1-1-2022

Serially Concatenated Continuous Phase Modulation with SOVA Turbo Decoding

Zhijie Guo

SERIALLY CONCATENATED CONTINUOUS PHASE MODULATION WITH SOVA TURBO DECODING

A Thesis
presented in partial fulfillment of requirements
for the degree of Master of Science
in the Department of Electrical and Computer Engineering
The University of Mississippi

by

Zhijie Guo

May 2022

ABSTRACT

For a Serially Concatenated Continuous Phase Modulation (SCCPM) system that concatenates a rate of 1/2 Convolutional Code (CC) and an M-ary full response continuous phase modulation (CPM) signal, we design a turbo decoding scheme using the Soft Output Viterbi algorithm (SOVA) and study the system performance. A decomposition model is used in CPM to reduce the number of states and separate the continuous phase encoder (CPE) with the modulator. As a soft-input soft-output (SISO) decoding algorithm, SOVA is used to generate and update the soft information of decoded signal symbols during the iterative process for both the CPM signal and the CC. Newly generated soft information from one component decoder will be used by the other component decoder to constitute an iterative, i.e., turbo, decoding process. Simulation results show that a decoding gain of at least 1 dB can be obtained by using turbo decoding compared to that without turbo decoding.

ACKNOWLEDGEMENTS

I would like to express the most profound appreciation to my advisor Prof Dr. Lei Cao, for his patience, invaluable advice, continuous support, and encouragement throughout my M.S. program. I am deeply grateful to Prof Dr.Ramanarayanan Viswanathan and Prof Dr. John Daigle for serving on the thesis committee and providing invaluable assistance and comments for the thesis.

I would like to thank my classmates and support staff members in the Department of Electrical and Computer Engineering for their help in my job and study. I also gratefully acknowledge the assistance of my friends and all other people who helped me.

I would like to express my heartfelt gratitude to my parents for supporting me all the time.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

CPM	continuous phase modulation
SCCPM	serially concatenated continuous phase modulation
CC	convolutional code
SOVA	soft output Viterbi algorithm
CPE	continuous phase encoder
SISO	soft-input soft-output
VA	Viterbi algorithm
MM	memoryless modulator
MAP	maximum a-posterior probability
MLE	maximum likelihood estimation
MLSE	maximum likelihood sequence estimation
LLR	log-likelihood ratio
LREC	a rectangular frequency pulse with duration time LT
LRC	a raise cosine pulse with duration time LT
CPFSK	continuous phase frequency shift keying
MSK	minimum shift keying
GMSK	Gaussian minimum shift keying
AWGN	additive white Gaussian noise
BER	bit error rate

LIST OF SYMBOLS

T	symbol interval
E_s	symbol energy
E_b	bit energy
h	modulation index of CPM
M	alphabet size of modulation scheme
L	memory of the CPM signal
LT	duration time of a frequency pulse for CPM
$M_c(\cdot)$	correlation Metric
$M_b(\cdot\ ,\cdot)$	branch metric for Viterbi decoder
$M_p(\cdot)$	path metric for Viterbi decoder
s_n^k	a path starts from starting stage and arrives state n at stage k in a trellis
$d(\cdot\ ,\cdot)$	the Euclidean distance between two vectors
$D(\cdot\ ,\cdot)$	the square Euclidean distance of two vectors
δ_{min}^2	the minimum Euclidean distance of two paths in the trellis
$\Lambda(\cdot\ ;\cdot)$	soft output information from one component SISO decoder
$\Lambda^*(\cdot\ ;\cdot)$	updated soft output information from one component SISO decoder
$L(u_k)$	log-likelihood ratio of decoded bit u_k
Δ_s^k	path metric difference of survival path s_s^k and discarded path at state s at stage k
XOR	exclusive-or operation

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 **INTRODUCTION**

The continuous phase modulation (CPM) is a class of digital phase modulation schemes with memory [1, 2] which is widely used in satellite communication, Bluetooth, etc. CPM has been extensively studied for its excellent bandwidth and energy efficiency due to the property of the continuous phase that narrows the main lobe of the spectrum. Aulin and Sundberg [2] gave a comprehensive description and analysis of the full-response CPM signal, power spectrum properties, and proposed a method to analyze the error performances by calculating the minimum Euclidean distance of the paths through the trellis. Later, Aulin [3] proved that the minimum Euclidean distance is a good performance measure for Viterbi detected CPM signal. One important property of CPM that has attracted research interest is the memory of phase. The memory property lets the signal phase possess a form of finite-state trellis, that is, a finite-state Markov chain. Therefore, some trellis-based [4] decoding algorithms can be implemented, such as the Viterbi algorithm (VA) [5] and the BCJR algorithm [6], where the VA is an implementation of maximum likelihood sequence estimation (MLSE) and the BCJR algorithm is a maximum a-posterior probability (MAP) algorithm.

A CPM signal can be decomposed into two separated parts: a linear encoder called continuous phase encoder (CPE) and a memoryless modulator (MM). This approach allows the coding/decoding process to be independent from the modulation/demodulation process. Rimoldi [7] proved that the CPE is in a form of convolutional code (CC) and proposed a form of tilted-phase trellis that simplifies the structure of the finite-state trellis of CPM with reduced number of states in the trellis. Much work on coded CPM has been done based on the CPM decomposition model [1, 8]. Various types of channel encoders can be cascaded with the CPE to constitute different serially concatenated continuous phase modulation

(SCCPM) systems. The channel code can increase the reliability of the transmission by correcting errors of decoding result of CPM. Conventionally decoding an SCCPM system is a sequential process with the decoding of CPM followed by decoding the channel code. In this thesis, we further develop a turbo decoding structure for an SCCPM system to obtain improved performance from an iterative process.

When the turbo coding [1,9,10] was first proposed, its performance, being close to the Shannon limit, attracted wide research interest. The idea of iterative decoding has become a popular research area in channel coding. The classical turbo coded system is designed with a parallel structure which iteratively exchanges the reliability information of decoding results between two component decoders. Later a serially coding and decoding structure was developed and it has been shown that the serially structure concatenated with convolutional codes has a better performance than the parallel structure [1, 11]. Based on the turbo idea, many attempts to apply the iterative decoding to the CPM system [1, 12–16], and the CPM works well with the turbo decoding due to the trellis property of the CPM [17, 18].

The Viterbi algorithm (VA) is implemented as an maximum likelihood sequence estimation (MLSE) detector, which was first proposed by Viterbi [5] as a method of decoding convolution code. Forney [19] gave a tutorial of the algorithm and indicated how to implement the VA. The VA is a well-known MLSE algorithm to utilize the Markov process and has become an important decoding component in the communication area [20]. Hagenaauer [20] proposed a Soft Output Viterbi Algorithm (SOVA) which can provide the reliability information, i.e., soft information, of the decoding results in a form of log-likelihood ratio (LLR). It has been shown [21] that the computation complexity of the SOVA algorithm is about half of the Max-Log BCJR (Max-log MAP) algorithm, where the Max-Log BCJR is a MAP algorithm with reduced complexity by invoking an approximation [10]. Hagenaauer proposed a modified SOVA in [22] that introduced the method of *Update Sequence* for implementing SOVA and proposed the modification of the branch/path calculation for SOVA implementation in turbo decoding in [23].

In this thesis, we consider an SCCPM system with the concatenation of a 1/2 CC and a 4-ary CPM. We design a turbo decoding process for this SCCPM system using two Soft Output VA (SOVA) decoders, one for CPM decoding and one for CC decoding. Soft information generated by one SOVA decoder will be utilized by the other SOVA decoder to construct a turbo decoding process. SOVA is adopted in this work as it can provide log-likelihood ratio (LLR) of decoded results close to the BCJR algorithm performance but with much-reduced complexity [1, 20, 24].

In this thesis, we give the implementation details of the turbo decoding process of the SCCPM system with SOVA. Decoding performance of the design is demonstrated through simulations. It shows that SCCPM can overperform CPM with about 1.5 dB in SNR without turbo decoding and achieve an additional one dB SNR gain with the designed turbo decoding.

The rest of the thesis is organized as follows. Chapter 2 describes the CPM system, modulation/demodulation, optimal receiver, and its error performance analysis. Chapter 3 presents the details of the coding, decoding, iterative structure of the SCCPM, addresses how to implement the SOVA in the SCCPM system, and presents the experiment results. Finally, Chapter 4 concludes the thesis.

CHAPTER 2 **CPM SYSTEM DESCRIPTION**

In this chapter, we describe the conventional CPM signal model, properties of the signal phases, and its decomposed model. We detailed how to use the Viterbi algorithm as an optimal receiver for CPM signals. Then we describe a method to analyze the error performance of the MLSE decoder for the CPM system.

2.1 Modulation of CPM Signal

2.1.1 CPM Signal

The conventional CPM signal can be expressed as [2]

$$s(t, \boldsymbol{\alpha}) = \sqrt{\frac{2E_s}{T}} cos(2\pi f_c t + \phi(t; \boldsymbol{\alpha}) + \phi_0) \tag{2.1}$$

where E_s is the *symbol energy*, T is the *symbol duration time* and the ϕ_0 is the *initial phase* which can be assumed as *zero* without loss of generality. The $\phi(t; \boldsymbol{\alpha})$ is the *information-carrying phase* which is defined by

$$\phi(t; \boldsymbol{\alpha}) = 2\pi h \sum_{k=-\infty}^{n} \alpha_k q(t - kT), \quad nT \leq t \leq (n+1)T \tag{2.2}$$

where the $\boldsymbol{\alpha}$ is the sequence of M-ary data symbols, α_k is the k-th symbol in the sequence and belongs to the alphabet set $\{\pm 1, \pm 3, \ldots, \pm(M-1)\}$, and h is the modulation index of the CPM signal. The modulation index h is a rational number for the sake of implementation and is defined as

$$h = \frac{K}{P} \tag{2.3}$$

where K and P are relatively prime positive integers. The $q(t)$ is the phase pulse which is defined as

$$q(t) = \int_0^t g(\tau)d\tau \tag{2.4}$$

where $g(t)$ is called frequency pulse which controls the shape of the phase pulse, for example, a rectangular frequency pulse which commonly referred to as $LREC$ is defined as

$$g(t) = \begin{cases} \frac{1}{2LT} & 0 \leq t \leq LT \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

where L is referred to as the memory of the signal [7] and LT is the duration of the pulse. When $L = 1$, the signal is called full-response CPM. When $L > 1$, the signal is called partial-response CPM.

Table 2.1 shows three commonly CPM pulse shape. The GMSK denotes a *Gaussian minimum shift keying* pulse and the B represents the 3 dB bandwidth of the Gaussian pulse which is referred to as bandwidth parameter. The LRC denotes a *raise cosine* pulse with pulse duration L. The $1REC$ CPM is also referred to as continuous phase frequency shift keying (CPFSK), and minimum shift keying (MSK) is a special case of binary CPFSK with modulation index $h = 0.5$.

Table 2.1: Three commonly used frequency pulse of CPM

LREC	$g(t) = \begin{cases} \frac{1}{2LT} & 0 \leq t \leq LT \\ 0 & \text{otherwise} \end{cases}$
LRC	$g(t) = \begin{cases} \frac{1}{2LT}\left(1 - \cos\frac{2\pi t}{LT}\right) & 0 \leq t \leq LT \\ 0 & \text{otherwise} \end{cases}$
GMSK	$g(t) = \frac{Q(2\pi B(t-\frac{T}{2}))-Q((2\pi B(t+\frac{T}{2})))}{\sqrt{\ln 2}}$

2.1.2 Phase State and State Transition

The (2.2) gives the definition of the *information-carrying phase*. It can be rewritten as

$$\begin{aligned}\phi(t;\boldsymbol{\alpha}) &= \pi h \sum_{k=-\infty}^{n-L} \alpha_k + 2\pi h \sum_{k=n-L+1}^{n} \alpha_k q(t-kT) \\ &= \theta_n + 2\pi h \sum_{k=n-L+1}^{n} \alpha_k q(t-kT), \quad nT \le t \le (n+1)T \end{aligned} \tag{2.6}$$

The first term on the right hand side of (2.6), i.e., θ_n, is the accumulated phase from the starting to the $(n-L)$-th symbol interval which is called *phase state*; the second term depends on the data symbol $(\alpha_n, \alpha_{n-1}, \ldots, \alpha_{n-L+1})$ which is called *correlative state vector*. When $L > 1$, the terminal phase state of the CPM signal at time $t = nT$ can be expressed as [25]

$$S_n = [\theta_n, \alpha_{n-1}, \alpha_{n-2}, \ldots, \alpha_{n-L+1}] \tag{2.7}$$

and we can get terminal phase state of the CPM when next symbol α_{n+1} received at $t = (n+1)T$:

$$S_{n+1} = [\theta_{n+1}, \alpha_n, \alpha_{n-1}, \ldots, \alpha_{n-L+2}] \tag{2.8}$$

where the phase state θ_n is obtained by:

$$\theta_{n+1} = \theta_n + \pi h \alpha_{n-L+1} \tag{2.9}$$

The terminal phase state of the CPM signal depends on the previous L symbols. Therefore, the CPM scheme is a modulation scheme with L symbol memories. For $L = 1$ CPM signal (full response CPM signal), the memory is 1 and the terminal phase state only depends on the last phase of symbol. Therefore, the full response CPM phase transition is a simple

Markov Chain. That is, for full response CPM, (2.9) can be rewritten as

$$\theta_{n+1} = \theta_n + \pi h \alpha_n$$

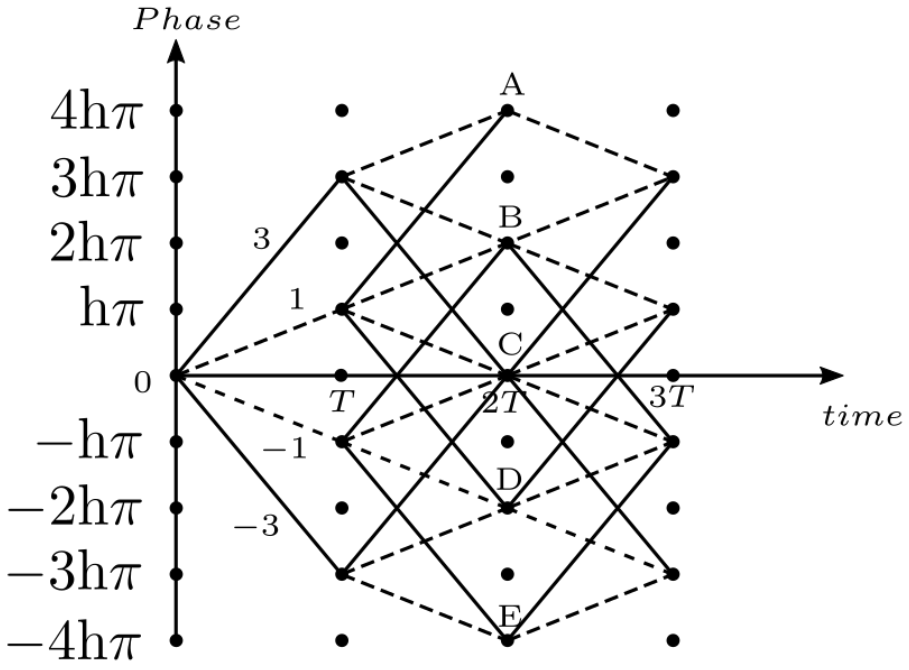

Figure 2.1: The phase trajectory for quaternary CPFSK (solid and dash lines) and binary CPFSK (dash lines)

Fig.2.1 displays a phase tree for CPFSK with M = 2 and M =4. When h = $\frac{1}{2}$, the phase trajectory with dashed lines in Fig.2.1 displays a phase trajectories for MSK signal. It can be observed that there are 4 possible phase states for MSK signal, that is, the possible phase states θ_s at one arbitrary time instant $t = nT$ can be

$$\theta_s \in \left\{0, \frac{1}{2}\pi, \pi, \frac{3}{2}\pi\right\} \tag{2.10}$$

where n is a positive integer greater than 1. It should be noted that the phases of CPM can

be always modulo 2π, that is, the period of phases is 2π. In the phase trellis of the CPM signal, there are PM^{L-1} phase states when K is even and there are $2PM^{L-1}$ phase states when K is odd, where the K and P are the numerator and denominator of h respectively. It can be noticed that when L increases, the number of states in the trellis will be increased exponentially.

2.1.3 CPM Decomposed Model

According to Rimoldi's CPM decomposition approach [7], a general CPM signal can be decomposed into one linear encoder with memory and one memoryless modulator (MM). The structure of phase trellis is modified into a form of time-invariant trellis, which is called physical tilted-phase. The physical tilted-phase trellis simplifies the structure of phase trellis and reduces the number of states. The CPE can be taken as a convolutional code (CC) with rate 1.

Consider an $LREC$ M-ary CPM signal with $h = K/P$, the symbols $u_k \in \{0, 1, ..., M-1\}$.

Let

$$V_n = R_P \left[2\pi h \sum_{k=0}^{n-L} u_k \right] \tag{2.11}$$

where $R_P[\cdot]$ is the "modulo P operator".

Let

$$t = \tau + nT \tag{2.12}$$

and

$$0 \leq \tau \leq T \tag{2.13}$$

The tilted-phase $\psi(t)$ during a symbol interval $[nT, (n+1)T]$ is given as [7]

$$\psi(\tau + nT, \mathbf{u}) = R_{2\pi} \left[V_n + 4\pi h \cdot \sum_{k=0}^{L-1} u_{n-k} q(\tau + kT) + \omega(\tau) \right] \tag{2.14}$$

where $R_{2\pi}[\cdot]$ is the "modulo 2π operator" and

$$\omega(\tau) = \pi h(M-1)\frac{\tau}{T} - 2\pi h(M-1)\sum_{k=0}^{L-1} q(\tau + kT) + \pi h(L-1)(M-1) \tag{2.15}$$

which is the data-independent term. It has been shown in [7] that the number of states for $\psi(t, \mathbf{u})$ will decrease to PM^{L-1} in the phase trellis while the number of states is $2PM^{L-1}$ for the original M-ary CPM when K is odd.

Comparing with the conventional phase trellis for MSK shown in Fig. 2.2, Fig. 2.3 shows the physical tilted-phase trellis for the MSK signal. MSK is the simplest signal of CPM, which is a binary, 1REC CPM with $h = 1/2$. It has 2 states in the physical tilted-trellis, whereas the original MSK has $2 \times 2 \times 2^{1-1} = 4$ states in the phase tellies. Let X_n

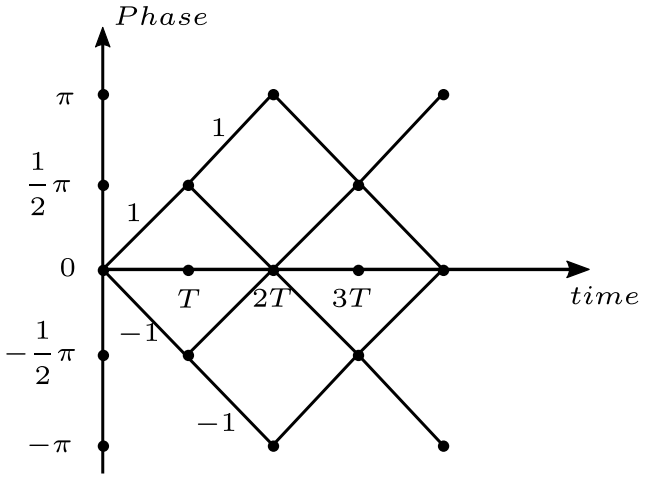

Figure 2.2: Conventional MSK phase trajectory diagram

denote the n-th encoded symbol by CPE, $X_n = [u_n,\ \ldots,\ u_{n-L+1},\ V_n]$, and for full-response CPM, $X_n = [u_n,\ V_n]$, where the u_n is the n-th information symbol and V_n is obtained from (2.11) which is the phase state of symbol u_{n-1}. Using (2.1) and following notations in [7], we write

$$\psi(\tau, X_n) \text{ instead of } \psi(\tau + nT, \mathbf{u}), \qquad 0 \le \tau \le T$$

and

$$s(\tau, X_n) \text{ instead of } s(\tau + nT, \mathbf{u}), \qquad 0 \le \tau \le T$$

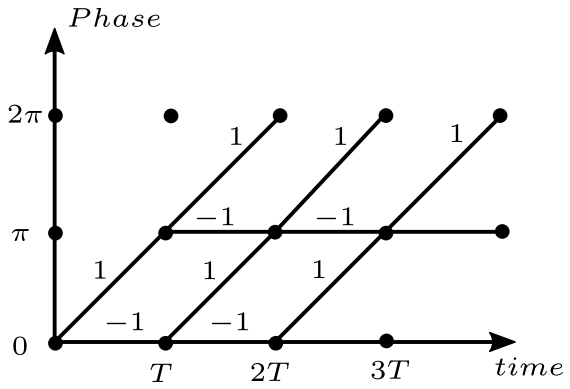

Figure 2.3: Physical tilted-phase MSK phase trajectory diagram

The CPM signal transmitting the n-th CPE symbol X_n with tilted phase can be expressed as

$$\begin{aligned} s(\tau, X_n) &= \sqrt{2E_s/T}\cos(2\pi f_1(\tau + nT) + \psi(\tau + nT, X_n) + \phi_0), \\ &= I(\tau, X_n)\Phi_I(\tau + nT) + Q(\tau, X_n)\Phi_Q(\tau + nT) \\ 0 &\le \tau \le T,\ t = \tau + nT \end{aligned} \tag{2.16}$$

where

$$\begin{aligned} I(\tau, X_n) &= \sqrt{E_S/t}\cos\psi(\tau, X_n) \\ Q(\tau, X_n) &= \sqrt{E_S/t}\sin\psi(\tau, X_n) \end{aligned} \tag{2.17}$$

and

$$\begin{aligned} \Phi_I(\tau + nT) &= \sqrt{1/2}\cos[2\pi f_1(\tau + nT) + \phi_0] \\ \Phi_Q(\tau + nT) &= -\sqrt{1/2}\cos[2\pi f_1(\tau + nT) + \phi_0] \end{aligned} \tag{2.18}$$

The f_1 is introduced to compensate for the offset between the tilted phase with original phase [7], which is obtained by

$$f_1 = f_c - h(M-1)/2T$$

Note that in (2.16), the signal is decomposed into in-phase and quadrature components. Fig .2.4 shows the structure of the memoryless modulator.

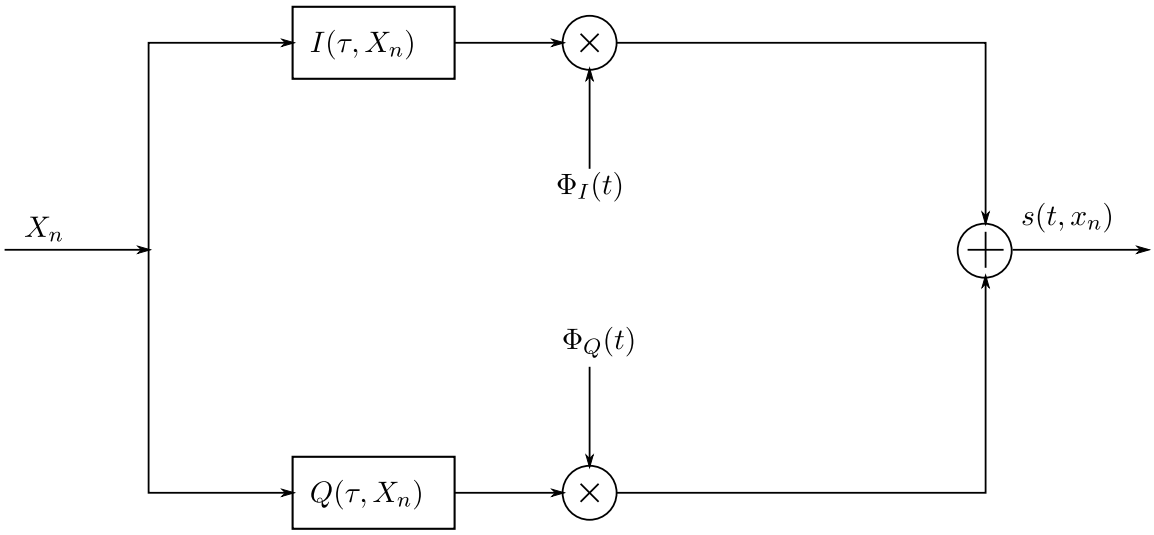

Figure 2.4: Memoryless modulator

2.2 Optimal Receiver and Demodulation of CPM Signal

2.2.1 Optimal Receiver

The CPM demodulation consists of a demodulator and a decoder that can implement some optimal detection algorithm based on the trellis structure.

An AWGN channel model over one symbol interval can be described as [25]

$$r(t) = s_m(t) + n(t), \quad 0 \le t \le T \tag{2.19}$$

where $s_m(t)$ is the transmitted signal which is selected from the set $\{s_m(t), m = 1,\ 2,\ \ldots, M\}$, $n(t)$ is a waveform of zero-mean Gaussian noise with variance $N_0/2$. We define an N-dimensional orthogonal space as a signal space characterized by N orthonormal basis functions $\{f_k(t)\}$. The basis functions satisfy that:

$$\int_0^T f_i(t) f_j(t) dt = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{2.20}$$

Each signal in the set $\{s_m(t), m = 1,\ 2,\ \ldots, M\}$ can be expressed as a linear combination

of N orthogonal waveforms $\{f_k(t)\}$

$$s_m(t) = \sum_{k=1}^{N} s_{m,k}\ f_k(t) \tag{2.21}$$

where

$$s_{m,k} = \int_{-\infty}^{\infty} s_m(t)\ f_k(t)dt \tag{2.22}$$

Note that (2.22) is the inner product of $s_m(t)$ and $f_k(t)$ which denotes the projection of $s_m(t)$ onto $f_k(t)$. Then the vector expression of the received signal (2.19) is given as:

$$\mathbf{r} = \mathbf{s_m} + \mathbf{n} \tag{2.23}$$

where all vectors are N-dimensional real vectors. We use m to denote the transmitted message selected from the set of possible messages $\{1, 2, \ldots, M\}$ according to probabilities P_m, the elements of noise component $\mathbf{n}$ are independent and identically distributed (iid) Gaussian random variables. It has been shown that the AWGN waveform channel (2.19) and the AWGN vector channel (2.23) are equivalent [25]. We use $\hat{m}$ to denote the estimated message of receiver. The receiver observes $\mathbf{r}$ and based on the observation makes its decision to determine which message was transmitted. The goal of an optimal receiver/detector is to maximize the probability of making the correct estimate based on the given observation $\mathbf{r}$.

The estimated message $\hat{m}$ can be optimal decided by:

$$\hat{m} = \underset{1 \le m \le M}{\arg\max}\, P\{m \mid \mathbf{r}\} \tag{2.24}$$

Note that transmitting message m is equivalent to transmitting s_m, the decision rule (2.24) can be expressed as:

$$\hat{m} = \underset{1 \le m \le M}{\arg\max}\, P\{\mathbf{s_m} \mid \mathbf{r}\} \tag{2.25}$$

The optimal decision rule given by (2.24) and (2.25) is referred to as *maximum a*

posterior probability (MAP). According to the Bayes's Theorem, the (2.24) can be written as

$$\hat{m} = \arg\max_{1 \le m \le M} \frac{P\{\mathbf{r} \mid m\} P_m}{P\{\mathbf{r}\}} \tag{2.26}$$

where P_m is called the *a prior* probability of message m, and the messages are assumed to be equiprobable in the CPM system, i.e., $P_m = 1/M$ for all $1 \le m \le M$. $P\{\mathbf{r} \mid m\}$ is referred to as the likelihood of m. And $P\{\mathbf{r}\}$ is same for all possible m, therefore, the optimal decision only depends on the likelihood of m when $\mathbf{r}$ is given. Then the (2.26) can be written as:

$$\hat{m} = \arg\max_{1 \le m \le M} P\{\mathbf{r} \mid m\} P_m \tag{2.27}$$

$$\hat{m} = \arg\max_{1 \le m \le M} P\{\mathbf{r} \mid m\} \tag{2.28}$$

The decision rule (2.28) is referred to as *maximum-likelihood estimate* (MLE).

For a system with trellis structure, the MLSE and MAP are two commonly used decision rules to design the optimal receiver [24, 25]. The MLSE selects the most likely path (sequence) over an observed signal sequences. The Viterbi algorithm (VA) performs the MLSE algorithm by searching the path (sequence) with the minimum Euclidean distance between the received signal and estimated sequences. A detailed description of the VA implementation will be presented later.

Assume that the received M-ary signal has a duration of K symbol intervals: $r(t) = s(t)+n(t)$. Then through the trellis, we can have M^K possible paths over K symbol intervals. To select the path with minimum metric, we need to calculate the Euclidean distance between the received signal and all possible paths. The vector model of received signal is represented as: $\mathbf{R} = \mathbf{S} + \mathbf{N}$, where $\mathbf{R} = [\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{r}_3\ \ldots\ \mathbf{r}_K]^T$, and transmitted signal is denoted as $\mathbf{S} = [\mathbf{s}_1\ \mathbf{s}_2\ \mathbf{s}_3\ \ldots\ \mathbf{s}_K]^T$, the AWGN $\mathbf{N} = [\mathbf{n}_1\ \mathbf{n}_2\ \mathbf{n}_3\ \ldots\ \mathbf{n}_K]^T$ where $\{\mathbf{r}_i, i = 1, 2, \ldots, K\}$, $\{\mathbf{s}_i, i = 1, 2, \ldots, K\}$, and $\{\mathbf{n}_i, i = 1, 2, \ldots, K\}$ have same definition with (2.23). Then we

can have

$$\mathbf{R} = \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \vdots \\ \mathbf{r}_K \end{bmatrix} = \begin{bmatrix} r_{1,1} & r_{1,2} & \dots & r_{1,N} \\ r_{1,1} & r_{1,2} & \dots & r_{1,N} \\ \vdots & \ddots & \ddots & \vdots \\ r_{K,1} & r_{K,2} & \dots & r_{K,N} \end{bmatrix}$$

and

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_1 \\ \mathbf{s}_2 \\ \vdots \\ \mathbf{s}_K \end{bmatrix} = \begin{bmatrix} s_{1,1} & s_{1,2} & \dots & s_{1,N} \\ s_{1,1} & s_{1,2} & \dots & s_{1,N} \\ \vdots & \ddots & \ddots & \vdots \\ s_{K,1} & s_{K,2} & \dots & s_{K,N} \end{bmatrix}$$

The Euclidean distance between the two signal $r(t)$ and $s(t)$ over one symbol interval which is denoted as $d\left(r(t), s(t)\right)$ is calculated by using

$$d\left(r(t), s(t)\right) = \sqrt{\int_0^T [r(t) - s(t)]^2 \, dt} \tag{2.29}$$

Then the Euclidean distance between two signals $r(t)$ and $s(t)$ over K symbol intervals is

$$\begin{aligned} d\left(r(t), s(t)\right) &= \sqrt{\int_0^{KT} [r(t) - s(t)]^2 dt} \\ &= \sqrt{\sum_{i=1}^{K} \int_{(i-1)T}^{iT} [r(t) - s(t)]^2 dt} \end{aligned} \tag{2.30}$$

Then the Euclidean distance between two paths $\mathbf{R}$ and $\mathbf{S}$ over K symbol intervals can be written as:

$$d(\mathbf{R}, \mathbf{S}) = \sum_{i=1}^{K} \|\mathbf{r}_i - \mathbf{s}_i\| \tag{2.31}$$

where

$$\|\mathbf{r}_i - \mathbf{s}_i\| = \sqrt{\sum_{j=1}^{N} (r_{i,j} - s_{i,j})^2}, \quad 1 \le i \le K \tag{2.32}$$

is the Euclidean distance of two vectors $\mathbf{r}_i$ and $\mathbf{s}_i$. The square of Euclidean distance $D(\cdot, \cdot)$

is used as the distance metric, that is,

$$D(\mathbf{R}, \mathbf{S}) = d^2(\mathbf{R}, \mathbf{S}) \tag{2.33}$$

Then the decision rule for MLSE can be expressed as

$$\hat{\mathbf{S}} = \arg \min_{m} \sum_{j=1}^{K} D(\mathbf{r}_j, \mathbf{s}_j^{(m)}) \tag{2.34}$$

where $\mathbf{s}_j^{(m)} = \{s_1^{(m)}, s_2^{(m)}, \ldots, s_K^{(m)}\}$ is the symbol sequence corresponding to the m-th path along the trellis. And

$$\hat{\mathbf{S}} = [\hat{s}_1 \ \hat{s}_2 \ \ldots \ \hat{s}_K]^T \tag{2.35}$$

Therefore, an MLSE detector can obtained the optimal estimated results by searching for the estimated sequence (path) that has minimum distance metric to the received signal.

For a M-ary CPM signal, the received signal $r(t)$ with a length of KT is:

$$r(t) = s(t, \mathbf{u}) + n(t) \tag{2.36}$$

and

$$\mathbf{R} = \mathbf{S} + \mathbf{N} \tag{2.37}$$

where $s(t, \mathbf{u})$ is the transmitted signal waveform and $\mathbf{u}$ is the sequence of transmitted data symbols, $n(t)$ is an AWGN with zero mean and variance $\sigma^2 = N_0/2$. (2.37) is the vector model of (2.36). There will be M^K possible paths(sequences) generated in the trellis during the KT intervals. The likelihood of the estimated signal $\hat{\mathbf{S}}$ given the received signal $\mathbf{R}$ can

be written as

$$\begin{aligned} P\{\mathbf{R} \mid \hat{\mathbf{S}}\} &= P\{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_K \mid \hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, \ldots, \hat{\mathbf{s}}_K\} \\ &= \prod_{i=1}^{K} P\{\mathbf{r}_i \mid \hat{\mathbf{s}}_i\} \\ &= (\frac{1}{\sqrt{2\pi\sigma^2}})^K \cdot \exp\left\{-\sum_{i=1}^{K} \frac{(\mathbf{r}_i - \hat{\mathbf{s}}_i)^2}{2\sigma^2}\right\} \end{aligned} \tag{2.38}$$

then take the nature logarithm, (2.38) can be written as:

$$\begin{aligned} \ln P\{\mathbf{R} \mid \hat{\mathbf{S}}\} &= K \cdot \ln(\frac{1}{\sqrt{2\pi\sigma^2}}) - \frac{1}{2\sigma^2} \sum_{i=1}^{K} (\mathbf{r}_i - \hat{\mathbf{s}}_i)^2 \\ &= K \cdot \ln(\frac{1}{\sqrt{2\pi\sigma^2}}) - \frac{1}{2\sigma^2} \cdot D(\mathbf{R}, \hat{\mathbf{S}}) \end{aligned} \tag{2.39}$$

The distance metric can be written as:

$$\begin{aligned} D(\mathbf{R}, \hat{\mathbf{S}}) &= \sum_{i=1}^{K} \mathbf{r}_i^2 + \sum_{i=1}^{K} \hat{\mathbf{s}}_i^2 - 2\sum_{i=1}^{K} \mathbf{r}_i \cdot \hat{\mathbf{s}}_i \\ &= \sum_{i=1}^{K} \mathbf{r}_i^2 + \sum_{i=1}^{K} \hat{\mathbf{s}}_i^2 - 2\sum_{i=1}^{K} M_c(\mathbf{r}_i, \hat{\mathbf{s}}_i) \\ &= \sum_{i=1}^{K} \mathbf{r}_i^2 + \sum_{i=1}^{K} \hat{\mathbf{s}}_i^2 - 2M_c(\mathbf{R}, \hat{\mathbf{S}}) \end{aligned} \tag{2.40}$$

According to (2.39) and (2.40), the MLSE for CPM is obtained by minimizing the distance metric $D(\mathbf{R}, \hat{\mathbf{S}})$, which is equivalent to maximizing the correlation metric $M_c(\mathbf{R}, \hat{\mathbf{S}})$.

2.2.2 Viterbi Algorithm

Classical Viterbi decoding consists of two parts: a forward process and a traceback (backward) process. The forward process consists of three steps: adding, comparing, and selecting. For simplicity, a trellis with two branches entering and leaving at each state node is assumed in this section, i.e., there are two paths that merge at each possible state node at each stage except the first and the second stage, and there are two branches heading to

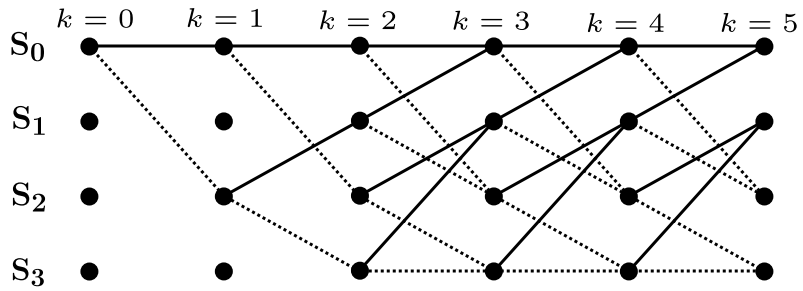

Figure 2.5: Example of the trellis diagram for a (2,1,3) convolutional code

two different states at the next stage. For example, Fig. 2.5 displays the trellis diagram for a simple convolution code with 1/2 code rate. Throughout this thesis, a solid line denotes a decoding bit -1 (i.e., 0) and a dashed line denotes a decoding bit $+1$. Assume that $\mathbf{s}_n^k$ denotes a possible path in the trellis from state 0 at stage 0 to state $S^k = n$ at stage k, and state $S^{k-1} = n'$ is the state at stage $k-1$, i.e., the previous state of the path. Let $M_b(\cdot,\cdot)$ denote branch metric, which is a metric corresponding to a transition from one stage to the next stage. For example, $M_b(S^{k-1}, S^k)$ is the metric corresponding to the transition from state $S^{k-1} = n'$ at stage $k-1$ to state $S^k = n$ at stage k. Let $M_p(\cdot)$ denote the path metric of a path arrived one stage, which is the accumulated branch metrics over a path from starting stage to the current stage. For example, $M_p(\mathbf{s}_n^k)$ is the accumulated branch metrics from stage 0 to stage k. In the procedure of decoding, for each state S^k at stage k, where $k \geq 0$,

- Add: the path metric M_p is obtained by calculating the branch metric M_b for all of the transitions (S^{k-1}, S^k) and adding the path metric of the previous state S^{k-1}.
- Compare: find the minimum (or the maximum) value of path metric for each terminal state node at each stage.
- Select: at each symbol interval, select one survival path for each terminal state and discard other paths.

Consider a state transition $(S_{n'}^{k-1} \rightarrow S_n^k)$ which is along a CC trellis (e.g., Fig. 2.5) transits from stage k-1 to stage k and the state transits from $S_{n'}$ to S_n, where $k \geq 1,\ n', n \in$

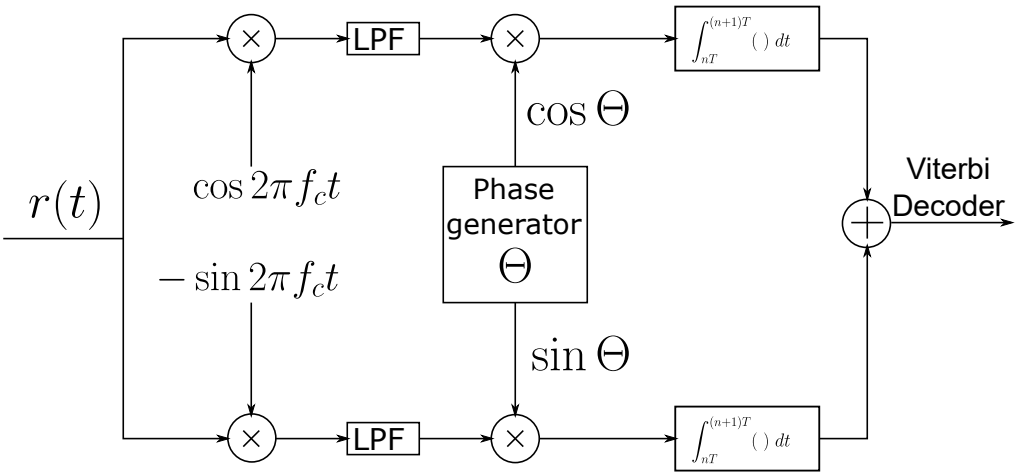

Figure 2.6: Demodulator for CPM

$\{0, 1, 2, 3\}$. We use $\mathbf{s}_n^k$ to denote a path that arrives state S_n at stage k. The path metric for the path $\mathbf{s}_n^k$ can be expressed recursively as:

$$M_p(s_n^k) = M_b((S_{n'}^{k-1}, S_n^k)) + M_p(s_{n'}^{k-1}) \tag{2.41}$$

In the traceback process, the decoded bit sequence can be obtained by tracing back the trellis from the final survival state node along the survival path. For example, in the Fig. 2.5, assume that the survival path is $[S_0^{k=0} \rightarrow S_0^{k=1} \rightarrow S_2^{k=2} \rightarrow S_1^{k=3} \rightarrow S_0^{k=4} \rightarrow S_0^{k=5}]$, then the decoding result [0 1 0 0 0] can be obtained by tracing back the survival path through the trellis.

2.2.3 Demodulator Description

Fig. 2.6 shows the structure of the demodulator for CPM signal. Assume that the received signal has a duration of K symbol intervals. The received signal $r(t)$ is converted into a pair of orthogonal baseband signal and the carrier frequency is suppressed by a low-pass filter (LPF). Let $r_I(t)$ and $r_Q(t)$ denote the in-phase and the quadrature components of the received baseband signal respectively. The branch metric for VA over one symbol

interval can be calculated by using correlation metric:

$$\begin{aligned} M_b(\Theta_{n-1}, \Theta_n) &= M_c(r(t), s(t, u_n)) \\ &= \int_{(n-1)T}^{nT} r_I(t)cos(\mathbf{\Theta}(t)) + r_Q(t)sin(\mathbf{\Theta}(t))dt,\ 1 \leq n \leq K \end{aligned} \tag{2.42}$$

where $\mathbf{\Theta}(t)$ denotes the waveform of a possible phase state in the trellis during one symbol interval, that is, the estimated sequence $\hat{\mathbf{u}}$ is represented by a sequence of phase states $\mathbf{\Theta}$ in the trellis in the CPM system, and $\Theta_n = \Theta_{n-1} + \pi h u_n$ by using the notations in (2.9). For example, in a MSK system, the possible phase states can be $\{0,\ \frac{\pi}{2},\ \pi,\ \frac{3\pi}{2}\}$, and for the transition during the interval $[T,\ 2T)$, by observing the Fig. 2.2, the possible phase states for this transition can be $\Theta = \left(\frac{\pi}{2} + \frac{1}{2}\pi\right)$ or $\Theta = \left(\frac{\pi}{2} + (-1)\frac{1}{2}\pi\right)$, note that in terms of phase, $-\pi$ and π are indistinguishable. The estimated phase states are generated by a phase generator. In the next chapter, the SCCPM implements a quaternary CPFSK, i.e., 1REC, $M = 4$ with $h = 1/4$. The waveforms of estimated phase states Θ corresponding to the phase transitions are presented in Fig. 2.7 and the phase state transition diagram is displayed in Fig. 3.8 in Chapter 3. In Fig. 2.7, blue lines and red lines are $\cos\Theta(t)$ and $\sin\Theta(t)$ respectively.

To implement the Viterbi algorithm for CPM demodulator, the branch metric and path metric of the CPM trellis at each possible state at each stage need to be calculated. Given the observed signal components $r_I(t)$ and $r_Q(t)$, the branch metric for the k-th symbol u_k which transits from phase state $S^{k-1} = \Theta_{k-1}$ to $S^k = \Theta_k$ can be calculated by using (2.42). Calculate the branch metrics for all of the possible transitions over the K symbol intervals and calculate the path metric that corresponds to each possible transition. A possible phase path of the received signal $r(t)$ from the first symbol to the k-th symbol can be denoted as $\mathbf{s^k}$, which is along the CPM trellis, then the path metric can be expressed as:

$$M_p(\mathbf{s^k}) = M_b(\Theta_{k-1}, \Theta_k) + M_p(\mathbf{s}^{k-1}) \tag{2.43}$$

The paths with maximum path metric at each possible state at each stage will be the

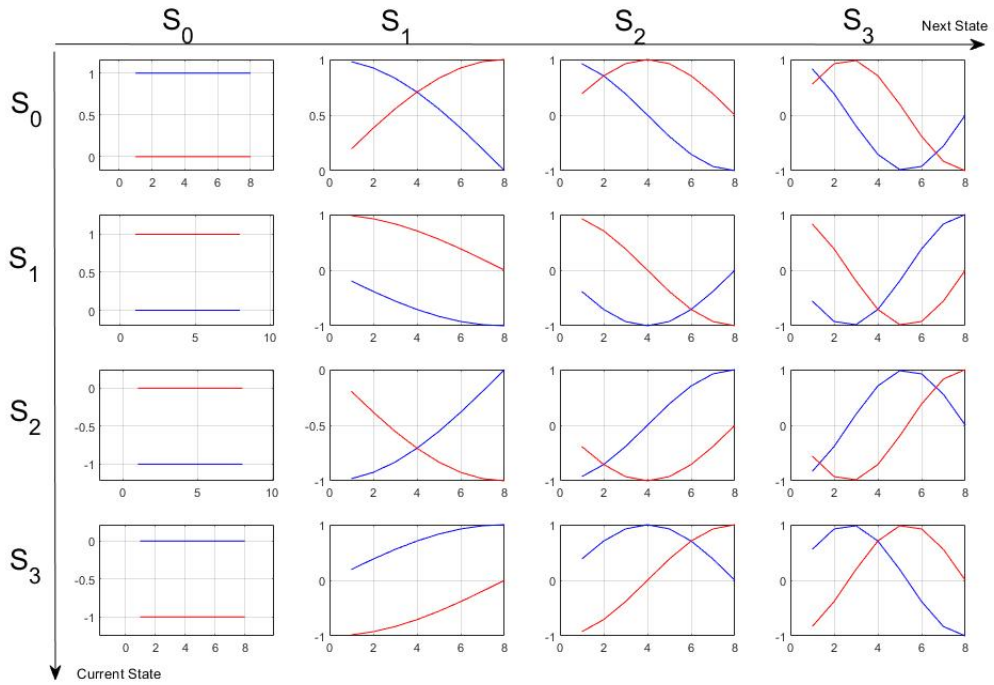

Figure 2.7: All possible $\cos\Theta$ and $\sin\Theta$ for 4-ary CPFSK with $h = 1/4$

survival path and other paths merged with the survival path at the same node in the trellis are discarded. Observe the phase trajectory diagram of CPM (e.g., Fig. 2.2), suppose that the initial state is undetermined, the numerator of the modulation index h is odd, $L = 1$, P is the denominator of the modulation index h. It can be noticed that there are PM^{k-1} possible paths at stage $k-1$ and there are M new branches generated from each possible state node of stage $k-1$ to stage k, then there are PM^k possible paths at the k-th stage. It requires a large storage to store all of these possible paths as the depth of decoding increases. By using the Viterbi decoder, in the "Select" step, the decoder will only store one survival path with optimal metric for each possible state at the current stage and then discard other paths. With this method, the number of survival paths at each stage of trellis is always equal to number of possible states at each stage which is PM^{L-1}. Another example is in the Fig. 2.5, at stage $k = 5$, there are 2^5 possible paths in the trellis, however, in the Viterbi decoder, the number of survival paths at the stages $k > 2$ is equal to 4. This method implemented in the VA reduces the memory cost significantly.

2.3 Error Performance of CPM Signal

To evaluate the error performance of the MLSE for CPM signal, a common method is to find the minimum Euclidean distance of paths through the trellis where the paths start at the same node and re-merge after several symbol intervals [2, 3, 25, 26]. The length of the symbol intervals is denoted as N and is referred to as observation length. It should be noted that the initial state of the trellis is zero. Consider two signals $s_i(t)$ and $s_j(t)$ corresponding to two paths in a CPM trellis $\phi(t; \mathbf{u}_i)$ and $\phi(t; \mathbf{u}_j)$. The sequence $\mathbf{u}_i$ and $\mathbf{u}_j$ are different at the first symbol, that is, they separate at the first symbol interval from state 0 and will re-merge at N symbol intervals and then they are coincident after N symbol intervals. Using the definition of Euclidean distance (2.30), the Euclidean distance between these two signals over interval of length NT, where T is the symbol duration time, can be calculated by:

$$\begin{aligned} d_{i,j}^2 &= \int_0^{NT} [s_i(t) - s_j(t)]^2 dt \\ &= \int_0^{NT} s_i^2(t)dt + \int_0^{NT} s_j^2(t)dt + 2\int_0^{NT} s_i(t)s_j(t)dt \\ &= NE_s - \frac{2E_s}{T}\int_0^{NT} \{\cos[4\pi f_c t + \phi(t; \mathbf{u}_i) + \phi(t; \mathbf{u}_j)] + \cos[\phi(t, \mathbf{u}_i) - \phi(t; \mathbf{u}_j)]\}dt \quad (2.44) \\ &= \frac{2E_s}{T}\int_0^{NT} \{1 - \cos[\phi(t, \mathbf{u}_i) - \phi(t; \mathbf{u}_j)]\}dt \\ &= \frac{2E_s}{T}\int_0^{NT} \{1 - \cos[\phi(t, \mathbf{u}_i - \mathbf{u}_j)]\}dt \end{aligned}$$

where E_s is the symbol energy and $E_s = E_b \log_2 M$ where E_b is the bit energy. It can be noticed that the Euclidean distance of the two paths is related to the difference of phases. Let $\boldsymbol{\xi} = \mathbf{u}_i - \mathbf{u}_j$, the $\phi(t, \boldsymbol{\xi})$ is referred to as the phase distance of the two signal over the observation interval NT. Then we define the normalized Euclidean distance:

$$\delta_{i,j}^2 = \frac{\log_2 M}{T}\int_0^{NT} [1 - \cos\phi(t, \boldsymbol{\xi})]dt \quad (2.45)$$

and

$$d_{i,j}^2 = 2E_b\delta_{i,j}^2$$

where $\boldsymbol{\xi}$ can take value from $\{0,\ \pm 2,\ \pm 4,\ \pm(M-1)\}$, but $\xi_o \neq 0$. The error performance for CPM is given by [25]:

$$P_e = K_\delta Q\left(\frac{E_b}{N_0}\delta_{min}^2\right) \tag{2.46}$$

where δ_{min}^2 is the minimum Euclidean distance and K_δ is the number of paths have the minimum distance. The minimum Euclidean distance can be find by:

$$\delta_{\min}^2 = \min_{\substack{i,j \\ i\neq j}} \delta_{i,j}^2 \tag{2.47}$$

For the M-ary 1REC CPM(i.e., M-ary CPFSK), the upper bound of the $\delta_{\min}$ can be obtained

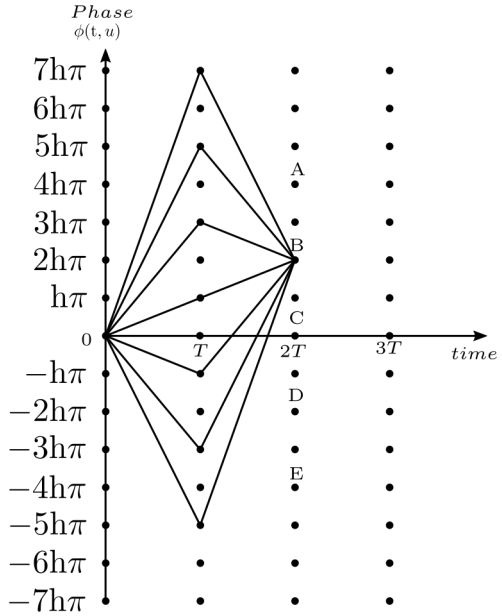

Figure 2.8: Example of 8-ary CPFSK for calculating the minimum Euclidean distance

by letting the two signal $\mathbf{u}_i$ and $\mathbf{u}_j$ separate at $t = T$ and merge at $t = 2T$, by observing the Fig. 2.1, we can notice that $\boldsymbol{\xi} = [\beta, -\beta, 0, \dots]$, where $\beta = \pm 2, \dots, \pm 2(M-1)$, and the phase distance at $t = T$ of any possible path pair that merge at points A, B, C, D, E can be written

as $2k\pi h,\ k = 1,\ 2,\ \ldots$. Fig. 2.8 gives an example of all possible path pairs that separate at $t = T$ and re-merge at $t = 2T$ at point B. Using (2.47) and calculating the integral in (2.45), then the upper bound of δ^2_{min} for M-CPFSK is given by:

$$d_B^2 = \min_{1 \le k \le M-1} \left\{ 2\log_2 M \left(1 - \frac{\sin 2k\pi h}{2k\pi h} \right) \right\} \tag{2.48}$$

Fig. 2.9 displays the upper bound for 4-ary CPFSK (dashed line) and the minimum Euclidean distance $\delta^2_{\min}$ with observation length $N = 1,\ 2,\ 3,\ 10$. Fig. 2.10 shows the upper bound for 8-ary CPFSK (dashed line) and the $\delta^2_{\min}$ with different modulation indexes, we can find that the larger observation interval N can make more values of h reach the upper bound, however, some values of h can never approach the upper bound even though the N is large enough (e.g., $h = 0.5,\ 0.8,\ 1$, etc.) for 8-ary CPFSK, these values of h are called *weak modulation indices* and they should be avoided when designing a CPM system. Fig. 2.11 shows the simulation results for 8-ary CPFSK error performance with different h through an AWGN channel with $E_b/N_0 = 0$, the error performance agrees with the result in Fig. 2.10, that is, the larger minimum Euclidean distance can provide more gain in error performance for MLSE detector. Fig. 2.12 shows the impact of modulation index for signal power spectrum. A simple conclusion is that increasing the value of h will increase the width of power spectrum, i.e., increasing more bandwidth. Hence, a tradeoff between the error performance and spectrum efficiency needs to be considered when selecting the modulation index h and alphabet size M to design the CPM system.

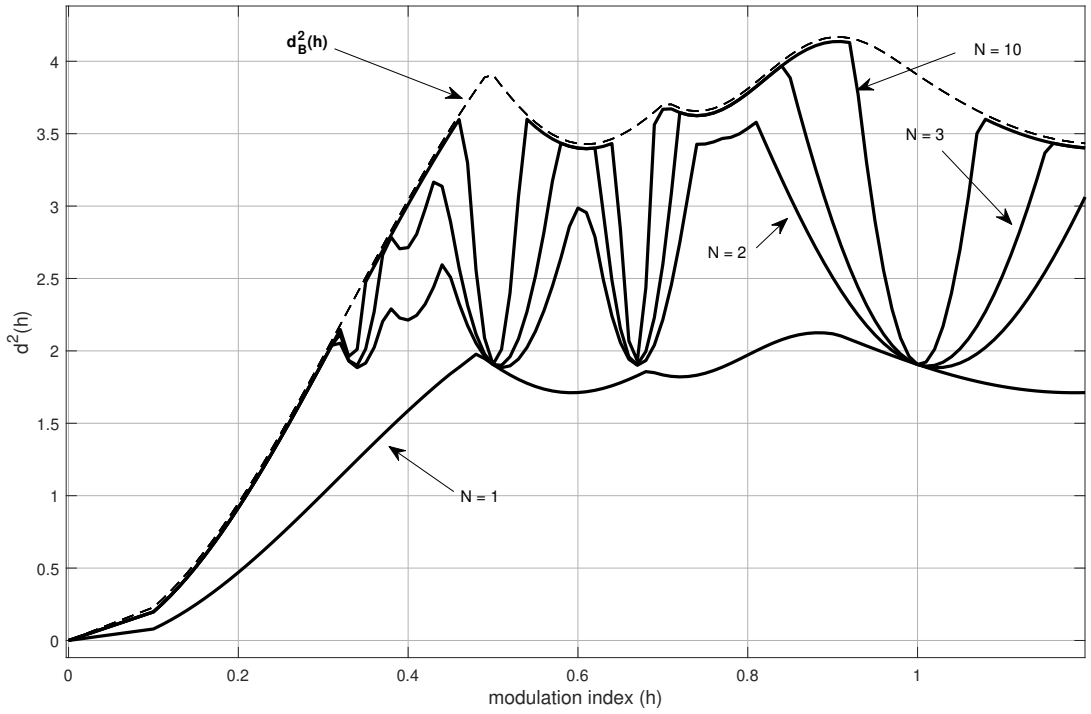

Figure 2.9: Square minimum Euclidean distance for 4-ary CPFSK with different h

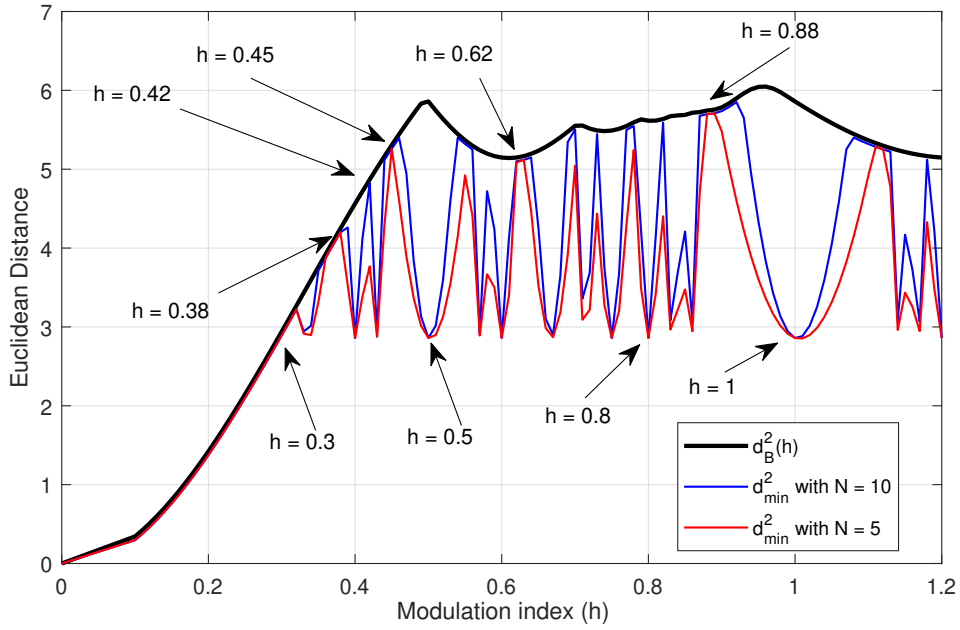

Figure 2.10: Square minimum Euclidean distance for 8-ary CPFSK with different h

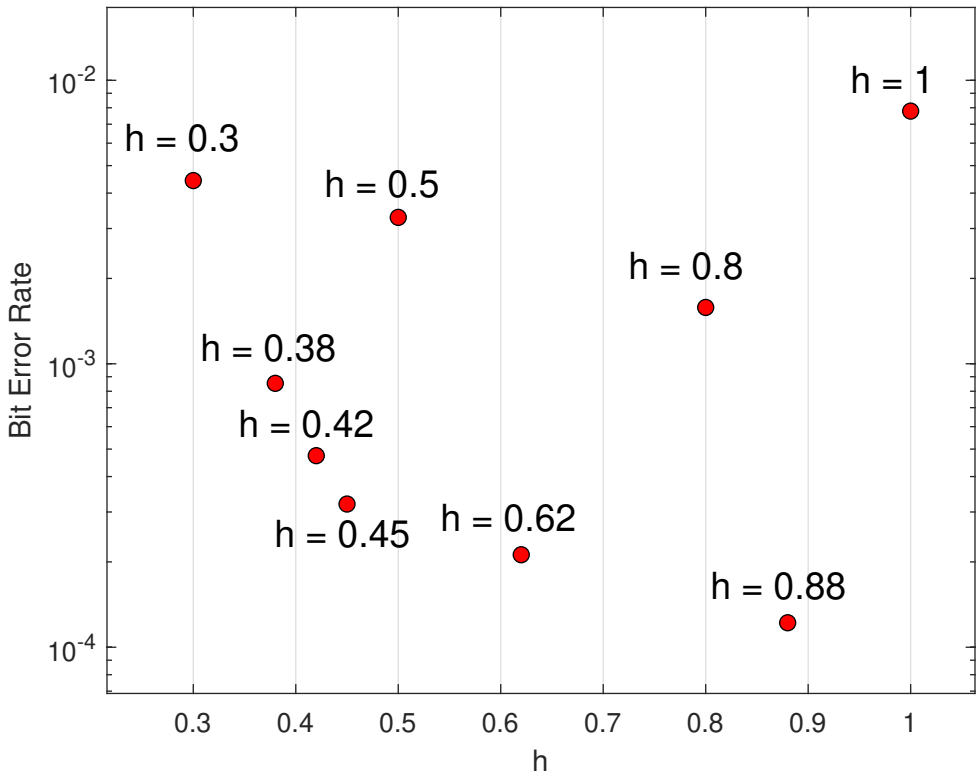

Figure 2.11: Impact of h for error performance of 8-ary CPFSK

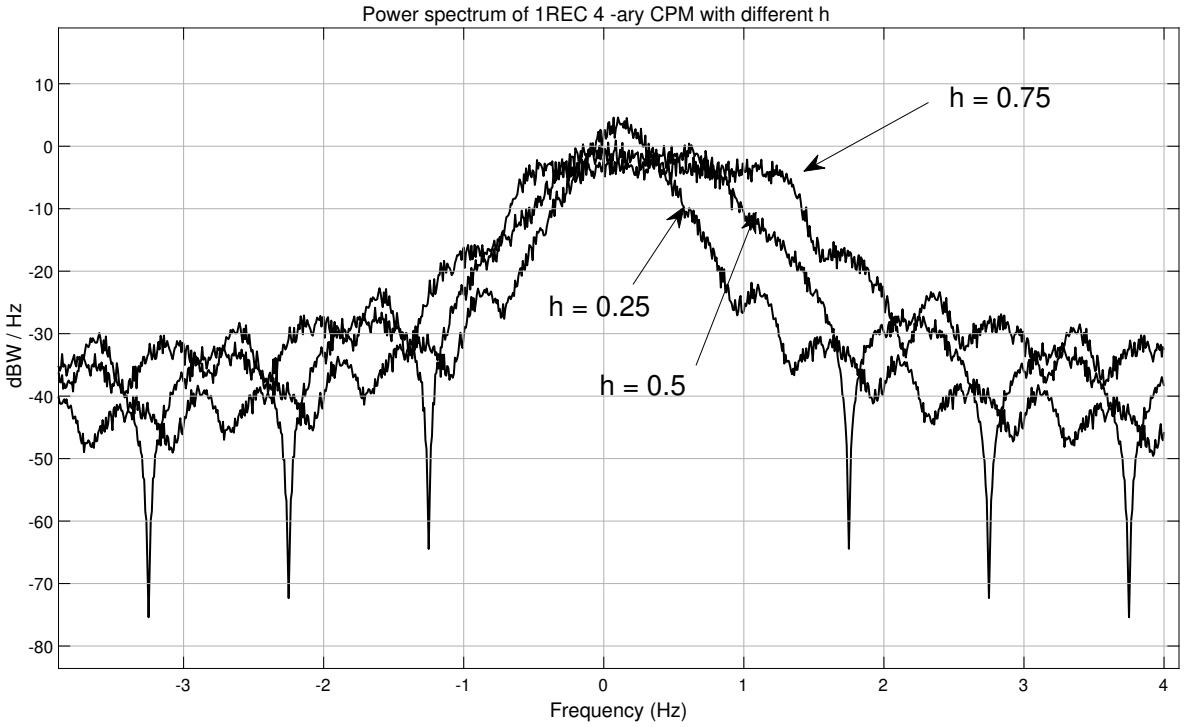

Figure 2.12: Power spectrum for 4-ary CPFSK with different h

CHAPTER 3 **SCCPM**

In this chapter, we design an SCCPM system with turbo decoding using SOVA as soft input soft output decoder. We have described the Viterbi algorithm in Chapter 2; in the SCCPM system, we use a modified Viterbi algorithm, i.e., SOVA, which can generate and update soft information to improve the error performance. We detail the mathematical description of SOVA and how to implement SOVA for CC and CPM in the designed SCCPM system. Finally, we show the simulation results.

3.1 SCCPM System Description

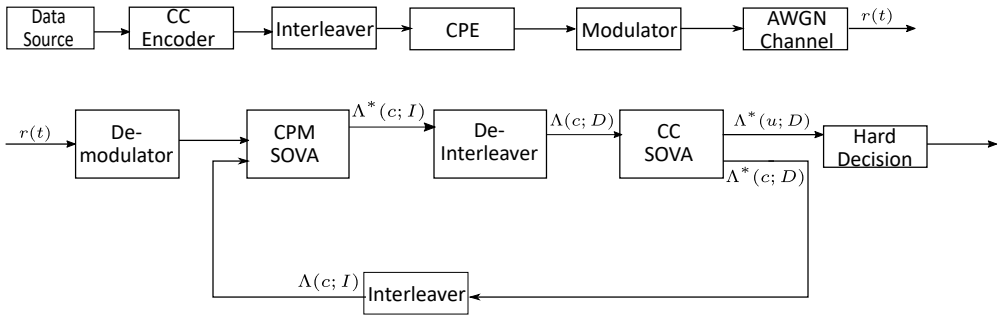

Figure 3.1: Block diagram of SCCPM with SOVA turbo decoding

Fig. 3.1 shows the block diagram of SOVA iterative decoding for SCCPM in an AWGN channel. The CC has a rate of 1/2. There is a pair of bit-wise pseudo-random interleaver and deinterleaver applied between CC and CPM encoder/decoder. Assume that the encoder will reset the state to 0 when each data frame is transmitted. We can find that the CC encoder/decoder and the CPE encoder/decoder are cascaded, such a structure is called serially concatenated code. The CC is called outer code and the CPE is called inner code of the concatenated coding system.

The iterative decoding uses soft-output Viterbi algorithm as the soft-input soft-output

(SISO) algorithm for both CC and CPM. Viterbi algorithm (VA) is the Maximum-Likelihood Sequence Estimate (MLSE) algorithm which finds the most likely path sequence in a finite-state Markov chain [20, 27]. The SOVA further gives the reliability information for each decoded symbol. It has been shown that the SOVA can provide 1-4 dB additional gains for the concatenated codes using hard decoding only [20].

In Fig. 3.1, the soft input and output of SOVA blocks are denoted by $\Lambda(\cdot\ ;\ \cdot)$ and $\Lambda^*(\cdot\ ;\ \cdot)$, $\Lambda(\cdot\ ;\ I)$ and $\Lambda(\cdot\ ;\ D)$ denote interleaved and deinterleaved sequences respectively. The $\Lambda(c\ ;\ \cdot)$ denotes the soft information of CC coded bit sequence which includes both information bits and parity bits, and $\Lambda(u\ ;\ \cdot)$ denotes the soft information of VA decoded bit sequence which only includes the information bits. The soft output of SOVA $\Lambda^*(\cdot\ ;\ \cdot)$ gives the LLR of selecting a correct branch for decoding bit u_k at stage k at the state on the survival branch [20]. The soft information from CPM SOVA will be the *a-priori* information for the CC SOVA decoder. The soft information outputs from CC SOVA will be fed back to the CPM SOVA decoder as the *a-priori* information, so that the CPM SOVA will use it to help improve decoding accuracy in the next round of decoding. This procedure is called iterative decoding and it continuous until no more decoding improvement is obtained.

3.2 Convolutional Code

In the designed SCCPM system, a convolutional code (CC) is used as the outer encoder. In this section, the basic idea of CC is presented.

A convolutional code is an error-correcting code with memory properties. A convolutional code can be generated by passing the information bits (i.e., the bit sequence that needs to be encoded) through a linear finite-state shift register. We can use a trellis and state diagram(Markov chain) to describe a CC. A CC is commonly defined by the constrained length K, the number of input bits k, and the number of output bits n. Then a CC can be specified by(K, k, n). The code rate is defined by:

$$R_c = k/n \tag{3.1}$$

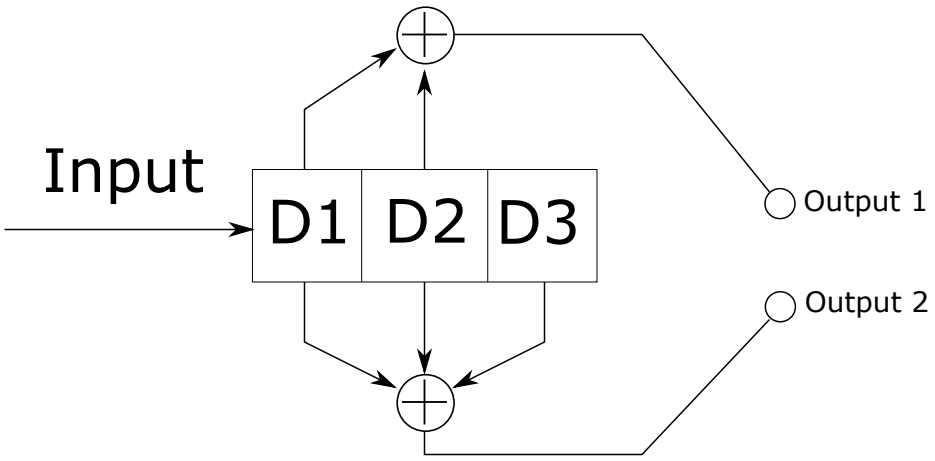

Figure 3.2: $K = 3$, $k = 1$, $n = 2$ convolutional encoder

Fig. 3.2 is the encoder structure of a (3,1,2) convolutional code. We can use a generator matrix to describe this CC encoder. Observe the encoder, it can be noted that it is a CC with 1/2 code rate, that is, one information bit will generate two encoded bits. The state of three registers are all zeros initially. We can see that for each information bit, the first output bit depends on the first and the second shift registers (from left to right), the second output bit depends on all of three registers. We denote the register as "1" if this register is used by output bit, and denote it as "0" if this register is not used by the output bit. Then we can get the generators for the two outputs as:

$$g_1 = [1\ 0\ 0] \tag{3.2}$$

and

$$g_2 = [1\ 1\ 1] \tag{3.3}$$

g_1 and g_1 together are referred to as the generator of this CC which can be described in octal form as $(6, 7)$. It can be noted that the g_1 and g_1 are the impulse responses of the two outputs. Then the coded sequence $\mathbf{c}$ corresponding to a information bit sequence $\mathbf{u}$ can be

obtained by:

$$\begin{aligned} \mathbf{c_1} &= \mathbf{u} * g_1 \\ \mathbf{c_2} &= \mathbf{u} * g_2 \end{aligned} \tag{3.4}$$

The final coded sequence can be obtained by interleaving:

$$\mathbf{c} = [c_1^{(1)}\ c_2^{(1)},\ c_1^{(2)}\ c_2^{(2)},\ c_1^{(3)}\ c_2^{(3)},\ \dots] \tag{3.5}$$

where $*$ is convolution operator, $c_i^{(j)}$ denotes the i-th output encoded bit corresponding to the j-th information bit, where i is the index of position of output of encoder and j is the index of the position of information bits. A generator matrix form of the CC can also be obtained as:

$$G = \begin{bmatrix} g_1^{(1)}g_2^{(1)} & g_1^{(2)}g_2^{(2)} & g_1^{(3)}g_2^{(3)} & & \\ & g_1^{(1)}g_2^{(1)} & g_1^{(2)}g_2^{(2)} & g_1^{(3)}g_2^{(3)} & \\ & & g_1^{(1)}g_2^{(1)} & g_1^{(2)}g_2^{(2)} & g_1^{(3)}g_2^{(3)} \\ & & \ddots & \ddots & \ddots \end{bmatrix} \tag{3.6}$$

where $g_j^{(i)}$ denote the i-th element (from left to right) of the j-th generator, and the blank areas are all zeros. For example, for the CC with generator (6,7), we can get the generator matrix G:

$$G = \begin{bmatrix} 11 & 11 & 01 & & \\ & 11 & 11 & 01 & \\ & & 11 & 11 & 01 \\ & & \ddots & \ddots & \ddots \end{bmatrix} \tag{3.7}$$

then the coded sequence $\mathbf{c}$ corresponding to an information bit sequence $\mathbf{u}$ is:

$$\mathbf{c} = \mathbf{u}G \tag{3.8}$$

We have known that the convolutional operation is equivalent to multiplication in the transform domain, for example, the convolution in the time domain equals to the multipli-

cation in the frequency domain. In the coding area, the "D transform" is commonly used. The D transform of an information bit sequence $\mathbf{u}$ is defined by:

$$\mathbf{u}(D) \triangleq \sum_{k=0}^{\infty} u_k D^k \tag{3.9}$$

for the CC with the generator (6,7), the transform function of the two impulse response $g_1 = [1\ 0\ 0]$ and $g_2 = [1\ 1\ 1]$ are:

$$\begin{aligned} g_1(D) &= 1 + D \\ g_2(D) &= 1 + D + D^2 \end{aligned} \tag{3.10}$$

and the transform of the encoder output is:

$$\mathbf{c}(D) = c^{(1)}(D^2) + Dc^{(2)}(D^2) \tag{3.11}$$

For example, if the information bit sequence $\mathbf{u} = [100101]$ is the input of the convolutional encoder with generator (6,7), we can have:

$$\begin{aligned} \mathbf{u}(D) &= 1 + D^3 + D^5 \\ c^{(1)}(D) &= (1 + D^3 + D^5)(1 + D) \\ &= 1 + D + D^3 + D^4 + D^5 + D^6 \\ c^{(2)}(D) &= (1 + D^3 + D^5)(1 + D + D^2) \\ &= 1 + D + D^2 + D^3 + D^4 + D^6 + D^7 \end{aligned}$$

and

$$\begin{aligned} \mathbf{c}(D) &= c^{(1)}(D^2) + Dc^{(2)}(D^2) \\ &= 1 + D + D^2 + D^3 + D^5 + D^6 + D^7 + D^8 + D^9 + D^{10} + D^{13} + D^{15} \end{aligned}$$

Table 3.1: State transition and output for (6,7) CC encoder

Current State	Next state and output when input bit = $0(-1)$	Next state and output when input bit = $+1$
S_0	$S_0 \mid 00$	$S_2 \mid 11$
S_1	$S_0 \mid 01$	$S_2 \mid 10$
S_2	$S_1 \mid 11$	$S_3 \mid 00$
S_3	$S_1 \mid 10$	$S_3 \mid 01$

Therefore the encoder output is:

$$\mathbf{c} = [1111011111100101]$$

Fig. 2.5 is the trellis of a (2,1,3) convolutional code. The solid lines and the dashed lines denote information bits -1 and $+1$ respectively, Table 3.1 shows the state transition of the CC and output corresponding to the transition. To encode the information sequence $\mathbf{u} = [010011]$, the output of the encoder is $\mathbf{c} = [00,\ 11,\ 11,\ 01,\ 11,\ 00]$. To decode the $\mathbf{c}$, we can use the trellis and implement the VA. In the VA for CC, when hard information, i.e., $+1$ or -1, is considered as the input, the Hamming distance is commonly used to get the branch metric, Hamming distance is the number of different elements in the corresponding positions of two equal length sequences/strings. In the CC decoding, the hamming distance of a state transition (branch) is the difference between the encoding result corresponding to the state transition and the received bit string at this transition. After searching all possible paths through the trellis, finding out the path with a minimum Hamming distance and selecting it as the survival path. A trace back process can be used through the trellis to get the final decoding result.

3.3 A Soft-Output Viterbi Algorithm

3.3.1 Log-likelihood Ratio

Throughout this chapter, we use u_k to denote the k-th binary symbol. The soft output from both components decoders are in term of Log-likelihood Ratio (LLR) which is a natural logarithm of two probability values. The LLR for the value of one decoded bit u_k is given by:

$$L(u_k) = \ln\left(\frac{P(u_k = +1)}{P(u_k = -1)}\right) \tag{3.12}$$

where $P(u_k = +1)$ and $P(u_k = -1)$ denote the probabilities of $u_k = +1$ and $u_k = -1$ respectively. At a receiver, the LLR for the value of the decoded bit u_k given the observed signal sequence $\mathbf{r}$ is represented by:

$$L(u_k \mid \mathbf{r}) = \ln\left(\frac{P(u_k = +1 \mid \mathbf{r})}{P(u_k = -1 \mid \mathbf{r})}\right) \tag{3.13}$$

which is called *a posterior* LLR of decoded bit u_k. By observing the Fig. 3.3, it can be

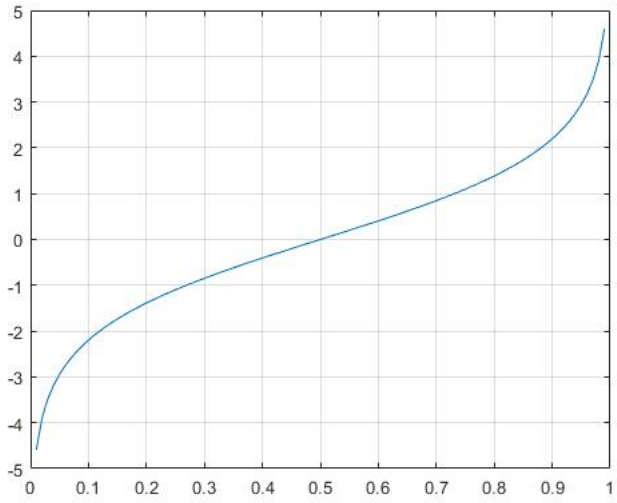

Figure 3.3: LLR for $P(u_k = +1)$

noticed that the sign of the LLR gives the estimated bit. By abuse of notation, the $L(u_k \mid \mathbf{r})$ is denoted as $L(u_k)$. Using Bayes' theorem and (3.12), (3.13), the *a posterior* LLR can be

rewritten as:

$$\begin{aligned} L(u_k) &= \ln \frac{P(\mathbf{r}, u_k = +1)/P(\mathbf{r})}{P(\mathbf{r}, u_k = -1)/P(\mathbf{r})} \\ &= \ln \frac{P(\mathbf{r}, u_k = +1)}{P(\mathbf{r}, u_k = -1)} \end{aligned} \tag{3.14}$$

Then the probability $P(u_k)$ can be written as:

$$\begin{aligned} P(u_k = \pm 1) =& \frac{e^{L(u_k)/2}}{1 + e^{-L(u_k)}} e^{\pm L(u_k)/2} \\ =& A_k \cdot e^{u_k \cdot L(u_k)/2} \end{aligned} \tag{3.15}$$

where A_k is a common factor for both $P(u_k = +1)$ and $P(u_k = -1)$:

$$A_k = \frac{e^{L(u_k)/2}}{1 + e^{-L(u_k)}} \tag{3.16}$$

and the nature logarithm is

$$\ln P(u_k) = \ln A_k + \frac{1}{2} \cdot u_k \cdot L(u_k) \tag{3.17}$$

3.3.2 Mathematical Description of the SOVA

Consider a state sequence $\mathbf{s_s^k}$ which is a survival path along the trellis from the starting stage to stage k, and state at stage k is $S^k = s$. Let $\mathbf{y}$ be the observed sequence, then the *a posterior* probability of the sequence given $\mathbf{y}$ is given by [10, 24]:

$$P(\mathbf{s_s^k} \mid \mathbf{y}) = \frac{P(\mathbf{s_s^k}, \mathbf{y})}{P(\mathbf{y})} \tag{3.18}$$

It can be noticed that the *a posterior* probability is proportional to $P(\mathbf{s_s^k}, \mathbf{y})$. Therefor, the metric $P(\mathbf{s_s^k}, \mathbf{y})$ should be maximized to obtain the optimal result. The metric can be calculated into a form of recursive, that is, it can be obtained by multiplying previous stage metric by the probability of transition from the state at stage $k-1$ to the state at stage k.

Let $\gamma(s', s)$ denote the probability of transition from one state to next state along the trellis:

$$\begin{aligned}\gamma(s', s) \triangleq & P(\mathbf{y}, S^k = s \mid S^{k-1} = s') \\ = & P(\mathbf{y}, s \mid s')\end{aligned} \tag{3.19}$$

Then the metric $P(\mathbf{s_s^k}, \mathbf{y})$ is:

$$\begin{aligned}P(\mathbf{s_s^k}, \mathbf{y}) = & P(\mathbf{s_{s'}^{k-1}}) P(s \mid s') \\ = & P(\mathbf{s_{s'}^{k-1}}) \gamma(s', s)\end{aligned} \tag{3.20}$$

The path metric for the survival path $\mathbf{s_s^k}$ is given by:

$$\begin{aligned}M_p(\mathbf{s_s^k}) \triangleq & \ln\left(P\left(\mathbf{s_s^k}, \mathbf{y}\right)\right) \\ = & M_p(\mathbf{s_{s'}^{k-1}}) + \ln\left(\gamma(s', s)\right)\end{aligned} \tag{3.21}$$

It can be noticed that the $\ln(\gamma(s', s))$ is the branch metric from state s' to state s during the internal $[k-1, k]$. The $\gamma(s', s)$ is calculated by using the definition (3.19) and using Bayes' rule:

$$\begin{aligned}\gamma_k(s', s) = & P\left(\mathbf{y}_k, s \mid s'\right) \\ = & P\left(\mathbf{y}_k \mid s, s'\right) P\left(s \mid s'\right) \\ = & P\left(\mathbf{y}_k \mid x_k\right) P\left(s \mid s'\right) \\ = & P\left(\mathbf{y}_k \mid x_k\right) P(u_k)\end{aligned} \tag{3.22}$$

where u_k is the symbol input the trellis during the interval $(k-1, k)$ that causes the state transition from s' to s, and $\mathbf{x}_k$ is the output of transition from state s' to s along the trellis, which is an M-ary symbol and $M = 2^n$. Then $\mathbf{y}_k$ is an M-ary symbol. We write $\mathbf{x}_k$ and $\mathbf{y}_k$ as sequences. In an AWGN channel, the probability $P\left(\mathbf{y}_k \mid \mathbf{x}_k\right)$ is given by:

$$P\left(\mathbf{y}_k \mid \mathbf{x}_k\right) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{1}{2\sigma^2} D(\mathbf{y}_k, \mathbf{x}_k)} \tag{3.23}$$

Then using (2.39), the nature logarithm of $P\left(\mathbf{y}_k \mid \mathbf{x}_k\right)$ is represented as:

$$\begin{aligned}
\ln\left(P\left(\mathbf{y}_k \mid \mathbf{x}_k\right)\right) =& n \cdot \ln(\frac{1}{\sqrt{2\pi\sigma^2}}) - \frac{1}{2\sigma^2} \cdot D(\mathbf{y}_k, \mathbf{x}_k) \\
=& n \cdot \ln(\frac{1}{\sqrt{2\pi\sigma^2}}) - \frac{1}{2\sigma^2} \cdot \left\{ \sum_{i=1}^{n} \mathbf{y}_i^2 + \sum_{i=1}^{n} \mathbf{x}_i^2 - 2\sum_{i=1}^{n} \mathbf{y}_i \cdot \mathbf{x}_i \right\} \\
=& n \cdot \ln(\frac{1}{\sqrt{2\pi\sigma^2}}) - \frac{1}{2\sigma^2} \cdot \left\{ \sum_{i=1}^{n} \mathbf{y}_i^2 + \sum_{i=1}^{n} \mathbf{x}_i^2 \right\} + \frac{1}{2\sigma^2} \cdot 2\sum_{i=1}^{n} \mathbf{y}_i \cdot \mathbf{x}_i \\
=& C_k + \frac{1}{2} L_c \sum_{i=1}^{n} \mathbf{y}_i \cdot \mathbf{x}_i
\end{aligned} \tag{3.24}$$

where

$$C_k = n \cdot \ln(\frac{1}{\sqrt{2\pi\sigma^2}}) - \frac{1}{2\sigma^2} \cdot \left\{ \sum_{i=1}^{n} \mathbf{y}_i^2 + \sum_{i=1}^{n} \mathbf{x}_i^2 \right\} \tag{3.25}$$

is independent on u_k or x_k so that it can be considered as a constant and omitted. L_c is referred as channel reliability measure and is given by [10, 23]:

$$L_c = 2a/\sigma^2 \tag{3.26}$$

where a is fading amplitude of channel and $a = 1$ when AWGN is assumed.

Substitute (3.17) and (3.24) into (3.22) and the nature logarithm of the branch metric $\gamma_k(s', s)$ is given by:

$$\ln \gamma_k(s;, s) = \frac{1}{2} L_c \sum_{i=1}^{n} \mathbf{y}_i \cdot \mathbf{x}_i + \frac{1}{2} \cdot u_k \cdot L(u_k) \tag{3.27}$$

where the constant term C_k and the common factor A_k are omitted because they do not affect the value of the metric. Then substitute (3.26) and (3.27) into (3.21):

$$\begin{aligned}
M_p(\mathbf{s}_s^k) =& M_p(\mathbf{s}_{s'}^{k-1}) + \ln\left(\gamma(s', s)\right) \\
=& M_p(\mathbf{s}_{s'}^{k-1}) + \frac{1}{2} L_c \sum_{i=1}^{n} \mathbf{y}_i \cdot \mathbf{x}_i + \frac{1}{2} \cdot u_k \cdot L(u_k)
\end{aligned} \tag{3.28}$$

which is the recursive equation for path metric of the Soft Output Viterbi algorithm. Comparing (3.28) with the conventional Viterbi algorithm path metric equation (2.43), the branch metric is modified by introducing *a-prior* information $u_k L(u_k)$ which is called soft information.

Now the calculation of soft information is discussed. In the SOVA algorithm, at stage k of the trellis, when the survival path $\mathbf{s}_s^k$ enter the state $S^k = s$, there is another path $\mathbf{s}_s^{k'}$ that merges into the same state node gets discarded by "Select" step in VA due to its path metric less than the survival path. After calculating the path metrics for both paths, the path metric difference between the survival path and the discarded path, which is denoted as Δ_s^k, is calculated and stored, that is:

$$\Delta_k^s = M_p(\mathbf{s}_s^k) - M_p(\mathbf{s}_s^{k'}) \geq 0 \tag{3.29}$$

The probability of the survival path is correctly selected is given by:

$$\begin{aligned} &P(\text{the selection of survival path is correct at stage k}) \\ =&P(\text{correct}) \\ =&\frac{P(\mathbf{s}_s^k)}{P(\mathbf{s}_s^k) + P(\mathbf{s}_s^{k'})} \end{aligned} \tag{3.30}$$

Using the definition of path metric (3.21), (3.30) can be rewritten as:

$$\begin{aligned} P(\text{correct}) =&\frac{e^{M_p(\mathbf{s}_s^k)}}{M_p(\mathbf{s}_s^k) + M_p(\mathbf{s}_s^{k'})} \\ =&\frac{e^{\Delta_s^k}}{e^{\Delta_s^k} + 1} \end{aligned} \tag{3.31}$$

then the LLR of selecting the correct survival path at stage k is given by:

$$L(u_k) = \frac{P(\text{correct})}{1 - P(\text{correct})} = e^{\Delta_s^k} \tag{3.32}$$

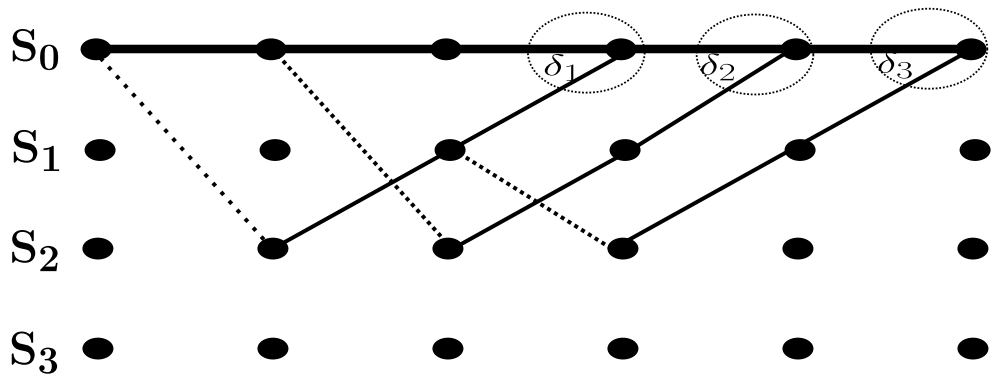

Figure 3.4: Example of survival path with metric differences for a CC trellis

where u_k is the decoded result corresponding to the survival path $\mathbf{s}_s^k$ at stage k. It can be noted that the reliability of selecting a correct path is simply given by Δ_s^k. It has been shown [20] that at stage l, all of the surviving paths in the trellis will have the same starting state node at stage $l-\sigma$, where usually the σ is at most five times of the constraint length of the convolutional code. However, at the interval from stage $k=l-\sigma$ to stage $k=l$, the decoding bit u_k may be affected by which survival path is selected. Therefore, the probability of selecting the incorrect survival path from stage $l-\sigma$ to stage l has to be considered, i.e., the difference of metrics for all states along the survival path from stage $l-\sigma$ to l have to be taken into account. Fig 3.4 is an example of a trellis with metric differences. Throughout this paper, a solid line denotes a decoding bit -1 (i.e., 0) and a dashed line denotes a decoding bit $+1$, and the heave line denotes the survival path. Along the survival path, for each node with metric difference, there is a non-surviving path discarded. Comparing the survival path and discarded path that starting from a same state node and merged at one state node again, if the survival path and discarded path have same decoded bits at some stages, there is no bit errors even though the survival path is not selected correctly. However, if some decoded bits at some stages are different between the survival path and the discarded path, the metric differences at these stages give the reliability of selecting the survival path correctly. It is

shown by Hagenauer in [2] that the LLR of selection a correct path can be approximated by:

$$L(u_k \mid \mathbf{y}) \approx \min_{\substack{i=k-\sigma,\ldots,k-1,k \\ u_s^i \neq u_d^i}} \Delta_{s_i}^i \tag{3.33}$$

where u_s^i and u_d^i denote the bits decoded at stage i based on the survival path and the discarded path, respectively.

3.3.3 Implementation of SOVA

In this section, we denote u_k the decoded bit at stage k at state s_k, and $\Delta_{s_k}^k$ the metric difference of state s_k at stage k. The method for implementation of SOVA in this paper is described in [10, 22].

3.3.3.1 A Soft-Output Viterbi Algorithm for Convolutional Code

A simple example of CC decoding by using SOVA is given in Figs. 3.5 and 3.6.

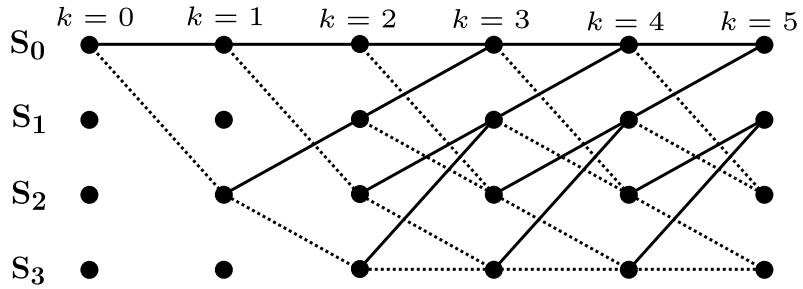

Figure 3.5: Example of the trellis diagram for a (2,1,3) convolutional code

It would be noted that in Figs. 3.5 and 3.6, there are only 5 decoded bits corresponding

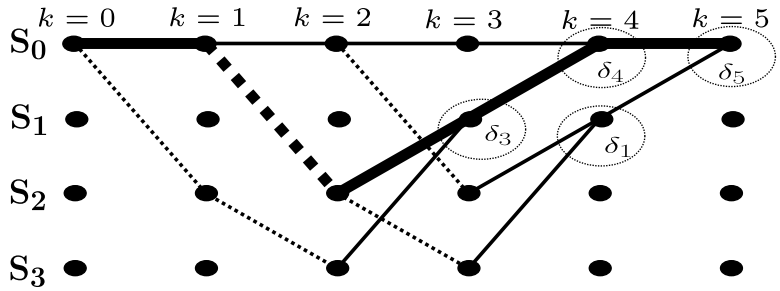

Figure 3.6: Example of survival path with metric differences for a CC trellis

to $k = 1, \ldots, 5$, $k = 0$ corresponds to the initial condition which means starting at S_0. Throughout this paper, a solid line denotes a decoding bit -1 (i.e., 0) and a dashed line denotes a decoding bit $+1$. In Fig. 3.6, the bold line denotes the optimum survival path. The value δ_3 denotes $\Delta_3^{S_1}$, i.e., it denotes the metric difference of survival path and discard path at state node S_1 at stage $k = 3$. In the same way, $\delta_4 = \Delta_4^{S_0}$, $\delta_1 = \Delta_4^{S_1}$, and $\delta_5 = \Delta_5^{S_0}$. Table. 3.2 shows the decoding procedure and decoding results with LLR. At each stage from $k = 3$ to $k = 5$, we only concern the metric differences of survival path, therefore, δ_1 at stage 4 is ignored. The '*Update Sequence*' in Table. 3.2 denotes that at stage k, when tracing back through the trellis to stage $k = 0$, the difference of decoding results between the survival path and the discard path. For example, at stage $k = 3$, the survival path is $[S_0 \rightarrow S_0 \rightarrow S_2 \rightarrow S_1]$ and the discarded path is $[S_0 \rightarrow S_2 \rightarrow S_3 \rightarrow S_1)]$, then the decoding results will be [010] and [110] respectively, the first decoding bit is different, hence, the *Update Sequence* at stage $k = 3$ is [100]. Therefore, the *Update Sequence* $= XOR$(*Decoding Results of two paths*). For convenience, we reverse the order from [100] to [001] for the '*Update Sequence*'. That is, the most left side bit represents the XOR value at stage k and the next bit on the right represents the XOR value at stage k-1 and so on. To obtain the '*Update Sequence*' for δ_5, we assume that the discarded path at stage $k = 5$ is $[S_0 \rightarrow S_0 \rightarrow S_0 \rightarrow S_2 \rightarrow S_1 \rightarrow S_0]$. Therefore the '*Update Sequence*' at stage $k = 5$ is [00110].

We can notice that there is no metric difference Δ for $k = 1$ and $k = 2$ from the trellis diagram Fig. 3.6. we can use the following strategy to get the LLR values of these two stages. To obtain the LLR value for stage $k = 1$, by looking at the Least Significant Bits (LSB) (i.e., the right most bit) of '*Update Sequence*' of stage $k = 3, 4, 5$, we can observe that only stage $k = 3$ has "1" at LSB of the '*Update Sequence*', then the LLR value of bit u_1 is the metric difference of 3rd stage δ_3. To get the LLR value of stage $k = 2$, by looking at the second LSB of '*Update Sequence*' for stage $k = 3, 4, 5$, we can notice that the second LSB of '*Update Sequence*' for stages $k = 4$ and $k = 5$ are "1"s, therefore, the LLR value of bit u_2 is the minimum value of metric differences of $k = 4$ and $k = 5$, i.e., the

minimum value of δ_4 and δ_5. For the LLR values of bits u_3, u_4, u_5, by looking at the '*Update Sequence*', to select a minimum value of metric differences Δ_k from those stages denoted as "1" in the '*Update Sequence*'. For example, for stage $k = 5$, the "1"s in the '*Update Sequence*' denote stage $k = 2$ and $k = 3$, then the LLR value for decoded bit u_5 is the minimum value of LLR value for $k = 2$ (which is denoted as $LLR(k = 2)$ in the Table. 3.2) and δ_3, i.e., $LLR(k = 5) = min[LLR(k = 2), \delta_3]$.

Table 3.2: SOVA output for CC example

Trellis stage k	Decoded Bit u_k	Metric Diff $\Delta_k^{s_k}$	Update Sequence	LLR
1	-1(0)	-	-	δ_3
2	1	-	-	$min(\delta_4,\delta_5)$
3	-1(0)	δ_3	001	$LLR(k = 1)$
4	-1(0)	δ_4	0010	$LLR(k = 2)$
5	-1(0)	δ_5	00110	$min(\delta_3, LLR(k = 2))$

3.3.3.2 A Soft-Output Viterbi Algorithm for CPM

In this section, a simplified SOVA for CPM is introduced [28].

CPM signal can be decomposed into a CPE and an MM. The CPE can be seen as a special CC with a rate of 1. Fig. 3.7 shows the physical tilted-phase trellis diagram for a MSK signal (binary full response 1REC CPM with $h = 1/2$). Fig. 3.8 shows the state transition diagram for 4-ary 1REC CPM with $h = 1/4$. In Fig. 3.8, the solid lines denote '00', the '$-\cdot$' lines denote '01', the '$--$' lines denote '10', and the '$\cdot\cdot$' lines denote '11'.

Fig. 3.9 is an example of trellis decoding of noised MSK with branch metrics. The bold lines are the survival paths, we can see that the phase state starts from state S_0 at stage $k = 0$. Table 3.3 shows the results of metric difference, decoding results, '*Update Sequence*' and LLR just like what we do in Table 3.2.

From the '*Update Sequence*' in Table 3.3, it is observed that the LLR values of decoded bits $u_k, k = 3, 4, 5$ only depend on the previous two stages, that is, at stage l, the surviving paths in the trellis will have a same starting state node at stage $l - 2$. Therefore, we can simplify the SOVA for CPM by storing the metric differences Δ_k at each stage and for u_k,

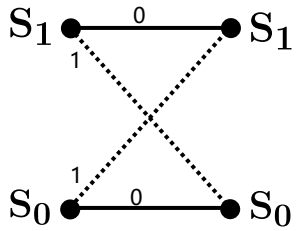

Figure 3.7: State transition diagram of tilted-phase MSK

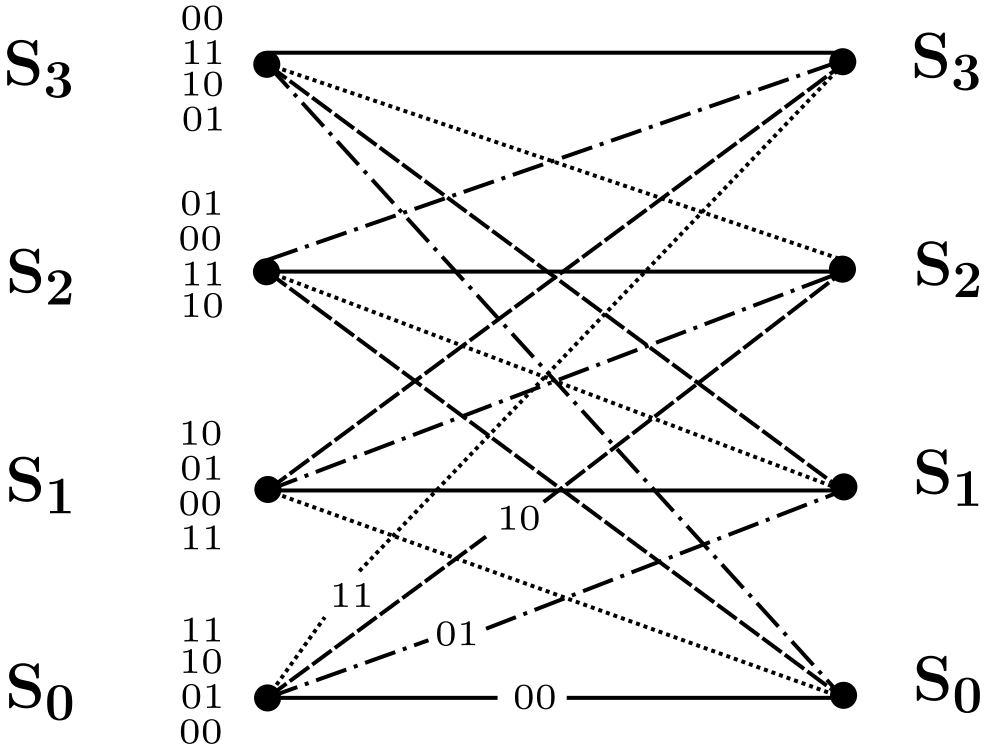

Figure 3.8: State transition diagram of physical tilted-phase 4-ary 1REC CPM with $h = 1/4$

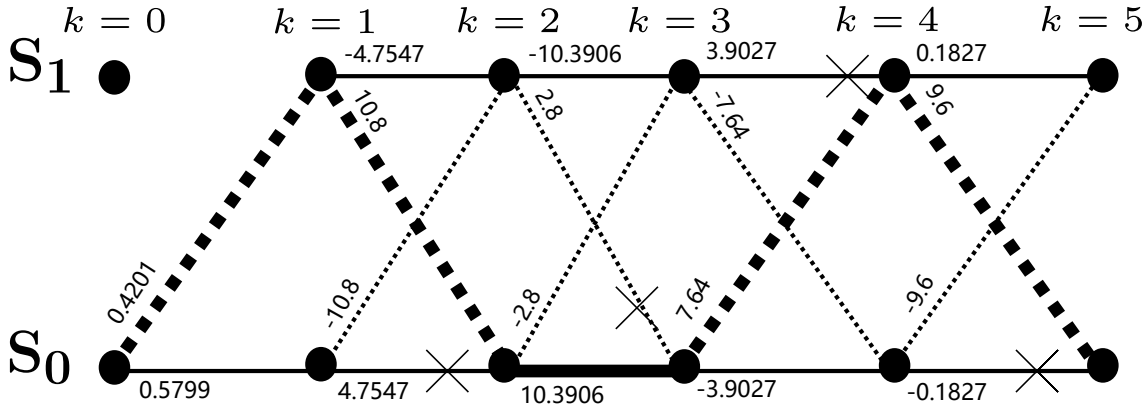

Figure 3.9: Example of SOVA trellis diagram for MSK

Table 3.3: Example of SOVA outputs for MSK

Trellis stage k	Decoded Bit u_k	Metric $\Delta_k^{s_k}$	Update Sequence	LLR
1	1	-	-	5.9155
2	1	5.9155	11	5.9155
3	-1(0)	23.1753	110	5.9155
4	1	16.9279	1100	16.9279
5	1	20.42	11000	16.9279

where $k \geq 2$, then the LLR for u_k will be the minimum value of Δ_k and Δ_{k-1}. Therefore, we can simply use $\sigma = 2$ in this example. To get the LLR value for stage $k = 1$, we use the method discussed in the CC SOVA example. Observing the '*Update Sequence*' in Table 3.3, the LLR value for $k = 1$ is the metric difference value of $k = 2$.

3.3.4 Implement SOVA in SCCPM with iterative decoding

A modified metric calculation method has been given by (3.28) which allows to take advantages of the *a-prior* information provided by SOVA decoder of the other component [10, 20, 23]. We rewrite it and expand the binary to M-ary symbol situation:

$$M_p(s_k^n) = M_p(s_{k-1}^{\hat{n}}) + \frac{L_c}{2} M_b(s_k^n) + \frac{1}{2} c_k L(c_k) \tag{3.34}$$

where $L(c_k)$ is the *a-prior* information of CC coded symbols c_k. The CC coded symbol c_k is a signed M-ary value, where $M = 2^m$, m is a positive integer. For example, for a 4-ary codeword, $c_k \in \{-1-1, -1+1, +1-1, +1+1\}$. The sequence of c_k will be updated in the CC SOVA decoder by storing the input symbols for each survival branches from the state transition diagram during the traceback process of VA. This step may correct errors of hard decoding results of CPM SOVA Decoder. In next Section, some simulation examples will be presented.

3.4 Simulation Result

In this experiment, the inner code is a 4-ary 1REC CPE with $h = 1/4$ and the outer code is a $(2, 1, 3)$ binary CC. Each CC data frame contains N bits data. Therefore, there are $N/2$ bits of source data per coded data frame. A pair of N-bit length bit-wise pseudo-random interleaver and deinterleaver are placed between the coding and decoding of the inner code and the outer code. We use Bit Error Rate (BER) as the performance measure. We simulate the error performance of this SCCPM system with different length of N and with different number of iterations.

1. *Comparison between CC coded CPM and uncoded CPM.* Fig. 3.10 shows that the CPM concatenated with a CC without using interative decoding can provide a $1.5 - 2$ dB coding gain compare with using the CPM scheme along.

2. *The effect of number of iterations.* Fig. 3.11 shows the impact of the number of iterations on the BER performance of the SCCPM. We can observe that the first iteration can improve the BER performance significantly (about 1 dB) and after the third iteration, the improvement is slight. Therefore, in this designed SCCPM, a 3 times of iterative decoding is enough to get the optimum decoding result.

3. *The effect of size of N.* A greater N can improve the BER performance of the system about $0.2 - 1$ dB (See Fig. 3.10, 3.12 and 3.13), which is due to the interleaving gain associated with a large N [11]. From Fig. 3.13, we can notice that the BER performance of $N = 2000$ without iterative decoding is close to the BER performance of $N = 400$ with 3 times iterative decoding when $E_b/N_0 = 5$. From Fig. 3.12, we can observe that the error performance of $N = 4000$ and $N = 2000$ without iterative decoding are close to $N = 2000$ and $N = 1000$ with 3 times of iterative decoding respectively when $E_b/N_0 = 5$dB. It would be noted that the decoding for a long data frame will introduce more complexity and more hardware memories. Hence, with same BER performance, the iteration decoding for a small N may be preferred.

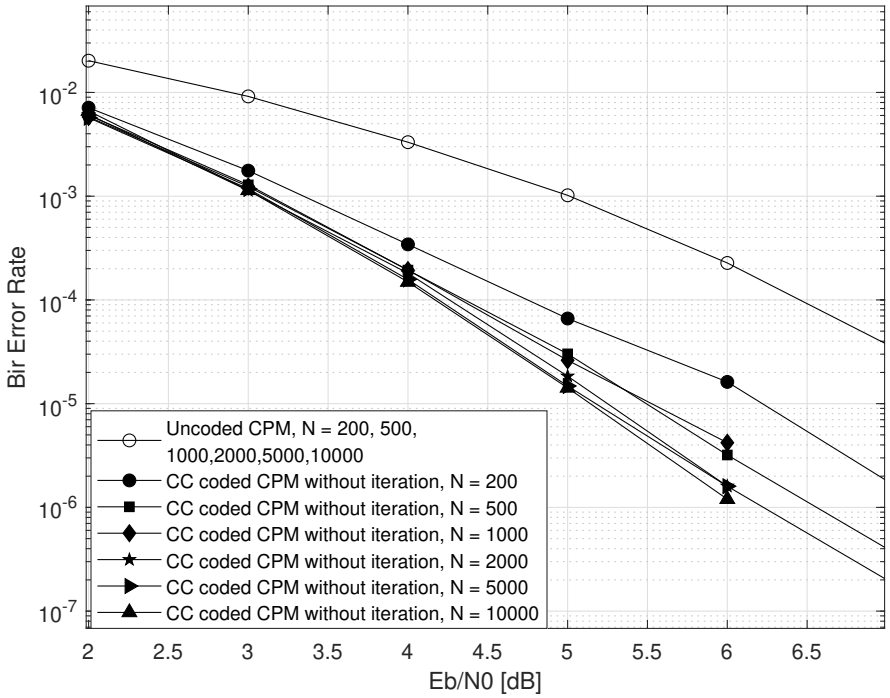

Figure 3.10: Impact of CC coding for BER performance of CPM

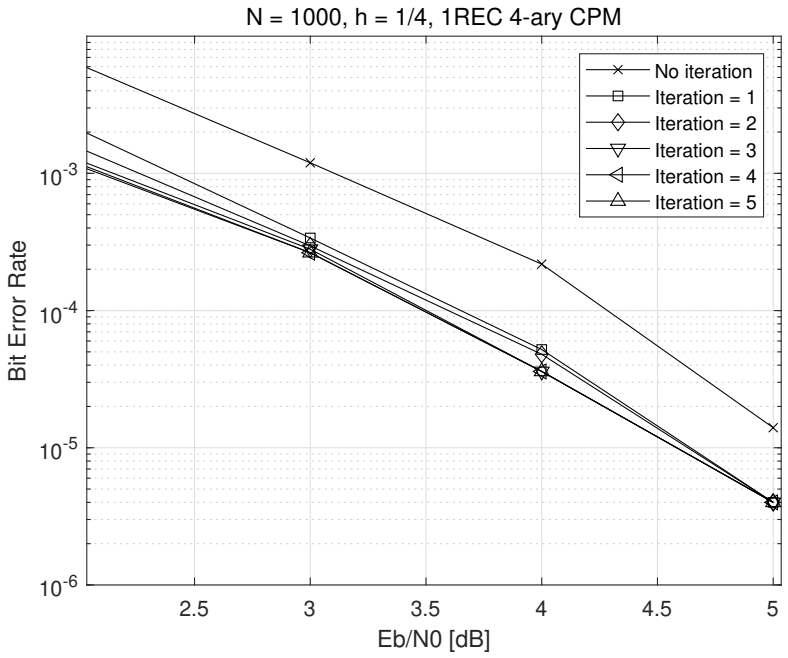

Figure 3.11: Impact of iteration times for error performance of SCCPM

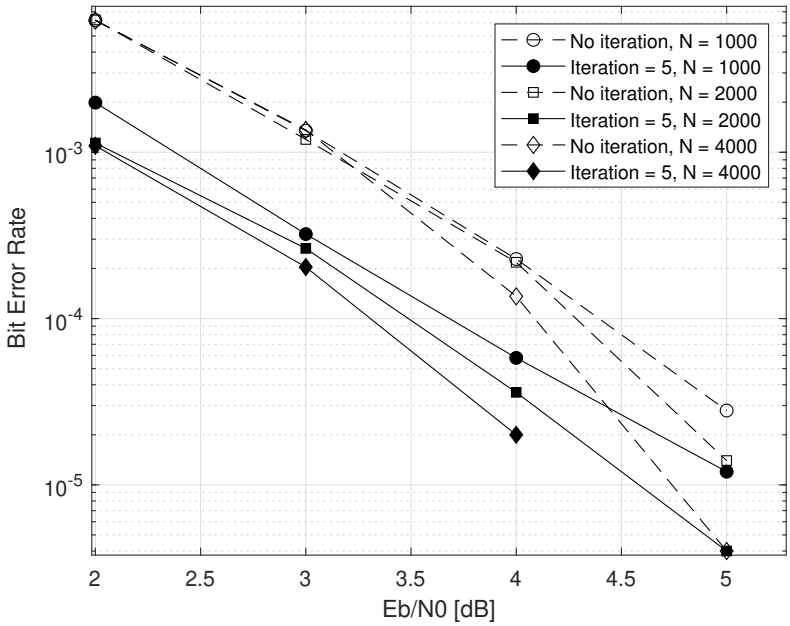

Figure 3.12: The error performance of SCCPM with different interleave length

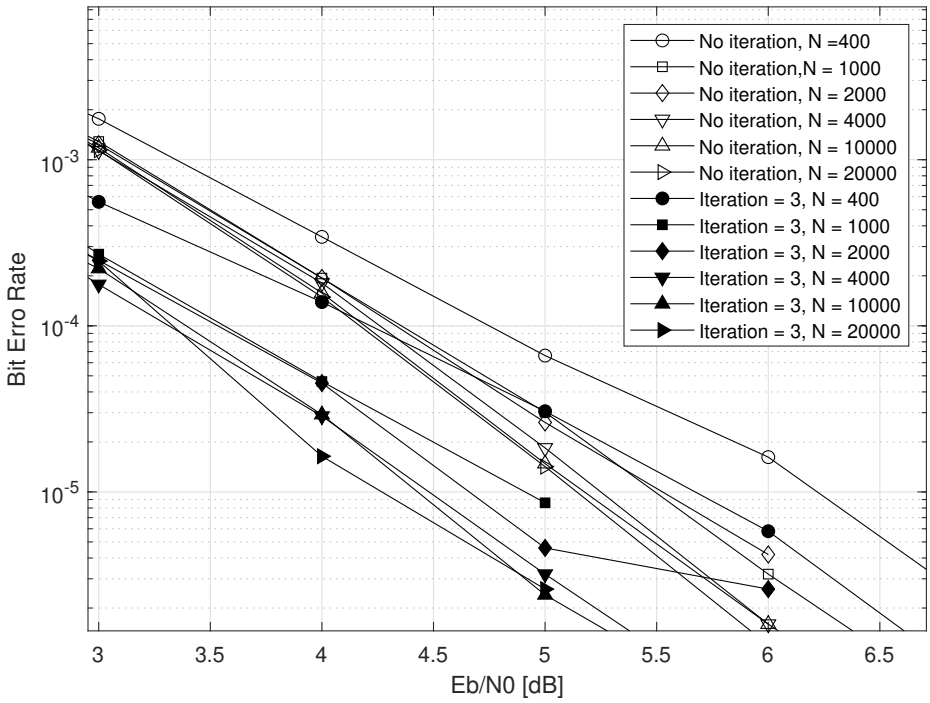

Figure 3.13: Impacts of turbo decoding and length of interleaver for BER performance of SCCPM

CHAPTER 4 **CONCLUSION**

In this thesis, first, the conventional full-response CPM signal and its decomposition module is described. By calculating the minimum Euclidean distance for M-ary CPFSK, the simulation of error performance for M-ary CPFSK with different modulation index verifies that the larger Euclidean distance in the phase trellis can provide more gain in the error performance. The minimum Euclidean distance calculation shows that there are several values of h providing a catastrophic BER performance due to the small Euclidean distance. Simulation result also shows that a greater h can cause more spectrum cost when $h < 1$.

Second, based on the decomposition module, a design of turbo decoding SCCPM system by using SOVA is presented. A practiced calculation method for SOVA is detailed, and a simplified metric calculation approach for CPE (CPM) SOVA decoding is provided. The simulation results show that a $1.5 - 2$ dB coding gain can be obtained by concatenating a CC with the CPM without iterative decoding compared to using CPM alone. The turbo decoding design can provide another 1 dB improvement of the BER performance for the CC-coded CPM. This turbo decoding SCCPM can be effectively implemented with a moderate frame size.

Possible future work may include the implantation of a more complicated modulation scheme (e.g., partial-response CPM) and concatenation of different channel coding (e.g., recursive systematic convolutional code, low-density parity-check code (LDPC), etc.) in the SCCPM with SOVA/MAP turbo decoding system.

LIST OF REFERENCES

[1] P. Moqvist and T.M. Aulin. Serially concatenated continuous phase modulation with iterative decoding. *IEEE Transactions on Communications*, 49(11):1901–1915, 2001.

[2] T. Aulin and C. Sundberg. Continuous phase modulation - part i: Full response signaling. *IEEE Transactions on Communications*, 29(3):196–209, 1981.

[3] T. Aulin. Symbol error probability bounds for coherently viterbi detected continuous phase modulated signals. *IEEE Transactions on Communications*, 29(11):1707–1715, 1981.

[4] Guo Tai Chen, Lei Cao, Lun Yu, and Chang Wen Chen. Test-pattern-reduced decoding for turbo product codes with multi-error-correcting ebch codes. *IEEE Transactions on Communications*, 57(2):307–310, 2009.

[5] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

[6] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate (corresp.). *IEEE Transactions on Information Theory*, 20(2):284–287, 1974.

[7] B.E. Rimoldi. A decomposition approach to CPM. *IEEE Transactions on Information Theory*, 34(2):260–270, 1988.

[8] B. Rimoldi and Quinn Li. Coded continuous phase modulation using ring convolutional codes. *IEEE Transactions on Communications*, 43(11):2714–2720, 1995.

[9] C. Berrou and A. Glavieux. Near optimum error correcting coding and decoding: turbo-codes. *IEEE Transactions on Communications*, 44(10):1261–1271, 1996.

[10] J.P. Woodard and L. Hanzo. Comparative study of turbo decoding techniques: an overview. *IEEE Transactions on Vehicular Technology*, 49(6):2208–2233, 2000.

[11] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara. Serial concatenation of interleaved codes: performance analysis, design, and iterative decoding. *IEEE Transactions on Information Theory*, 44(3):909–926, 1998.

[12] Pär Moqvist et al. Serially concatenated systems: An iterative decoding approach with application to continuous phase modulation. *Lic. Eng. thesis, Chalmers University of Technology, Gothenburg, Sweden*, 1999.

[13] Harold Ollivier and Jean-Pierre Tillich. Interleaved serial concatenation of quantum convolutional codes: gate implementation and iterative error estimation algorithm. In *Proceedings of the 26th Symposium on Information Theory in the Benelux*, page 149, 2005.

[14] V.F. Szeto and S. Pasupathy. Iterative decoding of serially concatenated convolutional codes and MSK. *IEEE Communications Letters*, 3(9):272–274, 1999.

[15] K.R. Narayanan and G.L. Stuber. Performance of trellis-coded CPM with iterative demodulation and decoding. *IEEE Transactions on Communications*, 49(4):676–687, 2001.

[16] T.L. Tapp and R.L. Mickelson. Turbo detection of coded continuous-phase modulations. In *MILCOM 1999. IEEE Military Communications. Conference Proceedings (Cat. No.99CH36341)*, volume 1, pages 534–537 vol.1, 1999.

[17] Ezio Biglieri, Dariush Divsalar, Marvin K Simon, and Peter J McLane. *Introduction to trellis-coded modulation with applications*. Prentice-Hall, Inc., 1991.

[18] John B Anderson, Tor Aulin, and Carl-Erik Sundberg. *Digital phase modulation*. Springer Science & Business Media, 2013.

[19] G.D. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[20] J. Hagenauer and P. Hoeher. A viterbi algorithm with soft-decision outputs and its applications. In *1989 IEEE Global Telecommunications Conference and Exhibition 'Communications Technology for the 1990s and Beyond'*, pages 1680–1686 vol.3, 1989.

[21] P. Robertson, E. Villebrun, and P. Hoeher. A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain. In *Proceedings IEEE International Conference on Communications ICC '95*, volume 2, pages 1009–1013 vol.2, 1995.

[22] J. Hagenauer. Source-controlled channel decoding. *IEEE Transactions on Communications*, 43(9):2449–2457, 1995.

[23] J. Hagenauer and L. Papke. Decoding "turbo"-codes with the soft output viterbi algorithm (sova). In *Proceedings of 1994 IEEE International Symposium on Information Theory*, pages 164–, 1994.

[24] Lei Cao and Chang Wen Chen. Constrained SOVA decoding in concatenated codes. In *2004 IEEE Wireless Communications and Networking Conference (IEEE Cat. No.04TH8733)*, volume 4, pages 2280–2284, 2004.

[25] J.G. Proakis and M. Salehi. *Digital Communications*. McGraw-Hill, 2008.

[26] John B Anderson, Tor Aulin, and Carl-Erik Sundberg. *Digital phase modulation*. Springer Science & Business Media, 2013.

[27] Lei Cao and Chang Wen Chen. A novel product coding and recurrent alternate decoding scheme for image transmission over noisy channels. *IEEE Transactions on Communications*, 51(9):1426–1431, 2003.

[28] Zhijie Guo and Lei Cao. Serially concatenated continuous phase modulation with SOVA turbo decoding. In *SoutheastCon 2022 (SoutheastCon22)*, Mobile, USA, March 2022.

VITA

Zhijie Guo, born in ShanXi Province, China in 1996, received the Bachelor degree of Engineering in Measurement Control Technology and Instrument from TianJin University of Technology in 2018. Starting from August 2019, he is working towards the M.S degree in Electrical Engineering at The University of Mississippi, University, MS. During the M.S program, He worked as a teaching assistant (instructor) of the circuit experiment course as well as a research assistant for the Department of Electrical and Computer Engineering. In 2022 April, his conference paper *Serially Concatenated Continuous Phase Modulation with SOVA Turbo Decoding* was accepted by IEEE SoutheastCon 2022 and was selected as one of the four finalists for IEEE-HKN Best Student Paper Award at IEEE SoutheastCon 2022.