The Effects of I/Q Imbalance on the Performances of QPSK in Uncoded and Coded Schemes: Imbalance Detection and Compensation Techniques

Yuan Dong Wang

A Thesis

In

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Applied Science at

Concordia University

Montreal, Québec, Canada

August, 2002

0-612-72916-8

ABSTRACT

The Effects of I/Q Imbalance on the Performances of QPSK in Uncoded and Coded Schemes: Detection and Compensation Techniques

Yuan Dong Wang, MASc
Concordia University, 2002

Analogue implementations of quadrature modulation circuits are usually imperfect and suffer from several impairments. Among these impairments, the amplitude imbalance and phase imbalance are the major source of signal distortions especially for direct conversion schemes. The imperfection contributes to the distortion of signal constellation and spurious tones at the transmitter, and also affects the steady state lock point at the receiver. The imperfect rendered from such a quadrature modulation scheme inevitably degrades the system performance.

In this thesis, two kinds of mathematic models are proposed. One of them is an equivalent baseband matrix model. Another is a modulated signal mathematic model. Base on our research, a least square estimation method is adapted at the transmitter by means of amplitude of modulated signal, and a compensation method is achieved backward in baseband before the quadrature modulator. At the receiver, maximum likelihood (ML) estimation is used to detect the imbalances from the transmitter and receiver. A compensation method reaches the solution of compensation at the receiver. We also prove the efficient algorithms by the simulation result.

Channel coding technologies can be a general error correction technology. With Viterbi decoder algorithm and Turbo code, the simulations for the imbalance effect on the performance of the system have been done. These simulation results show that Turbo code as well as Viterbi decoding has powerful correction ability to the I/Q imbalances.

Dedicated to my wife and son ....

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank my supervisor Dr. M.R. Soleymani. His constant guidance, encouragement and suggestions allowed me to make steady progress throughout the research phase. I would like to convey special thanks to one of the committee members, Dr. Rong Gang Qi for his vast help during all stages of my study. I also extend my gratitude to Dr. Shayan for reviewing my work and providing worthy feedback. I appreciate all the members of Wireless and Satellite Communication lab. Without their support this work would be sorely lacking. Finally and most importantly, I want to thank my wife Ying and my son Qiao for their patience and continuous support in overcoming the difficulties of these years.

TABLE OF CONTENT

TABLE of FIGURES

LIST OF ACRONYMS

QPSK	Quadrature Phase-Shift Keying
OQPSK	Offset Quadrature Phase-Shift Keying
IF	Intermediate Frequency
RF	Radio Frequency
AWGN	Additive White Gaussian Noise
LO	Local Oscillator
LS	Least Square
LMS	Least Mean Square
ML	Maximum Likelihood
RSC	Recursive Systematic Convolutional
PA	Power Amplitude
LNA	Low Noise Amplifier
SSB	Single Side Band
IM	InterModulation
BER	Bit Error Rate
SNR	Signal-Noise-Ratio
MAP	Maximum a Posteriori
MMSE	Minimum Mean Square Error
DA	Data-aided
DD	Decision-directed
NDA	Non-Data-aided
PSAM	Pilot Symbol Assisted Modulation
FEC	Forward Error Correction
VA	Viterbi Algorithm
ISI	InterSymbols Interference
PCCC	Parallel Concatenation Convolutional Codes
SOVA	Soft-Output Viterbi Algorithm
SISO	Soft-In Soft-Output

NSC	NonSystematic Convolutional
LLR	Logarithm of Likelihood Ratios

Chapter 1

Introduction

1.1 Communication System Model

A digital communication system is a means of transporting information from one party to another party. In recent years, there has been an increasing demand for more efficient and more reliable data transmission for this system and a great effort has been made. Figure 1.1 illustrates the basic components of such a system.

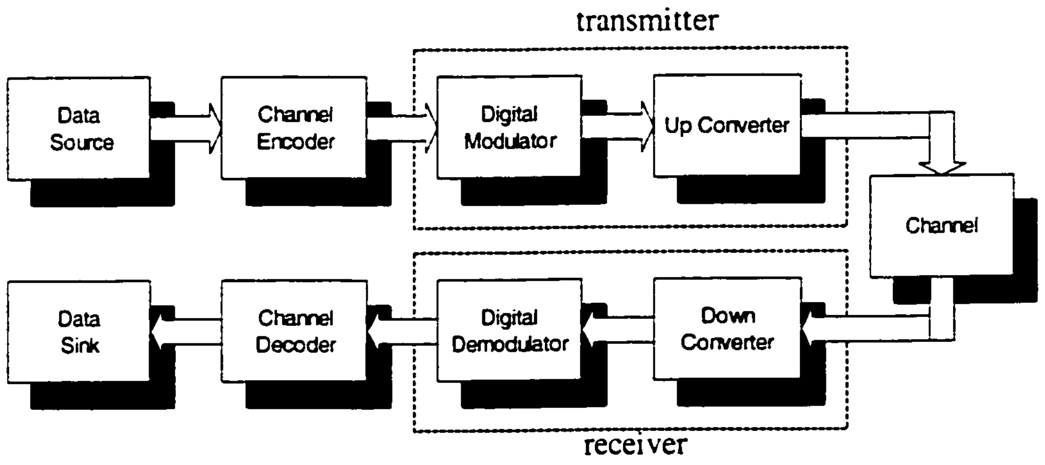


Fig. 1.1 Digital communication system block diagram

Data source feeds digital information to the channel encoder. Channel coding is performed to provide reliable transmission over the channel. The channel encoder introduces structured redundancy into the transmitted data stream and the channel decoder exploits the deliberately introduced redundancy in the data stream to correct errors in the received sequence, which has been corrupted by the channel. In this study, convolutional encoders with Viterbi decoders and Turbo codes have been used.

The transmitter includes a digital modulator and an up-converter. The function of the digital modulator is to map the encoded digital sequence into a train of analogue waveform. The up-converter shifts the analogue information-bearing waveform from the baseband first to an intermediate frequency (IF) then to a desired radio frequency (RF) or directly to RF and also raises the signal power to the desired level. This frequency band is normally positioned at a frequency much higher than the data rate. The resulting signal is then transmitted over a channel to a receiver. A receiver consists of a down-converter and a digital demodulator. At the receiver, the down-converter shifts the received signal in RF frequency to an intermediate frequency (IF) and then to baseband or directly to baseband. The digital demodulator is used to recover the transmitted data from the baseband signal. For a direct conversion scheme, the up-converter shifts the analogue information-bearing waveform from the baseband directly to a radio frequency (RF) and the down-converter reverses the procedure. There are many modulation schemes used in communication system. In this thesis, only Quadrature Phase-Shift Keying (QPSK) modulation scheme is adapted. However, some results obtained through this study may apply to other quadrature modulation schemes, such as Offset Quadrature Phase-Shift Keying (OQPSK).

The transmitted signal may have various channel distortions depending on applications. For instance, in mobile wireless communication channels, we may haves significant multi-path and fading distortions. On the other hand, it might be sufficient to use a simple Additive White Gaussian Noise (AWGN) channel model for many other applications (e.g., for satellite communications). For simplicity (but without loss of generality), we use the AWGN model for this study.

1.2 Gain and Phase Imbalances and DC Offset

QPSK modulation scheme has been widely used in digital communication systems for its good balance between bandwidth and power efficiencies. A QPSK up-converter is shown in Fig. 1.2. In Fig. 1.2, the information sequence is split evenly between the upper branch, referred to as the in-phase (I) channel, and the lower branch, the quadrature (Q) channel. On the transmitter side, the single-tone generated by the Local Oscillator (LO) is split into two equal-amplitude, 90° apart, single-tones prior to the mixing. The mixers receive the two outputs of the two low pass filters (LPF) and mix them with a Local Oscillator (LO) to an intermediate frequency (IF) or directly to

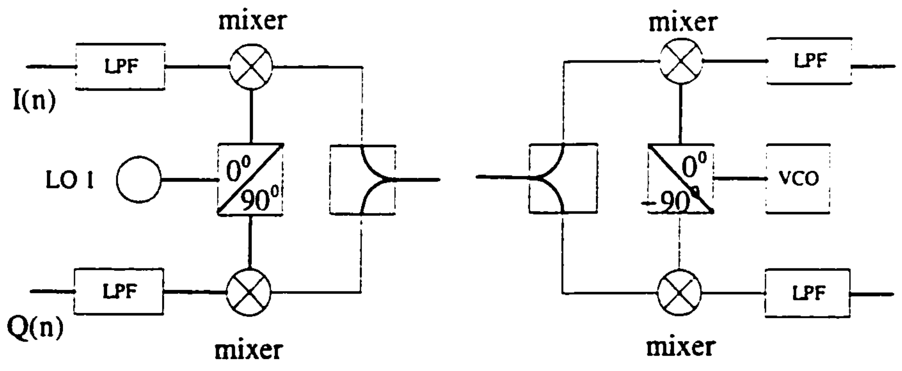


Fig. 1.2 Up converter and down converter diagram

radio frequency (RF). After the mixers the two resulted IF or RF signals are combined by a 0 ° power combiner to produce a real signal at an IF or RF. When used as a down converter, the process is reversed. As shown in right side of Fig. 1.2, the received IF (or RF) signal is split into the in-phase (I) and the quadrature (Q) signals by mixing it with two single-tones of a 90-degree phase difference. The baseband I and Q signals are obtained by using a pair of lowpass filters to remove the other side-band of the IF (RF) signal at a high frequency.

In mathematical terms, the above model has the following respesentations. On the transmitter side, the baseband signal at the discrete time n can be represented by:

$$x(t) = I(t) + jQ(t) \tag{1.1}$$

Where $I(t)$ and $Q(t)$ are the binary data streams for the I-channel and the Q-channel, respectively. The output of the splitter is given by:

$$l_1(t) = \cos(\omega_c t) - j\sin(\omega_c t) \tag{1.2}$$

where $\omega_c = 2\pi f_c$, and f_c is a carrier frequency. After the up-converter, the transmitted output signal is:

$$\begin{aligned} s(t) &= \mathrm{Re}\{x(t)l(t)\} \\ &= I(t)\cos(\omega_c t) + Q(t)\sin(\omega_c t) \end{aligned} \tag{1.3}$$

Where $I(t)\cos(\omega_c t)$ is the term from I channel. $Q(t)\sin(\omega_c t)$ is the term from the Q channel. The two channel signals are orthogonal to each other, which means that they have a 90° phase difference.

In the down converter, the received RF signal can be written as:

$$r(t) = s(t) + n(t) \tag{1.4}$$

Where $n(t)$ is the AWGN.

This signal is then mixed with the local oscillator signal:

$$l_2(t) = \cos(\omega_c t) + j\sin(\omega_c t) \tag{1.5}$$

Two outputs of local oscillator signal are also orthogonal to each other. Passing the low pass filters and sampling, we have the output of the two channels,

$$\begin{aligned} Y_I(n) &= I(n) + N_I(n) \\ Y_Q(n) &= Q(n) + N_Q(n) \end{aligned} \tag{1.6}$$

where $I(n)$ and $Q(n)$ are the samples of $I(t)$ and $Q(t)$ with a sample per symbol, respectively. This result is an ideal case. Unfortunately, analogue implementations of quadrature modulation circuits are usually imperfect and suffer from several impairments [1]-[9]. Among these impairments, the amplitude imbalance, phase imbalance, and DC

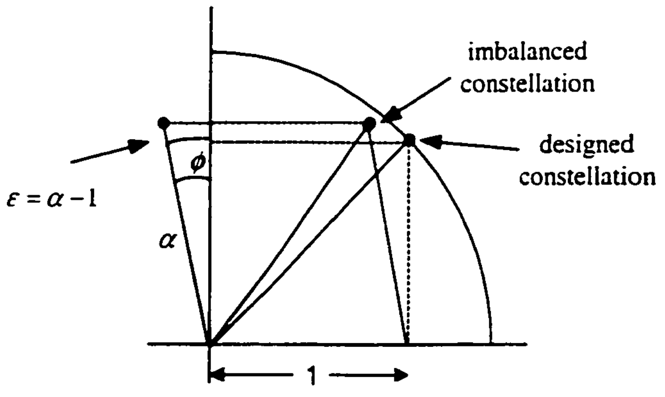


Fig. 1.3 Gain and phase imbanlance definition

offset are the major source of signal distortions especially for direct conversion schemes [1][2]. In the ideal case, the two channels have the same amplitude and a 90° phase difference. When gain and phase imbalances occur, the signal amplitudes of the two channels are not the same anymore and the phase difference is not 90°. Amplitude imbalance ε is defined as the amplitude difference between I channel and Q channel in the up-converter or the down-converter. Phase imbalance, ϕ, is a phase offset from the designed phase difference of 90° between two channels. In practice, I channel component is usually taken as a reference component (phase $=0^\circ$,and gain=1). These definitions of gain and phase imbalances are shown in Fig 1.3. If I channel amplitude is 1, the gain imbalance in the Q channel is $\varepsilon = \alpha - 1$, where α is the gain of the Q channel component. If the Q channel signal is 90° apart from the reference component at the ideal case, the phase offset ϕ in the Q channel component from 90° is referred as the phase imbalance. For example, if the desired carrier in I channel is $\cos\omega_c t$ as a reference, imbalanced Q channel component becomes $\alpha\sin(\omega_c t + \phi)$. DC offset is a dc component at the down-converter output.

The imperfection contributes to the distortion of signal constellation and spurious tones at the transmitter, and also affects the steady state lock point at the receiver [1][2][3][4]. The imperfection rendered from such a quadrature modulation scheme inevitably degrades the system performance [1]-[5]. In some cases, the deficiencies seriously affect the system performance [6][7]. Channel coding technology can be a general correction technology. In order to find other high efficient detection and compensation technologies, a deep analysis of I/Q imbalance is essential. Based on the analysis, some special correction technologies can be developed.

1.3 Brief Review of Literature about Gain and Phase Imbalances

Cavers and Liao offered a detailed analysis on the gain and phase imbalance in 1993 [1]. Based on their analysis model, they proposed an imbalance detection and compensation approach to counteract the gain and phase imbalance at both the transmitter and the receiver sides. The analysis includes spurious tone analysis and intermodulation product analysis at the transmitter, and gives simulation results of a degraded BER due to distortion of the signal constellation. They also proposed an adaptive method at the transmitter for compensation of quadrature modulator imbalance and another adaptive method at the receiver for compensation of both quadrature modulator and demodulator imbalance. Both methods are based on Least Mean Square (LMS) adaptive algorithm.

In 2000, Nguyen published "Improving QPSK Demodulator Performance for Quadrature Receiver with Information from Amplitude and Phase Imbalance Correction" [2]. It gave a linear Least Square (LS) estimator and a nonlinear estimator algorithm at the receiver with known phase and amplitude information. The simulation results in the paper shows that the imbalances greatly degrade the system performance. His linear LS estimator at a receiver greatly improves the performance impaired by the imbalances at a transmitter. Then Huang in [3] explored Gram-Schmit orthogonization procedure at a receiver to compensate only for phase imbalance. Tsou published two papers that analyzed the imbalance effect on steady state lock point based on the conventional Costas loop and further analyzed its degradation of system performance [4].

1.4 Thesis Preview

The purpose of this thesis is to identify phase and gain imbalance effects on the QPSK performance and to find efficient detection and compensation technologies. The objective is not only achieved in uncoded schemes, but also done in coded schemes.

In following chapter, we first analyze the sources of the phase and gain imbalances as well as imbalance levels in different transmitter and receiver architecture. Subsequently, two kind of mathematical models are derived. One of them is the equivalent matrix model that is used to analyze the effect on the constellation and to simulate the system performance. Another is a modulated signal mathematic model that can be used for the analysis of intermodulation products and signal spectrums. In this chapter, several aspects of effects from the phase and gain imbalances are analyzed. The analysis of imbalance effects include:

1. How the imbalances cause spurious tones in the transmitter signal.

2. How the intermodulation products are produced due to the imbalances.
3. The effects of the imbalances on steady state lock point.
4. The effects on the system performance.

In chapter 3, novel detection and compensation methods are proposed on the base of recent research. A least square estimation method is adopted in the transmitter by means of amplitude of modulated signal and compensation is achieved backward in baseband before the up-converter. In the receiver, maximum likelihood (ML) estimation is used to detect the imbalances from the transmitter and the receiver. One compensation method reach the solution of compensation at the receiver. We also prove the efficient algorithms by the simulation results.

Imbalance effect on the system with a convolutional encoder and a Viterbi decoder is presented in the chapter 4. Firstly, the convolutional code is discussed. Then, Viterbi decoder algorithm is introduced. Finally, simulation results for the imbalance effect on the performance of the system with Viterbi decoder algorithm is presented.

In Chapter 5, turbo code and imbalance effects on performance are addressed. In a turbo encoder, two identical Recursive Systematic Convolutional (RSC) encoders and a random interleaver are used. Then, we will describe turbo code MAP algorithm. The key of turbo decoding is the iterative algorithm. The turbo code performance with imbalances is investigated at last.

Finally, conclusion and suggestion for future research are made in chapter 6.

1.5 Summary of the Contribution of the Thesis

The original contribution of this thesis is described below:

1. Initial analysis of imbalance effects on the transmitted signal at frequency domain.
2. Using least square estimation algorithm compensates the imbalances based on data sequence at a transmitter.

3. Using Maximum Likelihood method compensates the imbalances based on training sequence at a receiver
4. Analysis of the imbalance effect on the performance of the system with convolutional encoder and Viterbi decoder.
5. First analysis of the imbalance effects on the performance of the system with turbo coding.

Chapter 2

Effects of I/Q Imbalances on the System

Phase and gain imbalances affect the signal constellation and also produce spurious tones as well as have an effect on the receiver phase synchronization. Eventually, the system performance is degraded. In this chapter, we are going to address the aspects of imbalance issues, including the sources of I/Q imbalances, math models and effects on the system performance. Before we start our work on the issues, some assumptions have to be made:

1. The channel bandwidth is infinitive. This means the transmitter is allowed to use a rectangle pulse shaping, and there is no inter-symbol-interference.
2. The I/Q imbalances are Wide Sense Stationary (WSS) processes.

2.1 Sources of Imbalances and DC Offset

In this section, we will discuss where the phase, gain imbalances and DC offset are produced.

2.1.1 Sources of Phase Imbalance

Phase shift introduced in a real quadrature frequency converter between the two channels is not always exactly 90°. There are mainly four sources of the phase imbalance:

1. A splitter is usually composed of a RC network that splits a local oscillator signal into two signals with a 90° phase difference. The phase difference between two output signals from the splitter may not be 90°. This is the main source of phase imbalance that is independent of I and Q signals.
2. A phase difference introduced by the two RF paths is one source of phase imbalance. Along the two RF paths, a little circuit asymmetry between the

two channels is inevitable. The circuit asymmetry will result in a high phase difference since two channel RF signals have very high frequency.

3. Phase shift may be caused by an impedance mismatch between a mixer and a splitter circuit as well as by the finite amount of isolation between the two output ports on the phase shift circuit. When the phase shifts in two channels are not same, phase imbalance occurs.
4. Phase differences may be caused by the different phase responses of I and Q channel low-pass filters. Higher data rate causes larger phase error

In brief, the sources of phase imbalance are frequency dependent. The higher the frequency of LO is, the larger the offset of phase is. A high data rate has a larger phase imbalance.

2.1.2. Sources of Amplitude Imbalance

Amplitude imbalance between I and Q channels has two main sources. The first is from RF parts. RF splitters and combiners have different insertion losses to two branches. The mixers also have different conversion losses in I and Q channels. The second source of gain imbalance is from the gain differences of the low pass filters between I and Q channels. Amplitude imbalance greatly increases the distortion of the constellation with phase imbalance although it is unlike phase imbalance, which causes a cross talk between I and Q channels.

2.1.3 Carrier Leakthrough and DC Offset

Real mixers are far away from the ideal mixer. Although a balanced up-converter, which is widely used to generate suppressed-carrier QPSK signals, is adapted, there are great many spurious products that arise from intermodulation terms and leakthrough from the local oscillator LO. The leakthrough of the LO signal is a particular problem in that it lies in the middle of the wanted pass band of the modulated signal and cannot be removed by filtering. It acts like DC offset in the receiver side. DC components are created out of the down converter mixers from both self-mixing of the receiver LO and the transmitter LO mixing with the receiver LO. DC offset is a problem in receivers that are directly coupled. especially in low data rate systems. DC offsets may occur in I and Q channels starting with the down-converter mixers through direct coupled amplifiers and active filters. Individual DC offsets, contributed by each of these components, will accumulate at the output of the down-converter.

2.2 Transceiver Architectures

Imbalance levels are different with different transceiver architectures. There are mainly two kinds of architectures that are widely used. One of them is heterodyne architecture. Another is homodyne architecture (also called direct-conversion architecture or zero IF architecture). They are discussed respectively below.

2.2.1 Heterodyne Architecture

Heterodyne transceiver is a well-known transceiver architecture; a transmitter as shown in Fig. 2.1 and a receiver as in Fig 2.2. In Fig.2.1, SAW represents Surface Acoustic Wave, PA means Power Amplifier and D/A is Digital to Analog converter. The other notations have been mentioned before. The heterodyne transmitter converts the baseband signal up in two steps so that the power amplitude (PA) output spectrum is far away from the LO1 frequency. Its quadrature modulation is performed at a lower

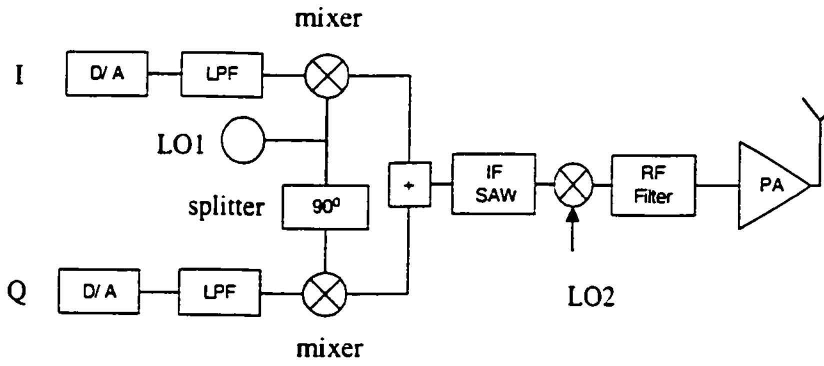


Fig. 2.1 Diagram of heterodyne structure transmitter

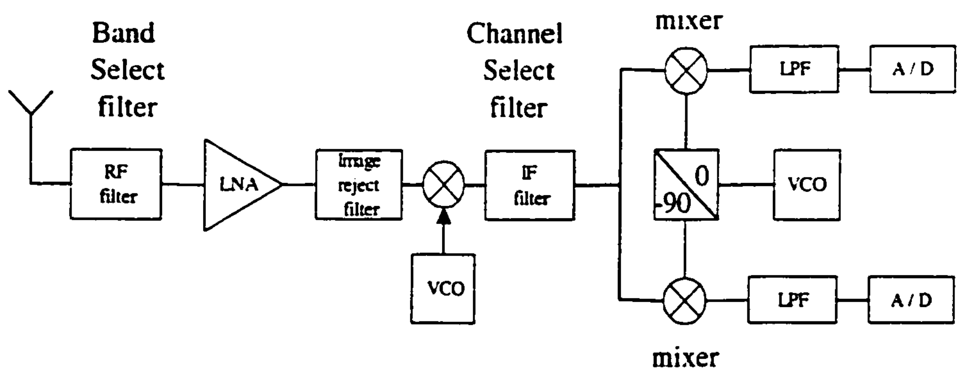


Fig. 2.2 Diagram of heterodyne structure receiver

frequency. The heterodyne receiver works in a reverse procedure. It converts the radio frequency (RF) signal down in two steps. Its quadrature demodulation works also at lower frequency. A major advantage of the structure is its adaptability to many different requirements. That is why it has been the dominant choice in RF systems for many decades. However, the complexity of the structure and the large number of external component (eg. The IF filter) cause problems if a high level of integration is necessary. It is also the major drawback if costs are concerned. Furthermore, amplification at some high IF can cause high power consumption. Fortunately, the imbalance issues are not so critical as other structures since quadrature modulation and demodulation are performed at a much lower frequency than RF [9].

2.2.2 Direct Conversion Architecture

Direct conversion architecture includes a direct conversion transmitter and a direct conversion receiver. A direct conversion transmitter can be seen in Fig 2.3 and a direct conversion receiver in Fig 2.4 [9]. In Fig. 2.4, LNA means Low Noise Amplifier. With this architecture, the carrier frequency of the direct conversion transmitter is equal to the local oscillator frequency, so up-conversion occurs in one step. The structure suffers from I/Q phase mismatch and carrier leakthrough, which has an effect similar to that of DC offset on the performance. The direct conversion receiver has the same imbalance problem and also suffers from DC offset. The I/Q imbalances and DC offset

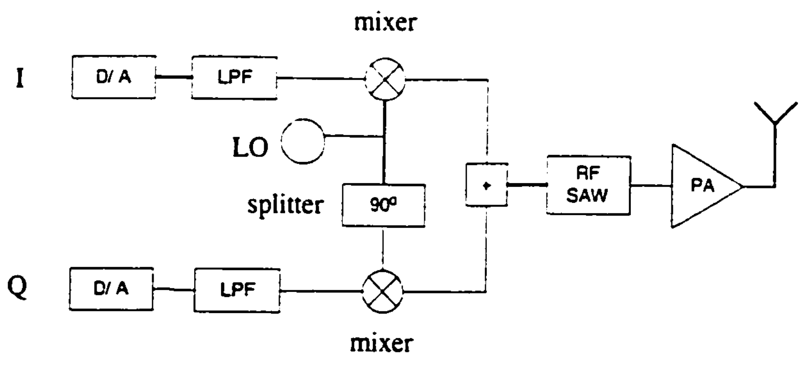


Fig. 2.3 Direct conversion transmitter

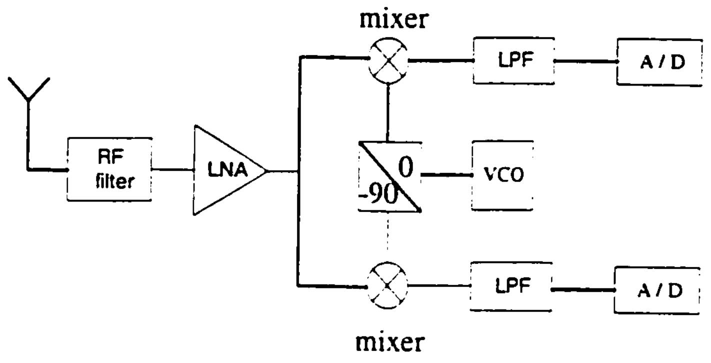


Fig. 2.4 Direct conversion receiver

with this architecture are much worse than the heterodyne architecture [1][2][3]. So I/Q imbalances and DC offset in the direct conversion structure are main concerns.

2.3 The Imbalance Models of a Quadrature Modulation System

From previous sections, it is known that all the devices on two branches of quadrature modulation and demodulation could cause gain and phase imbalances. In a transmitter, we can merge all of the imbalances with two parameters for convenience without loss of generality, and the same thing can be done in a receiver. The mathematical model diagram can be obtained as shown in Fig 2.5, where ϕ_T and α_T are

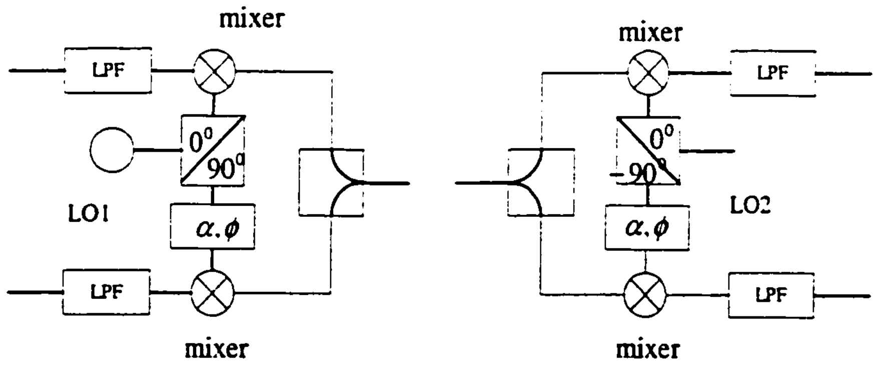


Fig. 2.5 I/Q imbalance model diagram

respectively phase and gain imbalances in the transmitter, ϕ_R and α_R respectively are phase imbalance and gain imbalances in the receiver.

2.3.1 The Imbalance Model of the Quadrature Modulation

In the up-converter, as can be seen in Fig 2.5, the two inputs to mixers from the local oscillator signal can be written into complex asymmetric form [6][7]:

$$l_1(t) = \cos(\omega_c t) - j\alpha_T \sin(\omega_c t + \phi_T) \tag{2.1}$$

Where all of gain and phase imbalances in the up-converter are covered. The input to the up-converter can also be represented in complex form:

$$x(t) = I(t) + jQ(t) \tag{2.2}$$

Where $I(t)$ and $Q(t)$ are the binary data streams for I-channel and the Q-channel, respectively. The real transmitted signal is:

$$\begin{aligned} s(t) &= \mathrm{Re}[x(t)l_1(t)] \\ &= [I(t) + \alpha_T Q(t)\sin(\phi_T)]\cos(\omega_c t) \\ &\quad + \alpha_T Q(t)\cos(\phi_T)\sin(\omega_c t) \end{aligned} \tag{2.3}$$

From Eq. (2.3), if there is no gain and phase imbalances, that is $\alpha_T = 1, \phi_T = 0$, then:

$$\begin{aligned} s(t) &= \mathrm{Re}[x(t)l_1(t)] \\ &= I(t)\cos(\omega_c t) + Q(t)\sin(\omega_c t) \end{aligned}$$

This is the ideal case.

In the receiver, the local oscillator signal is:

$$l_2(t) = \cos(\omega_c t) + j\alpha_R \sin(\omega_c t + \phi_R) \tag{2.4}$$

where α_R is an imbalanced gain and ϕ_R represents a phase imbalance.

If channel rotation is not considered, the received signal can be written as:

$$r(t) = \mathrm{Re}[x(t)l_1(t)] + n(t) \tag{2.5}$$

$$\begin{aligned} y(t) &= r(t)l_2(t) \\ &= \{\mathrm{Re}[x(t)l_1(t)] + n(t)\} * \{\cos(\omega_c t) + j\alpha_R \sin(\omega_c t + \phi_R)\} \end{aligned} \tag{2.6}$$

If the imbalances in the receiver are only considered, after removing high frequency components and sampling, the output of the demodulator may be represented by:

$$Y(n) = I(n) + j[\alpha_R I(n)\sin\phi_R + \alpha_R Q(n)\cos\phi_R] + N'_I(n) + jN'_Q(n) \tag{2.7}$$

where $I(n)$ and $Q(n)$ are the samples of the $I(t)$ and $Q(t)$, respectively. $N'_I(n)$ and $N'_Q(n)$ are the noise components of the two channels.

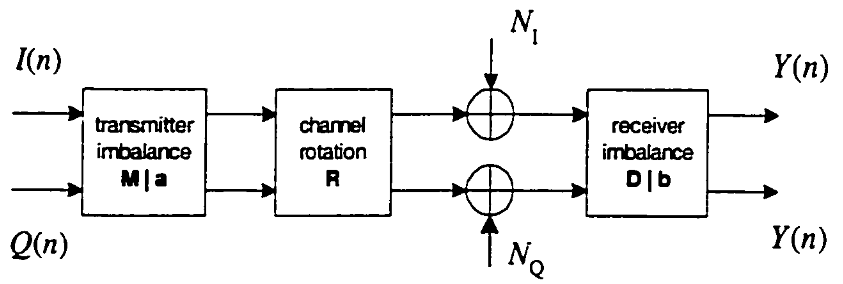


Fig. 2.6 Diagram of matrix model of trasmission system

2.3.2 Equivalent Matrix Model

Fig.2.6 shows a diagram of an equivalent matrix model. As shown in the figure. the matrixes **M** and **D** can be obtained easily from the previous section. The equivalent signal matrix in the transmitter is written as [1]:

$$\mathbf{S} = \mathbf{MX} + \mathbf{a} \tag{2.8}$$

Where **M** is the transmitter imbalance matrix, **X** is the baseband input, **S** is the transmitted equivalent matrix and **a** is the DC offset.

$$\mathbf{S} = \begin{bmatrix} S_I(t) \\ S_Q(t) \end{bmatrix} \tag{2.9}$$

where $S_I(t)$ and $S_I(t)$ are I and Q channel components of the transmitted signal.

$$\mathbf{X} = \begin{bmatrix} I(t) \\ Q(t) \end{bmatrix} \tag{2.10}$$

$$\mathbf{M} = \begin{bmatrix} 1 & \alpha_T \sin(\phi_T) \\ 0 & \alpha_T \cos(\phi_T) \end{bmatrix} \tag{2.11}$$

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \tag{2.12}$$

where a_1 and a_2, respectively, are DC offset components of the two channels.
A channel phase rotation is:

$$\mathbf{R} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{2.13}$$

The phase θ represents the difference in phase between the transmitter and receiver oscillator. The down-converter imbalance model is similar to the model of the transmitter:

$$\mathbf{D} = \begin{bmatrix} 1 & 0 \\ \alpha_R \sin(\phi_T) & \alpha_R \cos(\phi_R) \end{bmatrix} \tag{2.14}$$

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{2.15}$$

$\mathbf{b}=[b_1, b_2]^T$, is the DC offset in the down-converter.

The total system imbalance model is presented by following [1]:

$$\begin{aligned} \mathbf{Y} &= \mathbf{D}(\mathbf{RS} + \mathbf{N}) + \mathbf{b} \\ &= \mathbf{DRMX} + \mathbf{DRMa} + \mathbf{DN} + \mathbf{b} \end{aligned} \tag{2.16}$$

Where $\mathbf{N}$ is the noise matrix. Let.

$$\begin{aligned} \mathbf{H} &= \mathbf{DRM} \\ \mathbf{N}' &= \mathbf{DN} \end{aligned} \tag{2.17}$$

We have,

$$\mathbf{Y} = \mathbf{HX} + \mathbf{Ha} + \mathbf{b} + \mathbf{N}' \tag{2.18}$$

We assume that the compensation of DC offset has been done before imbalance detection and compensation in the section to come, i.e. **a=b=0**. This is because usually DC offset compensation is done at first.

2.4 Imbalance Effects on Transmitted Signal

The phase and gain imbalances have effects on the transmitted signal. They not only degrade the signal constellation, but also introduce spurious tone. I/Q imbalances cause the amplitude change of the modulated signal. This may interact with the power amplifier to cause more distortion.

2.4.1 Spurious Tones and Constellation Distortion

From Eq. (2.3), it can be seen that there is cross talk between I and Q channels. The product $\alpha_T Q(t)\sin(\phi_T)$ is a cross talk component caused by I/Q imbalances. This cross talk introduces constellation distortion. We can see it in the Fig. (2.7). This constellation has a signal offset in the transmitter.

We also see imbalance effect on a signal spectrum. The LO single tone may be rewritten to exponent form here:

$$\begin{aligned} l_1(t) &= \cos(\omega t) - j\alpha_T \sin(\omega t + \phi_T) \\ &= k_1 e^{-j\omega t} + k_2 e^{j\omega t} \end{aligned} \tag{2.19}$$

Where

$$\begin{aligned} k_1 &= \frac{1+\alpha_T e^{-j\phi_T}}{2} \\ k_2 &= \frac{1-\alpha_T e^{j\phi_T}}{2} \end{aligned} \tag{2.20}$$

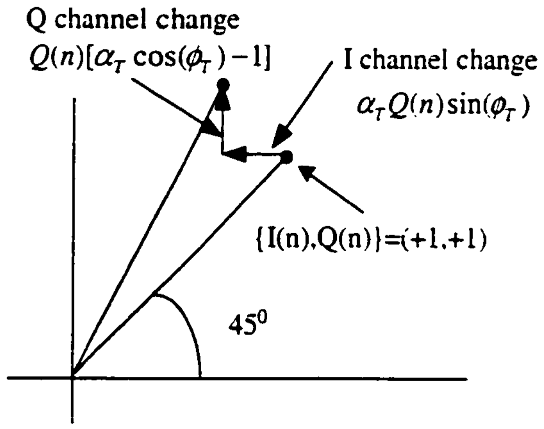


Fig. 2.7 Gain and phase imbalances effect on constellation

$$s(t) = \mathrm{Re}\{x(t)[k_1 e^{-j\omega t} + k_2 e^{j\omega t}]\} \tag{2.21}$$

When no imbalances occur, $k_2 = 0$.

If we assume x(t) is a single tone, that is:

$$x(t) = e^{j\omega_1 t} \tag{2.22}$$

then we have transmitted a real signal:

$$\begin{aligned} s(t) &= \mathrm{Re}\{e^{j\omega_1 t}[k_1 e^{-j\omega_c t} + k_2 e^{j\omega_c t}]\} \\ &= \mathrm{Re}[k_1 e^{-j(\omega_c - \omega_1)t} + k_2 e^{j(\omega_c + \omega_1)t}] \end{aligned} \tag{2.23}$$

first term second term

The second term, which is image tone of desired signal, is caused by I/Q imbalances. If gain and phase imbalances are zero, i.e. k_2=0, no second term is produced. As shown in Fig. 2.8, when a single tone is sent to the up-converter with imbalances, an extra modulated tone with image frequency of desired modulated signal is produced in addition to the designed signal. If a baseband signal with single-side-band (SSB) is inputted to the up-converter, its output signal will have an extra modulated signal, which has image spectrum of desired modulated signal.

Its power ratio is shown as:

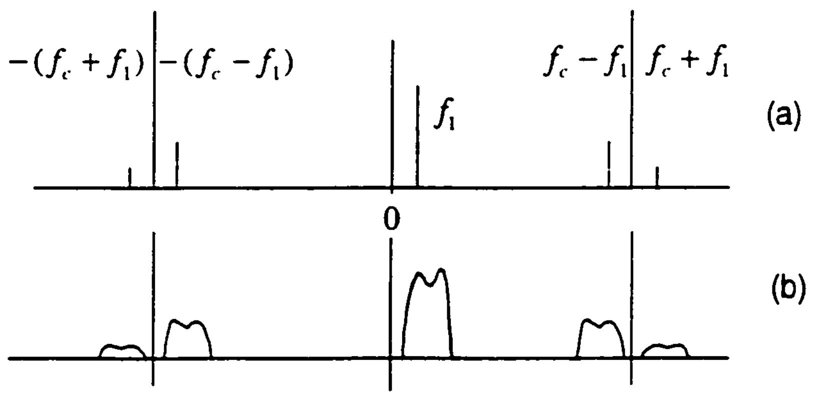


Fig. 2.8 Imbalance effect on the modulated signal spectrum. (a) effect of single tone, (b) effect of a baseband signal

$$\frac{P_{spur}}{P_{des}} = \frac{|k_2|}{|k_1|} = \frac{1+\alpha_T^2 - 2\alpha_T \cos\phi_T}{1+\alpha_T^2 + 2\alpha_T \cos\phi_T} \tag{2.24}$$

For a phase imbalance of 3°, the power ratio of imbalance is about -30.4 dB. In many situations, spurious tones at this level are not crucial unless they cause other problems. However, the mixers are not ideal and the power amplifier (PA) works close to nonlinear area so that the spurious tones cause many more other spurious tones.

2.4.2 Intermodulation Products

When the modulated signal with imbalance distortion is feed to a nonlinear power amplifier (PA), unexpected intermodulation (IM) products will be produced due to the phase and gain imbalances. For simplicity, we only consider third order intermodulation products (IP3). Form (2.23), we have:

$$\begin{aligned} IP3 &= [x(t)l_1(t)]^3 = [k_1 e^{j(\omega+\omega_1)t} + k_2 e^{-j(\omega-\omega_1)t}]^3 \\ &= k_1^3 e^{j3(\omega+\omega_1)t} + k_2^3 e^{-j3(\omega-\omega_1)t} + k_1^2 k_2 e^{j[2(\omega+\omega_1)-(\omega-\omega_1)]t} + k_1 k_2^2 e^{j[(\omega+\omega_1)-2(\omega-\omega_1)]t} \\ &= k_1^3 e^{j3(\omega+\omega_1)t} + k_2^3 e^{-j3(\omega-\omega_1)t} + k_1^2 k_2 e^{j(\omega+3\omega_1)t} + k_1 k_2^2 e^{-j(\omega-3\omega_1)t} \end{aligned} \tag{2.25}$$

Third and forth terms are caused by the imbalances and they may fall in-band. Although the second term is also produced by the imbalances, it can be filtered out easily. The fourth term can be negligible since it has a higher order of k_2. When there are other odd orders of intermodulation, many more spurious tones are produced. They greatly raise the IM floor.

We can see imbalance interaction with PA nonlinearity in a different way. Usually a constant envelope modulation is often used because it produces no IM even when the nonlinear amplifier works close to saturation. However, amplitude A of RF signals in the transmitter is not constant and fluctuates due to the imbalances. We can see it below:

$$\begin{aligned} A^2(t) &= [I(t) + \alpha_T Q(t)\sin(\phi_T)]^2 + [\alpha_T Q(t)\cos(\phi_T)]^2 \\ &= I^2(t) + \alpha_T^2 Q^2(t) + 2\alpha_T I(t)Q(t)\sin\phi_T \end{aligned} \qquad (2.26)$$

In this equation, the third term is an additional product that is from the phase imbalances. From this equation, we can see that the amplitude of the IF or RF signal in the transmitter is not constant, and it have relation with data stream I(t) and Q(t). When I(t)Q(t) is changing, amplitude A is changing with them. Higher the gain imbalance is, larger the amplitude fluctuation is. High phase imbalance causes high fluctuation of modulated amplitude. Together, they make the signal entering the power amplifier lose constant amplitude. On the other hand, the power amplifier usually works at saturate state or near it in order to get high power efficiency. The amplitude fluctuation interacting with PA non-linearity causes IM and degrades system performance [13].

2.5 the Imbalance Effect on Steady-state Lock Point of the Carrier-tracking Loop at Receiver

For the conventional Costas loop used to track the suppressed carrier of QPSK signals, the received signal is first mixed with the I-arm and Q-arm reference signals, e.g., $\cos(\omega_c t - \phi)$ and $\sin(\omega_c t - \phi)$, and then passed through an integrate-and-dump (I&D) filters on both arms of the Costas loop, as shown in Fig. (2.4) [4].

Assuming:

1. Costas loop is driven by a perfectly synchronized symbol clock.
2. For simplicity, only the imbalances in the transmitter side are considered.
3. The phase error signal of Costas loop is formed from an $Y_I Y_Q(Y_I^2 - Y_Q^2)$ type of product.

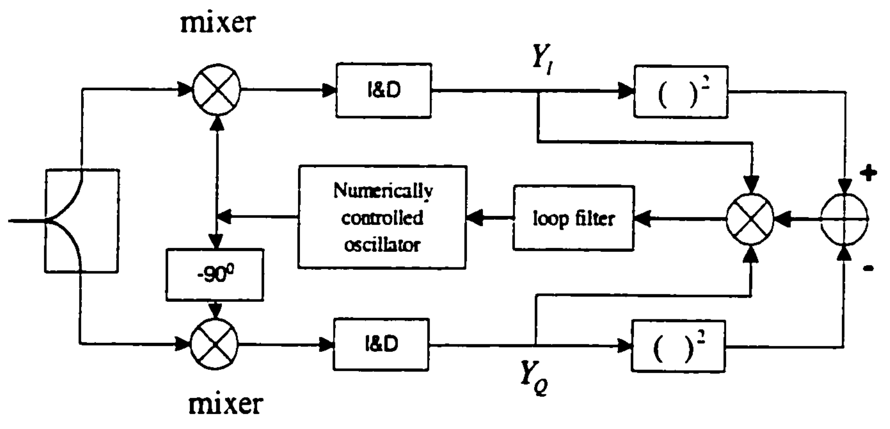


Fig. 2.9 Diagram of conventional Castal loop

Based on the baseband model shown in Eq. 2.3 and Eq. 2.4, we can obtain two arm baseband signals as:

$$\begin{aligned} Y_I(t) &= [I(t)+\alpha_T Q(t)\sin\phi_T]\cos\phi + \alpha_T Q(t)\cos\phi_T \sin\phi \\ &= I(t)\cos\phi + \alpha_T Q(t)\sin(\phi_T+\phi) \end{aligned} \tag{2.27}$$

$$\begin{aligned} Y_Q(t) &= -[I(t)+\alpha_T Q(t)\sin\phi_T]\sin\phi + \alpha_T Q(t)\cos\phi_T \cos\phi \\ &= -I(t)\sin\phi + \alpha_T Q(t)\cos(\phi_T+\phi) \end{aligned} \tag{2.28}$$

or in matrix form,

$$\mathbf{Y} = \begin{bmatrix} Y_I(t) \\ Y_Q(t) \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix}\mathbf{MX} \tag{2.29}$$

where ϕ is the initial phase delay. If ϕ is not zero, this is equivalent to rotating the received signal constellation by ϕ. The phase error of the Costas loop is obtained by solving $Y_IY_Q(Y_I^2-Y_Q^2)=0$. It is, however, not easy to find the solution at Steady-state Lock Point of the Carrier-tracking Loop in the receiver. To simplify the problem, we further assume $\alpha_T = 1$ and let:

$$Y_IY_Q[Y_I^2 - Y_Q^2] = 0 \tag{2.30}$$

It follows that:

$$\phi = -\frac{1}{2}\phi_T \tag{2.31}$$

This means that phase synchronization does not lock at $\phi = 0$ due to the I/Q imbalances. Instead, it locks at a phase that is a function of the phase and gain imbalances. So the rotation of constellation, caused by the offset of Steady-state Lock Point of the Carrier-tracking Loop, is determined by the I/Q imbalances. Moreover, phase synchronization does not change the gain and phase imbalances.

If the transmitter imbalance matrix, **M**, is multiplied by the rotation matrix caused by the offset of Steady-state Lock Point of the Carrier-tracking Loop, we can have another result. From Eq.(2.29), we have,

$$\mathbf{M}' = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \mathbf{M} \tag{2.32}$$

If we consider **M'** as an imbalance matrix, the difference between **M'** and **M** is that they have different imbalance references. Under our assumption of $\alpha_T = 1$, it is followed:

$$\mathbf{M}' = \begin{bmatrix} \cos(\phi_T/2) & \sin(\phi_T/2) \\ \sin(\phi_T/2) & \cos(\phi_T/2) \end{bmatrix} \tag{2.33}$$

This is a matrix from the case considering both the transmitter imbalances and the offset of steady loop lock point. The matrix **M'** just is a new model i.e. the symmetric imbalance matrix model. Eq. (2.32) and Eq. (2.33) shows that a symmetric imbalance matrix model can be obtained by a rotation matrix timing an asymmetric imbalance matrix model. No matter we choose the symmetric imbalance matrix model or the asymmetric imbalance matrix model, both of them represent the same imbalances. So either of them can be adapted.

The receiver imbalances also cause the offset of Steady-state Lock Point of the Carrier-tracking Loop. The rotation caused by the offset of Steady-state Lock Point of the Carrier-tracking Loop can also be merged with the channel rotation.

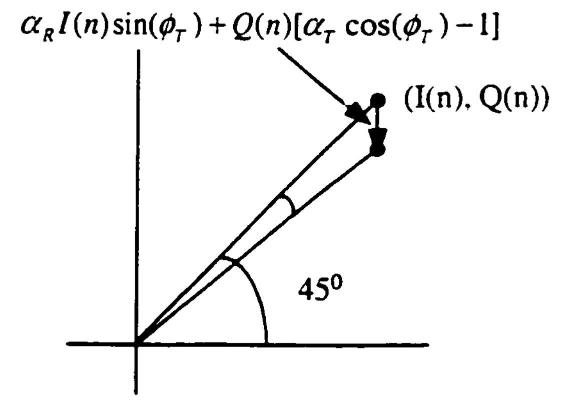


Fig 2.10 Effect of imbalances in the receiver on constellation

2.6 Effect of Imbalances in the Receiver

The receiver imbalances are discussed in this section. From the asymmetric model of the receiver imbalances, we have:

$$\mathbf{D}=\begin{bmatrix} 1 & 0 \\ \alpha_R\sin(\phi_R) & \alpha_R\cos(\phi_R) \end{bmatrix} \tag{2.34}$$

$$\mathbf{Y}=\mathbf{DX}+\mathbf{DN} \tag{2.35}$$

The two equations show that the I/Q imbalances in the receiver have an effect on constellation and two channel noises are not orthogonal anymore. We can see the imbalances in the receiver effect on the constellation, as shown in Fig. 2.5. The distortion of the constellation definitely degrades the system performance.

From Eq. (2.4), an exponent form is obtained:

$$\begin{aligned} l_2(t) &= \cos(\omega_c t)+j\alpha_R\sin(\omega_c t+\phi_R) \\ &= k_3 e^{-j\omega_c t}+k_4 e^{j\omega_c t} \end{aligned} \tag{2.36}$$

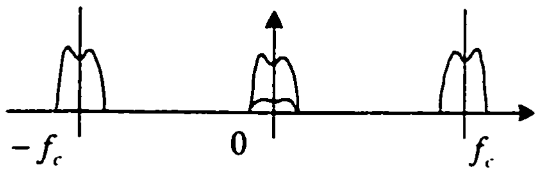


Fig. 2.11 Receiver imbalance effect on the baseband spectrum

$$\begin{aligned} k_3 &= \frac{1-\alpha_R e^{-j\phi_R}}{2} \\ k_4 &= \frac{1+\alpha_R e^{j\phi_R}}{2} \end{aligned} \tag{2.37}$$

When there are no imbalances in the receiver, $k_3 = 0$ and $k_4 = 1$. A demodulated signal is:

$$\begin{aligned} y(t) &= r(t)l_2(t) \\ &= \{\mathrm{Re}[x(t)e^{-j\omega_c t}] + n(t)\} * \{k_3 e^{-j\omega_c t} + k_4 e^{j\omega_c t}\} \\ &= \{\mathrm{Re}[x(t)e^{-j\omega_c t}]k_4 e^{j\omega_c t} + \mathrm{Re}[x(t)e^{-j\omega_c t}]k_3 e^{-j\omega_c t}\} + noise \end{aligned} \tag{2.38}$$

If receiver imbalances occur, $k_3 \neq 0$, then there is an additional signal produced in the baseband, as shown in Fig. 2.11.

In a multi-user system with a low-IF receiver, I/Q imbalances may introduce image tones from a neighboring channel [6][7]. This is a more challenging problem. Some digital signal processing algorithms are proposed in [6][7].

2.7 Imbalance Effect on Bit Error Rate (BER)

From the previous analysis, I/Q imbalances both in the transmitter and the receiver all affect the constellation. Sometimes the imbalances both in the transmitter and receiver complement each other so that total effect on the constellation is less than the effect from one of them alone. Sometimes the total effect on the constellation is larger than the effect from any one of them alone. It depends on the polar of phase imbalances both ain the transmitter and receiver. Fig 2.6 shows the worst case.

Gain and phase imbalances in the transmitter affect the signal constellation, furthermore affect the Bit Error Rate (BER). The performance is determined by the

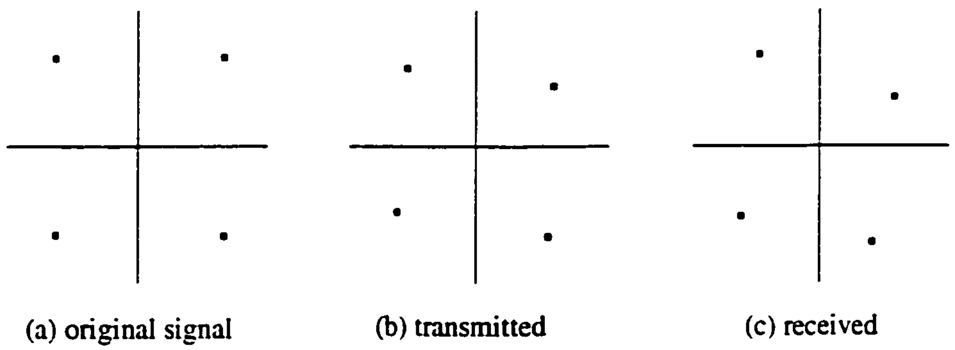


Fig. 2.12 Imbalances effect on the constellation

average bit-error probability associated with the demodulated data streams. If only the imbalances in the transmitter are considered, we have:

$$\begin{aligned} Y_I(n) &= I(n) + \alpha_T \sin(\phi_T) Q(n) + N_I(n) \\ Y_Q(n) &= \alpha_T \cos(\phi_T) Q(n) + N_I(n) \end{aligned} \tag{2.39}$$

When I(n), Q(n) and imbalances occur, The conditional probability of $Y_I(n)$, $Y_Q(n)$ is represented by:

$$\begin{aligned} &p(Y_I(n), Y_Q(n) \mid I(n), Q(n)) \\ &= \frac{1}{\pi N_0} \exp\{-\frac{\{Y_I(n) - [I(n) + \alpha_T \sin\phi_T Q(n)]\}^2 + \{Y_Q(n) - \alpha_T \cos\phi_T Q(n)\}^2}{N_0}\} \end{aligned} \tag{2.40}$$

Assuming a hard decision is made and I(n) and Q(n) has a uniform density distribution, respectively, the bit error probability of in-phase channel is written as:

$$\begin{aligned} P_{b,1}(\alpha_T, \phi_T) = &\frac{1}{8}\{1 - P(Y_I(n) > 0 \mid I(n) = +1, Q(n) = +1)\} \\ &+ \frac{1}{8}\{1 - P(Y_I(n) < 0 \mid I(n) = -1, Q(n) = +1)\} \\ &+ \frac{1}{8}\{1 - P(Y_I(n) < 0 \mid I(n) = -1, Q(n) = -1)\} \\ &+ \frac{1}{8}\{1 - P(Y_I(n) > 0 \mid I(n) = +1, Q(n) = -1)\} \end{aligned} \tag{2.41}$$

From Eq (2.39) and Eq (2.40), we obtain:

$$P_{b,1}(\alpha_T,\phi_T)=\frac{1}{8}Q(\sqrt{\frac{2E_b}{N_0}}(1+\alpha_T\sin\phi_T))+\frac{1}{8}Q(\sqrt{\frac{2E_b}{N_0}}(-1+\alpha_T\sin\phi_T)))$$
$$+\frac{1}{8}Q(\sqrt{\frac{2E_b}{N_0}}(-1-\alpha_T\sin\phi_T))+\frac{1}{8}Q(\sqrt{\frac{2E_b}{N_0}}(1-\alpha_T\sin\phi_T)) \quad (2.42)$$

Similarly, the bit error probability of quadrature channel is written as:

$$P_{b,2}(\alpha_T,\phi_T)=\frac{1}{4}Q(\sqrt{\frac{2E_b}{N_0}}(\alpha_T\cos\phi_T))+\frac{1}{4}Q(\sqrt{\frac{2E_b}{N_0}}(-\alpha_T\cos\phi_T))) \quad (2.43)$$

The average bit-error probability for QPSK signals is:

$$P_b(\alpha_T,\phi_T)=P_{b,1}+P_{b,2} \quad (2.44)$$

If α_T and ϕ_T are random variables,

$$P_b=\iint P_b(\alpha_T,\phi_T)f(\alpha_T,\phi_T)d\alpha_T d\phi_T \quad (2.45)$$

where $f(\alpha_T,\phi_T)$ is the joint probability density distribution of random variables α_T and ϕ_T.

Imbalance Effect on Bit Error Rate (BER) can be shown by simulation results. The effect of the transmitter imbalances on Bit-Error Rate (BER) is shown in Fig. 2.13. In Fig. 2.13, there are two lines, one line without imbalances, another with imbalances. The line with the I/Q imbalances is under the condition: gain imbalance ε is a constant with a value of 10% of the reference gain or 0.83 dB, and phase imbalance is a random variable of a uniform distribution with a range of [5.5°, 12.5°]. At this condition, Signal-Noise-Ratio (SNR) loses about 1.3 dB at Signal-Noise-Ratio (SNR) due to the imbalances. The simulation results indicate that imbalance degradation rises as the signal-noise-ratio is getting higher.

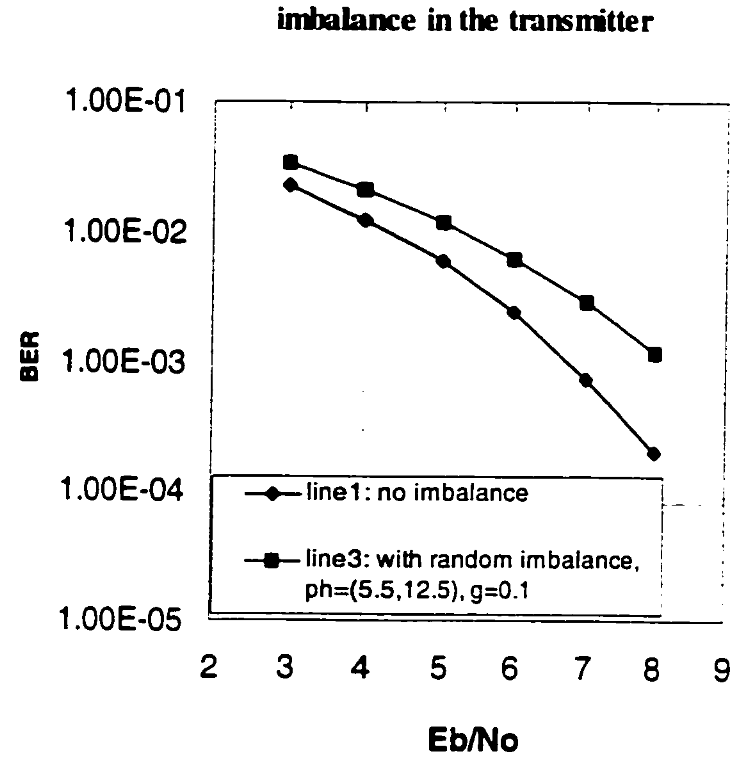


Fig. 2.13 Transmitter I/Q imbalance effect on performance

Fig. 2.14 shows simulation results of the receiver imbalances. In order to compare these with the simulation results of the transmitter imbalances, the receiver imbalances have the same statistic as that of transmitter imbalances. At this condition, Signal-Noise-Ratio (SNR) loses about 1.2 dB at Signal-noise-ratio of 6 dB due to the imbalances. This indicates that the receiver imbalances have a little bit less impairment than that of the transmitter imbalances.

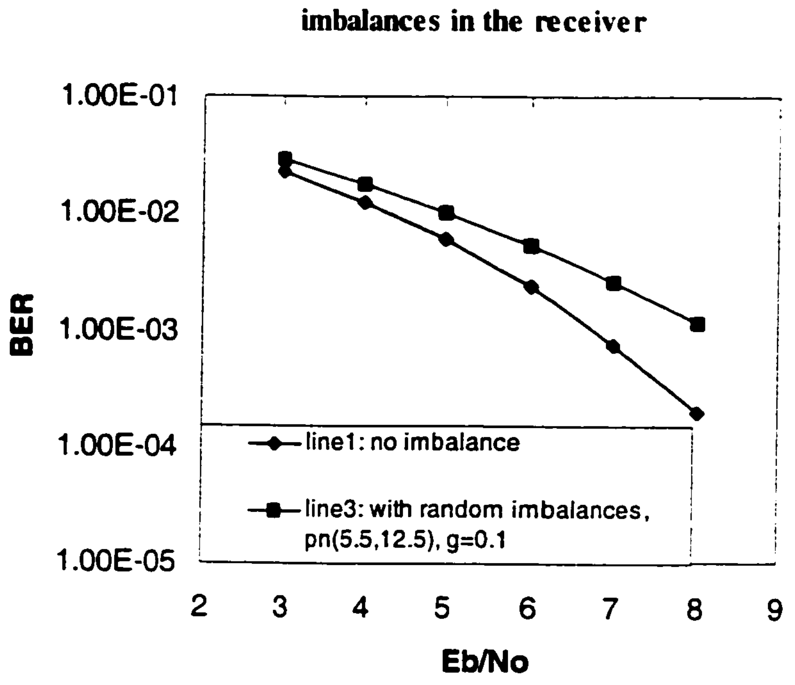


Fig. 2.14 Effect of I/Q imbalances of the receiver on performance

Chapter 3

Detection and Compensation of the Imbalances

In this chapter, general parameter estimation technologies are first introduced. Subsequently, I introduce our contribution to the imbalance detection and compensation.

3.1General Parameter Estimation Technologies

From previous analysis, I/Q imbalances can be seen as unknown random parameters. Before the imbalances are compensated, the random parameters have to be detected first. Imbalance detection is actually a parameter estimation issue. We can use parameter estimation to detect the imbalances. There are several estimation technologies that can solve this kind of problem, as shown in Fig. 3.1.

Parameter estimation technologies can be classified into three sorts:

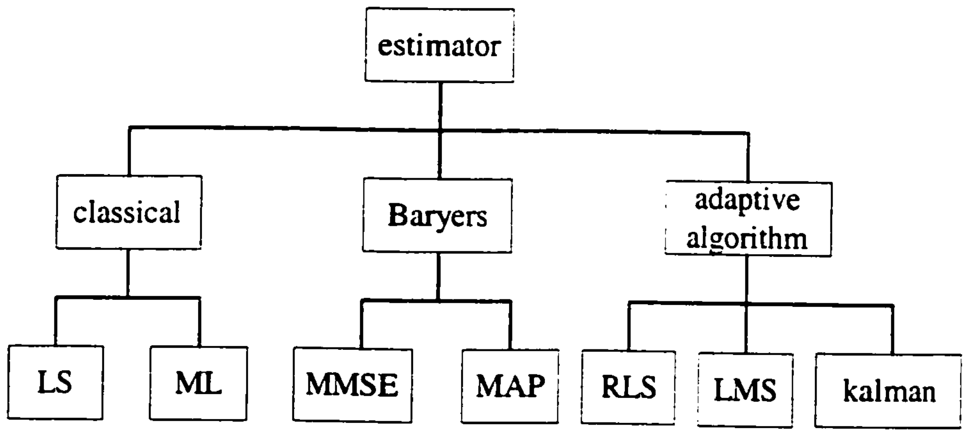


Fig. 3.1 Categories of estimation technologies

- Classical method: Least Square estimation (LS), Minimum Likelihood estimation (ML).
- Bayesian method: Maximum a Posteriori estimation (MAP), Minimum Mean Square Error estimation (MMSE).
- Adaptive algorithm: Recursive Least Square estimation, Least Mean Square estimation.

Least Square estimation (LS) and Minimum Likelihood estimation (ML) are still the most often used estimation methods because they do not need to know the statistic of the parameters.

In terms of data dependency, estimation technologies may be classified into:

- Data-aided (DA), Decision-directed (DD)
- Non-Data-aided (NDA).

3.1.1 Maximum Likelihood Estimation (ML)

It is well known that ML estimation is a special case of maximum a Posterior (MAP) estimate. So, we first consider MAP estimate. MAP estimation $\hat{\theta}_{MAP}$ may be viewed as a parameter vector that maximizes the posterior probability density function (pdf) $f_{\theta/Y}(\theta/y)$. The MAP estimation corresponds to a Bayesian estimation with a so-called uniform cost function defined as

$$C(\hat{\theta},\theta)=1-\delta(\hat{\theta},\theta) \tag{3.1}$$

where $\delta(\hat{\theta},\theta)$ is a delta function. Substitution of the cost function in the Bayesian risk equation yields:

$$\begin{aligned}\Re_{MAP}(\hat{\theta}/y) &= \int_{\Theta}[1-\delta(\hat{\theta},\theta)]f_{\Theta/Y}(\theta/y)d\theta \\ &= 1 - f_{\Theta/Y}(\hat{\theta}/y)\end{aligned} \tag{3.2}$$

From Eq. (3.2), the minimum Bayesian risk estimate corresponds to the maximum of the posterior function. Hence, the MAP estimate of the parameter vector θ is obtained from a minimization of the risk Eq. (3.2) as

$$\hat{\theta}_{MAP} = \arg\max_{\theta} f_{\Theta/Y}(\theta/y) \tag{3.3}$$

For most densities of interest, we can find $\hat{\theta}_{MAP}$ as a solution to the equation:

$$\left.\frac{\partial \ln p(\theta \mid y)}{\partial \theta}\right|_{\hat{\theta}=\theta_{MAP}(y)} = 0 \tag{3.4}$$

By the Bayers rule, we may rewrite Eq.(3.4) to separate the role of the observation y and the prior knowledge:

$$\left.\frac{\partial \ln p(y \mid \theta)}{\partial \theta}\right|_{\theta=\hat{\theta}(y)} + \left.\frac{\partial \ln p(\theta)}{\partial \theta}\right|_{\theta=\hat{\theta}(y)} = 0 \tag{3.5}$$

The first term gives the dependence on y and the second corresponds to the prior knowledge.

When second term $\left.\frac{\partial \ln p(\theta)}{\partial \theta}\right|_{\theta=\hat{\theta}(y)} = 0$, it means

$$\left.\frac{\partial \ln p(y \mid \theta)}{\partial \theta}\right|_{\theta=\hat{\theta}(y)} = 0 \tag{3.6}$$

In this case, MAP is ML. The maximum likelihood (ML) estimation $\hat{\theta}_{ML}$ is obtained as the parameter vector that maximizes the likelihood function $f_{Y/\Theta}(y/\theta)$. It is represented as:

$$\hat{\theta}_{ML} = \arg\max_{\theta} f_{y/\Theta}(y/\theta) \tag{3.6}$$

With a Bayesian framework, the main difference between the ML and MAP estimator is that the ML assumes that the prior pdf of θ is a uniform. A uniform prior is also used when the parameter prior pdf is unknown, or when the parameter is an unknown constant.

3.1.2 Minimum Mean Squared Error (MMSE) Estimation

The minimum mean squared error (MMSE) estimate is obtained as the parameter vector that minimizes a mean error cost function defined as:

$$\begin{aligned}\Re_{MSE}(\hat{\theta}/y) &= E\{(\hat{\theta}-\theta)^2/y\} \\ &= \int_{\theta}(\hat{\theta}-\theta)^2 f_{\Theta/Y}(\theta/\text{y})d\theta\end{aligned} \tag{3.7}$$

Assuming that the mean squared error risk function is differentiable and has a well defined minimum, the MMSE solution can be obtained by setting the gradient of the mean square error risk function to zero:

$$\frac{\partial}{\partial\hat{\theta}}\Re_{MSE}(\hat{\theta}/y) = 2\hat{\theta} - 2\int_{\theta}\theta\, f_{\Theta/Y}(\theta/\text{y})d\theta \tag{3.8}$$

MMSE solution is:

$$\hat{\theta}_{MMSE} = \int_{\theta}\theta\, f_{\Theta/Y}(\theta/\text{y})d\theta \tag{3.9}$$

For the cases where we do not have a pdf model of the parameter process, the minimum mean squared error estimate is obtained through minimization of a mean square error function $E[e^2(\hat{\theta}/\text{y})]$:

$$\hat{\theta}_{MMSE} = \arg\min_{\hat{\theta}} E[e^2(\hat{\theta}/\text{y})] \tag{3.10}$$

The MMSE estimation of Eq. (3.10) does not use any prior knowledge of the distribution of the signals and parameters. This can be considered a strength in situations where the pdfs are unknown.

In communication systems, the relation between the observation vector **y** and the parameter vector θ is described as a linear model:

$$\mathbf{y} = \mathbf{G\theta} + \mathbf{n} \tag{3.11}$$

Where **n** is Gaussian noise, y is an observation vector, and matrix G is known. If we know covariance $\mathbf{R}_n = N_0\mathbf{I}$ of vector **n** and covariance $\mathbf{R}_\theta$ of vector θ, the MMSE of θ is obtained [19]:

$$\hat{\theta}_{\text{MMSE}} = (\mathbf{G}^T\mathbf{G} + N_0\mathbf{R}_\theta^{-1})^{-1}\mathbf{G}^T\mathbf{y} \tag{3.12}$$

When $N_0\mathbf{R}_\theta^{-1} \to 0$ MMSE estimator approaches LS estimator in form:

$$\hat{\theta}_{\text{MMSE}} = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{y} = \hat{\theta}_{LS} \tag{3.13}$$

LS algorithm is the most often used estimation technology.

3.2 Imbalance Detection and Compensation in the Transmitter

If we detect and correct the imbalances before the nonlinear PA, intermodulation products due to the nonlinearity of PA may be decreased greatly. Considering there is only little thermal noise in the transmitter and gain and phase imbalances are random variables, Least square (LS) estimation performs as good as MMSE. Envelope detector can be used for decreasing complexity. From Eq. (2.26), passing sampling, the square of the amplitude is represented by A:

$$A(n) = 1 + a + bI(n)Q(n) \tag{3.14}$$

$$a = (\alpha_T)^2 \tag{3.15}$$

$$b = 2\alpha_T \sin\phi_T \tag{3.16}$$

It can be represented in matrix form:

$$\mathbf{A} = \mathbf{G}\boldsymbol{\theta} \tag{3.17}$$

Where

$$\mathbf{A} = \begin{bmatrix} A(1) \\ A(2) \\ \vdots \\ A(N) \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} 1 & I(1)Q(1) \\ 1 & I(2)Q(2) \\ \vdots & \vdots \\ 1 & I(N)Q(N) \end{bmatrix}, \quad \boldsymbol{\theta} = \begin{bmatrix} a \\ b \end{bmatrix} \tag{3.18}$$

From the Eq. (3.17), the mean square error of the estimation of $\boldsymbol{\theta}$ is:

$$\xi = E\{\|\mathbf{A} - \mathbf{G}\hat{\boldsymbol{\theta}}\|^2\} \tag{3.19}$$

Finding the estimation $\hat{\boldsymbol{\theta}}$ of minimizing Eq. (3.19) :

$$\frac{\partial\xi}{\partial\hat{\boldsymbol{\theta}}} = 0 \tag{3.20}$$

$$\hat{\boldsymbol{\theta}} = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{A} \tag{3.21}$$

Although this is Least Square estimation expression, it is also MMSE estimation result because of no noise.

From Eq. (3.18) and Eq. (3.21), we can obtain:

$$\begin{bmatrix} \hat{a} \\ \hat{b} \end{bmatrix} = \begin{bmatrix} 1 & \left(\frac{1}{N}\sum_{n=1}^{N} I(n)Q(n)\right) \\ \left(\frac{1}{N}\sum_{n=1}^{N} I(n)Q(n)\right) & 1 \end{bmatrix}^{-1} \begin{bmatrix} \left(\frac{1}{N}\sum_{n=1}^{N} A(n)\right) - 1 \\ \left(\frac{1}{N}\sum_{n=1}^{N} A(n)I(n)Q(n)\right) - \left(\frac{1}{N}\sum_{n=1}^{N} I(n)Q(n)\right) \end{bmatrix} \tag{3.22}$$

Assume we can make:

$$\frac{1}{N}\sum_{n=1}^{N} I(n)Q(n) = 0 \tag{3.23}$$

Eq. (3.22) is rewritten as:

$$\begin{bmatrix}\hat{a}\\ \hat{b}\end{bmatrix} = \begin{bmatrix}\left(\frac{1}{N}\sum_{n=1}^{N} A(n)\right) - 1\\ \left(\frac{1}{N}\sum_{n=1}^{N} A(n)I(n)Q(n)\right)\end{bmatrix} \tag{3.24}$$

$$\alpha^2{}_T \cong \hat{a} \tag{3.25}$$

$$\alpha_T \sin\phi_T \cong \frac{\hat{b}}{2} \tag{3.26}$$

$$\alpha_T \cos\phi_T = \sqrt{\alpha_T^2(1-\sin^2\phi_T)} \cong \sqrt{\hat{a} - (\frac{\hat{b}}{2})^2} \tag{3.27}$$

The imbalance matrix in the transmitter approximately is:

$$\mathbf{M} \cong \begin{bmatrix} 1 & \frac{\hat{b}}{2} \\ 0 & \sqrt{\hat{a} - (\frac{\hat{b}}{2})^2} \end{bmatrix} \tag{3.28}$$

The basic idea for compensation is to find a matrix **C** that makes **CM** approach unit matrix **I**. Obviously, **C** is the compensating matrix. Let,

$$\mathbf{C} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} = \mathbf{M}^{-1} \cong \begin{bmatrix} 1 & \frac{-\hat{b}}{\sqrt{4\hat{a} - \hat{b}^2}} \\ 0 & \frac{2}{\sqrt{4\hat{a} - \hat{b}^2}} \end{bmatrix} \tag{3.29}$$

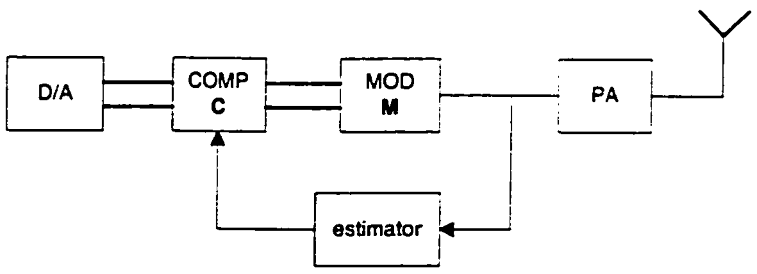


Fig. 3.2 Diagram for backward compensation at the transmitter

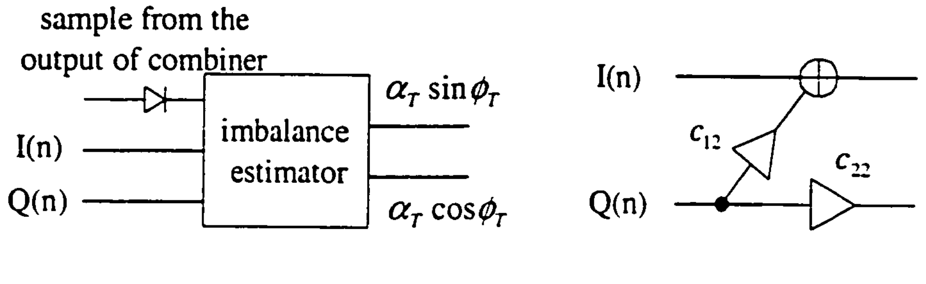


Fig. 3.3 (a) estimator, (b) compensator

Fig. 3.1 shows our compensation scheme, and the estimator and compensator are shown in Fig.3.2

Update of compensation parameter matrix is obtained by:

$$\mathbf{C} = \mathbf{C}_1\mathbf{C}_2 = \mathbf{M}_1^{-1}\mathbf{M}_2^{-1} \tag{3.30}$$

where $\mathbf{C}_1$ is the previous parameter compensator matrix, and $\mathbf{C}_2$ is a current parameter compensator matrix.

Simulation results of the compensation are shown in Fig 3.3. One line is the result without imbalances, and the other two lines are at the condition of imbalances: gain imbalance ε is a constant with a value of 10% of the reference gain or 8.3 dB, and phase imbalance is a random variable of a uniform distribution with a range of (5.5, 12.5). From the simulation, we can see the compensation gain is obtained about 1.0 dB at 7 dB.

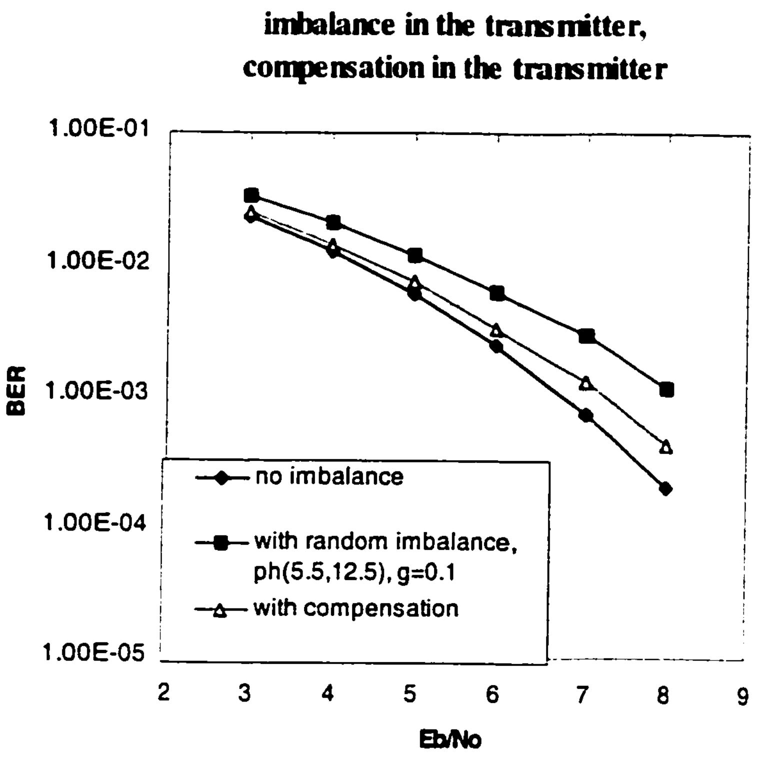


Fig. 3.4 Simulation result with compensation at the transmitter

3.3 Imbalance Detection and Compensation in the Receiver

In order to improve the system performance in the presence of the I/Q imbalances, we need to compensate the distortions due to the imbalances in the transmitter-receiver chain. A practical and effective compensation scheme is to compensate the imbalances in the receiver. In this section we propose a data-aided feed forward estimation method based on a ML estimation algorithm.

3.3.1 Estimation of Phase and Gain Imbalances at Receiver

In the receiver, we have the following relation according to Eq. (2.35):

$$\mathbf{Y} = \mathbf{HX} + \mathbf{DN} \tag{3.31}$$

For data aided algorithm, we have a N-symbol training sequence or N-symbol data from preambles or Pilot Symbol Assisted Modulation (PSAM), which can be represented by a vector:

$$[\mathbf{X}(1), \cdots \mathbf{X}(i) \cdots \mathbf{X}(\mathbf{N})] \qquad \mathbf{X}(i) = [I(i) \quad Q(i)]^T \tag{3.32}$$

Accordingly, the observation vector is:

$$[\mathbf{Y}(1), \cdots \mathbf{Y}(i), \cdots \mathbf{Y}(\mathbf{N})], \qquad \mathbf{Y}(i) = [Y_I(i) \quad Y_Q(i)]^T \tag{3.33}$$

In Eq. (3.32), the term of $\mathbf{DN}$ has not been an AWGN. Because the receiver imbalance has a smaller effect, we can still regard it as the AWGN for simplicity. We can find the conditional probability $f(\mathbf{Y}(1), \cdots \mathbf{Y}(\mathbf{N}) | \mathbf{X}(1) \cdots \mathbf{X}(\mathbf{N}), \mathbf{H})$:

$$f(\mathbf{Y}(1), \cdots \mathbf{Y}(N) | \mathbf{X}(1) \cdots \mathbf{X}(N), \mathbf{H}) \\ = \prod_{i=1}^{N} \frac{1}{(\pi N_0)^{1/2}} \exp\{-\frac{1}{N_0} [\mathbf{Y}(i) - \mathbf{HX}(i)]^T [\mathbf{Y}(i) - \mathbf{HX}(i)]\} \tag{3.34}$$

Where $\mathbf{H}$ is a 2×2 imbalance parameter matrix from EQ. (2.22) and noise covariance matrix $\mathbf{R}_n$ is a 2×2 dimensional matrix that is not orthogonal since the new noise matrix is obtained from Eq. (2.22). Taking log-likelihood of Eq.(3.34):

$$\log f(\mathbf{Y}(1), \cdots \mathbf{Y}(N) | \mathbf{X}(1) \cdots \mathbf{X}(N), \mathbf{H}) \\ = -N \log(\pi N_0)^{1/2} + \sum_{i=1}^{N} (-\frac{1}{N_0} (\mathbf{Y}(i) - \mathbf{HX}(i))^T (\mathbf{Y}(i) - \mathbf{HX}(i))) \tag{3.35}$$

Then, differentiating the log-likelihood, we get the equation below:

$$\frac{\partial}{\partial \mathbf{H}} \log f(\mathbf{Y}(1),\cdots \mathbf{Y}(N)|\mathbf{X}(1)\cdots \mathbf{X}(N),\mathbf{H}) = \sum_{i=1}^{N} (-\frac{1}{N_0}[\mathbf{Y}(i) - \mathbf{H}\mathbf{X}(i)]\mathbf{X}^T(i))\Big|_{\mathbf{H}=\hat{\mathbf{H}}} = 0 \qquad (3.36)$$

From this equation, the estimation $\hat{\mathbf{H}}$ is:

$$\hat{\mathbf{H}} = \left[\sum_{i=1}^{N} \mathbf{X}(i)\mathbf{X}^T(i)\right]^{-1} \sum_{i=1}^{N} \mathbf{Y}(i)\mathbf{X}^T(i) \qquad (3.37)$$

It is well known that ML estimation is the same as LS estimation in the case of the AWGN. So, we can obtain this result from LS estimation.

3.3.2 Compensation of Phase and Gain Imbalances

A way of compensating for the imbalance effects is to find out the inverse of the estimated **H** matrix and then use this inversed matrix to cancel out the imbalance effects in Eq. (3.31). Hence, we have,

$$\mathbf{Y}' = \hat{\mathbf{H}}^{-1}\mathbf{Y} = \mathbf{X} + \mathbf{N}' = \hat{\mathbf{X}} \qquad (3.38)$$

With the compensation method, the simulation result is shown in Fig. 3.4. The result is obtained under the same imbalance assumption as that in transmitter, i.e. the gain imbalance is a constant with a value of 10% of the reference gain or 8.3 dB, and phase imbalance is a random variable of a uniform distribution with a range of [5.5, 12.5]. From our simulation results, there is about 0.9 dB compensation gain from Signal-Noise-Ratio of 7 dB.

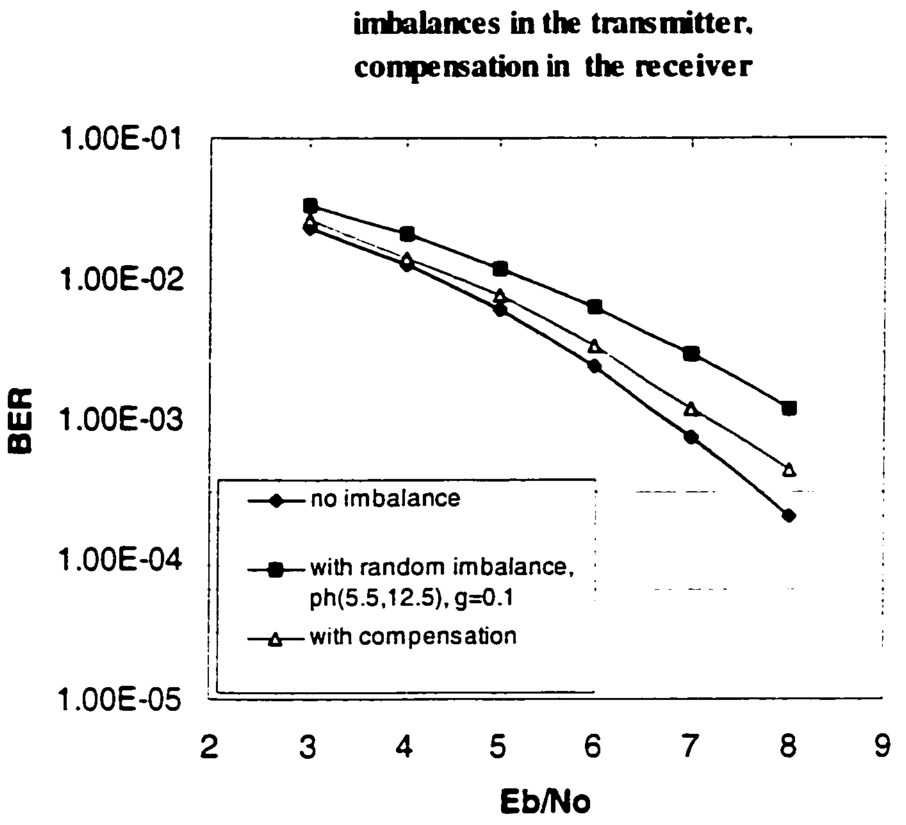


Fig. 3.5 Simulation result with compensation in the receiver

Chapter 4

Imbalance Effect on the Performance of the System with Convolutional Encoder and Viterbi Decoder

4.1 Introduction

In 1948. Shannon demonstrated that by proper encoding of the information, error induced by a noisy channel could be reduced to any desired level without sacrificing the rate of the information transmission. Since Shannon's work, a great deal of effort has been made on the problem of designing efficient encoding and decoding methods for error detection and correcting in a noisy environment.

Channel coding protects digital data from errors by selectively introducing redundancies in the transmitted data. Coding involves adding extra bits to the data stream so that the decoder can reduce or correct errors at the output of the receiver. However, the cost of the extra bits increases the data rate (bits/s) and, consequently, increases the bandwidth of the encoded signal.

Channel coding techniques can be divided into four categories: block codes, convolutional codes, concatenated codes, and turbo codes. We mainly focus on the convolutional codes with viterbi decode in this chapter and turbo codes in the following chapter.

Convolutional encoding with Viterbi decoding is a forward error correction (FEC) technique that is particularly suited to a channel in which the transmitted signal is corrupted mainly by AWGN. Convolutional codes were invented by Elias in 1955. Convolutional encoder contains memory and its outputs not only depend on the current inputs but also on previous input blocks. It converts the entire data stream into one single coded word. Most practical communication systems such as GSM, IS-95, etc. use convolutional codes. Viterbi algorithm performs the estimation of the input sequence of a discrete time finite-state Markov process observed in memoryless channel. The algorithm is a powerful and flexible means for decoding convolutional codes. This algorithm reduces the error probability of a sequence of received bits and has been of widespread use in varied applications involving maximum-likelihood estimation of Markov processes.

A significant discovery —turbo codes were introduced by Berrou, Glavieux and Thitimajshima in 1993 [14]. These powerful codes are capable of achieving near Shannon capacity. The codes are constructed by applying two or more constituent codes with different interleaved versions of the same information sequence. Decoding calls on iterative processing in which each component decoder takes advantage of the work of the other at the previous step. Since the introduction of turbo codes, numerous research efforts have been dedicated to optimize the components of turbo codes and their applications to wireless communication systems.

Phase and gain imbalances act as a spurious interference introduced by the mismatch of quadrature modulation. Since the imbalances are introduced between encoder and decoder, the total channel noise level increases. Therefore, the imbalances inevitably degrade the performance of the systems with coded scheme. Fig. 4.1 shows the system model for considering imbalances and channel coding applications.

In this chapter, convolutional encoder and Viterbi decoder will first be reviewed, and then the effect of gain and phase imbalances on the performance of the system with the coding scheme will be investigated.

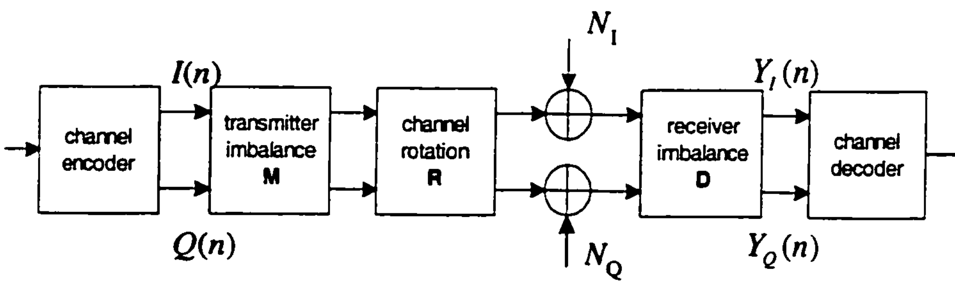


Fig. 4.1 Block diagram of a system with imbalances and AWGN channel

4.2 Encoding of Convolutional Codes

Fig. 4.2 shows a convolutional encoder with a one-bit input, a two-bits output and two memory elements. Usually we use (n, k, m) notation to describe a convolutional code. Here, n is the number of encoder outputs, k is the number of encoder inputs, and m is the number of the code memory. The integer K=m+1 is a parameter known as the constraint length. R_c=k/n is called code rate. Fig. 4.2 shows (2, 1, 2) convolutional encoder with constrain length K=3 and code rate R_c=½.

As shown in Fig. 4.2, a binary input data stream **u** is fed into a shift register having 2 memory elements. The input stream **u** is given by:

$$\mathbf{u} = (u_0, u_1, u_2 \cdots u_k \cdots u_{N-1}) \tag{4.1}$$

The output streams is denoted by:

$$\mathbf{v}^{(i)} = (v_0^{(i)}, v_1^{(i)}, \cdots v_k^{(i)} \cdots v_{N-1}^{(i)}) \tag{4.2}$$

An input bit produces two-bits output and hence the code rate of this encoder is 1/2. Each encoded output stream is formed by modulo-2 addition of input bit with tapped values from memory elements in the shift register according to a fixed pattern. This fixed pattern of the tap positions is called the generator sequence of the encoder. The generator sequence $g_i^{(j)}$ is the impulse response obtained at the j^{th} output of the encoder by applying a single 1 at the i^{th} input followed by a string of zeros. Since the encoder in Fig.

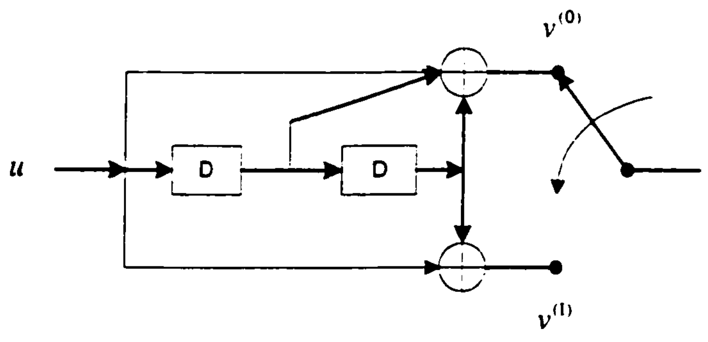


Fig. 4.2 a (n, k, m)=(2,1,2) convolutional encoder

4.2 has only one-bit binary input, the generator sequences of this encoder are given as:

$$g^{(0)} = (1\ 1\ 1)$$
$$g^{(1)} = (1\ 0\ 1) \tag{4.3}$$

The overall generator comprising of both the feed-forward generators is usually represented in octal. The octal generator for the encoder in Fig. 4.2 is (7,5). These generator sequences, thus, can be read from the encoder diagram through an examination of the tap positions. The output of the single-input encoder in terms of the generator sequences is given by:

$$v_l^{(j)} = \sum_{i=0}^{m} u_{l-i} g_i^{(j)} \tag{4.4}$$

The output is a discrete convolution of the input and the impulse response g, hence, the code is named "convolutional codes".

There are two methods that are often used to describe convolutional codes. These are the trellis diagram and the state diagram. The state diagram is simply a graph of the possible states of an encoder and possible transitions from one state to another. The state diagram of Fig. 4.2 is shown in Fig. 4.3. The encoder has two memory elements and hence, it can be in any of the four different states at a particular time. The diagram also shows the transitions from each state to the others. The transitions are marked as X/YY where X is the input bit that causes the transition and YY are the two bits produced at the output as a result of encoding the input bit.

The performance of a convolutional code is usually measured in terms of the code's minimum free distance. The minimum free distance, d_{free}, of a convolutional code is the minimum Hamming distance between all pairs of complete convolutional code words. The minimum free distance for the encoder in Fig. 4.2 can easily be evaluated from the state diagram in Fig. 4.3. Starting from state 0, d_{free} is the minimum total Hamming weight over all the paths traced by the encoder until it comes back to state 0. The Hamming weight of a particular path is the number of ones in the output associated with the path. Therefore, the path traced by the encoder leaving state 0 and coming back

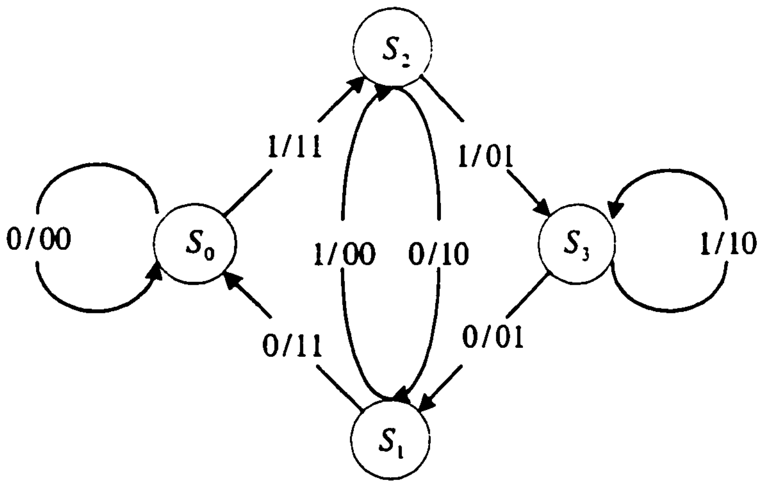


Fig. 4.3 state transition of convolutional code (2,1,2)

to state 0 such that the Hamming weight is minimum is $S_0 \to S_2 \to S_1 \to S_0$ and the $d_{free} = 5$.

The trellis diagram can be used to explicitly show the passage of time. Such a diagram for the encoder of Fig. 4.2 is shown in Fig. 4.4. The left side and right side of Fig 4.4 are the same trellis with the only difference that the right one labels the two-bit numbers of encoder outputs in order to examine the left explicitly. The dotted lines represent the cases where the encoder input is zero, and solid lines represent the cases where the encoder input is one. If state is 01 and input is 1 at time t, output shows 00 and next state will be 10.

Now let a state start from 00, and input a sequence 01011, that is, input 0 at t=0, input 1 at t=1, input 0 at t =2, input 1 at t=3, input 1 at t=4. The path corresponding the input message is shown in bold. The outputs are 00, 11, 10, 00, 01. When more bits input, the path on the trellis will be continuous to extend. If all inputs are zero, output bits become zero and all paths end at state 0 after first m input bits are zero.

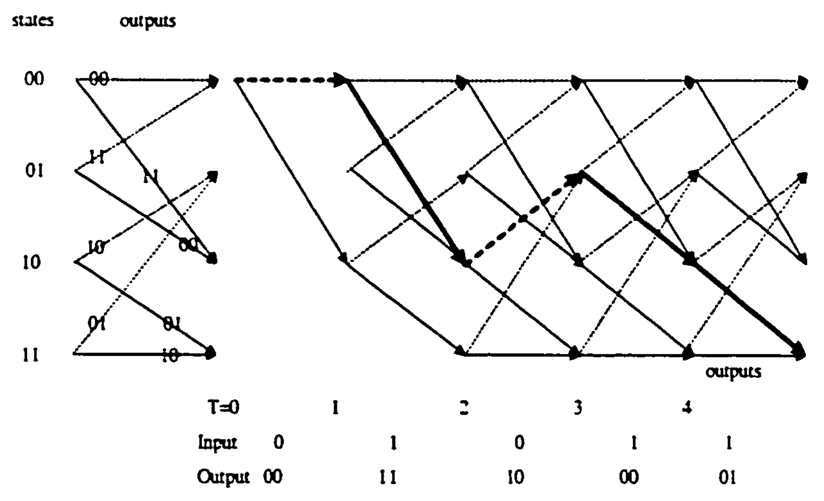


Fig. 4.4 Trellis diagram for convolutional code (2. 1. 2)

4.3 Decoding of Convolutional Codes: Viterbi Algorithm

As the encoder transforms the information sequence into a convolutional code word, the code word is modulated and transmitted across the channel. The corrupted version of the transmitted sequence is received at the receiver. The convolutional decoder at the receiver has to produce an estimate of the transmitted code word by evaluating the noisy received vector.

The Viterbi algorithm (VA) is a maximum-likelihood (ML) decoding algorithm, which produces an estimate $\hat{v}$ that maximizes the probability $p(\mathbf{y} / \mathbf{v})$ [22]. Let the input block $\mathbf{u}$ be of length L. The corresponding output code word is of length $n(L+m)$ where m is the number of memory elements in the encoder. The decoder receives the contaminated code word of length $n(L+m)$ and has to produce an estimate $\hat{v}$ of each of these code bits. The channel is assumed to be memoryless channel. Let $x_{l,j}$ represent the modulated data sequence, it is obtained:

$$x_{l,j} = 1 - 2 \times v_l^{(j)} \tag{4.5}$$

If the received contaminated data sequence correponding to $x_{l,j}$ is $y_{l,j}$. Viterbi algorithm maximizes the joint probability distribution below [13].

$$p(\mathbf{y}/\mathbf{x}) = \prod_{l=0}^{L+m-1} p(\mathbf{y}_l/\mathbf{x}_l) = \prod_{l=0}^{L+m-1}\prod_{j=0}^{n-1} p(y_{l,j}/x_{l,j}) \tag{4.6}$$

A log-likelihood function $\log P(\mathbf{r}/\mathbf{x})$ is denoted by:

$$\log p(\mathbf{y}/\mathbf{x}) = \sum_{l=0}^{L+m-1}\sum_{j=0}^{n-1} p(y_{l,j}/x_{l,j}) \tag{4.7}$$

For AWGN channel, the log-likelihood function is shown below:

$$\log P(\mathbf{y}/\mathbf{x}) = \sum_{l=0}^{L+m-1}\sum_{j=0}^{n-1} \log\frac{1}{\sqrt{2\pi}\sigma}\exp\{-\frac{(y_{l,j}-x_{l,j})^2}{2\sigma^2}\} \tag{4.8}$$

Maximizing the above equation is equivalent to minimizing its Euclidean distance. The Euclidean distance is represented by $M(\mathbf{y}/\mathbf{x})$:

$$M(\mathbf{y}/\mathbf{x}) = \sum_{l=0}^{L+m-1}\sum_{j=0}^{n-1}(y_{l,j}-x_{l,j})^2 \tag{4.9}$$

Omitting common parts, a Metric is written below:

$$M(\mathbf{y}/\mathbf{x}) = \sum_{l=0}^{L+m-1}\sum_{j=0}^{n-1} y_{l,j}x_{l,j} = \sum_{l=0}^{L+m-1} M(\mathbf{y}_l/\mathbf{x}_l) \tag{4.10}$$

where $M(\mathbf{y}_l/\mathbf{x}_l) = \sum_{j=0}^{n-1} y_{l,j}x_{l,j}$ is called a branch metric. Maximizing this metric is the same as the operation of maximizing log-likelihood function in Eq. (4.8)

From the trellis diagram, such as Fig. 4.4, the Viterbi algorithm is applied to compute the maximum likelihood (ML) path. For each state, there are 2^k branches entering the state where k is the number of encoder inputs. With each branch associated to the state, an accumulated metric can be computed by adding the branch metric with the corresponding previous state metric. Take maximum accumulated metric from 2^k accumulated metric as a survivor. The maximum survivor metric is the current state metric. Applying the procedure to all the states, we can obtain all the state metrics. Viterbi Algorithm is summarized below:

1. Set initial state, for example S_0=00 at time t=0.
2. Calculate the accumulated metric for each current state nod at time $t = l$. Take a state and find survivor metric for the state. Store the survivor metric as the accumulated metric. Then repeat above procedure to calculate accumulated metric for each state until all of states are computed.
3. If $t < L+m$, $t = l+1$ and return to step 2. At $t = L+m$, all the branches close to 0 state. With this state, only one branch survives.

4.4 Performance of Convolutional Codes

Usually, the received channel symbols are quantized by one or a few bits of precision before they are inputted to the decoder. If received symbols are quantized to one-bit precision, the process is called hard decision. Otherwise, it is called soft decision. A Viterbi decoder with soft decision inputs has better performance than that with hard decision [10].

The performance of convolutional codes with soft decision decoding is measured by the first error event probability P_e. P_e is the probability that an error event (when the decoder chooses a path on the trellis diverging from the correct path) begins during the current time interval. The first-error event probability P_e is represented as following [13]:

$$P_e \leq \sum_{d=d_{free}}^{\infty} a_d P_2(d) \qquad (4.11)$$

where a_d is the number of paths through the trellis with output weight d and d_{free} is the minimum distance of the code as described in section 4.2. $P_2(d)$ is called the pairwise error event probability, and is given as:

$$P_2(d) = Q\left(\sqrt{\frac{2E_b r_b d}{N_0}}\right) \tag{4.12}$$

where r_b is a code rate, and $r_b = 1/2$ in our application.

Although the first–event error probability provides a measure of the performance of a convolutional code, a more useful measure of performance is the bit error probability. The bit error probability for $k = 1$ is upper-bounded by:

$$P_b < \sum_{d=d_{free}}^{\infty} \beta_d P_2(d) \tag{4.13}$$

where $\beta_d = a_d f(d)$. The function $f(d)$ denotes a function of hamming distance d derived from the transfer function $T(D,N)$ [13].

The price of soft decision increases with the complexity of the decoder. A Viterbi decoder with soft decision data inputs quantized to three bits of precision can perform about 2dB gain better than one working hard decision inputs [10]. More quantized bits provide little additional improvement. In our study, a soft decision of three bits is used.

4.5 Gain and phase Imbalance Effects on the BER

In previous discuss of this chapter, a simple convolutional code with K=3 and Viterbi decoding are addressed. In a more practical application, a convolutional code with K=7 is often used. As shown in Fig. 4.5 the generator sequences of this encoder are given as:

$$g^{(0)} = (1,\ 1,1,1,\ 0,0,1)$$
$$g^{(1)} = (1,\ 0,1,1,\ 0,1,1)$$

When the gain and phase imbalances occur in the system, as shown in Fig. 4.1, with such an encoder and a Viterbi decoder, the system performance will degrade. The simulation results of the system with up-converter imbalances are shown in Fig. 4.6. The line 1

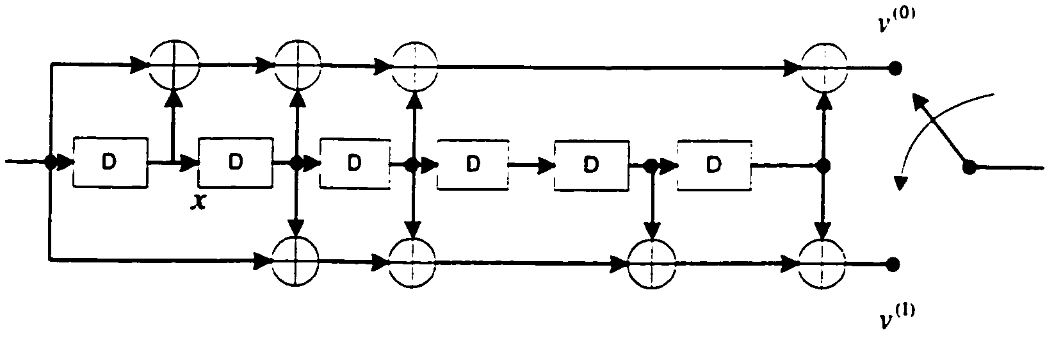


Fig. 4.5 convolutional encoder with constrain length K=7

shows the performance without imbalances. Considering the worst case, the maximum constant imbalance of phase is 10% of $90°$, i.e. $9°$. Line 2 shows the simulation result of a constant phase imbalance with a phase of $9°$. Comparing line 2 with line 1, we see that Viterbi decoding have a good correction to the phase imbalance. At Signal-Noise-Ratio of 5 dB, the performance is only degraded about 0.2 dB. Generally the phase imbalance is a random variable. For simplicity, we assume that the phase imbalance is a random variable with a uniform distribution. Line 3 shows the performance with the random imbalances, of which the phase imbalance has a uniform density distribution with a range of $[5.5°,12.5°]$, and the gain imbalance is a constant with a value of 10% of the reference gain or 0.83 dB. At Signal-to-Noise-Ratio of 5 dB, the performance is also degraded about 0.2 dB. Comparing the degradation in Fig 4.6 with the degradation of uncoded system in Fig. 2.13 at the same Bit-Error-Rate (BER), the Viterbi decoding has a powerful ability to correct the I/Q imbalances.

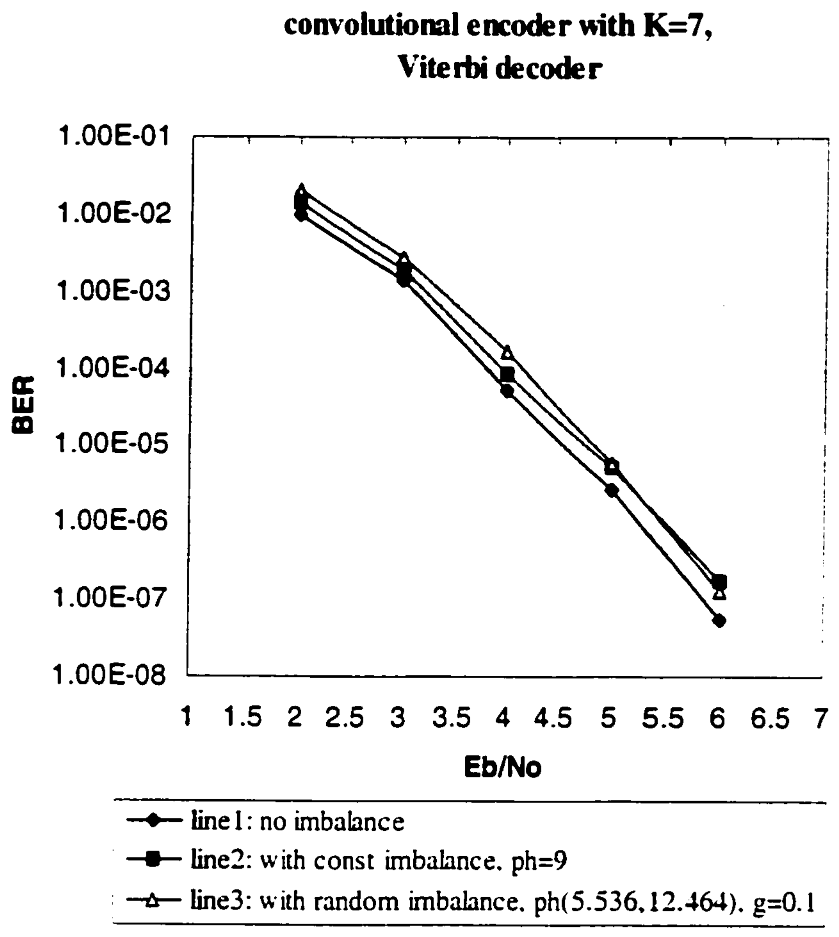


Fig. 4.6 The effect of imbalances

Chapter 5

Imbalances Effect on the Performance of the System with Turbo Code

5.1 Introduction

Turbo codes, introduced by Berrou *et al* [14] in 1993, were a major breakthrough towards realizing Shannon's channel capacity limit. Since its recent invention, turbo coding has evolved at an unprecedented rate and has reached a state of maturity within just a few years due to the intensive research efforts of the turbo coding community. As a result, turbo coding has also found its way into standardized systems, such as the recently standardized third generation (3G) mobile radio systems. Even more impressive performance gains can be attained with the aid of turbo coding in the context of video broadcast systems. Turbo code research not only involves in additive white gaussian noise (AWGN), but also extends to fading channel, intersymbols interference (ISI.). However, no one refers to phase and gain imbalance effect on the performance with turbo code. In this chapter, the issue about phase and gain imbalances will be investigated.

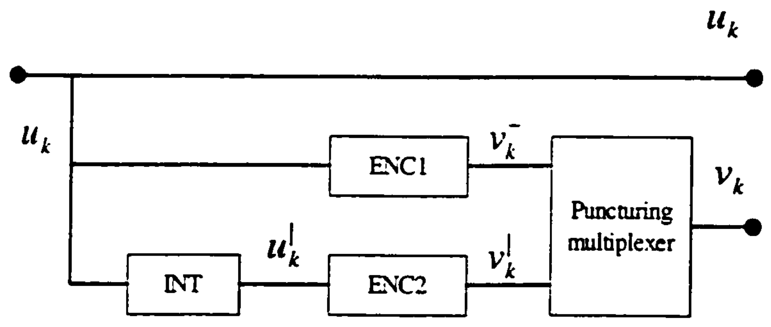


Fig. 5.1 Generic turbo encoder structure

Turbo code encoder structure is a parallel concatenation of two or more convolutional codes (PCCC), each separated by a non-uniform interleaver. A turbo code encoder with two convolutional codes is shown in Fig. 5.1. The constituent encoders used in turbo code are typically recursive systematic convolutional (RSC) encoders, also, in most turbo codes the constituent encoders used are the same. The interleaver is used to scramble the order of the input bits before feeding them into the second encoder (a deinterleaver in the receiver reverses the scrambling effect of the interleaver). In the general case, turbo code consists of two parts: the uncoded information bits and a set of parity sequences generated by passing interleaved versions of the information bits through convolutional encoders. In Fig. 5.1, two sets of parity bits are used, one is generated from the non-interleaved data sequence, and another is generated from an interleaved sequence. The parity bits are usually punctured in order to raise the code rate. The data sequence may or may not be terminated, usually depending on the kind of interleaver used.

An iterative decoder structure is shown in a simplified format in Fig. 5.2 [16]. The SISO decoder blocks use a modified Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm or a Soft-Output Viterbi Algorithm (SOVA). If the Turbo code consists of more components, it is just a matter of inserting the relevant deinterleaver, decoder, interleaver blocks. In the first iteration the first component decoder takes channel output values only, and produces a soft output as its extrinsic information to the second decoder. The a priori information is usually reset to be equiprobable before starting to decode a frame (unless we know that the information bits have a particular probability pattern). The extrinsic

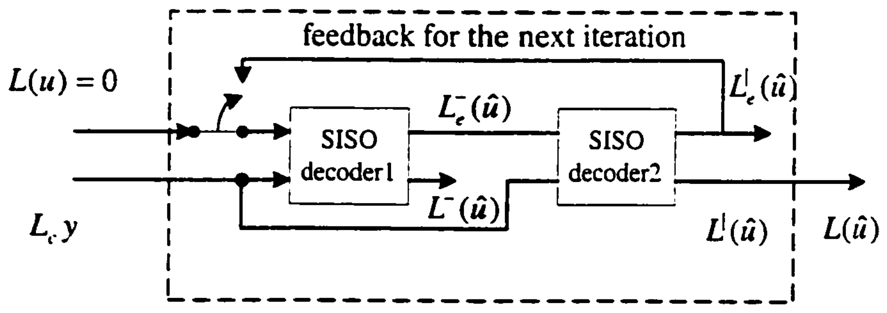


Fig. 5.2 Generic turbo decoder structure

information from the first decoder is then sent to the second decoder. Along with the channel outputs second decoder calculates its estimate of the data bits and delivers its extrinsic information. Successive iterations will use the extrinsic information from this iteration as *a priori* information. This cycle is repeated, and with more iteration the BER of the decoded bits tends to fall. However, the improvement in performance obtained with increasing numbers of iterations decreases as the number of iterations increases [14] [21]. Extrinsic information is obtained by removing the *a priori* information and channel information. It is essential that only extrinsic information be passed between decoders and between iterations for correct operation of the Turbo decoder. After final iteration the turbo code decoder has a hard-decision to output the decoded bits.

5.2 Turbo Code Encoder

5.2.1 Recursive Systematic Convolutional (RSC) Code

A recursive systematic convolutional (RSC) code can be constructed from a nonrecuesive nonsystematic convolutional (NSC) code (Fig 4.2). As shown in Fig. 5.3, one of the outputs from Fig. 4.2 is fed back to form the input of register, and the encoder is made systematically by setting output stream to be equal to the input stream. The RSC encoder in Fig. 5.3 can be represented by generator matrix:

$$g^{(0)} = (g_0^{(0)}, g_1^{(0)}, g_2^{(0)}) = (1,1,1) = 7,$$

$$g^{(1)} = (g_0^{(1)}, g_1^{(1)}, g_2^{(1)}) = (1,0,1) = 5, \qquad (5.1)$$

$$G = (1, \frac{g^{(1)}(D)}{g^{(0)}(D)}) = (1, \frac{5}{7})$$

Its state diagram is shown in Fig. (5.4). RSC codes perform better than their equivalent NSC codes at all SNRs for high code rates (rates > 2/3). Moreover, even for low code rates, the RSC codes have lower bit error rates than the NSC codes at low SNRs [14]. For turbo code, the main purpose of implementing RSC encoders as component encoders is to utilize the recursive nature of the encoders and not the fact the encoders are systematic. In Fig 5.3, D is the input of register in the convolutional encoder. It can be represented as following:

$$D_k = u_k + \sum_{i=1}^{m} D_{k-i} g_i^{(0)} \quad \text{modulo-2} \tag{5.2}$$

So the output of the RSC is:

$$v_k^{(0)} = u_k , \tag{5.3}$$

$$v_k^{(1)} = \sum_{i=0}^{m} D_{k-i} g_i^{(1)} \quad \text{modulo-2} \tag{5.4}$$

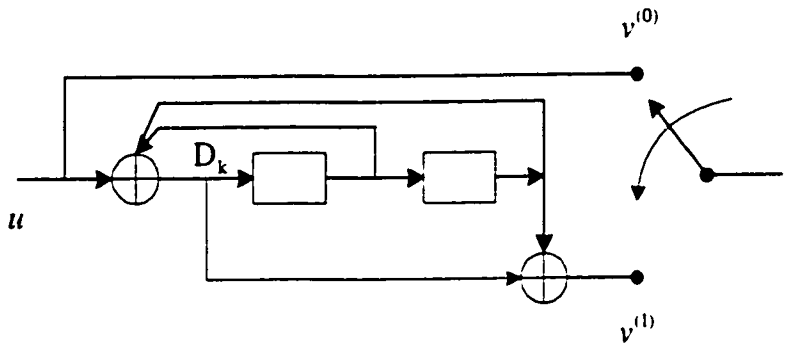


Fig. 5.3 A rate ½, (2,1,2) RSC encoder

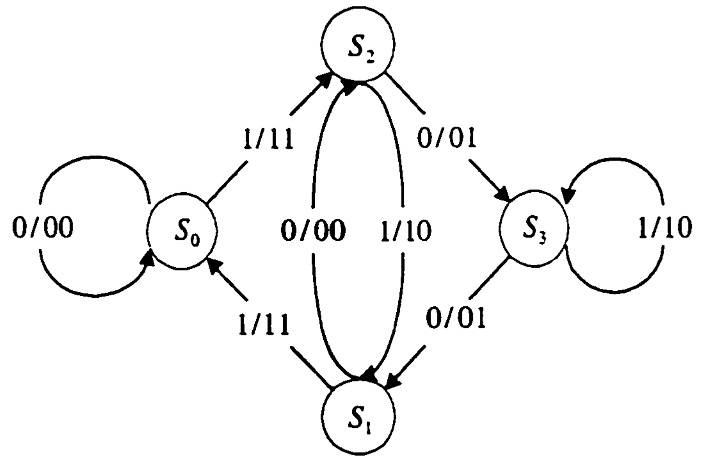


Fig. 5.4 State transition diagram in Fig 5.3

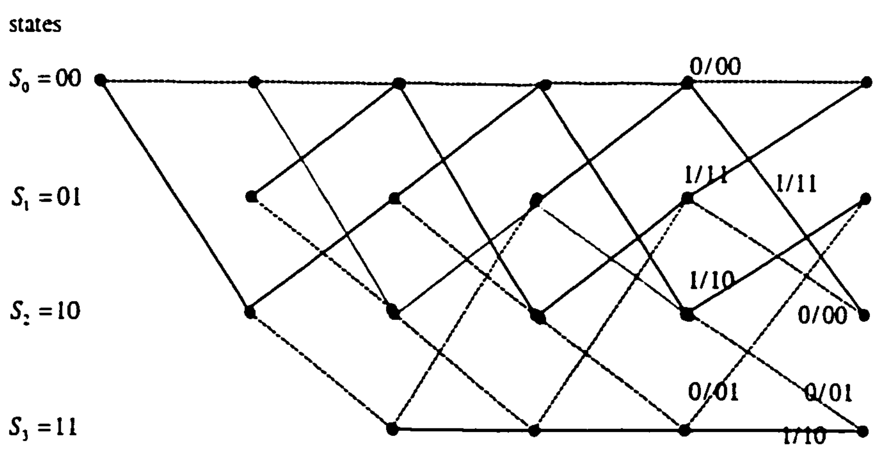


Fig. 5.5 Treilis diagram in Fig 5.3

From the above equations or Fig (5.3), we can easily obtain the state transition diagram of the encoder, as shown in Fig (5.4) and trellis diagram in Fig (5.5). We will use the trellis diagram to do MAP decoding.

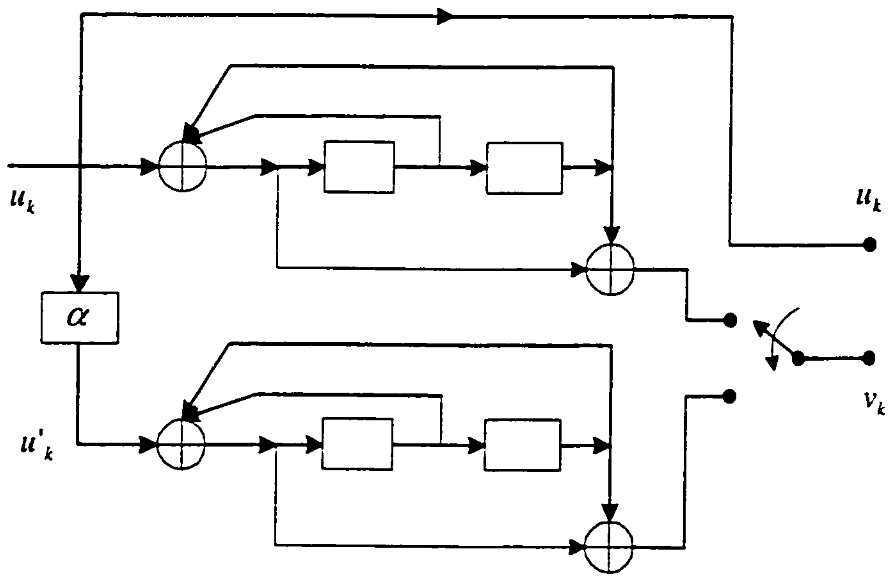


Fig. 5.6 Rate ½ turbo code with two RSC encoder

5.2.2 Turbo Code Encoder

A turbo code encoder is composed of two RSC encoders and an interleaver, as shown in Fig 5.6. In order to get high code rate, the systematic information bits u'_k of the second RSC encoder are not transmitted to the channel. Instead of doing them, the receiver can recover them at the decoder from the systematic information bits u_k of the first RSC encoder through the same pattern interleaver as at the transmitter. Due to RSC's infinite impulse response nature, the output of an RSC encoder usually has a fairly high Hamming weight. But, there are some input sequences that result in low Hamming weight outputs. By passing the same input through both RSC encoders in different orders, the probability of both encoders simultaneously producing outputs with low Hamming weights is reduced. Some situations may arise when both encoders produce low weight output sequences, but these occur at a relatively low rate, resulting in superior turbo code performance at low signal-to-noise ratio.

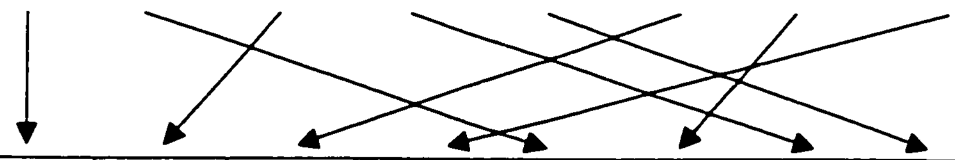
5.2.3 Interleaving

The interleaver in the turbo encoder mainly performs two important functions. Firstly, an interleaver decreases the number of low weight output's code words. As seen in the previous sections, even if there is an input pattern that causes a low weight code word at the output of one RSC encoder, the probability that the interleaved version of the input data will also cause a low weight code word at the output of the other RSC encoder is very low. This minimizes the number of low weight code words at the output of the overall turbo encoder. Secondly, an interleaver increases the performance of turbo code in the presence of burst errors. Burst errors (a chain of successive bits in error) are detrimental to the performance of a code. Since the interleaver scrambles the input data bits, adjacent bits in the original input data will be placed a distance away from each other after interleaving. Thus, the number of burst errors is reduced and performance of a turbo code is improved. Hence the interleaver design is a key factor which determines the good performance of a turbo code. The above issues motivate the design of efficient interleavers. Various interleaver designs have been considered and implemented with turbo codes. Block interleaver, Pseudo-Random Interleaver, Spread Interleaver are most commonly used interleavers. In this thesis, only Pseudo-random is involved.

Pseudo-random interleaver permutes the order of input data with random. It picks a random number $\alpha(i)$ within the size of the input data frame and maps the input bit to the i^{th} position in the interleaved frame as shown in Table 5.1. In the Table i is the index of interleaved information bits. $\alpha(i)$ is the number of original information input order. This approach can lead to good or bad interleavers, especially for small interleaver sizes. Choosing a good interleaver design is important for obtaining good turbo code performance, but the most significant parameter relating to the interleaver is its size. As the interleaver size increases, performance improves.

Table 5.1 pseudo-random interleaver function $\alpha(i)$ and data mapping

i	1	2	3	4	5	6	7	8
$\alpha(i) =$	1	3	6	8	2	7	4	5
$u_i =$	0	1	1	0	1	0	0	1



$u_i = u_{\alpha(i)}$	$u_1 = 0$	$u_3 = 1$	$u_6 = 0$	$u_8 = 1$	$u_2 = 1$	$u_7 = 0$	$u_4 = 0$	$u_5 = 1$

5.2.4 Output Puncturing

Many applications can benefit from the use of high rate convolutional codes. These applications also include those that require only moderate error correction but demand high throughput. There are two means to increase the code rate. One is to increase k, the number of inputs. However, an increase in the number of inputs corresponds to an exponential increase in the complexity of the decoding algorithm. Puncturing the code has another means of implementing high rate codes. This process increases the transmission rate of the code by periodically deleting bits from the output of the encoder according to a specific format called the puncturing pattern. Turbo coding system in our research is a two-level parallel concatenation system of two same structure recursive convolutional encoders with memory size equal to 2. In this case. turbo codes have original rate as 1/3 since two groups of parity bits are added in transmission and both of them have the same length as the information sequence. To achieve higher rate code, some of the bits in the codeword must be punctured. Since all information bits should be retained to obtain good results from iterative decoding, only the parity bits are punctured. For the code rate ½ .the used puncturing matrix is given by:

$$P = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{5.5}$$

According to the puncturing matrix, the parity check digits from the two component encoders are alternately deleted. The punctured turbo code symbol at a given time consists of an information digit followed by a parity check digit, which is alternately obtained from the first and the second component encoder. A puncturing pattern is used to select the bits to be punctured at the transmitter and the same pattern is made available to the receiver. At the receiver, zeros are inserted into the received sequence at the punctured positions.

5.2.5 Trellis Termination

A turbo encoder operating on each block of information bits can be regarded as a block encoder. At the end of each block, each component encoder is required to be the all-zero state. This ensures that the initial state for the next block is the all-zero state. Trellis termination means driving the encoder to the all-zero state. It is not possible to terminate the trellis by transmitting m zero tail bits due to the feedback of RSC. The tail termination bits for the RSC encoder depend on the state of the encoder and are very difficult to predict. Fig. 5.7 shows a simple strategy that has been developed in [23], which overcomes this problem. For encoding the input sequence, the switch is turned on to position A and for terminating the trellis, the switch is turned on to position B.

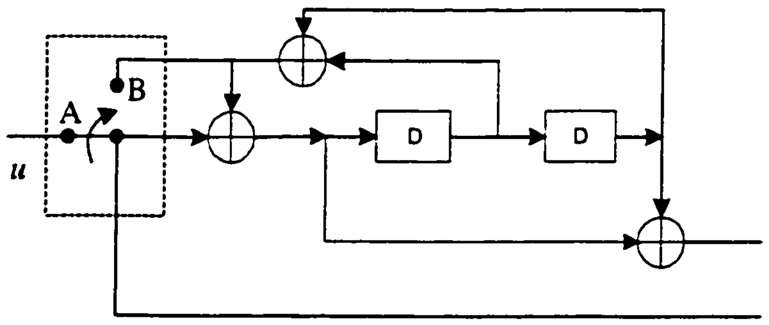


Fig. 5.7 Diagram for trellis termination

With a pseudo-random interleaver it is highly unlikely that this method terminates both encoders simutanuously with only m tail bits since the codes are recursive. Usually, only the first component encoder is forced to return to the all-zero state. The performance degradation produced by an unknown final state of the second encoder is negligible for a large interleaver size *N*.

5.3 Log-likelihood Map Algorithm

As mentioned previously, SISO decoder can be implemented based on MAP algorithm. MAP algorithm was presented by Bahl *et al.* in 1974 [15] known as BCJR algorithm that bears the name of the authors. This algorithm was introduced to provide an alternative to the Viterbi algorithm for decoding convolutional codes. The Viterbi algorithm is a maximum-likelihood decoding algorithm that minimizes the sequence error probability but does not necessarily minimize the bit error probability. MAP algorithm provides not only the estimated bit sequence, but also the probabilities for each bit that it has been decoded correctly. This is essential for the iterative decoding of turbo codes.

In order to simplify the algorithm (without loss of accuracy) for binary codes, the likelihood ratio of each bit is usually computed, rather than respectively computing the probabilities of the bit being zero or one. Furthermore, to reduce the range of values returned, the natural logarithm of the likelihood ratio is usually taken. This metric is called the Logarithm of Likelihood Ratios (LLR). The Log Likelihood Ratio of the information bit u_k conditioned only by the received symbol y_k, which corresponds to u_k, is:

$$\begin{aligned} L(u_k \mid y_k) &= \ln \frac{P(u_k = +1 \mid y_k)}{P(u_k = -1 \mid y_k)} \\ &= \ln \frac{P(y_k \mid u_k = +1) P(u_k = +1)}{P(y_k \mid u_k = -1) P(u_k = -1)} \\ &= \ln \frac{P(y_k \mid u_k = +1)}{P(y_k \mid u_k = -1)} + \ln \frac{P(u_k = +1)}{P(u_k = -1)} \end{aligned} \tag{5.6}$$

When the channel is AWGN, we have:

$$\begin{aligned} \ln \frac{P(y_k \mid u_k = +1)}{P(y_k \mid u_k = -1)} &= \ln \frac{\frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{E_s}{N_0}(y_k - 1)^2 \right]}{\frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{E_s}{N_0}(y_k + 1)^2 \right]} \\ &= L_c * y_k \end{aligned} \tag{5.7}$$

Where $L_c = 4 \cdot E_s / N_0$

$$L(\hat{u}_k) = L_c * y_k + L(u_k) \tag{5.8}$$

A soft-in Soft-out decoder is shown as in Fig 5.7. Let $\mathbf{u} = (u_1 \cdots u_k \cdots u_{N-m})$ for a frame of information bits produced by the source. N is interleaver size; m is the memory of RSC encoder. The value of u_k is (0, 1). The turbo encoder output is represented as $\mathbf{v}_1^N = (\mathbf{v}_1 \cdots \mathbf{v}_k \cdots \mathbf{v}_N)$ with $\mathbf{v}_k = (u_k, v_k)$, where the value of v_k is (0,1) and an alternatively punctured bit from two component RSC encoders. The encoded sequence is modulated and transmitted over the AWGN channel. The mapping relation is $1 - 2 * u_k$. For the simplification, we still use the notations for the modulated information. The noisy received sequence is given by $\mathbf{y}_1^N = (\mathbf{y}_1 \cdots \mathbf{y}_k \cdots \mathbf{y}_N)$ with $\mathbf{y}_k = (y_{s,k}, y_{p,k})^T$. $y_{s,k}$ is noisy received information corresponding to the systematic information. $y_{p,k}$ is noisy received information corresponding to parity check coded bit. Furthermore, we can also let

$\mathbf{y}_k = (y_{s,k}, y_{p,k})^T$. If the system at time k-1 is at state s' on the trellis and at time k is at state s, respectively, u_k is associated with the transition from time k-1 to k. If we consider the Log Likelihood Ratio of the information bit u_k conditioned by the whole observation sequence, the *LLR* output can be expressed:

$$L(\hat{u}_k) = \ln\frac{P(u_k = +1 \mid \mathbf{y}_1^N)}{P(u_k = -1 \mid \mathbf{y}_1^N)} = \ln\frac{P(u_k = +1, \mathbf{y}_1^N)}{P(u_k = -1, \mathbf{y}_1^N)} \tag{5.9}$$

From this Equation, we have:

$$\begin{aligned} &P(u_k, \mathbf{y}_1^N) = \sum_{(s,s')} P(u_k, s', s, \mathbf{y}_1^N) \\ &= \sum_{(s,s')} P(s', s, \mathbf{y}_1^{k-1}, \mathbf{y}_k, \mathbf{y}_{k+1}^N) \end{aligned} \tag{5.10}$$

$$\begin{aligned} &P(s', s, \mathbf{y}_1^{k-1}, \mathbf{y}_k, \mathbf{y}_{k+1}^N) \\ &= P(s', \mathbf{y}_1^{k-1}) P(s, \mathbf{y}_k \mid s') P(\mathbf{y}_{k+1}^N \mid s) \\ &= P(s', \mathbf{y}_1^{k-1}) P(s \mid s') P(\mathbf{y}_k \mid s', s) P(\mathbf{y}_{k+1}^N \mid s) \\ &= \alpha_{k-1}(s') \gamma_k(s', s) \beta_k(s) \end{aligned} \tag{5.11}$$

From Eq. (5.11), we have the forward recursion of MAP algorithm:

$$\begin{aligned} &\alpha_k(s) = P(s, \mathbf{y}_1^k) \\ &= \sum_{s'} P(s', s, \mathbf{y}_1^{k-1}, \mathbf{y}_k) \\ &= \sum_{s'} P(s', \mathbf{y}_1^{k-1}) P(s, \mathbf{y}_k \mid s') \\ &= \sum_{s'} \alpha_{k-1}(s') \gamma_k(s', s) \end{aligned} \tag{5.12}$$

The backward recursion is following:

$$
\begin{aligned}
\beta_{k-1}(s) &= P(\mathbf{y}_k^N \mid s') \\
&= \sum_s P(s, \mathbf{y}_k, \mathbf{y}_{k+1}^N \mid s') \\
&= \sum_s P(\mathbf{y}_k \mid s', s) P(\mathbf{y}_{k+1}^N \mid s) \\
&= \sum_s \gamma_k(s', s) \beta_k(s)
\end{aligned}
\tag{5.13}
$$

When the initial state starts at zero and the last state ends at zero in an observed sequence, the forward and backward recursion are initialized with $\alpha_{start}(0) = 1$, $\beta_{end}(0) = 1$. If a transition branch exits between s' and s, we have:

$$
\gamma_k(s', s) = p(\mathbf{y}_k \mid s's) P(s \mid s') = p(\mathbf{y}_k \mid s', u_k) P(u_k) \tag{5.14}
$$

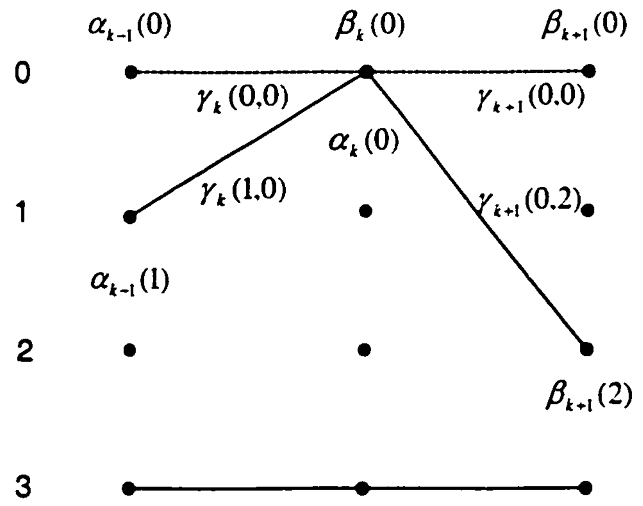
Fig.5.8 shows an example how to calculate recursively $\alpha_k(s)$ and $\beta_k(s)$ using values of $\alpha_{k-1}(s')$ and $\gamma_k(s', s)$ as well as $\beta_{k+1}(s)$ and $\gamma_{k+1}(s', s)$ for our example RSC code. As we are considering a binary trellis, only two previous states have paths to the current state. Therefore, the summation in Eq. (5.12) and Eq. (5.13) is over only two terms.

Next step we derive the relation between $P(u_k)$ and $L(u_k)$

$$
L(u_k) = \ln \frac{P(u_k = +1)}{P(u_k = -1)} = \ln \frac{P(u_k = +1)}{1 - P(u_k = +1)} \tag{5.15}
$$

$$
P(u_k = +1) = \frac{1}{1 + e^{-L(u_k)}} = \frac{e^{-\frac{1}{2}L(u_k)} \, e^{\frac{1}{2}u_k L(u_k)}}{1 + e^{-L(u_k)}} \tag{5.16}
$$

$$
L(u_k) = \ln \frac{P(u_k = +1)}{P(u_k = -1)} = \ln \frac{1 - P(u_k = -1)}{P(u_k = -1)} \tag{5.17}
$$



$$\alpha_k(0) = \alpha_{k-1}(0)\gamma_k(0,0) + \alpha_{k-1}(1)\gamma_k(1,0)$$

$$\beta_k(0) = \beta_{k+1}(0)\gamma_{k+1}(0.0) + \beta_{k-1}(2)\gamma_{k+1}(0.2)$$

Fig 5.8 Recursive calculation of $\alpha_k(s)$ and $\beta_k(s)$

$$P(u_k = -1) = \frac{e^{-L(u_k)}}{1+e^{-L(u_k)}} = \frac{e^{-\frac{1}{2}L(u_k)}\, e^{\frac{1}{2}u_k L(u_k)}}{1+e^{-L(u_k)}} \tag{5.18}$$

From Eq. (5.16) and Eq. (5.18), we obtain:

$$P(u_k) = \frac{e^{-L(u_k)}}{1+e^{-L(u_k)}} = \frac{e^{-\frac{1}{2}L(u_k)}\, e^{\frac{1}{2}u_k L(u_k)}}{1+e^{-L(u_k)}} = A_k e^{\frac{1}{2}u_k L(u_k)} \tag{5.19}$$

$$\begin{aligned} p(\mathbf{y}_k \mid s', u_k) &= B_k' \exp\left\{-\frac{E_s}{N_0}\left[(y_{s,k}-u_k)^2 + (y_{p,k}-v_k)^2\right]\right\} \\ &= B_k \exp\left[\frac{1}{2}L_c y_{s,k} u_k + \frac{1}{2}L_c y_{p,k} v_k\right] \end{aligned} \tag{5.20}$$

where B_k is a common term related with states

From Eq.(5.19) and Eq. (5.20) we have:

$$\gamma_k(s',s) = p(\mathbf{y}_k \mid s's)P(s \mid s') = p(\mathbf{y}_k \mid s',u_k)P(u_k)$$

$$= A_k B_k \exp\left[\frac{1}{2}L_c y_{s,k}u_k + \frac{1}{2}L_c y_{p,k}v_k\right]\exp(\frac{1}{2}u_k L(u_k))$$

$$= A_k B_k \exp\left[\frac{1}{2}L_c y_{s,k}u_k + \frac{1}{2}L_c y_{p,k}v_k + \frac{1}{2}u_k L(u_k)\right] \qquad (5.21)$$

where v_k depends on s' and u_k. From Eq. (5.21) only medium term is related with state s.

$$L(\hat{u}_k) = \ln\frac{\sum_{(s,s')}\alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s)}{\sum_{(s,s')}\alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s)}$$

$$= \ln\frac{\sum_{\substack{(s,s')\\u_k=1}}\alpha_{k-1}(s')\exp\left[\frac{1}{2}L_c y_{s,k} * 1 + \frac{1}{2}L_c y_{p,k}v_k + \frac{1}{2} * 1 * L(u_k)\right]\beta_k(s)}{\sum_{\substack{(s,s')\\u_k=-1}}\alpha_{k-1}(s')\exp\left[\frac{1}{2}L_c y_{s,k} * (-1) + \frac{1}{2}L_c y_{p,k}v_k + \frac{1}{2} * (-1) * L(u_k)\right]\beta_k(s)}$$

$$= \ln\frac{\exp\left[\frac{1}{2}L_c y_{s,k} * 1 + \frac{1}{2} * 1 * L(u_k)\right]\sum_{\substack{(s,s')\\u_k=1}}\alpha_{k-1}(s')\exp\left[\frac{1}{2}L_c y_{p,k}v_k\right]\beta_k(s)}{\exp\left[\frac{1}{2}L_c y_{s,k} * (-1) + \frac{1}{2} * (-1) * L(u_k)\right]\sum_{\substack{(s,s')\\u_k=-1}}\alpha_{k-1}(s')\exp\left[\frac{1}{2}L_c y_{p,k}v_k\right]\beta_k(s)}$$

$$= L_c y_{s,k} + L(u_k) + \ln\frac{\sum_{\substack{(s,s')\\u_k=1}}\alpha_{k-1}(s')\exp\left[\frac{1}{2}L_c y_{p,k}v_k\right]\beta_k(s)}{\sum_{\substack{(s,s')\\u_k=-1}}\alpha_{k-1}(s')\exp\left[\frac{1}{2}L_c y_{p,k}v_k\right]\beta_k(s)}$$

$$= L_c y_l(k) + L(u_k) + L_e(u_k) \qquad (5.22)$$

where $L(u_k)$ is the a priori information of the data bit u_k. L_c is called the channel reliability measure and

$$L_e(u_k) = \ln \frac{\sum\limits_{\substack{(s,s')\\ u_k=1}} \alpha_{k-1}(s')\exp\left[\frac{1}{2}L_c y_{p,k} v_k\right]\beta_k(s)}{\sum\limits_{\substack{(s,s')\\ u_k=-1}} \alpha_{k-1}(s')\exp\left[\frac{1}{2}L_c y_{p,k} v_k\right]\beta_k(s)} \qquad (5.23)$$

is the LLR information contributed by the parity associated with the input data bits and is called extrinsic LLR information.

5.4 Max-log-Map Algorithm

The Map algorithm has excellent performance, but it requires large memory and a large number of operations involving exponentiations and multiplications. For implementation consideration, it is too complex in many communication systems. One way of simplifying computational complexity is achieved by the replacement of all multiplication operations by addition operations in the log-domain. However, all addition operations also have to be replaced by a maximization operation:

$$\ln(xy) = \ln(x) + \ln(y)$$

and

$$\begin{aligned} \ln(e^x + e^y) &= \ln e^x(1+e^{y-x}) = \mathrm{x} + \ln(1+e^{y-x}) \\ &= \ln e^y(1+e^{y-x}) = y + \ln(1+e^{y-x}) \\ &\approx \max(\mathrm{x},\mathrm{y}) \end{aligned} \qquad (5.24)$$

For the Log Likelihood Ratio of the information bit u_k conditioned by the whole observation sequence:

$$\begin{aligned} L(\hat{u}_k) &= \ln \frac{\sum\limits_{\substack{(s,s')\\ u_k=+1}} \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s)}{\sum\limits_{\substack{(s,s')\\ u_k=-1}} \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s)} \\ &= \ln \sum_{\substack{(s,s')\\ u_k=+1}} \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s) + \ln \sum_{\substack{(s,s')\\ u_k=-1}} \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s) \end{aligned} \qquad (5.25)$$

We first consider first term. Same result can apply to second term.

$$\begin{aligned}&\ln \sum_{\substack{(s,s')\\u_k=+1}} \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s)\\&= \ln \sum_{\substack{(s,s')\\u_k=+1}} \exp[\ln \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s)]\\&= \underbrace{\max}_{\substack{(s,s')\\u_k=+1}} (\ln \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s))\end{aligned} \tag{5.26}$$

With the trellis shown in Fig 5.5, there are four terms in above equation. Maximum one of them is kept, and other three terms are left out.

Through the equation below, multiplication operations are replaced by addition operations:

$$\begin{aligned}&\ln \alpha_{k-1}(s')\gamma_k(s',s)\beta_k(s)\\&= \ln \alpha_{k-1}(s') + \ln \gamma_k(s',s) + \ln \beta_k(s)\end{aligned} \tag{5.27}$$

furthermore,

$$\begin{aligned}\ln \alpha_k(s) &= \ln \sum_{s'} \alpha_{k-1}(s')\gamma_k(s',s)\\&= \ln \sum_{s'} \exp \ln(\alpha_{k-1}(s')\gamma_k(s',s))\\&= \underbrace{\max}_{s'} \ln(\alpha_{k-1}(s')\gamma_k(s',s))\end{aligned} \tag{5.28}$$

and also:

$$\begin{aligned}&\ln(\alpha_{k-1}(s')\gamma_k(s',s))\\&= \ln \alpha_{k-1}(s') + \ln \gamma_k(s',s)\end{aligned} \tag{5.29}$$

where

$$\ln \gamma_k(s',s) = \frac{1}{2}L_c y_{s,k} u_k + \frac{1}{2}L_c y_{p,k}(k) v_k + \frac{1}{2}u_k L(u_k) \tag{5.30}$$

Similarly, $\ln \beta_k(s)$ can be found.

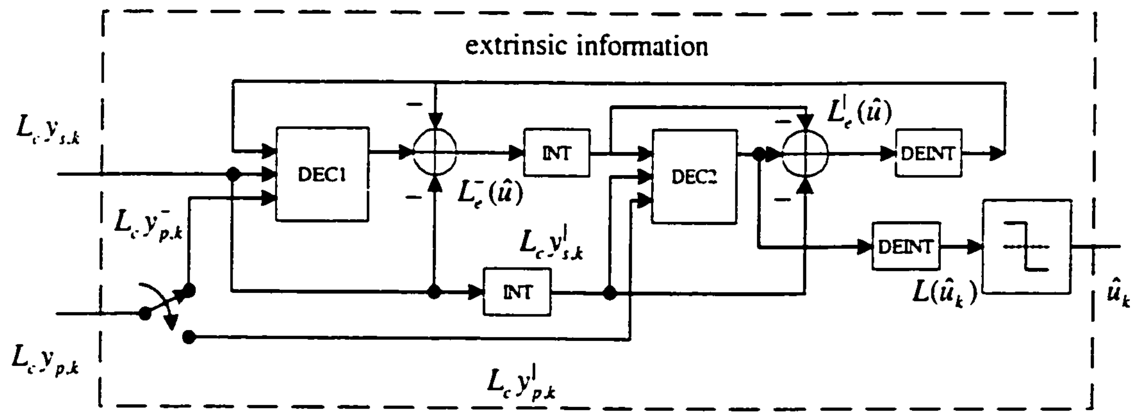


Fig. 5.9 diagram for iterative decoding of turbo code

When calculating the Log Likelihood Ratio of the information bit u_k conditioned by the whole observation sequence using Max-Log-Map, we can track back from the last step until finding the Log Likelihood Ratio.

5.5 Iterative Decoding of Turbo Code

In this section, we will address the concepts of extrinsic and intrinsic information and highlight how the MAP algorithm described in the previous section, and soft-in soft-out decoders, are used in the iterative decoding of turbo codes. The diagram for iterative decoding of turbo code is shown in Fig. 5.9, which shows the course of decoding of turbo code. For a systematic code such as a RSC code, the output from the MAP decoder, given by Eq. (5.22), is rewritten here:

$$L(\hat{u}_k) = L_c y_{s,k} + L(u_k) + L_e(u_k) \tag{5.31}$$

From Eq. (5.31), we can see that the a-posteriori log likelihood ratio (LLR) calculated with the MAP algorithm can be thought of as comprising of three terms-- $L_c y_{s,k}$, $L(u_k)$ and $L_e(u_k)$. The first term in Eq. (5.31) is the soft output of the channel for the systematic bit u_k, which was directly transmitted across the channel and received as $y_{s,k}$. When the channel signal-to-noise (SNR) is high, the channel reliability value L_c

will be high and this systematic bit will have a large influence on the a-posteriori LLR $L(u_k \mid \mathbf{y}_1^N)$. Conversely, when the channel is poor and L_c is low, the soft output of the channel for the received systematic bit u_k will have less impact on the a-posteriori LLR $L(u_k \mid \mathbf{y}_1^N)$ delivered by the MAP algorithm. The a-priori LLR term $L(u_k)$ comes from $P(u_k)$ in the expression for the branch transition probability in Eq.(5.14). This probability should come from an independent source. In most cases we will have no independent or a-priori knowledge of the likely value of the bit u_k, so the a-priori LLR $L(u_k)$ will be zero, corresponding to an a-priori probability. However, in the case of an iterative turbo decoder, each component decoder can provide the other decoder with an estimate of the a-priori LLR, as described later. The final term $L_e(u_k)$ in Eq. (5.31) is derived from the a-priori information sequence $L(\mathbf{u})$ and the received channel information sequence $\mathbf{y}_1^N$, excluding the received systematic bit $y_{s,k}$ and the a priori information $L(u_k)$ for the bit u_k. Hence, it is called the extrinsic LLR for the bit u_k. Eq. (5.31) shows that the extrinsic information from a MAP decoder can be obtained by subtracting the a-priori information and the received systematic channel input from the soft output of the decoder. Equations similar to Eq. (5.31) can be derived for the other component decoders, which are used in iterative turbo decoding.

We can summarize below what is meant by the terms a-priori, a-posteriori, and extrinsic information, which are central concepts behind the iterative decoding of turbo codes used throughout this treatise. The a-priori information about a bit is information known before decoding starts, from a source other than the received sequence or the code constraints. It is also sometimes referred to as intrinsic information to contrast with the extrinsic information described next. The extrinsic information about a bit is the information provided by a decoder based on the received sequence and on a-priori information excluding the received systematic bit $y_{s,k}$ and the a-priori information for the bit u_k. Typically, the component decoder provides this information using the constraints imposed on the transmitted sequence by the code used. It processes the received bits and a-priori information surrounding the systematic bit u_k, and uses this information and the code constraints to provide information about the value of u_k. The a-posteriori information about a bit is the information that the decoder gives taking into account *all* available sources of information about u_k. It is the *a-posteriori* LLR, $L(u_k \mid \mathbf{y}_1^N)$, that the MAP algorithm gives as its output.

We now describe how the iterative decoding of turbo codes is carried out. Consider initially the first component decoder in the first iteration. This decoder receives the channel sequence containing the received versions of the transmitted systematic bits $y_{s,k}$, and the parity bits $y^{-}_{p,k}$, from the first encoder. Usually, to obtain a half rate code, half of these parity bits will have been punctured at the transmitter, and so the turbo decoder must insert zeros in the soft channel output for these punctured bits. The first component decoder can then process the soft channel inputs and produce its estimate $L_{11}(\hat{u}_k)$ of the conditional LLRs of the data bits u_k. In this notation, the subscript 11 in $L_{11}(\hat{u}_k)$ indicates that this is the *a-posteriori* LLR in the first iteration from the first component decoder. Note that in this first iteration the first component decoder will have no *a-priori* information about the bits, and hence $P(u_k)$ in (5.14) giving will be 0.5.

Next, the second component decoder comes into operation. It receives the channel sequence containing the *interleaved* version of the received systematic bitsu_k, and the parity bits $y^{\mathrm{I}}_{p,k}$ from the second encoder. Again the turbo-decoder will need to insert zeroes into this sequence if the parity bits generated by the encoder are punctured before transmission. However, now, in addition to the received channel sequence, the decoder can use the conditional LLR $L_{11}(\hat{u}_k)$ provided by the first component decoder to generate *a-priori* LLRs $L(u_k)$ to be used by the second component decoder. These *a-priori* LLRs $L(u_k)$—which are related to bit u_k—would be provided by an "independent conduit of information, in order to have two independent channel-impaired opinions" concerning bit u_k. This would provide a "second channel-impaired opinion" as regards to bit **u**. In an iterative turbo decoder, the extrinsic information from the other component decoder is used as the *a-priori* LLRs $L(u_k)$, after being interleaved to arrange the decoded data bits **u** in the same order as they were encoded by the second encoder. The second component decoder thus uses the received channel sequence and the *a-priori* LLRs $L(u_k)$ (derived by interleaving the extrinsic LLRs $L_e(u_k)$ of the first component decoder) to produce its *a-posteriori* LLRs $L_{12}(\hat{u}_k)$. This is then the end of the first iteration.

For the second iteration the first component encoder again processes its received channel sequence, but now it also has *a-priori* LLRs $L(\hat{u}_k)$ provided by the extrinsic portion of the *a-posteriori* LLRs $L_{12}(\hat{u}_k)$ calculated by the second component encoder, and hence it can produce an improved *a-posteriori* LLR $L_{21}(\hat{u}_k)$. The second iteration then continues with the second component decoder using the improved *a-posteriori* LLRs $L_{21}(\hat{u}_k)$ from the first encoder to derive, through (5.31), improved *a-priori* LLRs $L(u_k)$. The second component uses the improved *a-priori* LLRs $L(u_k)$ in conjunction with its received channel sequence to calculate $L_{22}(\hat{u}_k)$. This iterative process continues, and with each iteration on average the BER of the decoded bits will fall. However, the improvement in performance for each additional iteration carried out falls as the number of iteration increases. Hence, for complexity reasons usually only around six to eight iterations are carried out, as no significant improvement in performance is obtained with a higher number of iterations.

When the series of iterations finishes the output from the turbo decoder is given by the de-interleaved *a-posteriori* LLRs $L_{i2}(\hat{u}_k)$ of the second component decoder, where i is the number of iterations used. The sign of these *a-posteriori* LLRs gives the hard decision output, i.e., whether the decoder believes that the transmitted data bit was 1 or -1, and in some applications the magnitude of these LLRs, which gives the confidence the decoder has in its decision, may also be useful. Ideally, for the iterative decoding of turbo codes, the *a-priori* information used by a component decoder in the context of bit should be based on a "conduit of information" independent from the channel outputs used by that decoder. More explicitly, for example, an original systematic information bit and the parity-related information smeared across a number of bits by the encoder due to the code constraints imposed are affected by the channel differently. Hence, even if the original systematic information bit was badly corrupted by the channel, the surrounding parity information may assist the decoder in obtaining a high-confidence estimate concerning the channel-impaired original systematic bit. However, in turbo decoders the extrinsic LLR $L_e(u_k)$ for the bit u_k, as explained above, uses all the available received parity bits and all the received systematic bits except the received value $y_{s,k}$ associated with the bit u_k. The same received systematic bits as that of the first component decoder are also used by the other component decoder, which uses the interleaved or de-interleaved version of $L_e(u_k)$ as its *a-priori* LLRs. Hence, the *a-priori* LLRs are not truly independent from the channel outputs used by the component decoders. However, due to the fact that the component convolutional codes have a short memory, usually of only 4 bits or less, the extrinsic LLR $L_e(u_k)$ is only significantly affected by the received systematic bits, which are relatively close to the bit u_k. When this extrinsic LLR $L_e(u_k)$ is used as the *a-priori* LLR by the other component decoder, because of the interleaving used, the bit and its neighbors will probably have been well separated. Hence, the dependence of the *a-priori* LLRs $L(u_k)$ on the received systematic channel values $L_c y_{s,k}$ which are also used by the other component decoder will have relatively little effect, and the iterative decoding provides good results.

5.5 Simulation with Imbalances

In our research, two recursive systematic convolutional encoders with a constrain length $K = 3$ and a random interleaver are adapted. The interleaver has a data length of 1024. In iterative decoder, Max-log-Map Algorithm is used. Simulation results of the performance with the imbalances are shown in Fig. 5.10. Line1 is the performance without imbalances. Line2 is the performance of constant imbalances. Line3 is the performance under random imbalances. In order to compare with the performance of the system with Viterbi decoding, we have the same imbalance statistic as that of the system with Viterbi decoding. It is repeated here. The line 1 also shows the performance without imbalances. The line 2 shows the simulation result of a constant phase imbalance with a phase of 9^0. The line 3 shows the performance with random imbalances of which the phase imbalance has a uniform density distribution with a range of [5.5, 12.5], and the gain imbalance is a constant with a value of 10% of the reference gain or 0.83dB. From the figure, it is seen that the turbo code has a powerful correct ability for the imbalances, specially, when SNR is getting high. Fig. 5.11 shows the effects of different iterative numbers of a turbo code on the performance with imbalances. Increasing iterative number 6 to 8 can almost achieve the same performance as that of no imbalances if SNR is larger than 2.5 dB.

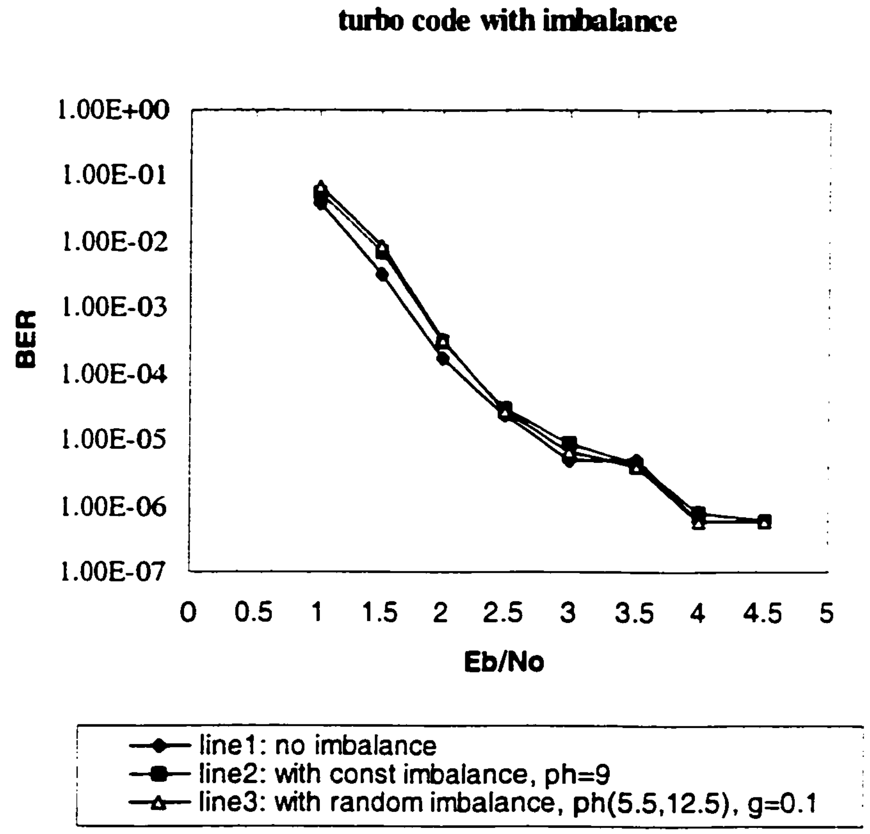


Fig 5.10 The performance with different imbalance levels

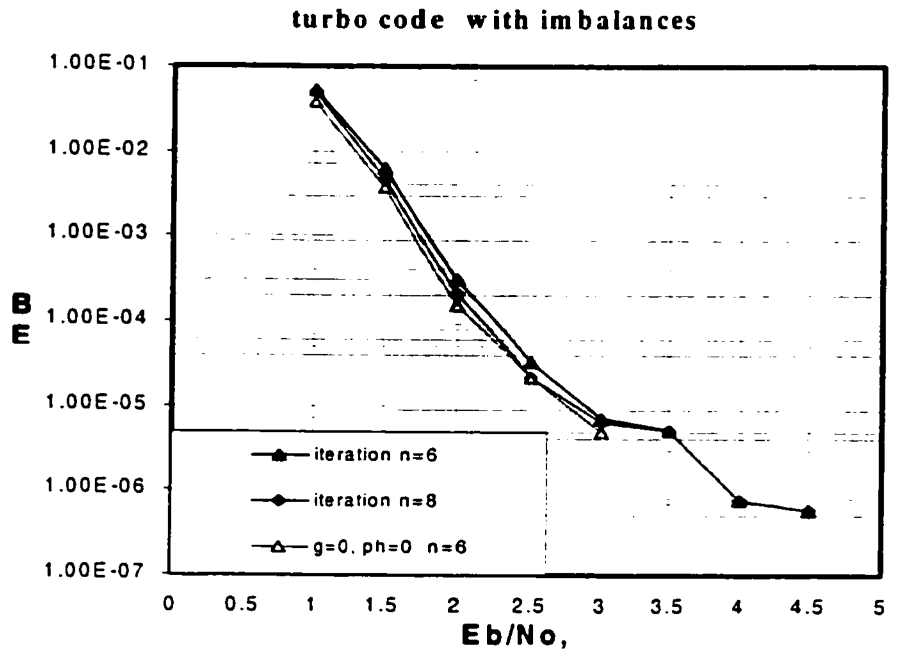


Fig 5.11 The performance with different iterative numbers

Chapter 6

Conclusions and Suggestions for the Future Research

6.1 Conclusion Remarks

The works described in my thesis has involved the investigation and analysis of quadrature modulation imbalances. The major objective of this research is to invest the imbalance effect on the performance in uncoded and coded scheme and also to develop efficient detection and compensation technologies to solve the problem.

To reach the objective, we analyze the constellation of signals with I/Q imbalances as well as the frequency spectrum of the signals in the transmitter. The research shows gain and phase imbalances not only affect signal constellation, but also cause many other spurious signals, such as IP3 and etc., which make the noise floor rise greatly. In order to reduce the signal distortion, the detection and compensation to the imbalances have to do at the transmitter. We develop detection and compensation method to eliminate the imbalance effects. The observation is taken from the amplitude of RF before the power amplifier (PA). A linear Least Square (LS) estimation is used based on data sequence or test tone. The estimation is also equivalent to Minimum Mean Squared Error (MMSE) estimation in the absence of the channel noise. Compensation is done by pre-offsetting the I/Q imbalance. The pre-offset matrix is the reverse of the estimated matrix of imbalances. The simulation result of this compensation shows that there is a compensation gain of 0.9 dB at a BER of 2.0×10^{-3}.

A receiver also has the phase and gain imbalances so that the system performance degrades further. From our analysis, the I/Q imbalances cause the signal constellation changed. From the analysis of spectrum, the extra imbalance signal has a compressed amplitude image spectrum of the designed signal. The simulation results with receiver imbalances prove the effect on the performance.

Because of the imbalances in the receiver and the transmitter, the steady-state lock point of loop has not been locked at the state of the channel phase of 0°, but there is the phase delay, determined by the imbalance level, at the output of the down-converter. So, the phase synchronization cannot completely eliminate the channel rotation between the up-converter and the down-converter, also cannot change the I/Q imbalances at all.

A receiver imbalance estimation algorithm and a compensation technology are proposed in this thesis based on our study. Since it is difficult and not necessary to estimate individual imbalance parameter respectively, four parameters in **H** matrix, which combine all of the imbalances in the transmitter and the receiver as well as the channel rotation, are considered as the estimated parameters. ML estimation algorithm is adapted to estimate the four parameters. The simulation of this estimation method achieves the same compensation gain at the BER of 2.0×10^{-3} under the same conditions.

Channel coding as a general technology of channel error correction can also correct the I/Q imbalances. A convolutional encoder with Viterbi decoder achieves a good correction of I/Q imbalances. The simulation of Viterbi decoding with I/Q imbalances shows that the effect of I/Q imbalances on the performance is greatly decreased.

Turbo code is a significant discovery towards realizing Shannon's channel capacity limit. It is also a powerful tool to remove the effects of I/Q imbalances. With the conditions we used, turbo code is much insensitive to the I/Q imbalances than the Viterbi decoding. When signal-noise ratio goes to the floor region, the effect of I/Q imbalances on the performance is almost completely removed out. Increasing iterative time can achieve better performance with I/Q imbalances. Not much correction space of I/Q imbalance is left with turbo code if BER is less than 10^{-5}.

6.2 Suggestions for the Future Research

In this thesis, we use preamble data to estimate the I/Q imbalances of transmitter-receiver chain. In the absence of a preamble, there is no effective method to measure I/Q imbalances until now. Unlike phase estimation, imbalance estimation cannot use a conventional Non-data-aided method to estimate I/Q imbalance because I/Q imbalance is not a phase rotation on a constellation. Finding an efficient method of I/Q imbalance estimation in the absence of aided data is a more challenging task.

References

1. James K. Cavers, Maria W. Liao, "Adaptive Compensation for Imbalance and Offset Losses in Direct Conversion Transceivers" IEEE Trans. On Vehicular Technology, VOL. 42, NO.4 Nov. 1993
2. Hung Nguyen, "Improving QPSK Demodulator Performance For Quadrature Receiver with Information from Amplitude and Phase Imbalance Correction" 2000 IEEE WCNC Conference V.3 P_{1440}~P_{1444}.
3. Huang, X., "On Transmitter Gain/Phase Imbalance Compensation At Receiver," IEEE Communications Letters, November, 2000, Vol. 4, No. 11.
4. H. Tsou, "The Effect of Phase and Amplitude Imbalance on the Performance of BPSK/QPSK communication Systems", TDA Progress 42-130
5. H. Tsou, "The Effect of Phase and Amplitude Imbalance on the Performance of Offset Quadrature Phase-Shift-Keyed (OQPSK) Communication Systems", TDA Progress 42-135
6. M. Valkman, M Renfors "Digital filter Design for I/Q Imbalance Compensation" in Proc. Xeuropean Signal Processing Conf., Tampere, Finland, Sept. 2000 PP. 1497~1500
7. Mikko Valkama, Markku Renfors: Advanced DSP for I/Q Imbalance Compensation in a Low-IF Receiver. 2000 IEEE International Conference on Communications, pp. 768-772
8. M., Renfors, M., Koivunen, " On the Performance of Interference Canceller Based I/Q Imbalance Compensation", ICASSP'2000, Vol. 5, pp. 2885-2888
9. R. Weigel, L. Maurer, D. Pimingsdorfer, A. Springer, "RF Transceiver Architectures for W-CDMA Systems Like UMTS: State of the Art and Future Trends", http://www.iis.ee.ethz.ch/nwp/lemon/pub/Weigel_SAW_Symposium_2001.pdf
10. Stephen B. Wicker, Error Control Systems for Communication and Storage. Prentice Hall 1995
11. Saeed V. Vaseghi, "Advanced Signal Processing and Digital Noise Reduction". Wiley & Teubner
12. Berard Mulgrew, "Adaptive filters and Equlisers" Kluwer adademic Publishers

13. John G. Proakis, "Digital Communications," New York: McGraw-Hill.
14. C. Berrou, A. Glavieux, and P. Thitimasjshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes(1)," in *Proc., IEEE Int. Conf. on Commun.*, (Geneva, Switzerland), May 1993, pp. 1064-1070.
15. L.R.Bahl, J.Cocke, F.Jelinek, and J.Racic, "Optimal decoding of linear codes for minimizing symbol error rate," IEEE Trans, Inform. Theory, Vol. IT-20, pp.284-287, 1974.
16. Joachim Hangenauer, "Iterative decoding of Binary Block and Convolutional codes" IEEE trans on information Theory VOL42. No2 1996
17. S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Soft output decoding algorithms in iterative decoding of Turbo codes", TDA progress report 42-124, February 15, 1996.
18. Matthew C. Valent*i*, and Brian D. Woerne*r*, "Iterative Channel Estimation and Decoding of Pilot Symbol Assisted Turbo Codes Over Flat-Fading channels"
19. http://www.cs.tut.fi/kurssit/83090/Mammela.pdf
20. S. Adrian Barbulescu, TURBO CODES: a tutorial on a new class of power error correcting coding schemes, Part I and Part II. Revised: 26 October, 1998
21. Jason P. Woodard and Lajos Hanzo, "Comparative Study of Turbo Decoding Techniques: An Overview" IEEE TRANS. ON VEHICULAR TECHNOLOGY, VOL. 49, NO. 6, NOVEMBER 2000
22. A. J. Viterbj, "Error bounds for convolutional codes and an asymtotically optimum decoding algorithm," IEEE Trans. Inform. Theory, vol. IT-13, pp. 260-269, Apr. 1967.
23. D. P. DiVincenzo, Quantum computation, Science 270 (1995), 255