Notes on Coding and TCM

1. Introduction

Trellis Coded Modulation (TCM) is based on convolutional coding, therefore it is appropriate to introduce the basic concepts in coding and make a partial introduction of convolutional coding.

The philosophy of coding is to (artificially) increase the dimensionality of the signal space and leave some signal vector positions unoccupied, so the minimum distance between the adjacent signal vector ends is increased, thus reducing probability of error.

We illustrate this point by Example 9.4.1 of Proakis 2002.

Example 1.1 : Assume that we have 4 PSK whose constellation diagram is as shown in Fig. 1.1 together with (common) length of signal vectors the respective distances between them.

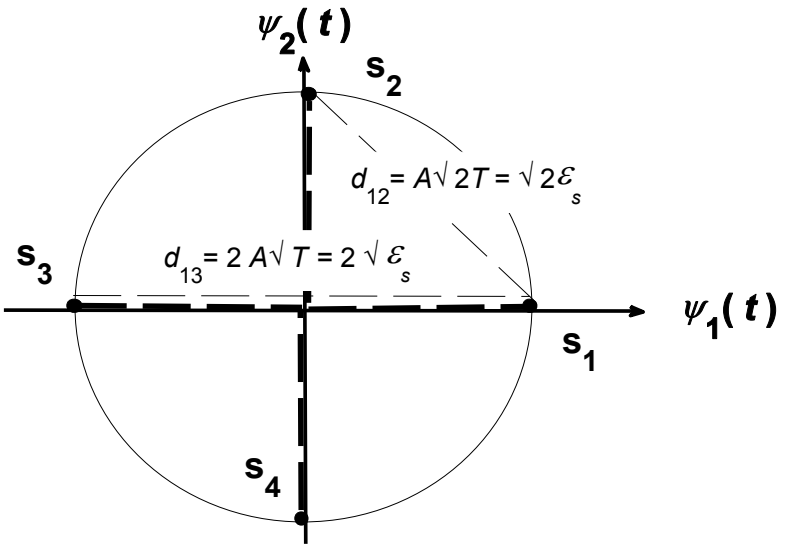

Fig. 1.1 Constellation diagram for 4 PSK (two dimensional case).

For clarity, we state below the properties of 4 PSK shown in Fig. 1

$$d_{2\min 4\mathrm{PSK}} = d_{12} = d_{14} = d_{23} = A\sqrt{2T}\ ,\ d_{13} = d_{24} = 2A\sqrt{T}$$
$$|\mathbf{s}_1| = |\mathbf{s}_2| = |\mathbf{s}_3| = |\mathbf{s}_4| = A\sqrt{T}\ \ ,\ \ \varepsilon_1 = \varepsilon_2 = \varepsilon_3 = \varepsilon_4 = \varepsilon_s = A^2T \qquad (1)$$

Suppose now that we wish to design another constellation diagram which has a larger minimum distance between the adjacent signal vector ends than $d_{2\min 4\mathrm{PSK}}$ given in (1.1) (Keeping the signal vector lengths or the total or the average energy the same). One way to do this is to increase the dimensionality of the signal space from two to three. Such a configuration is shown in Fig. 1.2.

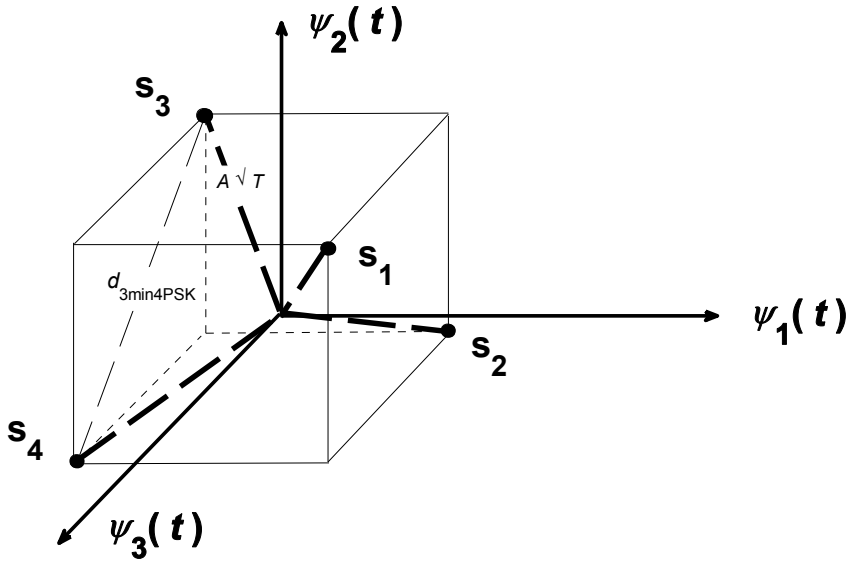

Fig. 1.2 Constellation diagram for 4 PSK (three dimensional case).

In Fig. 1.2, we have placed the signal vectors on the vertices (corners) of a cube whose centre coincides with the origin of the three dimensional signal space. Additionally, it is arranged that the distances between the signal vector ends are maximized by placing our signals diagonally on cube sides and leaving the other diagonals unoccupied. To see the amount of increased separation we have gained between signal vector ends by going from two to three dimensions, we write for the signal vector components of the three dimensional case as follows

$$\begin{aligned}
\mathbf{s}_1 &= \left[+\sqrt{\varepsilon_c},\ +\sqrt{\varepsilon_c},\ +\sqrt{\varepsilon_c}\right] = \sqrt{\varepsilon_c}\left[+1,\ +1,\ +1\right] \\
\mathbf{s}_2 &= \left[+\sqrt{\varepsilon_c},\ -\sqrt{\varepsilon_c},\ -\sqrt{\varepsilon_c}\right] = \sqrt{\varepsilon_c}\left[+1,\ -1,\ -1\right] \\
\mathbf{s}_3 &= \left[-\sqrt{\varepsilon_c},\ -\sqrt{\varepsilon_c},\ +\sqrt{\varepsilon_c}\right] = \sqrt{\varepsilon_c}\left[-1,\ -1,\ +1\right] \\
\mathbf{s}_4 &= \left[-\sqrt{\varepsilon_c},\ +\sqrt{\varepsilon_c},\ -\sqrt{\varepsilon_c}\right] = \sqrt{\varepsilon_c}\left[-1,\ +1,\ -1\right]
\end{aligned} \tag{1.2}$$

It is easy to find what ε_c should be in terms of $\varepsilon_s = A^2T$ so that there is energy equivalence between the constellation of Figs. 1.1 and 1.2. Hence

$$\varepsilon_1 = \|\mathbf{s}_1\|^2 = 3\varepsilon_c = A^2T,\ \ \varepsilon_c = \frac{A^2T}{3} \tag{1.3}$$

This way, we can find $d_{3\min 4PSK}$ of the constellation in Fig. 1.3 and compare it to $d_{2\min 4PSK}$ of constellation of Fig. 1.2 as

$$d^2_{3\min 4\text{PSK}} = 8\varepsilon_c = \frac{8\varepsilon_s}{3} = \frac{8A^2T}{3}$$

$$d_{3\min 4\text{PSK}} = 2A\sqrt{\frac{2T}{3}} = 1.633A\sqrt{T},\ d_{2\min 4\text{PSK}} = A\sqrt{2T} = 1.4142A\sqrt{T}$$

$$\frac{d_{3\min 4\text{PSK}}}{d_{2\min 4\text{PSK}}} = 1.1547 \tag{1.4}$$

As seen from (1.4), we have an increase by a factor of 1.1547 in the minimum distance, upon going from two dimensions to three dimensions. It is interesting to note that, in the present arrangement, four vertices of the cube in Fig. 1.2 remain unoccupied. This is exactly what coding does, that is, coding increases the dimensionality of the signal space, leaving the number of symbols in our alphabet the same, i.e. Figs. 1.1 and 1.2 both refer to the case of 4 PSK. Another interpretation is that Fig. 1.2 represents a signal space of eight symbols, but only four symbols ("codewords" in the terminology of coding) out of the eight are valid symbols, the remaining four symbols are invalid symbols of the message signal. We describe coding in terms of binary waveforms (bits) rather than symbols. Hence in the present example, when going from 4 PSK to 8 symbols configuration, we have extended the existing two bit length of the message signal to three bits. In general we represent the length of the uncoded message signal as k bits, while the coded message will have a length of n bits, where it is always the case that $n > k$.

If we were to fill the unoccupied signal positions in Fig. 1.2, we would obtain something similar to 8 PSK, but in three dimensions. Using Fig. 3.5 of the Notes in Dimensionality of Signals_Sept 2012 document and Fig. 1.2, it is possible to estimate the minimum distances in the 8 PSK cases of two and three dimensions as follows

$$d_{3\min 8\text{PSK}} = 2A\sqrt{\frac{T}{3}} = 1.1547A\sqrt{T},\ d_{2\min 8\text{PSK}} = A\sqrt{T\left(2-\sqrt{2}\right)} = 0.7654A\sqrt{T}$$

$$\frac{d_{3\min 8\text{PSK}}}{d_{2\min 8\text{PSK}}} = 1.5087 \tag{1.5}$$

Thus increasing the dimensionality always increases the minimum distance, but we should keep in mind that with increasing number of dimensions, our bandwidth requirements and receiver complexity rise as well.

There exist some rather simplified coding schemes such as linear and cyclic codes. Here we are interested in codes that show dependence on the present as well past symbols, thus have memory. One such scheme is convolutional coding which we study next.

2. Convolutional Codes

The general block diagram of a convolutional encoder is shown in Fig. 2.1 (pasted from Fig. 9.24 of Proakis 2002).

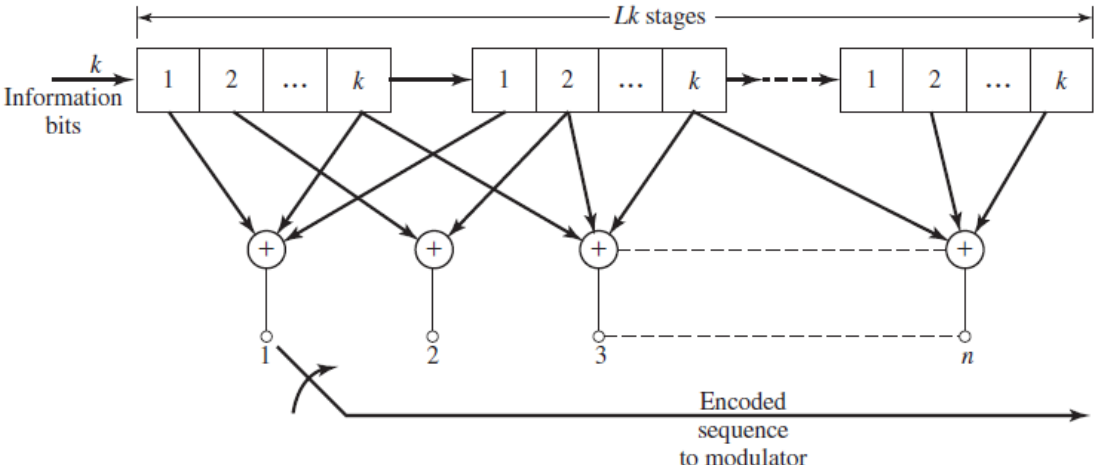

Fig. 2.1 General block diagram of a convolutional encoder.
According to Fig. 2.1, we take as message sequence (binary) information bits of length k at a time and store those in L stages of shift registers. Then from the adders that are connected to these shift registers, we read an output of length n, where k bits are due to the message, thus the extra $n-k$ bits appear because of the coding.

Generally for traceable results, we prefer much reduced configurations than the one depicted in Fig. 2.1. Such an example, based on Example 9.7.1 of Proakis 2002, is given below.

Example 2.1 : For the convolutional encoder shown below, find the output, if the input message sequence is $\mathbf{x} = [1,\ 1,\ 0,\ 1,\ 0,\ 1,\ 1]$

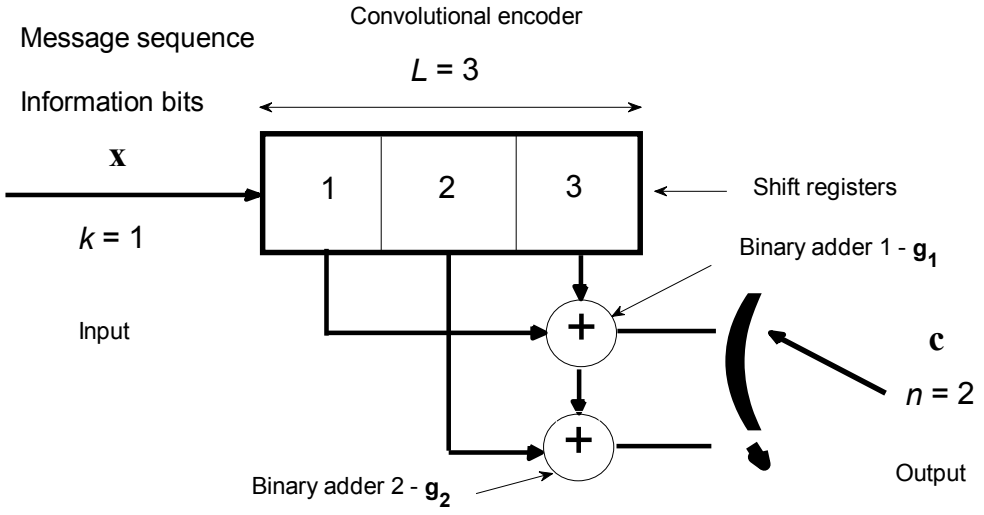

Fig. 2.1 Convulutional encoder for Example 2.1.

Solution for Example 2.1 : We can find the output in three ways,

a) By noting the contents of shift registers at successive loadings from the input side and performing the act of adding at the binary adders by observing the binary addition rules and finally scanning the output of the adders in the shown direction (downwards). Before the first

binary entry (which is the leftmost binary value in $\mathbf{x}$), we assume shift registers 1 and 2 are loaded with zeros. In fact, to create the same conditions for the next entry, we append two zeros to the end of the existing input message such that our actual input becomes $\mathbf{x}=[1,\ 1,\ 0, 1,\ 0, 1,\ 1, 0, 0]$. Then by manual tracing we find the output from the convolutional encoder of Fig. 2.1 is $\mathbf{c}=[1,\ 1,\ 1, 0, 1,\ 0, 0, 0, 0, 1,\ 0, 0, 1, 0, 1, 0, 1, 1]$. Comparing $\mathbf{c}$ with $\mathbf{x}$, we see no resemblance between the two. This means, for a decoder that has no knowledge of the encoder drawn in Fig. 2.1, it is almost impossible to go back from $\mathbf{c}$ to $\mathbf{x}$.
b) The output of the convolutional encoder can also b found by interlacing method. For this, we write the polynomial representation of the adders connections to the shift registers as follows

$$\begin{aligned}\mathbf{g}_1&=[1,\ 0, 1], \qquad \mathbf{g}_2=[1,\ 1, 1]\\ \mathbf{g}_1(p)&=p^2+1,\ \mathbf{g}_2(p)=p^2+p+1\end{aligned} \tag{2.1}$$

As apparent from (2.1), for the $\mathbf{g}$ representation, if there is connection of the adder to the shift register, we write one, otherwise we write zero. Then we convert this representation to $\mathbf{g}(p)$ polynomial expressions as shown on the second line of (2.1). Using the same convention, we write the input message sequence (without the added zeros at the end) as a polynomial expression as follows

$$\begin{aligned}&\quad 1\times p^6, 1\times p^5,\ 0\times p^4,\ 1\times p^3,\ 0\times p^2,\ 1\times p^1,\ 1\times p^0\\ \mathbf{x}=&\ [1, \quad 1, \quad 0, \quad 1, \quad 0, \quad 1, \quad 1]\\ \mathbf{x}(p)&=p^6+p^5+p^3+p+1\end{aligned} \tag{2.2}$$

Now by observing binary addition rules, we proceed as follows

$$\begin{aligned}\mathbf{x}(p)\mathbf{g}_1(p)&=(p^6+p^5+p^3+p+1)(p^2+1)=p^8+p^7+p^6+p^2+p+1\\ \mathbf{x}(p)\mathbf{g}_2(p)&=(p^6+p^5+p^3+p+1)(p^2+p+1)=p^8+p^4+1\end{aligned} \tag{2.3}$$

Then the output of the encoder will be obtained by the interlacing $\mathbf{x}(p)\mathbf{g}_1(p)$ with $\mathbf{x}(p)\mathbf{g}_2(p)$ as shown below

$$\begin{aligned}&p^8 \text{ of } \mathbf{x}(p)\mathbf{g}_1(p) \quad \cdots \quad p^4 \text{ of } \mathbf{x}(p)\mathbf{g}_1(p) \quad \cdots\\ &\quad \downarrow \qquad\qquad\qquad \downarrow\\ &\mathbf{c}=[1, \quad 1,\ 1, 0, 1,\ 0, 0, 0, 0, \quad 1,\ 0, 0, 1, 0, 1, 0, 1, 1]\\ &\qquad\ \uparrow \qquad\qquad\qquad \uparrow\\ &p^8 \text{ of } \mathbf{x}(p)\mathbf{g}_2(p) \quad \cdots \quad p^4 \text{ of } \mathbf{x}(p)\mathbf{g}_2(p) \quad \cdots\end{aligned} \tag{2.4}$$

As seen, the output given in (2.4) is in perfect agreement with the one found by hand tracing in a).

c) As a final option, we can find the output of the convolution encoder using Matlab. For this we must introduce the construction of convolutional encoder in Fig. 2.1 to Matlab in the notation of Matlab. This is done by specifying the arguments in the function named

"poly2trellis(ConstraintLength, CodeGenerator)". In the configuration of Fig. 2.1, we have one layer of shift registers, since $k=1$. Then the argument, ConstraintLength will be represented by the number of shift registers in the single layer $L=3$. For the CodeGenerator argument, we benefit from $\mathbf{g}_1$ and $\mathbf{g}_2$ given in (2.1) convert those the binary elements there to octal to obtain,

$$\begin{array}{rcl} & \mathbf{g}_1=[1,\ 0,\ 1] \quad \mathbf{g}_2=[1,\ 1,\ 1] & \\ \text{convert 101 to } 1\times 2^0+0\times 2^1+1\times 2^2=5 \downarrow & & \downarrow \text{ convert 111 to } 1\times 2^0+1\times 2^1+1\times 2^2=7 \\ \text{CodeGenerator}= \quad [5, & & 7] \end{array} \tag{2.5}$$

Thus we can specify the arguments of the function poly2trellis for the encoder configuration of Fig. 2.1 as follows

$$\text{poly2trellis}\left(3,\ [5,\ 7]\right) \tag{2.6}$$

In general, ConstraintLength is $1\times k$ row array, indicating L values for each layer. CodeGenerator is $k\times N_a$ matrix denoting the connections of the adders to the layers of shift registers, where N_a corresponds to the number of adders. Note that in the case $k>1$, in CodeGenerator matrix, the connections of each individual adder each layer of shift registers should be defined one by one. Note further that in Matlab the first shift register is not drawn, but the numeric value of L is defined in the same way as indicated in Fig. 2.1.

Exercise 2.1 : Using the m file Conv12_Exp2.m available on the course webpage, test that the message sequence of $\mathbf{x}=[1,\ 1,\ 0,\ 1,\ 0,\ 1,\ 1,\ 0,\ 0]$ gives an output (codeword) of $\mathbf{c}=[1,\ 1,\ 1,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 1,\ 0,\ 1,\ 0,\ 1,\ 1]$. Try at least five different message sequences, compare the outputs obtained from Matlab and one of the methods above, i.e. by hand tracing or interlacing.

3. Trellis Coded Modulation (TCM)

TCM was first proposed by Ungerboeck in 1981. The basic idea is to get better performance in PSK and QAM via the use of convolutional coding.

Trellis coded modulation can best be described by the following simplest example. Assume that we have a message signal that we can group as binary waveforms (bits) to make 4 PSK. Here instead of converting our message signal to 4 PSK symbols, we take the binary form of the message grouped as two bits and use convolutional encoding to generate three bits per symbol. Thus in coding notation, $k=2,\ n=3$. Such an operation can be performed by the encoder shown below in Fig. 3.1

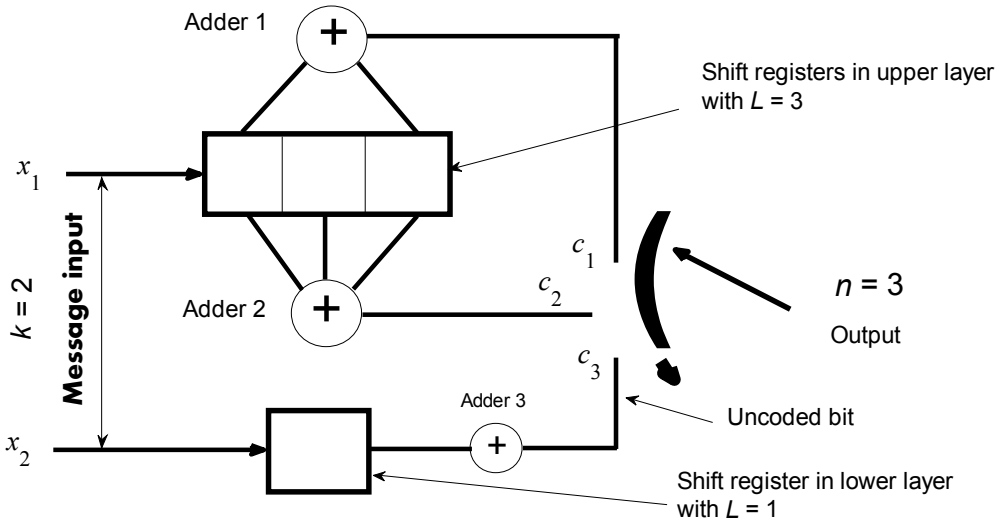

Fig. 3.1 Convolutional encoder for 8 PSK TCM.

Note that in order to achieve a coding rate of $k/n = 2/3$, in Fig. 3.1, we have had to encode only one bit (upper bit) of the message and let the other bit (lower) bit pass uncoded. The encoder configuration of Fig. 3.1 must be introduced to Matlab in its entirety, i.e. including the lower layer that illustrates that no coding is to be performed on the message input bit x_2. From the guidelines given in Example 2.1 c), it is easy to find that the encoder configuration in Fig. 3.1 can be defined in Matlab notation as

$$\text{poly2trellis(ConstraintLength, CodeGenerator)} = \text{poly2trellis}\left([3,\ 1],\ [5,\ 7,\ 0;0,\ 0,\ 1]\right) \qquad (3.1)$$

Now we come to the placement of the symbols obtained as a result of the coding operation of Fig. 3.1in a constellation. Assuming that PSK is selected for this purpose, then it is easy to guess that 8 PSK would have to be utilized in order to accommodate the three bit grouping (i.e. eight level signalling). The ordering of symbols, i.e. signal vectors, also called mapping, in an (conventional, uncoded) 8 PSK constellation can be in two ways as depicted in Fig. 3.2.

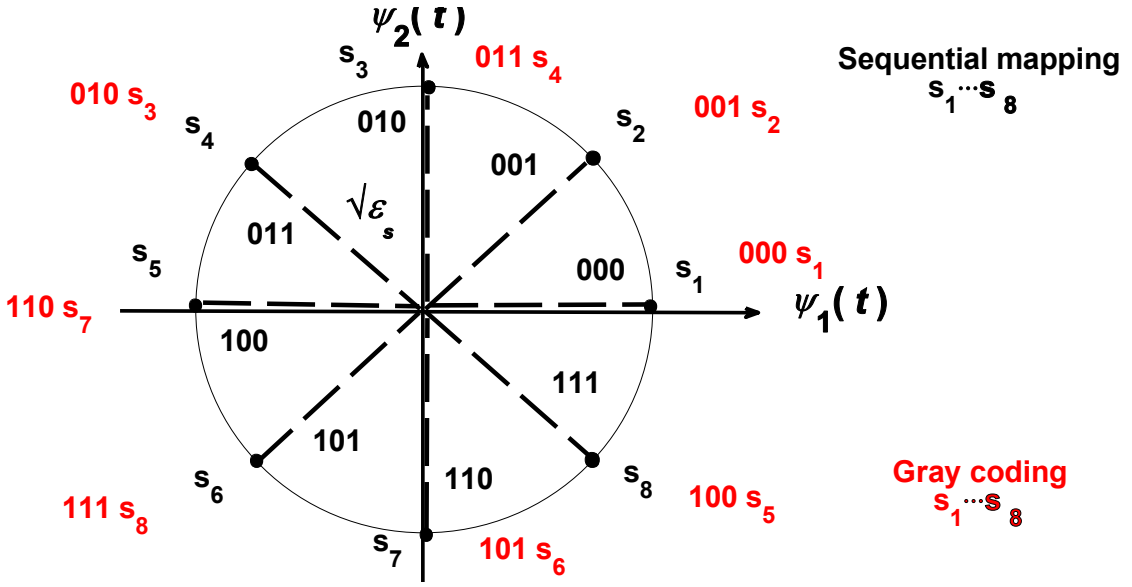

Fig. 3.2 Two ways of mapping the three bits to 8 PSK symbols.

As seen from Fig 3.2, the ordering on the inner side constellation circle is simply the sequential mapping (in ascending numeric order) of symbols from (level) 0 to (level) 7. Taking into account that most likely errors come from one transmitted signal being incorrectly detected as the adjacent signals, it is possible to arrange the mapping of bits to symbols as those placed outside the constellation circle (in red). The idea in the latter mapping is that the adjacent symbols should differ only by one bit, resulting in most errors being bit errors, rather than symbol errors. Such an arrangement is known as Gray coding. Although Gray coding brings some improvement over the simple sequential ordering of symbols around constellation circle, a substantial increase can only be achieved by taking into account the property embedded into TCM via coding.

To arrive at 8 PSK TCM constellation diagram, we must first construct the state trellis diagram for the encoder illustrated in Fig. 3.1. This is done by either hand tracing or using Matlab and the result is shown below in Fig. 3.3.

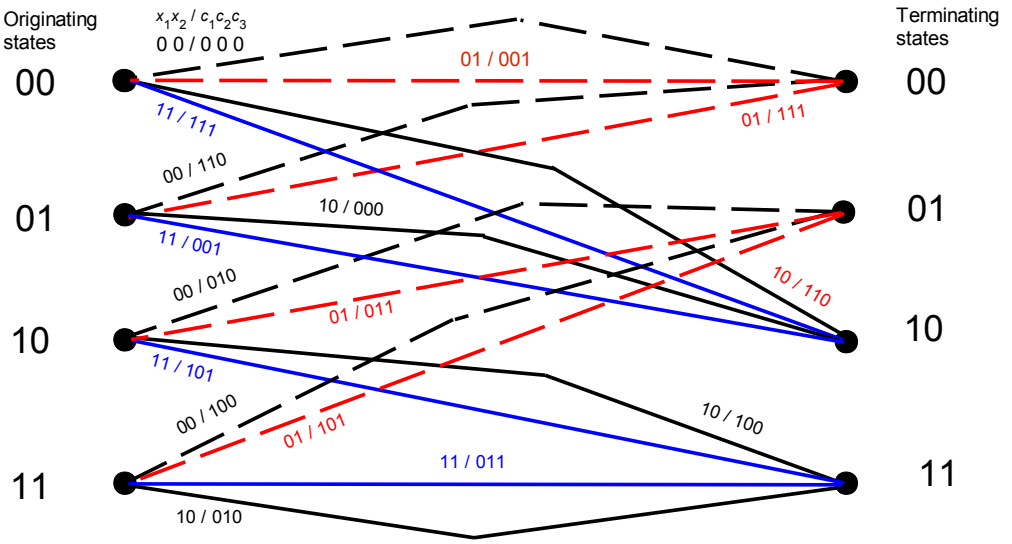

Fig. 3.3 State trellis diagram for the convolutional encoder of Fig. 3.1.

We know that in convolutional coding, the states are determined by the contents of the shift registers which contain bits from the past inputs. In the configuration of Fig. 3.1, this is applicable to the shift registers in the upper layer, since the lower layer does not perform any coding. This way, our state remains the same against the changes in the second bit (i.e. x_2). This phenomena leads to the emergence of parallel paths, consisting of two links. Such parallel paths start from the same originating state and merge into the same terminating state, as demonstrated by Fig. 3.3. On the paths of Fig. 3.3, the related two input bits and three output bits are also indicated with a separator sign of " / ".

Exercise 3. 1 : Verify Fig. 3.3 either by hand tracing in Fig. 3.1 or by modifying the m file Conv12_Exp2.m according to (3.1). If $\mathbf{x} = [0,\ 0,\ 0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 0,\ 1,\ 1,\ 0,\ 1,\ 0,\ 0,\ 0,\ 1,\ 1,\ 1,\ 1,\ 1,\ 0,\ 1,\ 1,\ 0,\ 1,\ 0,\ 1]$, the full state trellis diagram of Fig. 3. 3 is generated.

In order to arrive at TCM constellation, it is instructive to organize the information offered in Fig. 3.3 in the form of two tables, one for the originating states, the other for the terminating states. These are listed below.

Originating state	Output symbol	**s** designation	**B** designation	Parallel paths
00	000	$\mathbf{s}_1$	$\mathbf{B}_1$	Parallel
00	001	$\mathbf{s}_2$	$\mathbf{B}_1$	
00	110	$\mathbf{s}_7$	$\mathbf{B}_1$	Parallel
00	111	$\mathbf{s}_8$	$\mathbf{B}_1$	
01	110	$\mathbf{s}_7$	$\mathbf{B}_1$	Parallel
01	111	$\mathbf{s}_8$	$\mathbf{B}_1$	
01	000	$\mathbf{s}_1$	$\mathbf{B}_1$	Parallel
01	001	$\mathbf{s}_2$	$\mathbf{B}_1$	
10	011	$\mathbf{s}_4$	$\mathbf{B}_2$	Parallel
10	010	$\mathbf{s}_3$	$\mathbf{B}_2$	
10	100	$\mathbf{s}_5$	$\mathbf{B}_2$	Parallel
10	101	$\mathbf{s}_6$	$\mathbf{B}_2$	
11	100	$\mathbf{s}_5$	$\mathbf{B}_2$	Parallel
11	101	$\mathbf{s}_6$	$\mathbf{B}_2$	
11	010	$\mathbf{s}_3$	$\mathbf{B}_2$	Parallel
11	011	$\mathbf{s}_4$	$\mathbf{B}_2$	

Terminating state	Output symbol	**s** designation	B designation	Parallel paths
00	000	$\mathbf{s}_1$	$\mathbf{B}_1$	Parallel
00	001	$\mathbf{s}_2$	$\mathbf{B}_1$	
00	110	$\mathbf{s}_7$	$\mathbf{B}_1$	Parallel
00	111	$\mathbf{s}_8$	$\mathbf{B}_1$	
01	010	$\mathbf{s}_3$	$\mathbf{B}_2$	Parallel
01	011	$\mathbf{s}_4$	$\mathbf{B}_2$	
01	100	$\mathbf{s}_5$	$\mathbf{B}_2$	Parallel
01	101	$\mathbf{s}_6$	$\mathbf{B}_2$	
10	110	$\mathbf{s}_7$	$\mathbf{B}_1$	Parallel
10	111	$\mathbf{s}_8$	$\mathbf{B}_1$	
10	000	$\mathbf{s}_1$	$\mathbf{B}_1$	Parallel

10	001	$\mathbf{s}_2$	$\mathbf{B}_1$	
11	100	$\mathbf{s}_5$	$\mathbf{B}_2$	Parallel
11	101	$\mathbf{s}_6$	$\mathbf{B}_2$	
11	010	$\mathbf{s}_3$	$\mathbf{B}_2$	Parallel
11	011	$\mathbf{s}_4$	$\mathbf{B}_2$	

Table 3.1 Organization of the information in Fig. 3.3 in the form of two tables.

Table 3.1 as illustrates that the output symbols (signal vectors, $\mathbf{s}_1 \cdots \mathbf{s}_8$) falling into the same $\mathbf{B}$ designations are

$$\mathbf{B}_1 = (000,\ 001,\ 110,\ 111) \quad , \quad \mathbf{B}_2 = (010,\ 011,\ 100,\ 101)$$
$$\mathbf{B}_1 = (\ \mathbf{s}_1,\ \mathbf{s}_2,\ \mathbf{s}_7,\ \mathbf{s}_8) \quad , \quad \mathbf{B}_2 = (\ \mathbf{s}_3,\ \mathbf{s}_4,\ \mathbf{s}_5,\ \mathbf{s}_6) \qquad (3.2)$$

It is observed from Table 3.1 (and from Fig. 3.3) that for a given originating or a terminating state, the paths carry the symbols either from $\mathbf{B}_1$ or $\mathbf{B}_2$. We now assign the signal vectors contained in $\mathbf{B}_1$ and $\mathbf{B}_2$ separately to the vertices of the two squares oriented at 45^0 with respect to each other. In this assignment, no mixing of signal vectors between $\mathbf{B}_1$ and $\mathbf{B}_2$ is allowed. Furthermore, in the diagonal positions of vertices of the squares, the output symbols which belong to the parallel paths are placed. Eventually, these two squares will yield the 8 PSK TCM constellation as shown in Fig. 3.4.

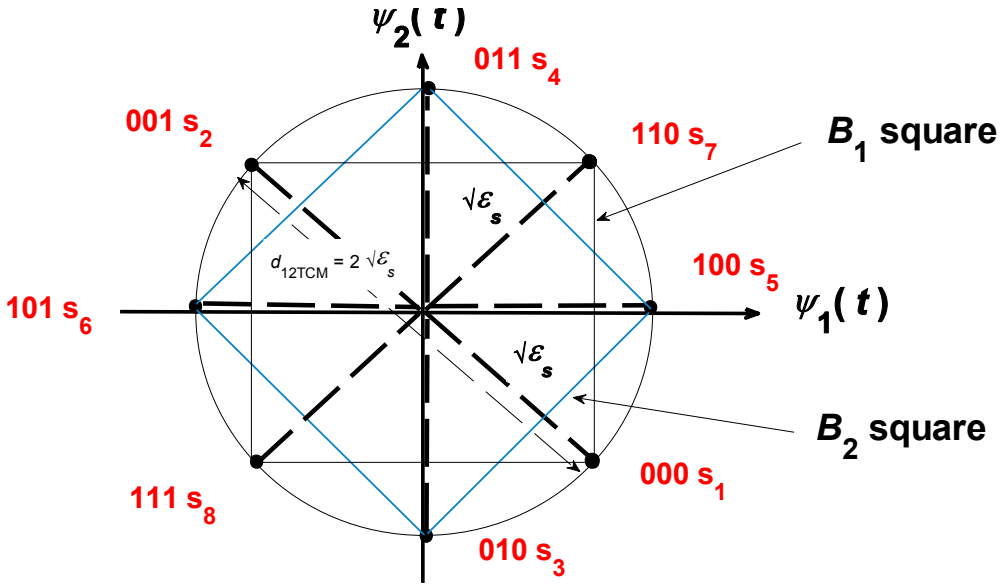

Fig. 3.4 8 PSK TCM constellation

Note that diagonal exchanges between the symbols of the same square (within the same $\mathbf{B}$ designation) has no effect on the performance as will be shown later in our Matlab runs. Note also that Fig. 9.48 on page 652 of Proakis 2002 does not agree with Fig. 3.4. The rules applied when drawing up the constellation of Fig. 3.4 are named as set partitioning rules by Ungerboeck.

Now we are in a position to compare 8 PSK TCM and 4 PSK in terms of probability of error performance. For high signal to noise ratio (SNR) settings, such a comparison can be quite safely be based on the consideration of the increase in the minimum distance by going from 4 PSK to 8 PSK TCM. In TCM, an error will be made by the routes that will divert from the same state and later merge at the same state. By examining Fig. 3.3, we see that the shortest of such routes are parallel paths. From Fig. 3.4, it is found that these parallel paths (all) have the same distance of $d_{12\text{TCM}} = 2\sqrt{\varepsilon_s}$. Thus $d_{12\text{TCM}} = 2\sqrt{\varepsilon_s}$ will be the minimum distance to be encountered between the symbols of 8 PSK TCM constellation. On the equal average energy basis, if we compare minimum distance of 8 PSK TCM to the minimum distance 4 PSK (shown in Fig. 1.1), and also relate this to probability of error formulations (denoted by P), we find that

$$d_{\min 8\text{PSKTCM}} = d_{12\text{TCM}} = 2\sqrt{\varepsilon_s} \quad , \quad d_{\min 4\text{PSK}} = d_{12} = \sqrt{2\varepsilon_s} \quad , \quad \frac{d_{\min 8\text{PSKTCM}}}{d_{\min 4\text{PSK}}} = \sqrt{2}$$

$$P_{8\text{PSKTCM}} \propto Q\left(d_{\min 8\text{PSKTCM}}\right) \quad , \quad P_{4\text{PSK}} \propto Q\left(d_{\min 4\text{PSK}}\right) \quad , \quad P_{8\text{PSKTCM}} < P_{4\text{PSK}} \tag{3.3}$$

The comparison in (3.3) means that under high SNR conditions, a 8 PSK TCM system having the signal energies of $1/\sqrt{2}$ times that of 4 PSK will have equivalent probability of error performance, or 8 PSK TCM will achieve the same probability of error performance as that of 4 PSK with less energy.

Next we examine a more sophisticated case of converting uncoded 8 PSK to 16 QAM TCM. For this, we take the middle encoder of shown Fig. 9 of Ungerboeck's paper [3], which is given below in the notation of these notes and in the configuration to be submitted to Matlab.

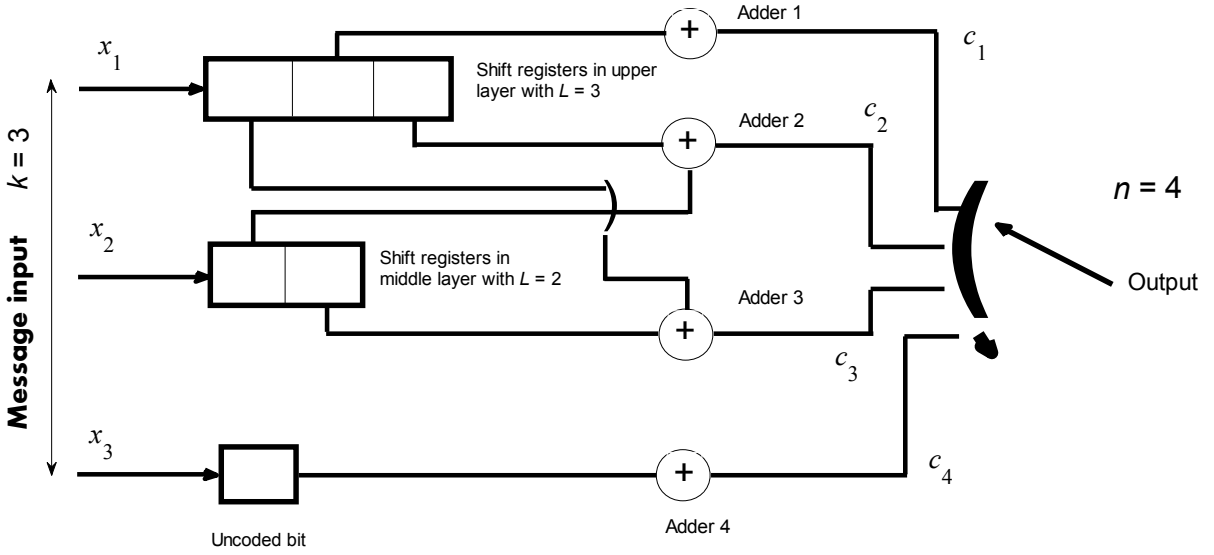

Fig. 3.5 Convolutional encoder for 16 QAM TCM.

By following the steps described above for 8 PSK TCM case, poly2trellis Matlab function for the encoder of Fig. 3.5 can be written as

$$\text{poly2trellis}\left(\left[3,\ 2,\ 1\right],\ \left[2,\ 1,\ 4,\ 0;0,\ 2,\ 1,\ 0;0,\ 0,\ 0,\ 1\right]\right) \tag{3.4}$$

By hand tracing on Fig. 3.5 or by using (3.4) in Matlab, the following state trellis diagram is constructed.

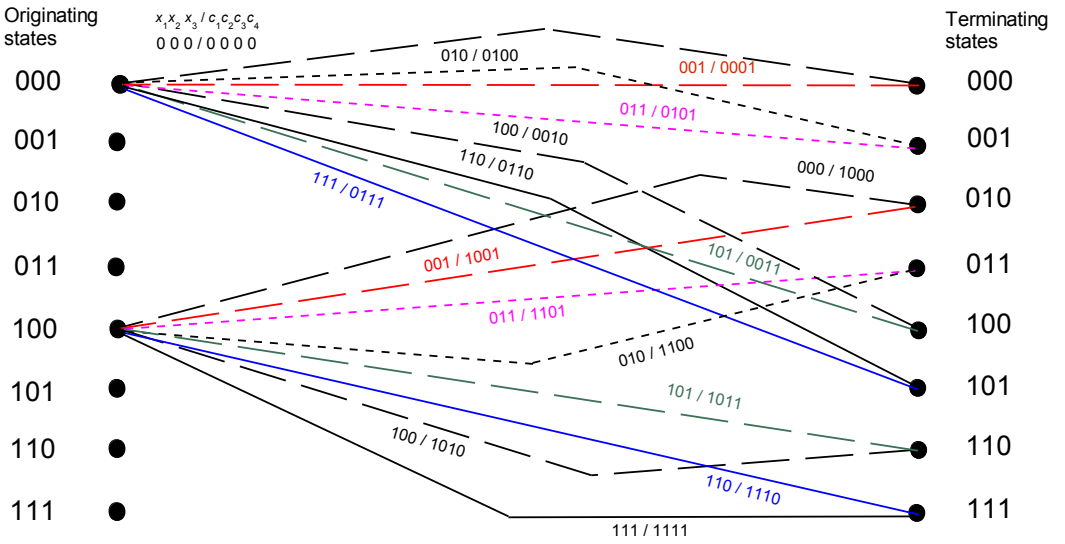

Fig. 3.6 State trellis diagram for the convolutional encoder of Fig. 3.5.

Exercise 3. 2 : Verify the shown paths in Fig. 3.6 either by hand tracing in Fig. 3.5 or by modifying the m file Conv12_Exp2.m according to (3.5).

In Fig. 3.6, due to space limitation, not all details are given. So we construct a table similar to Table 3.1, displaying the outgoing and terminating states (in summarized fashion to save space) separately coupled with **C** and **D** designations.

Originating state	Output symbol	**s** designation	**C** designation	**D** designation	Parallel paths
000	0000	$\mathbf{s}_1$	$\mathbf{C}_1$	$\mathbf{D}_1$	Parallel
000	0001	$\mathbf{s}_2$	$\mathbf{C}_1$	$\mathbf{D}_1$	
000	0100	$\mathbf{s}_5$	$\mathbf{C}_1$	$\mathbf{D}_2$	Parallel
000	0101	$\mathbf{s}_6$	$\mathbf{C}_1$	$\mathbf{D}_2$	
000	0010	$\mathbf{s}_3$	$\mathbf{C}_2$	$\mathbf{D}_3$	Parallel
000	0011	$\mathbf{s}_4$	$\mathbf{C}_2$	$\mathbf{D}_3$	
000	0110	$\mathbf{s}_7$	$\mathbf{C}_2$	$\mathbf{D}_4$	Parallel
000	0111	$\mathbf{s}_8$	$\mathbf{C}_2$	$\mathbf{D}_4$	
001	0010	$\mathbf{s}_3$	$\mathbf{C}_2$	$\mathbf{D}_3$	Parallel
001	0011	$\mathbf{s}_4$	$\mathbf{C}_2$	$\mathbf{D}_3$	
001	0110	$\mathbf{s}_7$	$\mathbf{C}_2$	$\mathbf{D}_4$	Parallel
001	0111	$\mathbf{s}_8$	$\mathbf{C}_2$	$\mathbf{D}_4$	
001	0000	$\mathbf{s}_1$	$\mathbf{C}_1$	$\mathbf{D}_1$	Parallel
001	0001	$\mathbf{s}_2$	$\mathbf{C}_1$	$\mathbf{D}_1$	

001	0100	$\mathbf{s}_5$	$\mathbf{C}_1$	$\mathbf{D}_2$	Parallel
001	0101	$\mathbf{s}_6$	$\mathbf{C}_1$	$\mathbf{D}_2$	
010	0100	$\mathbf{s}_5$	$\mathbf{C}_1$	$\mathbf{D}_2$	Parallel
010	0101	$\mathbf{s}_6$	$\mathbf{C}_1$	$\mathbf{D}_2$	
010	0000	$\mathbf{s}_1$	$\mathbf{C}_1$	$\mathbf{D}_1$	Parallel
010	0001	$\mathbf{s}_2$	$\mathbf{C}_1$	$\mathbf{D}_1$	
010	0110	$\mathbf{s}_7$	$\mathbf{C}_2$	$\mathbf{D}_4$	Parallel
010	0111	$\mathbf{s}_8$	$\mathbf{C}_2$	$\mathbf{D}_4$	
010	0010	$\mathbf{s}_3$	$\mathbf{C}_2$	$\mathbf{D}_3$	Parallel
010	0011	$\mathbf{s}_4$	$\mathbf{C}_2$	$\mathbf{D}_3$	
011	0110	$\mathbf{s}_7$	$\mathbf{C}_2$	$\mathbf{D}_4$	Parallel
011	0111	$\mathbf{s}_8$	$\mathbf{C}_2$	$\mathbf{D}_4$	
011	0010	$\mathbf{s}_3$	$\mathbf{C}_2$	$\mathbf{D}_3$	Parallel
011	0011	$\mathbf{s}_4$	$\mathbf{C}_2$	$\mathbf{D}_3$	
011	0100	$\mathbf{s}_5$	$\mathbf{C}_1$	$\mathbf{D}_2$	Parallel
011	0101	$\mathbf{s}_6$	$\mathbf{C}_1$	$\mathbf{D}_2$	
011	0000	$\mathbf{s}_1$	$\mathbf{C}_1$	$\mathbf{D}_1$	Parallel
011	0001	$\mathbf{s}_2$	$\mathbf{C}_1$	$\mathbf{D}_1$	
100	1000	$\mathbf{s}_9$	$\mathbf{C}_3$	$\mathbf{D}_5$	Parallel
100	1001	$\mathbf{s}_{10}$	$\mathbf{C}_3$	$\mathbf{D}_5$	
100	1100	$\mathbf{s}_{13}$	$\mathbf{C}_3$	$\mathbf{D}_6$	Parallel
100	1101	$\mathbf{s}_{14}$	$\mathbf{C}_3$	$\mathbf{D}_6$	
100	1010	$\mathbf{s}_{11}$	$\mathbf{C}_4$	$\mathbf{D}_7$	Parallel
100	1011	$\mathbf{s}_{12}$	$\mathbf{C}_4$	$\mathbf{D}_7$	
100	1110	$\mathbf{s}_{15}$	$\mathbf{C}_4$	$\mathbf{D}_8$	Parallel
100	1111	$\mathbf{s}_{16}$	$\mathbf{C}_4$	$\mathbf{D}_8$	
101	1010	$\mathbf{s}_{11}$	$\mathbf{C}_4$	$\mathbf{D}_7$	Parallel
101	1011	$\mathbf{s}_{12}$	$\mathbf{C}_4$	$\mathbf{D}_7$	
101	1110	$\mathbf{s}_{15}$	$\mathbf{C}_4$	$\mathbf{D}_8$	Parallel
101	1111	$\mathbf{s}_{16}$	$\mathbf{C}_4$	$\mathbf{D}_8$	
101	1000	$\mathbf{s}_9$	$\mathbf{C}_3$	$\mathbf{D}_5$	Parallel
101	1001	$\mathbf{s}_{10}$	$\mathbf{C}_3$	$\mathbf{D}_5$	
101	1100	$\mathbf{s}_{13}$	$\mathbf{C}_3$	$\mathbf{D}_6$	Parallel
101	1101	$\mathbf{s}_{14}$	$\mathbf{C}_3$	$\mathbf{D}_6$	

110	1100	$\mathbf{s}_{13}$	$\mathbf{C}_3$	$\mathbf{D}_6$	Parallel
110	1101	$\mathbf{s}_{14}$	$\mathbf{C}_3$	$\mathbf{D}_6$	
110	1000	$\mathbf{s}_9$	$\mathbf{C}_3$	$\mathbf{D}_5$	Parallel
110	1001	$\mathbf{s}_{10}$	$\mathbf{C}_3$	$\mathbf{D}_5$	
110	1110	$\mathbf{s}_{15}$	$\mathbf{C}_4$	$\mathbf{D}_8$	Parallel
110	1111	$\mathbf{s}_{16}$	$\mathbf{C}_4$	$\mathbf{D}_8$	
110	1010	$\mathbf{s}_{11}$	$\mathbf{C}_4$	$\mathbf{D}_7$	Parallel
110	1011	$\mathbf{s}_{12}$	$\mathbf{C}_4$	$\mathbf{D}_7$	
111	1110	$\mathbf{s}_{15}$	$\mathbf{C}_4$	$\mathbf{D}_8$	Parallel
111	1111	$\mathbf{s}_{16}$	$\mathbf{C}_4$	$\mathbf{D}_8$	
111	1010	$\mathbf{s}_{11}$	$\mathbf{C}_4$	$\mathbf{D}_7$	Parallel
111	1011	$\mathbf{s}_{12}$	$\mathbf{C}_4$	$\mathbf{D}_7$	
111	1100	$\mathbf{s}_{13}$	$\mathbf{C}_3$	$\mathbf{D}_6$	Parallel
111	1101	$\mathbf{s}_{14}$	$\mathbf{C}_3$	$\mathbf{D}_6$	
111	1000	$\mathbf{s}_9$	$\mathbf{C}_3$	$\mathbf{D}_5$	Parallel
111	1001	$\mathbf{s}_{10}$	$\mathbf{C}_3$	$\mathbf{D}_5$	

Originating state	Terminating paths, outputs
000	$\mathbf{C}_1, \mathbf{C}_2 \quad \mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_4 \quad \mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_5, \mathbf{s}_6, \mathbf{s}_3, \mathbf{s}_4, \mathbf{s}_7, \mathbf{s}_8$
001	$\mathbf{C}_1, \mathbf{C}_2 \quad \mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_4 \quad \mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_5, \mathbf{s}_6, \mathbf{s}_3, \mathbf{s}_4, \mathbf{s}_7, \mathbf{s}_8$
010	$\mathbf{C}_3, \mathbf{C}_4 \quad \mathbf{D}_5, \mathbf{D}_6, \mathbf{D}_7, \mathbf{D}_8 \quad \mathbf{s}_9, \mathbf{s}_{10}, \mathbf{s}_{13}, \mathbf{s}_{14}, \mathbf{s}_{11}, \mathbf{s}_{12}, \mathbf{s}_{15}, \mathbf{s}_{16}$
011	$\mathbf{C}_3, \mathbf{C}_4 \quad \mathbf{D}_5, \mathbf{D}_6, \mathbf{D}_7, \mathbf{D}_8 \quad \mathbf{s}_9, \mathbf{s}_{10}, \mathbf{s}_{13}, \mathbf{s}_{14}, \mathbf{s}_{11}, \mathbf{s}_{12}, \mathbf{s}_{15}, \mathbf{s}_{16}$
100	$\mathbf{C}_1, \mathbf{C}_2 \quad \mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_4 \quad \mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_5, \mathbf{s}_6, \mathbf{s}_3, \mathbf{s}_4, \mathbf{s}_7, \mathbf{s}_8$
101	$\mathbf{C}_1, \mathbf{C}_2 \quad \mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{D}_4 \quad \mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_5, \mathbf{s}_6, \mathbf{s}_3, \mathbf{s}_4, \mathbf{s}_7, \mathbf{s}_8$
110	$\mathbf{C}_3, \mathbf{C}_4 \quad \mathbf{D}_5, \mathbf{D}_6, \mathbf{D}_7, \mathbf{D}_8 \quad \mathbf{s}_9, \mathbf{s}_{10}, \mathbf{s}_{13}, \mathbf{s}_{14}, \mathbf{s}_{11}, \mathbf{s}_{12}, \mathbf{s}_{15}, \mathbf{s}_{16}$
111	$\mathbf{C}_3, \mathbf{C}_4 \quad \mathbf{D}_5, \mathbf{D}_6, \mathbf{D}_7, \mathbf{D}_8 \quad \mathbf{s}_9, \mathbf{s}_{10}, \mathbf{s}_{13}, \mathbf{s}_{14}, \mathbf{s}_{11}, \mathbf{s}_{12}, \mathbf{s}_{15}, \mathbf{s}_{16}$

Table 3.2 Organization of the information in Fig. 3.6 in the form of two tables.

From Table 3.2, from the rule of all originating and terminating paths from a state should be on the corners of $\mathbf{C}_1$ and $\mathbf{C}_2$ or $\mathbf{C}_3$ and $\mathbf{C}_4$ designated squares and parallel paths with $\mathbf{D}$ designations must be placed in diagonal positions of the $\mathbf{C}$ designated squares, we get the allocations of the signal vectors in the 16 QAM TCM constellation as depicted in Fig. 3.7.

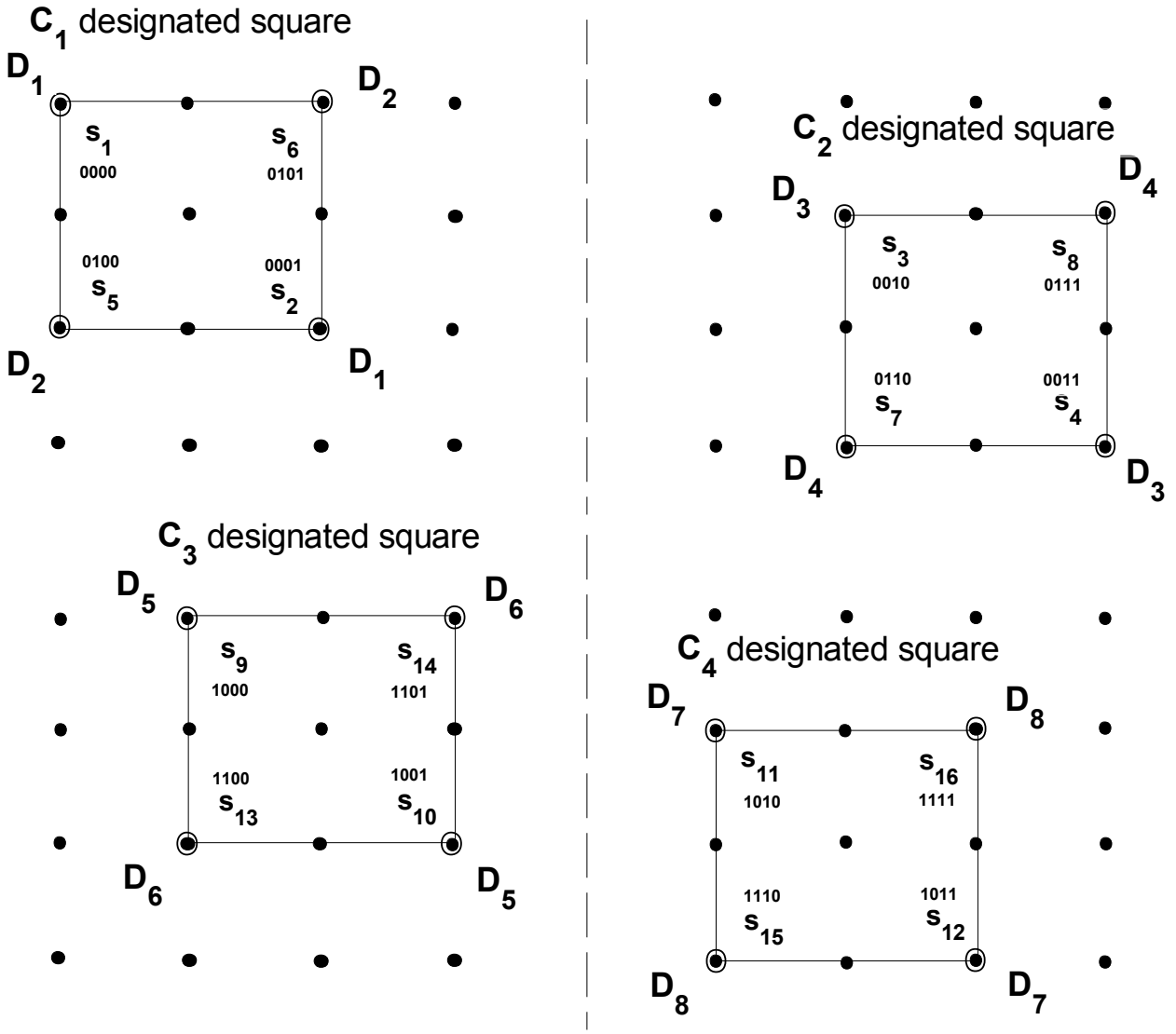

Fig. 3.7 Allocation of signal vectors in 16 QAM TCM constellation based on the information contained in Table 3.2.

In order to compare the performance of 16 QAM TCM against the uncoded 8 PSK, we must equate the average energy in both constellations. This is done by placing the 8 PSK signal vectors on a circle of unity radius adopting the same unity circle for eight signal vectors of 16 QAM TCM and assigning less than unit energy for the most inner ones and more energy for the four outer ones. In the end, the signal vector arrangement of Fig. 3.8 is obtained for 16 QAM TCM.

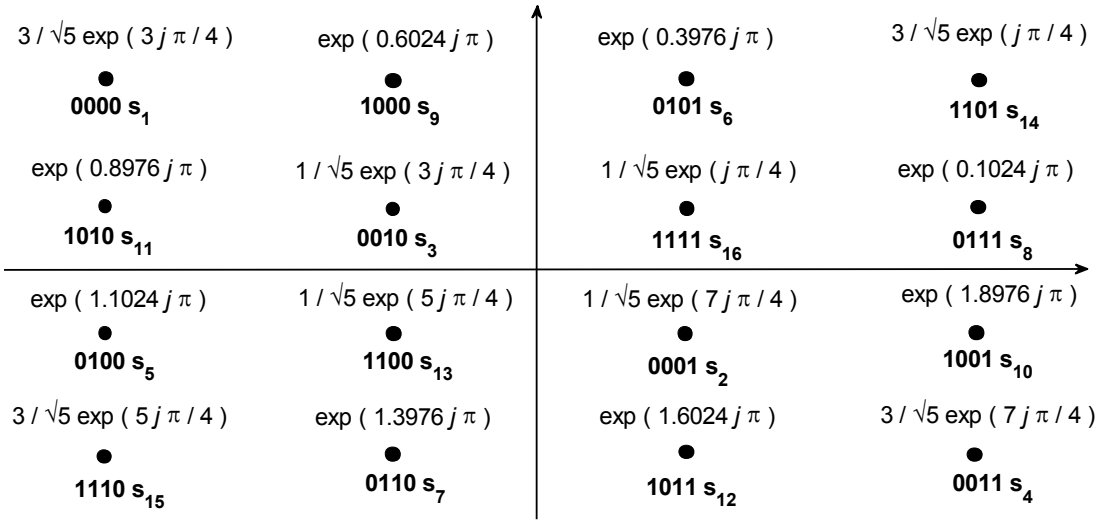

Fig. 3.8 Signal constellation of 16 QAM TCM.

Exercise 3.3 : By running TCM_PeGeneral.m which compares the probability of error performance of 8 PSK TCM against uncoded 4 PSK and TCMPe16QAM.m which compares the probability of error performance of 16 QAM TCM against uncoded 8 PSK using the encoder configurations and constellations of the above, find the relevant probability of error curves. Note the improvements. Verify that performance degrades, when the positions of signal vectors in the constellations are changed. Use the model files TCM_Dec.mdl and TCM_Enc.mdl during these runs. All Matlab files are available on course webpage.

References

1. John G. Proakis, Masoud Salehi, "Communication Systems Engineering" 2nd Ed., Prentice Hall 2002, ISBN : 0-13-061793-8, Chapter 9.
2. Bernard Sklar, "Digital Communications Fundamentals and Applications", 2nd Ed. Prentice Hall 2002, ISBN : 0-13-084788-7, Chapter 9.
3. G. Ungerboeck, "Channel Coding with Multilevel / Phase Signals, IEEE Transactions on Information Theory Vol. IT-28, No. 1, January 1982.