease in bandwidth efficiency, as is the case when we limit our choice of modulation to BPSK. This was a welcome result for modem designers, who had been frustrated in their attempts to go beyond data rates of 9600 bps. Since the International Telecommunications Union's ITU-T

V.29 modem standard was adopted in 1976 little progress was made in increasing the speed and quality of data transmission over voice-grade telephone lines until the appearance of the V.32 and V.33 standards in 1986 (see Example 18.14). The V.29 standard used uncoded 16-QAM and a 2400 symbols/second signaling rate to achieve a spectral efficiency of $\eta = 4.0$ bits/symbol and a transmission speed of 9600 bps in a half-duplex (one-way) mode. Owing to the bandwidth constraints of the channel, signaling rates higher than 2400 symbols/second were not considered feasible. Thus, the only avenue to increased data rates was to expand the size of the signal constellation; however, because of the SNR constraints of the channel, this meant that signals had to be packed closer together, resulting in degraded performance. Thus, a clear need developed for a scheme that could allow constellation expansion at the same signaling rate, thus achieving higher data rates, and yet provide a coding gain to at least recover the noise margin lost by the closer packing of signals. TCM proved to be just such a scheme and, combined with some sophisticated signal-processing techniques, has resulted in a series of improvements that have pushed modem speeds to 56 Kbps.

PROBLEMS

18.1 Prove equation (18.8).

18.2 Find, as functions of the parameter d, the AEWEs $\Delta_{\mathbf{e}}^2(X)$ and the MEWEs $\delta_{\mathbf{e}}^2(X)$ for the two signal set mappings shown in Figure P-18.2, and determine if they are uniform. Assume each constellation has unit average energy.

18.3 Determine if an isometry exists between the subsets $Q(0)$ and $Q(1)$ for the two signal set mappings in Problem 18.2.

18.4 Use Lemma 18.1 to prove that for uniform mappings, $A_{av}(X)$ can be computed by labeling the error trellis with the AEWEs and finding the transfer function of the modified state diagram.

18.5 Construct a counterexample to show that Lemma 18.1 does not necessarily hold for rate $R = k/(k+2)$ codes. State a rate $R = k/(k+2)$ code lemma, similar to Lemma 18.1, specify the conditions for uniformity, and prove the lemma.

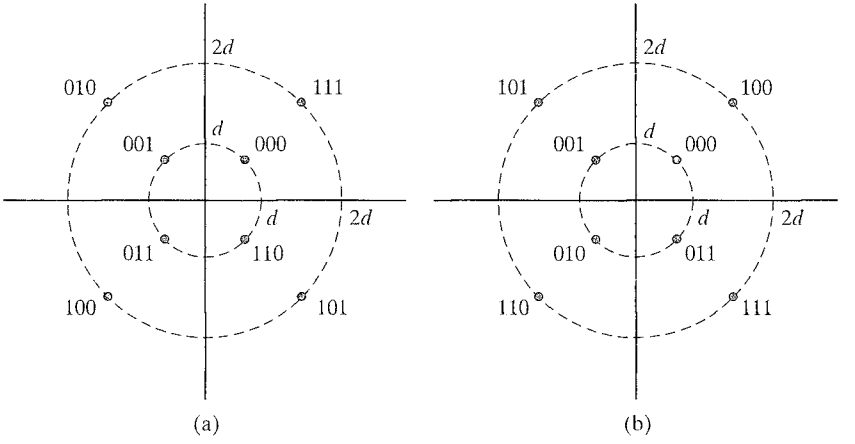


FIGURE P-18.2

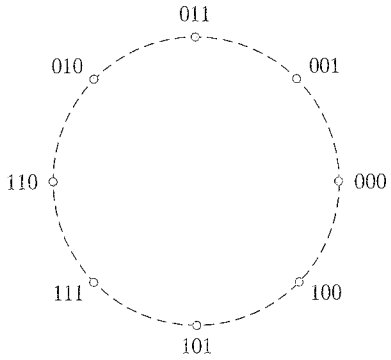


FIGURE P-18.8

18.6 Determine the AEWEs $\Delta_e^2(X)$ and the MEWEs $\delta_e^2(X)$ for Gray- and naturally mapped 4-AM and show that they are both uniform mappings.

18.7 Consider mapping a rate $R = 2/3$ convolutional code into 8-AM using natural mapping.

a. Determine the AEWEs $\Delta_e^2(X)$ and the MEWEs $\delta_e^2(X)$ for this mapping.

b. Determine if the mapping is uniform.

c. Find the coding gain (or loss) γ for the three 4-state, rate $R = 2/3$ convolutional codes of Example 18.4 compared with uncoded QPSK.

d. Can you find a 4-state, rate $R = 2/3$ convolutional code with a better coding gain when used with naturally mapped 8-AM?

18.8 Show that the Gray mapping of the 8-PSK signal set shown in Figure P-18.8 is not uniform.

18.9 Repeat Example 18.4, finding the MFSE distances and asymptotic coding gains for three rate $R = 2/3$ trellis-coded 8-PSK systems, if natural mapping is replaced by the uniform mapping of Figure 18.2(a). Compare the results with natural mapping.

18.10 Repeat Example 18.5 by finding a counterexample to the rate $R = k/(k+1)$ code lemma for the nonuniform signal set mapping in Problem 18.2(a).

18.11 Repeat Example 18.4, finding the MFSE distances and asymptotic coding gains for three rate $R = 2/3$ trellis-coded 8-PSK systems, if natural mapping is replaced by the nonuniform Gray-mapped 8-PSK signal set in Problem 18.8. (In this case, since the rate $R = k/(k+1)$ code lemma is not satisfied, the distances between all possible path pairs must be considered.) Compare the results with natural mapping.

18.12 Show that set partitioning of the infinite two-dimensional integer lattice $\mathbb{Z}^2$ results in a regular mapping.

18.13 Apply mapping by set partitioning to the 32-CROSS signal constellation and determine the error vectors $\mathbf{e}$ for which (18.26) is not satisfied with equality.

18.14 Construct an example in which (18.25) and (18.27) do not give the same result.

18.15 Apply mapping by set partitioning to the 8-AM signal constellation and determine the MSSDs Δ_i^2, $i = 0, 1, 2$. Find the asymptotic coding gain γ and the average number of nearest neighbors $A_{d_{free}}$ when the 4-state code of Table 18.6(a) is applied to 8-AM. Repeat for the one-dimensional integer lattice $\mathbb{Z}^1$.

18.16 Compute, as functions of the parameter d, the asymptotic coding gains of the 16-QAM codes in Table 18.6(b) compared with the following uncoded constellations:

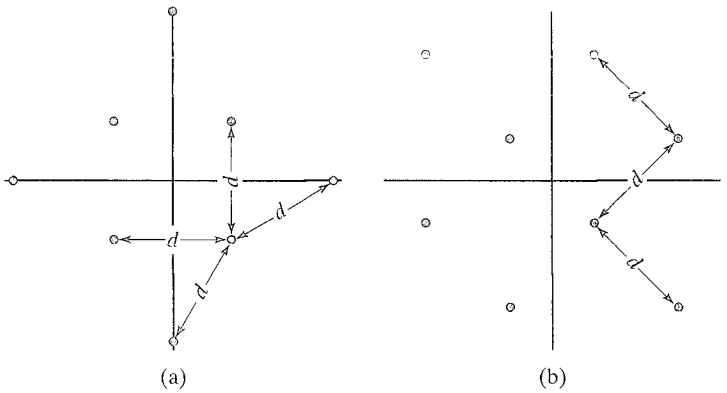


FIGURE P-18.16

(i) the 8-CROSS constellation shown in Figure P-18.16(a) and (ii) the 8-QAM constellation shown in Figure P-18.16(b). Assume each constellation has unit average energy.

18.17 Calculate $A_{av}^{t}(W, X)$ for Example 18.9.

18.18 Apply mapping by set partitioning to the 8-QAM signal constellation shown in Problem 18.16 and determine the MSSDs $\Delta_i^2, i = 0, 1, 2$. Find $A_{av}^{t}(W, X)$ and $A_{av}^{p}(W, X)$ for the code of Example 18.9 using this constellation.

18.19 Let $\mathbb{1}(D) = 1 + D + D^2 + D^3 + \cdots$ in Example 18.10 and recalculate (18.37) and (18.40). Are the conditions for rotational invariance affected?

18.20 Derive general conditions on the number of terms in $\mathbb{h}^{(1)}(D)$ and $\mathbb{h}^{(0)}(D)$ to satisfy (18.46).

18.21 Show that (18.52) is still satisfied when the rotated binary sequences for naturally mapped QPSK given in (18.38) are substituted into the equation, and $\mathbb{h}^{(0)}(D)$ has an odd number of nonzero terms.

18.22 Verify that (18.52) is satisfied for the encoder of (18.53) when the rotated binary sequences for naturally mapped QPSK given in (18.38) are substituted into the equation.

18.23 Find minimal encoder realizations for the 90° rotationally invariant $\nu = 4$ and $\nu = 5$ nonlinear rate $R = 1/2$ codes based on the parity-check matrices

$$\mathbb{H}(D) = [(D^3 + D)/(D^4 + D + 1) \quad 1]$$

and

$$\mathbb{H}(D) = [(D^4 + D)/(D^5 + D^2 + 1) \quad 1],$$

respectively. Show that the $\nu = 5$ case cannot be realized with 32 states.

18.24 Derive general conditions on the number of nonzero terms in $\mathbb{h}^{(2)}(D)$, $\mathbb{h}^{(1)}(D)$, and $\mathbb{h}^{(0)}(D)$ to satisfy (18.59).

18.25 Show that the 45° rotated binary code sequences for naturally mapped 8-PSK are given by $\mathbb{v}_r^{(2)}(D) = \mathbb{v}^{(2)}(D) \oplus \mathbb{v}^{(1)}(D) \circ \mathbb{v}^{(0)}(D)$, $\mathbb{v}_r^{(1)}(D) = \mathbb{v}^{(1)}(D) \oplus \mathbb{v}^{(0)}(D)$, and $\mathbb{v}_r^{(0)}(D) = \mathbb{v}^{(0)}(D) \oplus \mathbb{1}(D)$.

18.26 Show that (18.61) is still satisfied when the rotated binary sequences for naturally mapped 8-PSK given in Problem 18.25 are substituted into the equation, and $\mathbb{h}^{(0)}(D)$ has an odd number of nonzero terms.

18.27 Show how $f(D)$ in (18.65b) can be rewritten to correspond to the encoder realization shown in Figure 18.25.

18.28 Use the method of Euclidean weights to show that $d^2_{free} = 3.515$ for 16-state, rate $R = 2/3$, trellis-coded 8-PSK with $\mathbb{h}^{(2)} = (1\ 6)$, $\mathbb{h}^{(1)} = (1\ 2)$, and $\mathbb{h}^{(0)} = (2\ 3)$.

18.29 Prove (18.88); that is, the partition level p equals the sum of the redundancies of the I linear block codes that define the subcode $\Lambda_p(0)$.

18.30 Draw the appropriate signal set mappers, similar to Figure 18.30, for the three 3 × 8-PSK partitions of Example 18.17.

18.31 Draw the complete encoder diagram for the 8-state, 2×16-PSK, $\eta = 3.5$ bits/symbol encoder listed in Table 18.15(b), including differential encoding of appropriate input bits. Use (18.93) to express the first three 2×16-PSK encoder output signals in both binary and integer form, assuming the input sequence $\mathbb{u} = (\mathbb{u}_0, \mathbb{u}_1, \mathbb{u}_2, \cdots) = (1011001, 0001110, 1101110, \cdots)$, and the encoder starts in the all-zero state.

18.32 Use the approach of Example 18.19 to determine the rotational invariance of the two 16-state, 3×8-PSK, $\eta = 2.33$ bits/symbol encoders listed in Table 18.15(a).

18.33 Draw the augmented, modified state diagram and find the AIOWEFs in Example 18.20. Include sketches of $P_b(E)$ versus E_s/N_0 and uncoded QPSK, and estimate the real coding gain at a BER of 10^{-5}.

18.34 Repeat Example 18.20 for (1) a 2×16-QAM system with $\eta = 3.5$ bits/symbol and (2) trellis-coded $\mathbb{Z}^4$, using the 8-state code listed in Table 18.15(c) with $q = 0$.

18.35 Assuming a distance of d between neighboring signal points, calculate the average energies of the 192-point signal set in Figure 18.34 and of its shaped 4-D version as functions of d, and compute the shaping gain.

18.36 Assuming a distance of d between neighboring signal points, compute the CERs (compared with 2-AM) and the PARs of the 1-D signal sets 2-AM, 4-AM, and 8-AM as functions of d.

18.37 Draw the encoder corresponding to the rate $R = 2/3$ parity-check matrix of (18.112), and show that it is equivalent to the encoder in Figure 18.35.

BIBLIOGRAPHY

1. G. Ungerboeck, "Channel Coding with Multilevel/Phase Signals," *IEEE Trans. Inform. Theory*, IT-28: 55–67, January 1982.

2. G. Ungerboeck and I. Csajka, "On Improving Data-Link Performance by Increasing the Channel Alphabet and Introducing Sequence Coding," in *IEEE Int. Symp. Inform. Theory (ISIT 1976) Book of Abstracts*, p. 53, Ronneby, Sweden, June 1976.

3. G. Ungerboeck, "Trellis-Coded Modulation with Redundant Signal Sets. Part I: Introduction," *IEEE Commun. Mag.*, 25: 5–11, February 1987.

4. _____, "Trellis-Coded Modulation with Redundant Signal Sets. Part II: State of the Art," *IEEE Commun. Mag.*, 25: 12–21, February 1987.

5. J. L. Massey, "Coding and Modulation in Digital Communications," in *Proc. 1974 Zurich Seminar on Digital Communications*, pp. E2(1)–E2(4), Zurich, Switzerland, March 1974.

6. J. B. Anderson and D. P. Taylor, "A Bandwidth-Efficient Class of Signal Space Codes," *IEEE Trans. Inform. Theory*, IT-24: 703–12, November 1978.

7. G. D. Forney, Jr., R. G. Gallager, G. R. Lang, F. M. Longstaff, and S. U. Qureshi, "Efficient Modulation for Band-Limited Channels," *IEEE J. Select. Areas Commun.*, SAC-2: 632–47, September 1984.

8. A. R. Calderbank and J. E. Mazo, "A New Description of Trellis Codes," *IEEE Trans. Inform. Theory*, IT-30: 784–91, November 1984.

9. A. R. Calderbank and N. J. A. Sloane, "New Trellis Codes Based on Lattices and Cosets," *IEEE Trans. Inform. Theory*, IT-33: 177–95, March 1987.

10. G. D. Forney, Jr., "Coset Codes—Part I: Introduction and Geometrical Classification," *IEEE Trans. Inform. Theory*, IT-34: 1123–51, September 1988.

11. _____, "Coset Codes—Part II: Binary Lattices and Related Codes," *IEEE Trans. Inform. Theory*, IT-34: 1152–86, September 1988.

12. D. Divsalar and M. K. Simon, "The Design of Trellis Coded MPSK for Fading Channels: Performance Criteria," *IEEE Trans. Commun.*, COM-36: 1004–12, September 1988.

13. _____, "The Design of Trellis Coded MPSK for Fading Channels: Set Partitioning for Optimum Code Design," *IEEE Trans. Commun.*, COM-36: 1013–21, September 1988.

14. _____, "Multiple Trellis Coded Modulation (MTCM)," *IEEE Trans. Commun.*, COM-36: 410–19, April 1988.

15. C. Schlegel and D. J. Costello, Jr., "Bandwidth Efficient Coding for Fading Channels," *IEEE J. Select. Areas Commun.*, JSAC-7: 1356–68, December 1989.

16. R. H. Deng and D. J. Costello, Jr., "High Rate Concatenated Coding Systems Using Bandwidth Efficient Trellis Inner Codes," *IEEE Trans. Commun.*, COM-37: 420–27, May 1989.

17. _____, "High Rate Concatenated Coding Systems Using Multi-Dimensional Bandwidth Efficient Trellis Inner Codes," *IEEE Trans. Commun.*, COM-37: 1091–96, October 1989.

18. S. L. Goff, A. Glavieux, and C. Berrou, "Turbo-Codes and High Spectral Efficiency Modulation," in *Proc. IEEE Int. Conf. Commun. (ICC 1994)*, pp. 645–49, New Orleans, La., May 1994.

19. S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Bandwidth Efficient Parallel Concatenated Coding Schemes," *IEE Electron. Lett.*, 31: 2067–69, November 1995.

20. P. Robertson and T. Wörz, "Coded Modulation Scheme Employing Turbo Codes," *IEE Electron. Lett.*, 31: 1546–47, August 1995.

21. E. Biglieri, D. Divsalar, P. J. McLane, and M. K. Simon, *Introduction to Trellis-Coded Modulation with Applications*, MacMillan, New York, 1991.

22. G. D. Forney, Jr. and G. Ungerboeck, "Modulation and Coding for Linear Gaussian Channels," *IEEE Trans. Inform. Theory*, IT-44: 2384–2415, October 1998.

23. G. D. Forney, Jr., "Geometrically Uniform Codes," *IEEE Trans. Inform. Theory*, IT-37: 1241–60, September 1991.

24. E. Zehavi and J. K. Wolf, "On the Performance Evaluation of Trellis Codes," *IEEE Trans. Inform. Theory*, IT-32: 196–202, March 1987.

25. M. Rouanne and D. J. Costello, Jr., "An Algorithm for Computing the Distance Spectrum of Trellis Codes," *IEEE J. Select. Areas Commun.*, JSAC-7: 929–40, August 1989.

26. S. S. Pietrobon, G. Ungerboeck, L. C. Perez, and D. J. Costello, Jr., "Rotationally Invariant Nonlinear Trellis Codes for Two-Dimensional Modulation," *IEEE Trans. Inform. Theory*, IT-40: 1773–91, November 1994.

27. L. F. Wei, "Rotationally Invariant Convolutional Channel Coding with Expanded Signal Space. Part I: 180°," *IEEE J. Select. Areas Commun.*, SAC-2: 659–71, September 1984.

28. _____, "Rotationally Invariant Convolutional Channel Coding with Expanded Signal Space. Part II: Nonlinear Codes," *IEEE J. Select. Areas Commun.*, SAC-2: 672–86, September 1984.

29. _____, "Trellis-Coded Modulation with Multidimensional Constellations," *IEEE Trans. Inform. Theory*, IT-33: 483–501, July 1987.

30. H. Imai and S. Hirakawa, "A New Multilevel Coding Method Using Error-Correcting Codes," *IEEE Trans. Inform. Theory*, IT-23: 371–77, May 1977.

31. G. J. Pottie and D. P. Taylor, "Multilevel Codes Based on Partitioning," *IEEE Trans. Inform. Theory*, IT-35: 87–98, January 1989.

32. V. V. Ginzburg, "Multidimensional Signals for a Continuous Channel," in *Probl. Inform. Transm.*, 23: 20–34, January 1984.

33. S. S. Pietrobon, R. H. Deng, A. LaFanechere, G. Ungerboeck, and D. J. Costello, Jr., "Trellis Coded Multi-Dimensional Phase Modulation," *IEEE Trans. Inform. Theory*, IT-36: 63–89, January 1990.

34. S. S. Pietrobon and D. J. Costello, Jr., "Trellis Coding with Multidimensional QAM Signal Sets," *IEEE Trans. Inform. Theory*, IT-39: 325–36, March 1993.

35. L. F. Wei, "Rotationally Invariant Trellis-Coded Modulations with Multidimensional M-PSK," *IEEE J. Select. Areas Commun.*, SAC-7: 1281–95, December 1989.

36. S. Benedetto, R. Garello, M. Mondin, and G. Montorsi, "Geometrically Uniform Partitions of L × MPSK Constellations and Related Binary Trellis Codes," *IEEE Trans. Inform. Theory*, IT-39: 1773–98, November 1993.

37. M. D. Trott, S. Benedetto, R. Garello, and M. Mondin, "Rotational Invariance of Trellis Codes. Part I: Encoders and Precoders," *IEEE Trans. Inform. Theory*, IT-42: 751–65, May 1996.

38. S. Benedetto, R. Garello, M. Mondin, and M. D. Trott, "Rotational Invariance of Trellis Codes. Part II: Group Codes and Decoders," *IEEE Trans. Inform. Theory*, IT-42: 766–78, May 1996.

39. S. Benedetto, R. Garello, M. Mondin, and G. Montorsi, "Geometrically Uniform TCM Codes over Groups Based on L × MPSK Constellations," *IEEE Trans. Inform. Theory*, IT-40: 137–52, January 1994.

40. G. D. Forney, Jr., and L. F. Wei, "Multidimensional Constellations. Part I: Introduction, Figure of Merit and Generalized Cross Constellations," *IEEE J. Select. Areas Commun.*, SAC-7: 877–91, August 1989.

41. G. D. Forney, Jr., "Multidimensional Constellations. Part II: Voronoi Constellations," *IEEE J. Select. Areas Commun.*, SAC-7: 941–58, August 1989.

42. A. R. Calderbank and L. Ozarow, "Nonequiprobable Signaling on the Gaussian Channel," *IEEE Trans. Inform. Theory*, IT-36: 726–40, July 1990.

43. G. D. Forney, Jr., "Trellis Shaping," *IEEE Trans. Inform. Theory*, IT-38: 281–300, March 1992.

44. R. Laroia, N. Farvardin, and S. A. Tretter, "On Optimal Shaping of Multidimensional Constellations," *IEEE Trans. Inform. Theory*, IT-40: 1044–56, July 1994.

45. M. V. Eyuboglu, G. D. Forney, Jr., P. Dong, and G. Long, "Advanced Modulation Techniques for V. Fast," *Eur. Trans. Telecommun.*, 4: 243–56, May 1993.

46. G. D. Forney, Jr., L. Brown, M. V. Eyuboglu, and J. L. Moran, III, "The V.34 High-Speed Modem Standard," *IEEE Commun. Mag.*, 34: 28–33, December 1996.