The Effects of Carrier Frequency Offsets on the Performance of OFDM Systems

by

Stephen David Burfoot
B. Eng., University of Sheffield, 1974

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

MASTER OF APPLIED SCIENCE

In the Department of Electrical and Computer Engineering

We accept this thesis as conforming to the required standard

[redacted]

Dr. P. F. Driessen, Supervisor (Department of Electrical and Computer Engineering)

[redacted]

Dr. R. L. Kirlin, Department Member (Department of Electrical and Computer Engineering)

[redacted]

Dr. C. E. Picciotto, Outside Member (Department of Physics)

[redacted]

Dr. R. Lueck, External Examiner (School of Earth and Ocean Sciences)

Supervisor: Dr. P. F. Driessen

ABSTRACT

OFDM communications systems are known to be sensitive to Carrier Frequency Offsets (differences in frequency between transmitter and receiver). We develop a novel and simple formula to estimate the Bit Error Rate of OFDM systems subject to Carrier Frequency Offsets and Additive Gaussian White Noise. We also develop an improved formula to estimate the Degradation caused by CFOs and AWGN.

Examiners:

[redacted]

Dr. P. F. Driessen, Supervisor (Department of Electrical and Computer Engineering)

[redacted]

Dr. R. L. Kirlin, Department Member (Department of Electrical and Computer Engineering)

[redacted]

Dr. C. E. Picciotto, Outside Member (Department of Physics)

[redacted]

Dr. R. Lueck, External Examiner (School of Earth and Ocean Sciences)

Table of Contents

List of Figures

List of Abbreviations

		First Used
ADSL	Asymmetric Digital Subscriber Line	7
ASK	Amplitude Shift Keying	1
AWGN	Additive White Gaussian Noise	3
BER	Bit Error Rate	3
BPSK	Binary Phase Shift Keying	34
CDMA	Code division Multiple Access	1
CFO	Carrier Frequency Offset	3
DAB	Digital Audio Broadcasting	6
DFT	Discrete Fourier Transform	20
DSP	Digital Signal Processing	7
DVB	Digital Terrestrial Television Broadcast	6
FDM	Frequency Division Multiplexing	1
FFT	Fast Fourier Transform	6
HDSL	High-bit rate Digital Subscriber Lines	7
HIPERLAN/2	High Performance Local Area Network Type 2	6
ICI	Inter Channel Interference	3
IID	Independent, Identically Distributed	22
IFT	Inverse Fourier Transform	31
IFFT	Inverse Fast Fourier Transform	6
ISI	Inter Symbol Interference	7

LAN	Local Area Network	6
MCM	Multi Carrier Communications	2
OFDM	Orthogonal Frequency Division Multiplexing	2
OQAM	Offset QAM	6
PAPR	Peak to Average Power Ratio	8
PSD	Power Spectral Density	22
PSK	Phase Shift Keying	1
QAM	Quadrature Amplitude Modulation	1
QPSK	Quadrature Phase Shift Keying	13
SNR	Signal to Noise Ratio	17
SS	Spread Spectrum	35
TDM	Time Division Multiplexing	1
UWB	Ultra Wide Band	35
WLAN	Wireless Local Area Network	7
W-OFDM	Wideband OFDM	67

Nomenclature

Symbol		First used
a_k :	*k*th complex symbol	16
A :	amplitude of signal at output of correlator	36
b:	number of bits per subchannel symbol	18
B:	OFDM bandwidth	21
c_m :	complex amplitude of subchannel signal	46
d_k :	*k*th signal bit	16
$E[.]$:	expected value operator	50
E_b :	average bit energy	34
E_s :	average symbol energy	36
E_N :	energy in block of N samples	37
f :	frequency	22
f_c :	carrier frequency	16
j:	square root of -1	16
k:	subchannel index (subscript)	16
M_s :	number of samples per symbol or OFDM frame	37
n:	sample index (subscript)	18
$n(t)$:	continuous time noise signal	18
N_m :	Gaussian noise component of demodulated signal	19

N_{1m} :	phase corrected Gaussian noise component	49
N_0 :	one-sided noise spectral density	34
N :	number of OFDM subchannels	12
P :	signal power	39
P_{av} :	average signal power	22
P_k :	signal power in *k*th subchannel	22
P_{pk} :	peak signal power	22
$P(f)$:	PSD of signal	22
$P_k(f)$:	PSD of signal in subchannel k	22
r_m :	demodulated symbol of *m*th subchannel	19
r'_m :	phase corrected demodulated symbol	49
$r(n)$:	received discrete time signal	19
$r(t)$:	received continuous time signal	18
R:	digital bit rate	34
$R(t,t+\tau)$:	autocorrelation function of continuous time function	76
$R_N(n,n+\eta)$:	autocorrelation function of discrete time noise sequence	74
s(t):	continuous time real signal	15
t:	time (continuous)	15
t_s :	sample interval	38
T_d:	high-speed digital bit duration	12
T :	symbol and OFDM frame duration	12

V_{ICI} :	variance of demodulated ICI	51
V_n :	variance of demodulated noise	51
V_s :	variance of demodulated signal	50
$z(n)$:	discrete time complex baseband signal	19
$z(t)$:	continuous time complex baseband signal	18
$z_N(n)$:	complex baseband white noise sequence	19
β :	one-sided subchannel bandwidth	21
δ :	subchannel spacing	21
$\delta(t)$:	Dirac delta function, Kronecker delta	78
$\delta(k)$:	unit impulse function	19
Δf :	subchannel spacing	13
ΔFT :	normalized Carrier Frequency Offset	46
η :	discrete time delay	77
ϕ :	subchannel signal	78
θ_0 :	initial phase offset between carrier and receiver	46
σ :	standard deviation of noise sequence	36
τ :	continuous time delay	76

Acknowledgements

I would like to express my gratitude and thanks to my supervisor, Dr. P. F. Driessen, for giving me the opportunity to study with him, for his help and support throughout the course of my studies, and for the helpful suggestions that inspired and guided my topic of research.

I wish also to express my appreciation to the members of the examining committee, Dr. R. L. Kirlin, Dr C. E. Picciotto and Dr. R. Lueck, who have taken the time to review this work, and to Dr. N. Ruff for agreeing to Chair my examination.

Finally, my thanks go to the many students, faculty and staff at the University of Victoria, who have in some way influenced this work, and who have made my time here an enjoyable and productive one.

Dedication

To my Parents

Chapter 1

1 Introduction

In the never-ending effort to transmit higher and higher amounts of data reliably through an increasingly crowded spectrum three basic transmission strategies stand out.

The first is a development of the original radio ideas of Frequency Division Multiplexing (FDM), familiar to us through the analog amplitude or frequency modulation of a single carrier carrying voice signals. In the modern digital incarnation of this idea, data is modulated onto a carrier by either amplitude shift keying (ASK), phase shift keying (PSK), or a combination of both such as Quadrature Amplitude Modulation (QAM). The data rate may be increased by increasing the size of the signal constellation, at the cost of increased power, or by simply increasing the symbol rate, at the cost of increased bandwidth. A variation on this scheme is Time Division Multiplexing (TDM), in which a number of users are allocated specific time slots to transmit their data. While on the face of it these implementations seem to be attractive due to their apparent simplicity, in practice they are complicated by the need for sophisticated channel equalization and error correction schemes to compensate for channel impairments, and so achieve reliable communication. In consequence, there has been much effort spent to overcome the limitations of this approach.

A more recent strategy is spread spectrum modulation, also known as Code Division Multiple Access (CDMA). Here, many users share a common large frequency band, but distinguish their own data through the use of a code sequence used to encode the information bearing signal. The most common variations on this idea are Direct Sequence CDMA, Frequency or Time Hopping, or a combination of all these.

The third of the three basic strategies is Multi Carrier Communications (MCM), where a digital data stream is split into substreams that are then transmitted in parallel over a number of subchannels. A popular implementation of this idea uses overlapping subbands to improve spectral efficiency and is known as Orthogonal Frequency Division Multiplexing (OFDM), the subject of this thesis.

During this thesis we will discuss OFDM systems in general terms: their background and history; some implementations; and the mathematics behind them. In particular, we shall be interested in the effects of Carrier Frequency Offsets on their performance. We shall also discuss some of the issues relating to the simulation of OFDM systems.

The thesis is organized as follows:

Chapter 2 presents a historical background and overview of Orthogonal Frequency Division Multiplexing, together with some of the issues that arise in practical implementations.

Chapter 3 is a section devoted to the mathematics behind the implementation and simulation of OFDM systems, together with a discussion of their benefits and drawbacks. A major part of this chapter is devoted to showing how to generate simulated noise samples which, when added to the simulated signal, give rise to the desired value of Signal to Noise Ratio (SNR).

Chapter 4 deals with the Matlab code that is used to simulate the various OFDM systems covered in this thesis and to check the results of the theoretical analysis.

Despite its advantages, OFDM is known to be sensitive to Carrier Frequency Offsets (CFOs), which are differences in frequency between receiver and transmitter. Chapter 5 looks at this important practical problem, and a novel and simple formula for the Bit Error Rate (BER) of an OFDM system that is operating with a Carrier Frequency Offset and in the presence of Additive Gaussian White Noise (AWGN) is derived. During the development of this formula, we show that the average Inter Channel Interference (ICI) power in each subchannel is a constant independent of the subchannel position, contrary to previous beliefs. Using the formula, we show that there is an error floor below which the BER cannot drop, even by increasing the signal power. We also develop a better formula than that found in the literature [43] to estimate the SNR Degradation caused by CFOs, and show that it reduces to the previous one for small values of SNR and CFO.

Chapter 6 compares the results of a number of simulations with the new theoretical formula for the Bit Error Rate under a variety of operating conditions. The new and old formulas for Degradation are compared.

The conclusion, Chapter 7 summarizes the preceding chapters and some suggestions for further work are made.

Finally, an Appendix gathers a number of mathematical results about OFDM systems (that are not easily found in the literature) in one place. A copy of the simulation code can also be found.

Chapter 2

2 Background

2.1 History

The basic MCM idea of splitting a high-speed bitstream up into several low speed bitstreams, that are then transmitted in parallel over narrow frequency multiplexed subchannels, (with inverse operations to recover the data in the receiver), has been around since at least the late 1950's with the military Kiniplex system [1].

The advantages of this approach are several-fold. First, since the duration of each of the parallel symbols is longer than that of the original high-speed symbols, the system is obviously more resistant to impulsive noise or to multipath interference, since a given amount of interference lasts a smaller proportion of the symbol length. Second, while the overall signal band may suffer from significant distortion that would require extensive and costly channel equalization in a single carrier system; in a multichannel environment with a sufficient number of subchannels each subchannel is essentially flat across its band, and only requires simple amplitude and phase correction. Third, MCM is robust against frequency selective fading or narrowband interference as only a small number of subchannels will be affected, and error correction coding may be used to combat this type of interference. Finally, there is the possibility of distributing the available signal power over the subchannels in such a way that the maximum capacity of the overall channel is realized, in accordance with the Water-Pouring Theorem [2].

Of course, despite the above advantages, there are always drawbacks. The first MCM systems used non-overlapping independent subchannels with guard bands between them to avoid Inter Channel Interference. This makes inefficient use of the available spectrum. In a groundbreaking paper in 1966, Chang [3], devised a system that used overlapping subchannels, which approximately doubled the spectrum efficiency relative to previous systems. In order to successfully demodulate the received overlapping symbols using this system, the frequencies of the subchannels must be mathematically orthogonal (see Appendix section A.3) to one another, and this system became known as Orthogonal Frequency Division Multiplexing, or OFDM.

Orthogonality in OFDM is achieved by choosing the frequencies of the subcarriers such that the spacing between them is inversely proportional to the symbol duration, so that there are always an integer number of subcarrier cycles in each symbol duration. Increasing the number of subchannels (for a given overall data rate) increases the OFDM symbol duration and also decreases the subchannel bandwidth. We will see how this all works in Chapter 3.

In Chang's system, the baseband data in each of the parallel bitstreams is Nyquist filtered before frequency multiplexing so as to band-limit the subchannels. A further large part of his contribution was to derive conditions on the baseband filter response which allow orthogonality to be maintained in the filtered signal. In a variation on this idea, Salzburg [4], studied an OFDM system that used time-staggered Offset Quadrature Amplitude

Modulation (OQAM) of the subcarriers to reduce signal envelope variations and the related need for high linearity amplifiers.

Other workers developed and implemented these ideas in subsequent years, but the complexity of the hardware required for Nyquist filtering and frequency multiplexing banks of subchannels in the transmitter (with similar operations required in the receiver) proved expensive, and these systems did not become popular. An alternative was provided by Weinstein and Ebert in 1971 [5], who noted that the operation of frequency multiplexing data onto subchannels is equivalent to performing a Discrete Fourier Transform on the data symbols, and suggested that the implementation complexity could be consideratly reduced by replacing the hardware by a Fast Fourier Transform (FFT) performed on a computer. In the receiver, an Inverse Fast Fourier Transform (IFFT) would recover the transmitted data. The separate filtering operations on each baseband signal are omitted, but the complete OFDM symbol may be filtered so as to band-limit it before transmission. If this is the case, subchannels that are outside the passband of the filter carry no useful data, and are usually padded with zeros in the FFT operation.

Work continued in analyzing and implementing OFDM systems throughout the 1970's and 1980's [6], [7], [8], but without any great popular success. However, with the advent of cheap Digital Signal Processing at the beginning of the 1990's, it was pointed out by Bingham [9], in an influential paper, that 'the time has come' for Multi-Carrier Modulation.

Since that time OFDM has been adopted as the new European Digital Audio Broadcasting (DAB) standard, and for the Digital Terrestrial Television Broadcast (DVB) standard [48, 49]. Also in Europe, the ETSI BRAN HiperLAN/2 Wireless LAN standards (WLAN) are currently based on OFDM [48, 49]. In the U.S it is used for High-bit rate Digital Subscriber Lines (HDSL) and Asymmetric Digital Subscriber Lines (ADSL), and OFDM forms the basis of IEEE standard 802.11a, as well as the proposed IEEE standard 802.16.3. Recently (December, 1999), the OFDM Forum [54] has been formed by various hardware manufacturers, software companies and other users, to promote a single worldwide OFDM standard based on Wi-Lan OFDM technology in suitable fields, such as Fixed Wireless Access, Wireless LAN and Home Multi Media, and Broadband Mobile Wireless. An alternative group, Flarion Technology's flash-OFDM™ Alliance, has also been formed to promote a different OFDM technology.

2.2 Implementation Considerations

2.2.1 Cyclic Extension

As mentioned above, modern OFDM implementations are based on Digital Signal Processing (DSP) techniques and use the Inverse Fast Fourier Transform (IFFT) to modulate data symbols onto subcarriers.

An additional procedure can almost completely eliminate Inter Symbol Interference (ISI) caused by multipath delays, by adding a guard time to each OFDM symbol that is greater than the expected delay spread of the signal. The OFDM symbol is cyclically extended into the guard time before each symbol (the cyclic prefix), and also possibly extended after each symbol (the cyclic postfix). The prefix is composed of the last few samples of the OFDM data symbol, while the postfix is composed of the first few samples. Due to the periodic nature of the OFDM symbol, the prefix and postfix blend smoothly with the data samples, without any increase in bandwidth.

In the receiver, the extension is stripped from the data and discarded. The data is then demodulated using a Fast Fourier Transform (FFT) in the normal way. The extra power requirement to transmit the extension is usually negligible. For example, the 25% extension used for IEEE 802.11a leads to an increase in transmitted power of less than 1 dB.

A cyclic extension also allows an alternative to a bandwidth-limiting filter operation that is performed on the whole symbol. Instead of filtering, a time-domain window that reduces the symbol amplitude smoothly to zero at the symbol end-points is applied to the prefix and postfix (but not to the data) [48]. As this only requires relatively few multiplications of window samples with corresponding symbol samples in the rolloff region of the extension, implementation complexity is much reduced compared with filtering the entire symbol. Since the cyclic extension is discarded in the receiver, windowing it has no effect on the received data. In some cases, the prefix of one symbol is allowed to overlap with the postfix of the previous symbol, to give an approximately constant signal envelope.

2.2.2 Peak to Average Power Ratio

One of the disadvantages of OFDM is its high Peak to Average Power Ratio (PAPR), compared to other signaling schemes. This comes about when certain combinations of data symbols cause the peaks of many, or all, of the subcarriers to coincide, and hence to sum to a high amplitude [31]. The problem manifests itself when the signal is passed through the power amplifier, which needs a high degree of linearity to pass the signal unchanged. If this is not the case, intermodulation between the subcarriers and out of band radiation will result. To get the required degree of linearity in a conventional class

A or AB amplifier, it must be operated with significant backoff. This implies poor power efficiency in the amplifier.

There are several approaches that may be taken to combat this problem:

One is to apply a different overall phase to each subcarrier, chosen such that the peaks of the subcarriers can never coincide [10 – 12]. This can be shown to give useful reductions in PAPR without affecting the signaling efficiency or bandwidth.

Two other related techniques that have received a lot of attention are known as Selective Mapping and Partial Transmit Sequences. These both apply several sets of pre-chosen phases to the data symbols and compute a resulting OFDM symbol for each set. The OFDM symbol with the smallest PAPR is then the one transmitted, with details of the set that gave rise to it transmitted as side information. The difference between them is that the Selective Mapping technique applies a different phase to each subcarrier, while the Partial Transmit Sequences technique applies a common phase to blocks of subcarriers. Obviously, neither of these techniques can guarantee that low PAPR symbols will always be transmitted, but they do increase the probability of this.

If a symbol with a high PAPR is in fact generated, it can be deliberately clipped before transmission and the resulting distortion in the receiver corrected using the forward error correction coding that is always present in any practical system implementation. Since the high PAPR symbols can be shown to occur with low probability even without application of the previous techniques, clipping on its own is also a viable approach to the PAPR problem.

Another strategy is to identify the OFDM symbols with large Peak to Average ratios and, by coding the input data so that the data sequences that give rise to these symbols do not occur, simply not transmit them [13 - 15]. However, the coding required for this can be shown to be non-linear and so the use of large look-up tables will probably be required. Nevertheless, the results are quite good and there is the possibility that the coding can be combined with conventional forward error control coding for better implementation efficiency [16, 17].

Some other techniques that have recently been investigated include companding [27 – 29] and trellis shaping [30] to reduce PAPR.

Finally, it must be noted that the development of highly linear, yet efficient, amplifiers is proceeding apace, so the need for low PAPR signals may be reduced in the near future. Techniques such as fixed or adaptive feed-forward correction [39], fixed or adaptive predistortion [40], and Cartesian loop feedback [41] have been investigated.

2.2.3 Synchronization

As with any digital modulation system, Orthogonal Frequency Division Multiplexing requires carrier frequency, phase and timing synchronization to work. However, it may be shown that compared to Single Carrier modulation, OFDM is much more sensitive to carrier frequency offsets and phase noise [42].

Carrier frequency offsets give rise to a shift of the entire received signal in the frequency domain. Although the subcarriers are still mutually orthogonal, this leads to Inter Channel Interference due to components from different subcarriers being demodulated along with the desired signal and an increase in the bit error rate. Similarly, phase noise will also cause an increase in bit error rate due to ICI.

Time synchronization errors, by contrast, can lead to Inter Symbol Interference due to adjacent symbols overlapping in the receiver FFT window, although this effect is mitigated by use of a cyclic extension. Also, a shift in the FFT window will cause phase errors in the OFDM symbols, again leading to BER degradation.

Synchronization algorithms are usually divided into two phases. In the first (acquisition) phase, neither frequency nor timing information is known, and a coarse estimate for both quantities must be found. This is often performed using a synchronization burst of a few known symbols at the beginning of a transmission, and by the inclusion of pilot symbols within the data. In the second (tracking) phase, a fine estimate of frequency and timing is found and maintained for the duration of a transmission. This too may be performed using pilot symbols included in the data, and also by exploiting the inherent autocorrelation properties of OFDM symbols due to the cyclic extension.

Chapter 3

3 OFDM Mathematics

In this chapter we discuss the mathematics behind OFDM systems. Section 3.1 is devoted to a basic mathematical model of the transmission and reception of OFDM symbols in both the real and complex baseband representations; together with the calculation of some parameters of interest. In Section 3.2 we discuss how this simple model is modified and enhanced for real system implementations; and in Section 3.3 we talk about the simulation of communication systems in general and OFDM systems in particular.

3.1 Basic Signal Model

3.1.1 Introduction

The basic idea behind OFDM is to split up a single high rate digital bitstream, of bit duration T_d, into a number N of low rate symbol streams, of symbol duration T, that are modulated onto separate subcarriers and transmitted in parallel. In the receiver the inverse operations take place, with each subcarrier being individually demodulated, its data extracted, and the resulting parallel bitstreams merged into a single high rate bitstream once more. This process is illustrated in the block diagram of Figure 3.1.

In order to make efficient use of spectrum, the frequency bands of adjacent OFDM subcarriers are made to overlap. By choosing the subcarrier frequencies such that in one data symbol period there are always an integer number of subcarrier cycles, we ensure orthogonality of each subfrequency band. This may be achieved by choosing the subchannel spacing, Δf , to be

$$\Delta f = \frac{1}{T} \tag{3.1}$$

For example, the individual subchannel spectra of a four carrier Quadrature Phase Shift Keying (QPSK) modulated OFDM system are shown in Figure 3.2

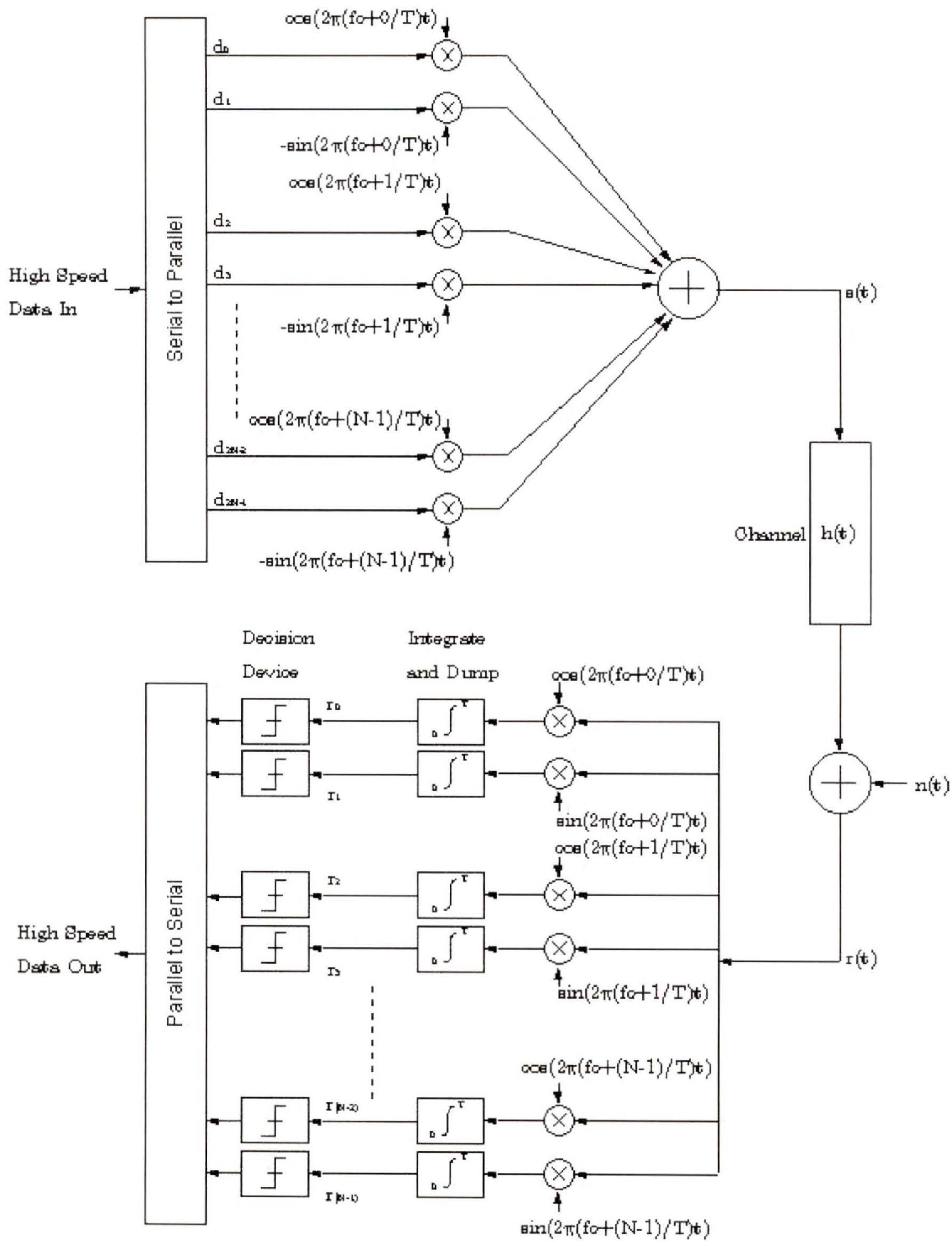

Figure 3.1 **Basic Real Passband Signal Model**

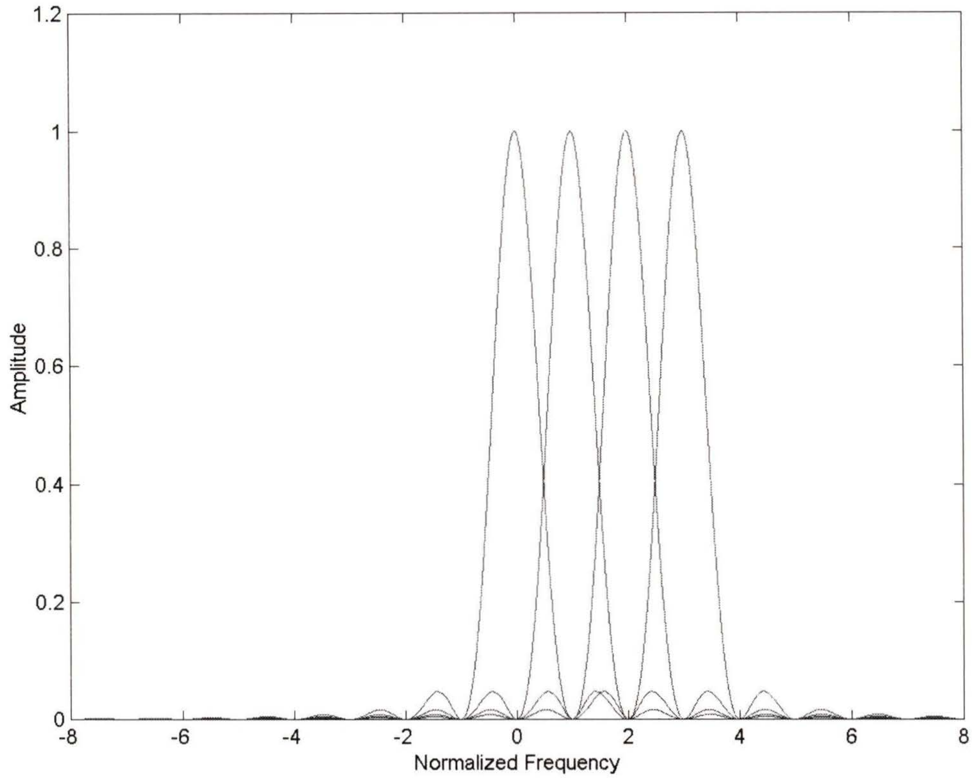

Figure 3.2 Individual Subchannel Spectra

3.1.2 Real Signal and Complex Baseband Representations of Transmitted Signal

The real passband transmitted signal for a single OFDM symbol, in which each subchannel is QAM or PSK encoded, may be written (see Figure 3.1) as

$$s(t) = \begin{cases} \dfrac{1}{T}\displaystyle\sum_{k=0}^{N-1}\left(d_{2k}\cos\left[2\pi\left(f_c+\frac{k}{T}\right)t\right]-d_{2k+1}\sin\left[2\pi\left(f_c+\frac{k}{T}\right)t\right]\right), & 0\le t\le T \\ 0, & \text{t<0 or t>}T \end{cases} \tag{3.2}$$

where f_c is the carrier frequency. The in-phase data symbol of subchannel k is d_{2k}, and the quadrature data symbol is d_{2k+1}. The subcarrier spacing is $1/T$, which we show in (3.13) gives subcarrier orthogonality so that we are able to recover the data.

For $0 \le t \le T$, (3.2) may be expressed as

$$\begin{aligned} s(t) = &\frac{1}{T}\left[\sum_{k=0}^{N-1}\left(d_{2k}\cos\left(\frac{2\pi kt}{T}\right) - d_{2k+1}\sin\left(\frac{2\pi kt}{T}\right)\right)\right]\cos\left(2\pi f_c t\right) \\ &- \frac{1}{T}\left[\sum_{k=0}^{N-1}\left(d_{2k}\sin\left(\frac{2\pi kt}{T}\right) + d_{2k+1}\cos\left(\frac{2\pi kt}{T}\right)\right)\right]\sin\left(2\pi f_c t\right), \end{aligned} \tag{3.3}$$

The complex baseband representation of this signal is, by definition

$$\begin{aligned} z(t) = &\frac{1}{T}\left[\sum_{k=0}^{N-1}\left(d_{2k}\cos\left(\frac{2\pi kt}{T}\right) - d_{2k+1}\sin\left(\frac{2\pi kt}{T}\right)\right)\right] \\ &+ j\frac{1}{T}\left[\sum_{k=0}^{N-1}\left(d_{2k}\sin\left(\frac{2\pi kt}{T}\right) + d_{2k+1}\cos\left(\frac{2\pi kt}{T}\right)\right)\right], \end{aligned} \tag{3.4}$$

which may be written as

$$z(t) = \frac{1}{T}\sum_{k=0}^{N-1}\left(d_{2k} + jd_{2k+1}\right)\left(\cos\left(\frac{2\pi kt}{T}\right) + j\sin\left(\frac{2\pi kt}{T}\right)\right), \tag{3.5}$$

or

$$z(t) = \frac{1}{T}\sum_{k=0}^{N-1} a_k \exp\left(j\frac{2\pi kt}{T}\right) \tag{3.6}$$

where

$$a_k = d_{2k} + jd_{2k+1} \tag{3.7}$$

is a complex QAM or PSK data symbol. Thus we arrive at the simplified complex baseband representation shown in **Figure 3.3**.

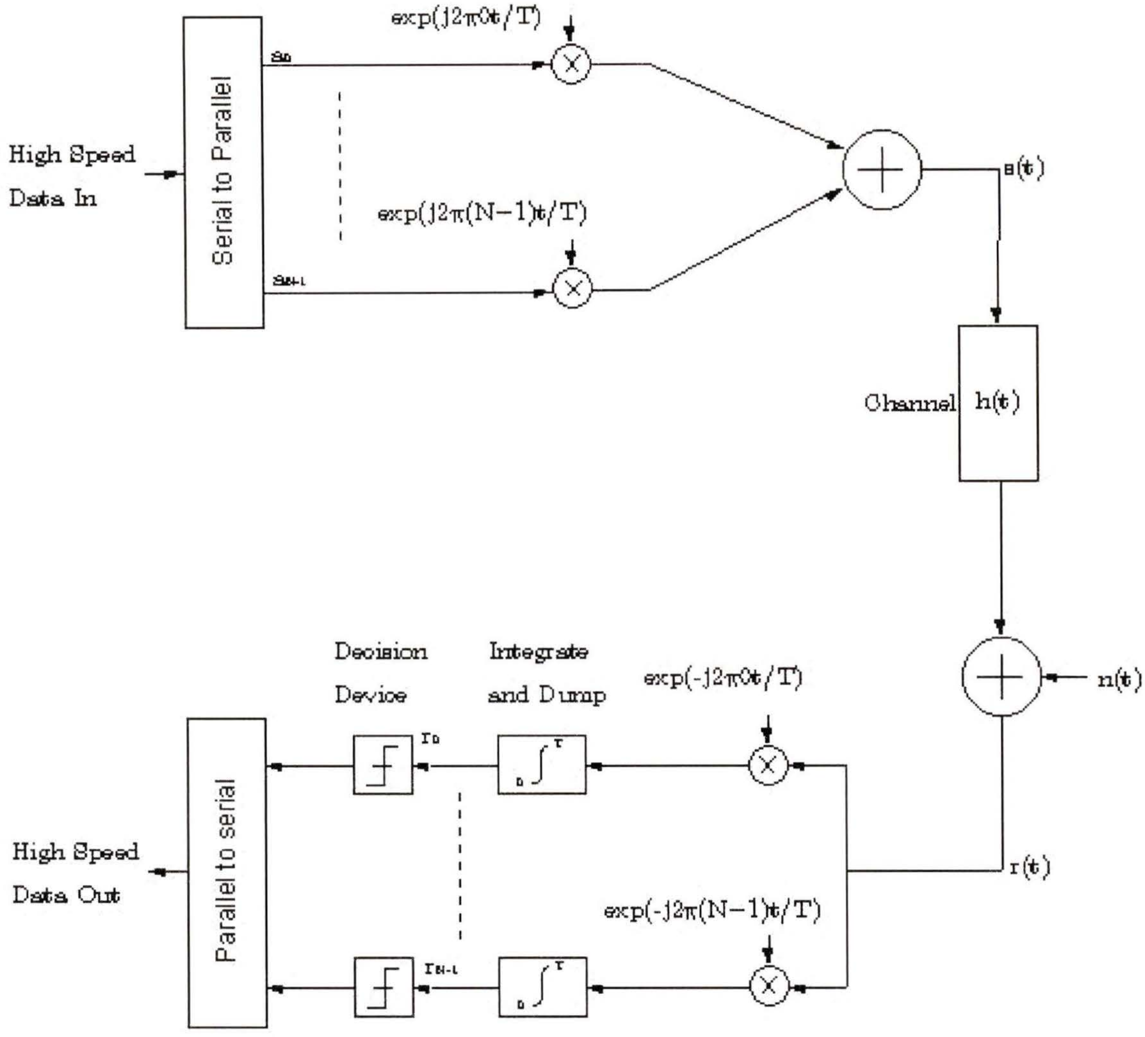

Figure 3.3 Complex Baseband Representation

This representation discards information about the carrier frequency that is not relevant, provided that the carrier frequency is much greater than the bandwidth of the real signal, and hence simplifies the analysis. It may be shown [18, 50] that many of the useful parameters (e.g. bandwidth, Signal to Noise Ratio (SNR), PAPR) are equal in both the real passband and complex baseband representations.

From (3.6) we see that the complex baseband representation is just the Inverse Fourier Transform of the complex data symbols, a_k. We also note that if the number of data bits per subchannel symbol is b, we have

$$T_d = \frac{T}{Nb} \tag{3.8}$$

For Digital Signal Processing (DSP) applications, and for simulations of continuous time systems, we use a discrete time representation of (3.6). We divide the symbol duration, T, into a number, N, of subintervals starting at times t_0, t_1, ..., t_{N-1}, and substitute the discrete integer parameter n for the continuous parameter t using

$$t_n = \frac{n}{N}T, \tag{3.9}$$

Choosing the number of subintervals to be the same as the number of subchannels, we get the following discrete time complex baseband representation

$$z(n) = \frac{1}{N}\sum_{k=0}^{N-1} a_k \exp\left(j\frac{2\pi kn}{N} \right) \tag{3.10}$$

(The factor $1/N$ is chosen to simplify later formulas).

This is recognized as being the Inverse Discrete Fourier Transform of the complex data symbols, a_k, and may be implemented by the well-known Inverse Fast Fourier Transform.

3.1.3 Channel Noise

In the channel, Additive White Gaussian Noise is added to the transmitted signal to form the received signal. For continuous and discrete time representations, respectively, these are

$$r(t) = z(t) + n(t) \tag{3.11}$$

and

$$r(n) = z(n) + z_N(n) \tag{3.12}$$

where $r(t)$ and $n(t)$, $r(n)$ and $z_N(n)$ represent the received signal and noise for continuous and discrete time representations respectively.

3.1.4 Demodulation

The received data symbol in the mth subchannel, r_m, may be recovered by applying the Fourier Transform to each received OFDM symbol. For a continuous time signal this is

$$\begin{aligned}
r_m &= \int_0^T \exp\left(-j\frac{2\pi mt}{T}\right) r(t) dt \\
&= \int_0^T \exp\left(-j\frac{2\pi mt}{T}\right)\left(\frac{1}{T}\sum_{k=0}^{N-1} a_k \exp\left(j\frac{2\pi kt}{T}\right) + n(t)\right)dt \\
&= \sum_{k=0}^{N-1} a_k \frac{1}{T}\int_0^T \exp\left(j\frac{2\pi(k-m)t}{T}\right)dt \\
&\qquad + \frac{1}{T}\int_0^T n(t)\exp\left(-j\frac{2\pi mt}{T}\right)dt \\
&= \sum_{k=0}^{N-1}\left(a_k\delta(k-m)\right) + N_m \\
&= a_m + N_m
\end{aligned} \tag{3.13}$$

where N_m is the received, demodulated noise; while its discrete equivalent is

$$
\begin{aligned}
r_m &= \sum_{n=0}^{N-1} \exp\left(-j\frac{2\pi mn}{N}\right) r(n) \\
&= \sum_{n=0}^{N-1} \exp\left(-j\frac{2\pi mn}{N}\right)\left(\frac{1}{N}\sum_{k=0}^{N-1} a_k \exp\left(j\frac{2\pi kn}{N}\right) + z_n(n)\right) \\
&= \sum_{k=0}^{N-1} a_k \frac{1}{N}\sum_{n=0}^{N-1} \exp\left(j\frac{2\pi(k-m)n}{N}\right) \\
&\qquad\qquad + \frac{1}{N}\sum_{n=0}^{N-1} z_n(n)\exp\left(-j\frac{2\pi mn}{N}\right) \\
&= \sum_{k=0}^{N-1}\left(a_k\delta(k-m)\right) + N_m \\
&= a_m + N_m
\end{aligned}
\tag{3.14}
$$

This last equation is just the Discrete Fourier Transform (DFT) of the signal, $z(n)$, and can be efficiently implemented by the Fast Fourier Transform. The received symbol from either (3.13) or (3.14) is then presented to a decision device in the normal way.

3.1.5 Bandwidth of OFDM Signal

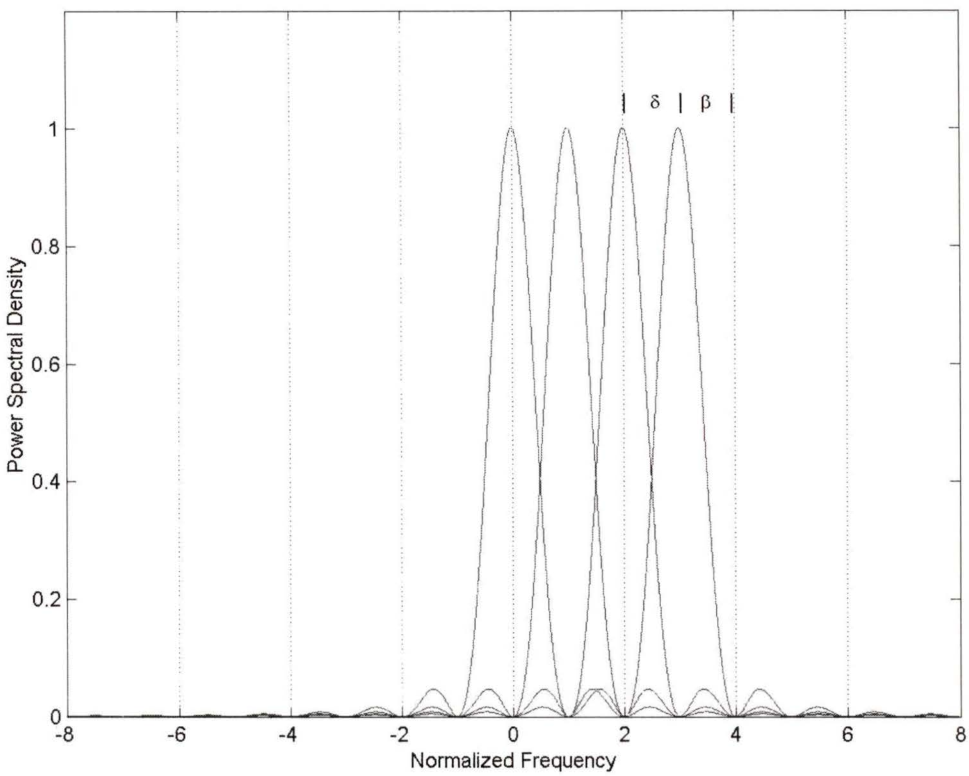

Figure 3.4 Signal Bandwidth

The bandwidth of the composite signal is seen, from Figure 3.4, to be

$$B = (N-1)\delta + 2\beta, \tag{3.15}$$

where δ is the subchannel spacing, and β is the one-sided bandwidth of each subcarrier. The frequency axis has been normalized relative to the subcarrier spacing. For subcarriers that are not filtered in any way, we may take the bandwidth to be equal to the distance to the first null, so $\beta = \delta$ and hence

$$B = (N+1)\delta. \tag{3.16}$$

Since the subchannel spacing, δ, is chosen such that:

$$\delta = \frac{1}{T} = \frac{1}{NbT_d}, \tag{3.17}$$

in order to make the subcarriers orthogonal, we get

$$B = \frac{N+1}{NbT_d} = \left(1+\frac{1}{N}\right)\frac{1}{bT_d}, \tag{3.18}$$

so as $N \to \infty$,

$$B \to \frac{1}{bT_d}. \tag{3.19}$$

This is the same as for ideal Nyquist signaling.

3.1.6 Power Spectrum of Composite Signal

In Appendix A.1 we show that that the Power Spectral Density (PSD) of any multicarrier signal, (not just an OFDM signal), is equal to the sum of the PSDs of the individual subcarriers, provided the crosscorrelation between any two different subcarriers is zero. In other words

$$P(f) = \sum_{k=0}^{N-1} P_k(f) \tag{3.20}$$

where P_k is the PSD of the signal in subchannel k.

The crosscorrelation condition will be fulfilled if the each subcarrier is modulated by Independent, Identically Distributed (IID) QAM or PSK symbols from (possibly different) symmetrical signal constellations. (See Appendix A.3). We may use this result to easily calculate the spectrum of a complicated OFDM system, in which different signal constellations are used to modulate each different subcarrier, from the spectra of the individual subcarriers. A complicated system like this can make more efficient use of the spectrum than one with the same fixed symbol constellation for each subcarrier, in accordance with the water-pouring theorem.

By way of example, we display in Figure 3.5 the overall PSD of the OFDM signal of Figure 3.2

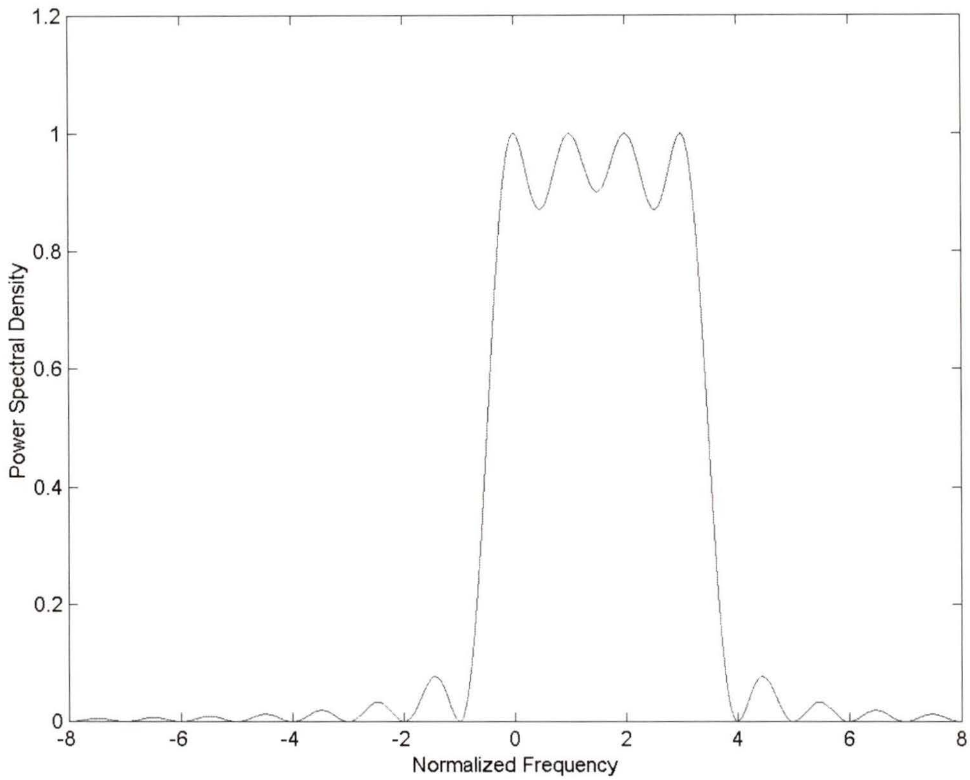

Figure 3.5 Spectrum of Composite OFDM Signal

3.1.7 Peak to Average Power Ratio

One of the disadvantages of OFDM is that is has a large Peak to Average Power Ratio (PAPR), which occurs when, in certain OFDM symbols, the phases of the subcarriers become equal, so constructive interference takes place. This can lead to difficulties in the power amplifier of the transmitter, which must be highly linear over a wide range of signal amplitudes.

We now show that the PAPR of an OFDM signal modulated by QPSK symbols is equal to N, the number of subchannels.

From (A.2) in the Appendix, we know that the overall signal power is the sum of the separate subchannel powers. For QPSK the data symbols a_k may be chosen from the set {1, -1, j, -j}, so $|a_k|^2 = 1$, and then for subchannel *k*, the average signal power is

$$P_k = \frac{1}{T}\int_0^T \left| a_k \exp\left(j2\pi k \frac{t}{T} \right) \right|^2 \qquad (3.21)$$
$$= 1$$

Hence

$$P_{av} = \sum_{k=0}^{N-1} P_k = N \qquad (3.22)$$

The peak power is given by

$$P_{pk} = \max_t \left[\left| \sum_{k=0}^{N-1} a_k \exp\left(j2\pi k \frac{t}{T} \right) \right|^2 \right] \qquad (3.23)$$

A peak occurs when all the subchannel phases are equal at *t*=0, and all the data symbols are the same (say $a_k = 1$), so

$$P_{pk} = \left| \sum_{k=0}^{N-1} 1 \times 1 \right|^2 = N^2 \qquad (3.24)$$

Hence the PAPR is given by

$$PAPR = \frac{P_{pk}}{P_{av}} = \frac{N^2}{N} = N \qquad (3.25)$$

The PAPR of a QAM modulated system is even larger than this, since we must take into consideration the peak to average ratio of the signal constellation itself.

For a system with a large number of subchannels, it may be shown [48] using the Central Limit Theorem that the real and imaginary values of the amplitude of the signal each independently approach a Gaussian distribution with zero mean. The signal envelope consequently approaches a Rayleigh distribution, and the power distribution approaches a central chi-square distribution with two degrees of freedom. The number of symbols that actually achieve the peak power is very small. Details may be found in [12–31].

3.2 OFDM Implementations

3.2.1 Introduction

The basic signal model of the previous section must be modified in any real implementation of OFDM for reasons of bandwidth efficiency, implementation complexity and so forth. We will now discuss three implementations: the first being the classic implementation originally conceived by Chang in his seminal paper [3] and (second) a variation on it pioneered by Saltzberg [4]; and the third being the modern FFT based implementation first discussed by Weinstein and Ebert [5].

3.2.2 Classical OFDM Implementations

The following systems were designed for hardware implementations, but may also be designed as DSP based systems.

3.2.2.1 QAM-OFDM

This implementation, shown in Figure 3.6, is based on the one first described by Chang [3], who just considered Amplitude Modulated (AM) subchannels. We use QAM, and call the resulting system QAM-OFDM.

The main difference between QAM-OFDM and the basic system model of Figure 3.1 is that each of the parallel baseband signals is filtered to restrict the bandwidth before being modulated onto subcarriers. In the receiver, a bank of parallel matched filters is used to detect the signals, which are then passed to decision devices in the normal way.

The overall transmitted OFDM signal then has a finite bandwidth, as opposed to the basic model, whose band edges fall off rather slowly as $1/f$. In the receiver, a matched filter is easier to implement in hardware than a correlator, and sensitivity to timing errors is reduced due to the filtering. This implementation too may be shown to have optimal bandwidth efficiency if the number of subchannels is large [3].

In addition to the usual Nyquist requirements on the transmit and receive filters to remove Inter Symbol Interference (ICI), they must also be chosen such that the filtered signals, when modulated onto subcarriers, retain their orthogonality properties so as to prevent Inter Channel Interference (ICI). Fortunately this is easily achieved, the details being found in Chang [3].

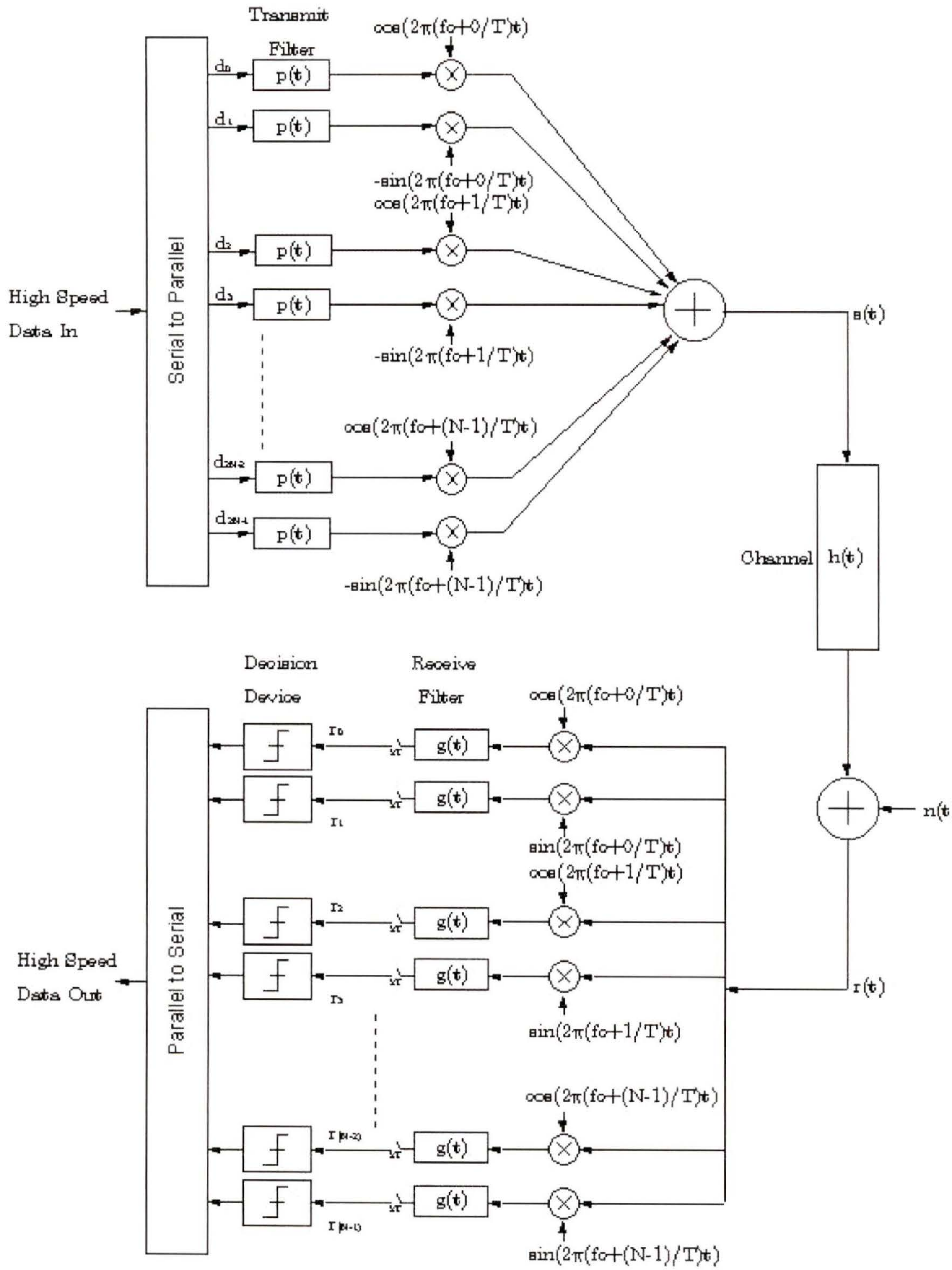

Figure 3.6 **QAM – OFDM Implementation**

3.2.2.2 OQAM-OFDM

A variation on the previous implementation we call OQAM-OFDM was first described by Salzberg [4]. It has the same beneficial effect on the signal envelope as does conventional Offset QAM (OQAM), due to the staggering of the real and imaginary symbols. See Figure 3.7

The implementation is as follows: the real and imaginary parts of each subchannel transmit symbol are individually filtered, and the imaginary part is delayed by half a symbol period as in conventional OQAM. The resulting real and imaginary baseband signals are then combined, modulated on to subcarriers (with an appropriate phase offset for orthogonality), and summed to form the OQAM-OFDM signal. In the receiver, the opposite operations are performed, but with a half-symbol delay in the real parts of the received subchannel symbols to bring the real and imaginary parts back into synchronization. The details may be found in Saltzberg [4].

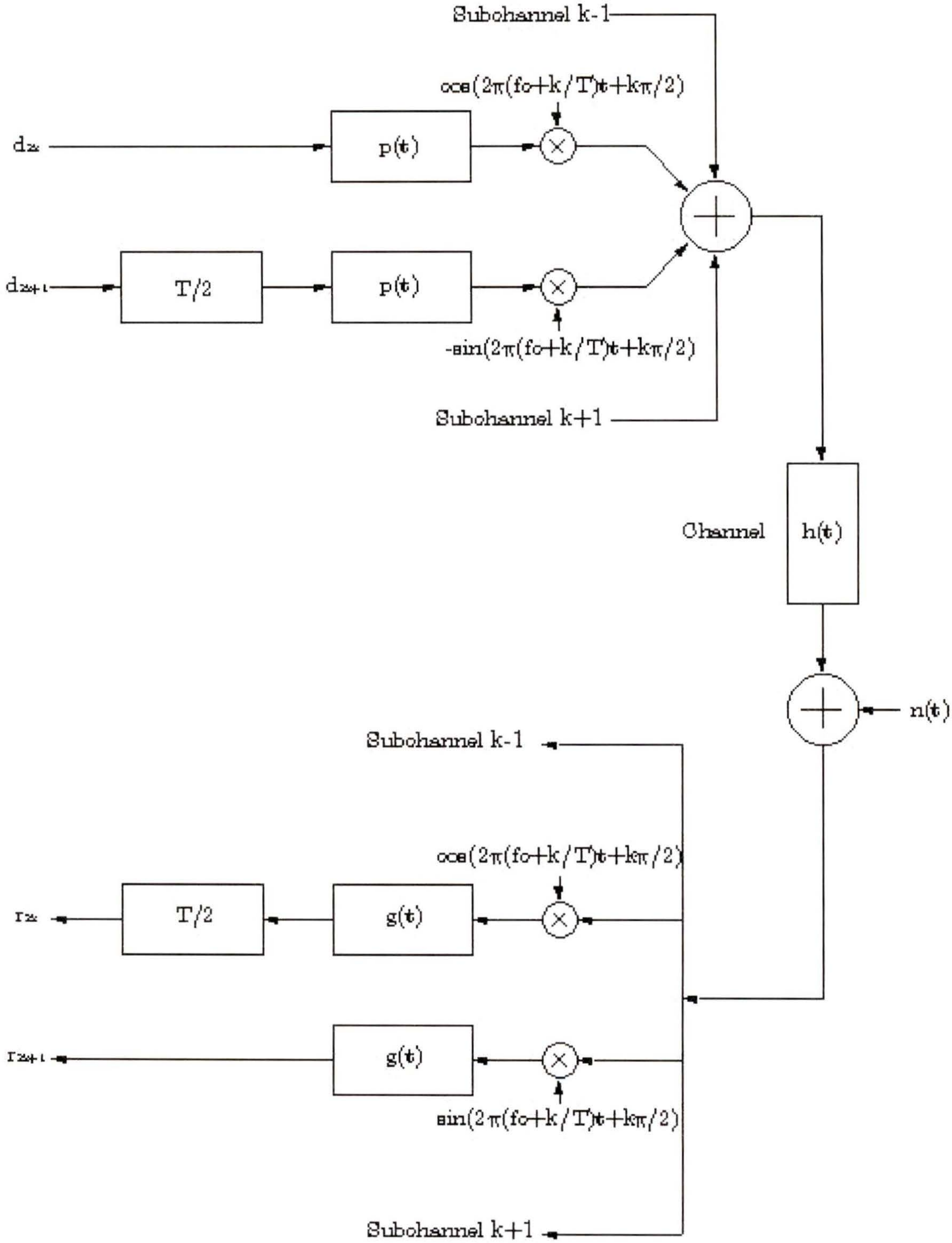

Figure 3.7 OQAM – OFDM Implementation

3.2.3 FFT Based OFDM

As previously noted, an OFDM signal is just the Inverse Fourier Transform (IFT) of the data symbols. Hence, OFDM is well suited for modern DSP based implementations; particularly since the efficiency of the FFT operation can reduce system complexity relative to the previously described systems. The following implementation is based on one described by Weinstein and Ebert [5], whose system used an FFT operation in the transmitter and an IFFT in the receiver. Modern practice is to reverse the order of these operations, which gives the same overall results.

For this implementation, sufficient high-speed data to form an OFDM frame is converted from serial to parallel format and used as the input to an IFFT operation. The transformed parallel data is converted back to serial order and a cyclic extension, described further in 3.2.4 below, is added to each symbol. The cyclic extension is used to reduce interference due to multipath delay spread to negligible levels. The entire symbol is then either filtered to reduce overall bandwidth, or, (more efficiently), undergoes a windowing operation for the same effect. See Figure 3.8.

In the receiver the cyclic extension is stripped off and discarded, and the data undergoes an FFT operation to form the received data symbols.

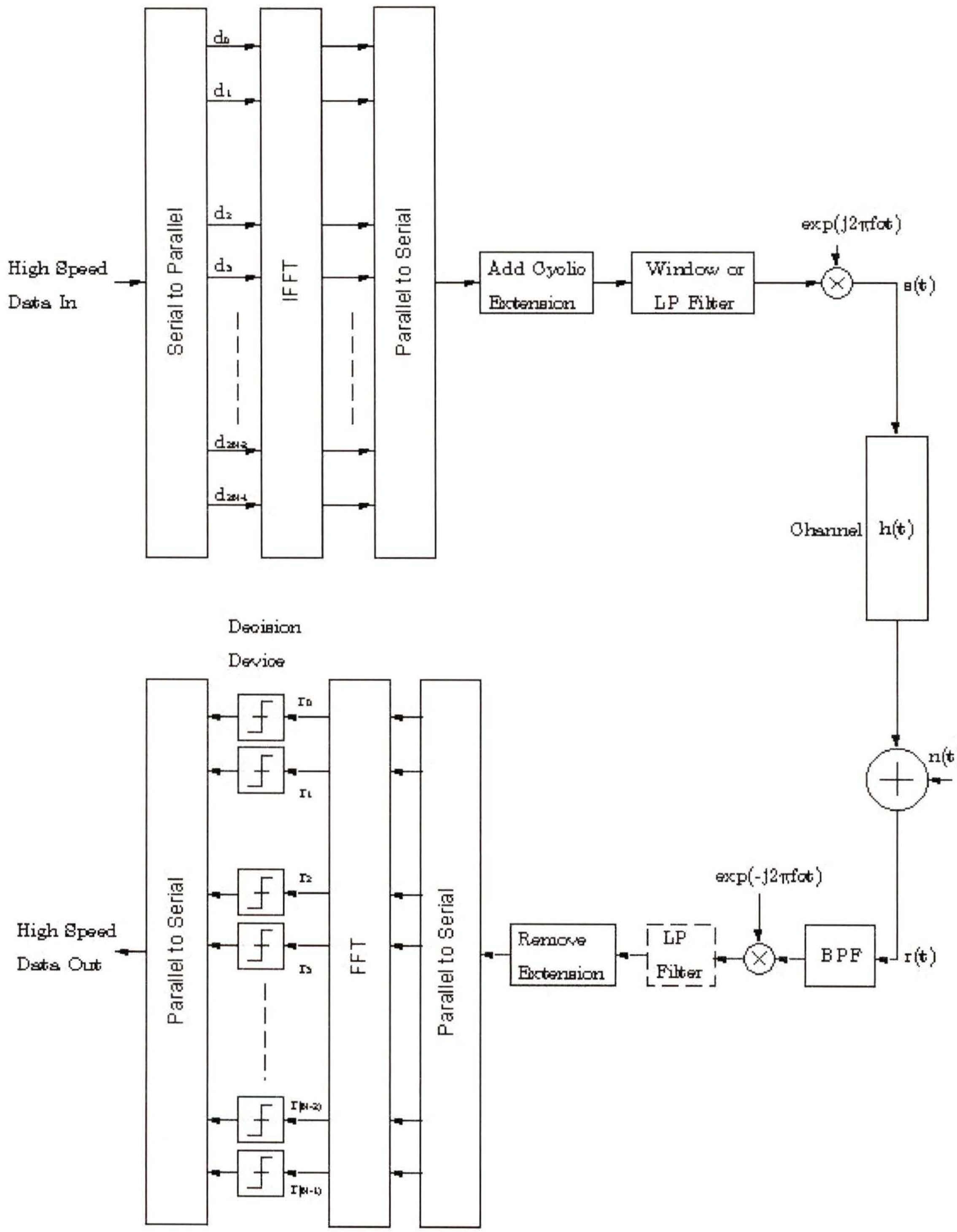

Figure 3.8 **FFT Based Implementation**

3.2.4 Guard Time and Cyclic Extension

One of the big advantages of OFDM is that by increasing the duration of each symbol by a given factor, any Inter Symbol Interference caused by multipath delay spreading is proportionally reduced by the same factor.

The effects of multipath delay may be further reduced by introducing a guard time between each symbol that is longer than the expected delay spread. If the guard time were to be composed of no signal at all, this would introduce Inter Channel Interference into the received signal. However, a cyclic extension composed of a cyclic prefix, formed from the last part of the data symbol, and a cyclic postfix, formed from the first part of the data symbol, eliminates ICI. The bandwidth of the signal is not directly affected by adding a cyclic extension, but obviously the overall data rate must drop if an extension is added. If the data rate is increased to compensate, this will require further bandwidth.

Because the resulting signal has sidelobes that do not roll off very quickly, the overall OFDM signal may be filtered to reduce bandwidth. Typically, those subchannels that fall within the passband of the filter will be modulated as normal, but subchannels falling outside the passband will not be modulated. This is achieved by padding the input to the IFFT with zeros in positions corresponding to out-of-band subchannels.

Alternatively, the cyclic extension may have windowing applied to reduce bandwidth. This is computationally more efficient than filtering, since during windowing only the small number of samples in the window need to be modified, while during filtering every sample in the signal is operated upon.

3.3 Simulation Considerations

3.3.1 Addition of Noise to the Signal

In order to compare the efficiencies of different receivers, both with each other and with theoretically optimal receivers, we may simulate adding a given amount of Additive White Gaussian Noise to the signal in the channel, and then check the resulting Bit Error Rate at the output of each receiver. This section is concerned with generating a noise sequence that will give rise to the desired Signal to Noise Ratio of the signal sequence combined with the noise sequence.

3.3.2 Definition of Signal to Noise Ratio

3.3.2.1 Traditional Definition of SNR

The traditional (analog) definition of SNR is just

$$SNR = \frac{\text{Signal Power}}{\text{Noise Power}} \tag{3.26}$$

This is adapted for a digital setting by noting that the $\text{Signal Power} = \text{P} = E_b R$, where E_b is the average transmitted energy per digital signal bit and R is the digital bit rate; and that the $\text{Noise Power} = N = N_0 B$, where $N_0/2$ is the two-sided Gaussian noise spectral density and B is the two-sided signal bandwidth. A traditional analysis goes on to say that in a 'well-designed system' we have that $R \simeq B$, leading to

$$SNR = \frac{P}{N} = \frac{E_b R}{N_0 B} = \frac{E_b}{N_0}. \tag{3.27}$$

However, while it is true that for early Binary Phase Shift Keying (BPSK) or QPSK systems the bit rate does indeed approximately equal the signal bandwidth, this is no longer necessarily the case for modern systems, so this reasoning is somewhat specious. For example, modern Spread-Spectrum (SS) or Ultra Wide Band (UWB) systems have signal bandwidths that are much larger than bit rate ($R \ll B$); while going the other way a large signal constellation QAM system may have a relatively large bit rate but a narrow bandwidth ($R \gg B$).

For this reason, the modern approach is as follows.

3.3.2.2 Modern Definition of SNR

For a given signaling scheme in the presence of AWGN, it may be shown on information theoretic grounds [2] that the Bit Error Rate (more properly, the Probability of Bit Error) depends only on the ratio of E_b to N_0. In general, the dependence of the BER on E_b/N_0 will be different for different signaling schemes, but it does not depend on either R or B. (Of course, if we are considering factors other than just AWGN, such as channels with impulsive noise or Rayleigh fading, then the BER may then be a function of both R, B, or other factors. We shall not consider this type of channel in what follows.) Consequently, the BER performance of a signaling scheme is conveniently plotted against an SNR defined as

$$SNR = \frac{E_b}{N_0}. \tag{3.28}$$

This definition of SNR also allows the relative efficiencies of signaling schemes with different bandwidths or symbol rates, but the same overall bit rate, to be compared with each other in a simulation.

3.3.2.3 Other definitions of SNR

It may be shown [51] that optimal reception of a digital signal is achieved using either a correlation receiver or a matched filter. Both of these structures optimize the ratio $E\left[A^2\right]/\sigma^2$, where A is the amplitude of the signal at the output of the correlator or matched filter at the symbol strobe time, and σ^2 is the variance of the noise in the strobe sample. Consequently, this ratio is sometimes used as a definition of SNR:

$$SNR = \frac{E\left[A^2\right]}{\sigma^2}, \tag{3.29}$$

particularly if our interest is in the analysis of the signal at the correlator or matched filter output

In Appendix A.2, we show that

$$\frac{E\left[A^2\right]}{\sigma^2} = \frac{2E_s}{N_0} = \frac{2bE_b}{N_0} \tag{3.30}$$

where b is the number of data bits per symbol and E_s is the average energy per symbol, which demonstrates the link between this definition of SNR and the previous one.

Finally, we note that that some authors use the definition

$$SNR = \frac{E_s}{N_0}, \tag{3.31}$$

especially if attention is directed at the probability of symbol error, rather than the probability of bit error.

3.3.3 Generation of Signal and Noise Sequences with chosen SNR

3.3.3.1 Generation of Signal

The first step is to calculate the value of E_b for the signal in the channel. We begin by generating a large number of pseudorandom data bits that are then transmitted according to the chosen signaling scheme and give rise to a block of N signal samples. The energy, E_N, of a block, $z(n)$, of N samples of the signal, spaced at intervals of t_s seconds, is given by

$$E_N = t_s \sum_{n=0}^{N-1} \left\| z(n) \right\|^2 \tag{3.32}$$

Note that in a simulation the sampling rate, and hence t_s, is not known to the computer, since all it sees is a file containing numbers that represent the sample values. However, we show below that t_s cancels out in the final formula for E_b/N_0.

The average energy per symbol, E_s, is

$$E_s = \frac{E_N}{(N/M_s)} = \frac{t_s M_s}{N} \sum_{n=0}^{N-1} \left\| z(n) \right\|^2 \tag{3.33}$$

where M_s is the number of samples per symbol; and the average energy per bit E_b is given by

$$E_b = \frac{t_s M_s}{bN} \sum_{n=0}^{N-1} \left\| z(n) \right\|^2 \tag{3.34}$$

and b is the number of bits per symbol.

3.3.3.2 Generation of Noise

Now we show how to generate a Gaussian noise sequence, $z_N(n)$, with a chosen PSD of $N_0/2$ that is white across a bandwidth B determined by the sampling rate. (Note that in this section B is the bandwidth of the entire simulation, not the bandwidth of the simulated signal).

The noise power of a sequence of samples taken at intervals of t_s seconds from a Gaussian random noise generator having variance σ^2 is

$$\text{Noise Power} = \sigma^2 = \frac{N_0}{2}B \tag{3.35}$$

where B is the two-sided bandwidth of the simulation.

The period of the highest frequency in a simulation with sampling period t_s is clearly $2t_s$; so the two-sided bandwidth goes from $-1/2t_s$ to $+1/2t_s$, and hence

$$B = \frac{1}{t_s}. \tag{3.36}$$

Using this value in (3.35) and rearranging gives

$$N_0 = 2\sigma^2 t_s. \tag{3.37}$$

3.3.3.3 Calculation of SNR

Equations (3.34) and (3.37) together give the result

$$\begin{aligned}\frac{E_b}{N_0} &= \frac{1}{2\sigma^2 t_s}\cdot\frac{t_s M_s}{bN}\sum_{n=0}^{N-1}\left\|z(n)\right\|^2 \\ &= \frac{M_s}{2\sigma^2 b}\left(\frac{1}{N}\sum_{n=0}^{N-1}\left\|z(n)\right\|^2\right) \\ &= \frac{M_s P}{2\sigma^2 b}.\end{aligned} \tag{3.38}$$

where P is an estimate of the power of the signal sequence, $z(n)$. Note that the sampling interval, t_s, does not appear in the formula for E_b/N_0.

3.3.3.4 Choice of Noise Variance

Finally, we calculate the value of σ that gives rise to the desired value of E_b/N_0.

$$\sigma = \sqrt{\frac{M_s P}{2b\left(E_b/N_0\right)}}. \tag{3.39}$$

In this expression, M_s, b and E_b/N_0 are known quantities; while P may be calculated from the transmitted signal sample sequence $z(n)$. It may be desirable to calculate P using only a subsequence of samples from the middle of the original sample sequence, in order to avoid the effects of start-up transients.

The desired noise sequence, $z_N(n)$, may now be generated from the output of a random Gaussian noise generator of variance 1 and length N samples, by multiplying each sample of the output sequence by the value of σ obtained from (3.39) above. The variance of the resulting sequence, σ^2, is the desired noise power.

In the Appendix, section A.2, we show that this noise sequence is white across the entire simulation bandwidth B. Clearly it is Gaussian from the way it was generated.

Chapter 4

4 Simulation Code

In this Chapter we present the Matlab code that was used to simulate the systems discussed in the previous Chapter and to check on the results of the theoretical analysis of Carrier Frequency Offsets in Chapter 5.

We give an overview of the structure and use of the code in the next section, and relegate the code itself to an Appendix.

4.1 Overview

The code is organized into modules with standard interfaces that link to one another. A block diagram is shown in Figure 4.1.

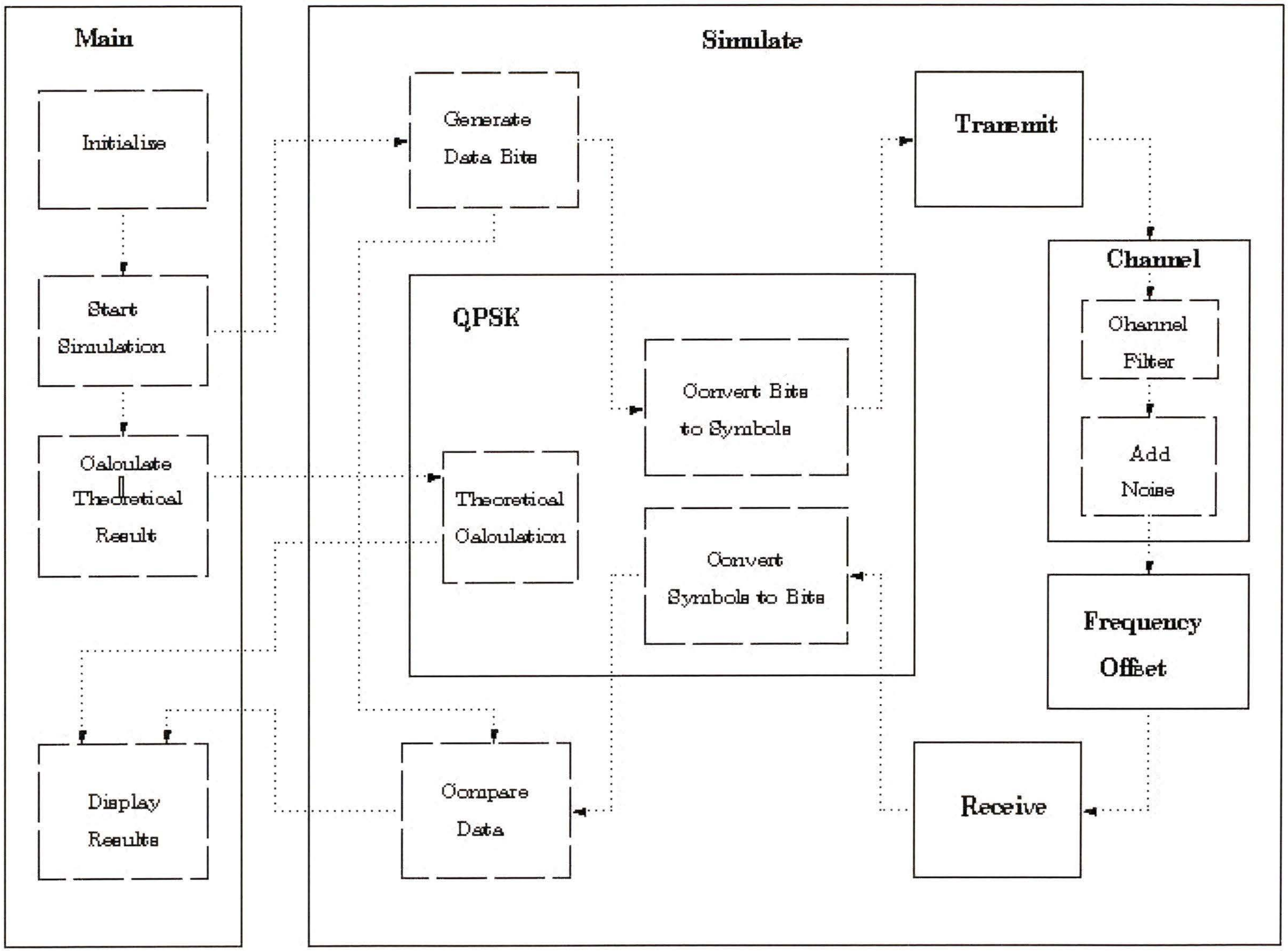

Figure 4.1 Simulation Code Block Diagram

4.2 Basic Modules

4.2.1 Main

This is the highest level module for the simulation. Here, the user enters the desired input parameters for the simulation (e.g. number of bits, signaling scheme, chosen SNRs, etc) and then runs the code. The software calculates a number of derived parameters and initializes some global variables. Then, for the chosen values of signal to noise ratio, the 'Simulate' submodule is called to simulate the transmission and reception of data through a communication channel. The bit error rate for each value of SNR is calculated and stored. The theoretical BER is calculated using the appropriate routine in the submodule appropriate to the chosen modulation scheme. Finally, the results of each simulation are plotted, along with the theoretical results.

4.2.2 Simulate

This module simulates the transmission and subsequent reception of a digital bit stream through an OFDM communication channel for a particular value of signal to noise ratio. The module contains two routines; one that performs the simulation, and the other that initializes the module.

The initialization operation is performed before any simulation takes place. During this operation, a number of variables are set up and some parameters dependent on the chosen modulation scheme are calculated.

The simulation function takes as a parameter a particular value of SNR, performs the simulation, and returns the BER calculated for this value of SNR.

To perform the simulation, a sequence of random digital bits, long enough to form a single OFDM frame, is generated. The bit sequence is converted to a symbol sequence using the chosen modulation scheme. A series to parallel operation on the symbol sequence is followed by subchannel modulation and addition of subchannels. Finally, the cyclic extension is added to form the transmitted signal.

The transmitted signal may optionally be filtered to simulate a channel frequency response, after which samples of Additive White Gaussian Noise (AWGN) are added. Before receiver operations are simulated, the received signal may optionally be modulated by a small carrier frequency offset.

Then subchannel demodulation, cyclic extension stripping, symbol detection, and parallel to serial conversion take place, followed by symbol to bit demapping.

Finally the received bitstream is compared with a copy of the transmitted bitstream and the number of bit errors logged. The steps above are repeated for subsequent frames until transmission and reception of the desired number of bits have been simulated. The module then return the overall bit error rate.

The Simulate module calls on the following submodules:

4.2.3 Transmit

The module receives as input a sequence of complex modulation symbols, sufficient for a single OFDM symbol. The OFDM symbol is formed, a cyclic extension is (optionally) added, and the symbol is either windowed or filtered. The output is a sequence of time domain samples.

4.2.4 Channel

The Channel module simulates the effect of channel distortion and addition of Additive White Gaussian Noise. The input is a sequence of time domain samples representing a single OFDM symbol. These may be filtered to simulate the effect of frequency dependant amplitude and phase distortion of the signal in the channel. Then, samples of randomly generated noise from a Gaussian distribution are scaled and added to each input sample. The scaling is chosen so that the ratio of the average signal and noise powers is equal to the chosen SNR. The output is another sequence of time domain samples.

4.2.5 Frequency Offset

This module simulates the effect of a Carrier Frequency Offset. The input is a sequence of time domain samples that are each multiplied by a complex number of modulus one to form the frequency shifted output sequence.

4.2.6 Receive

The Receive module inputs a sequence of data samples. The cyclic extension is stripped off and discarded. The remaining data samples are demodulated and an estimate of the transmitted symbol in each subchannel is determined. The output is a sequence of complex data symbols.

4.2.7 QPSK

The QPSK module encapsulates a number of routines that are dependant on the chosen modulation scheme (i.e. QPSK). If a different modulation scheme is desired, (e.g. QAM, MPSK), only these routines will need to be modified.

The routines handle data bit to complex symbol mapping, received symbol estimation, and complex symbol to data bit demapping. Further routines calculate the theoretical BER for the chosen modulation scheme and initialize the module.

4.2.8 Signal Power

This module is called by the Main module to estimate the transmitted signal power. A long sequence of OFDM symbols are generated using random bit data. The energy of each symbol is calculated and accumulated, and the average power of the whole sequence determined.

Chapter 5

5 Theoretical Calculations

5.1 Bit Error Rate

In this section we derive an expression for the Bit Error Rate of an OFDM system as a function of the Signal to Noise Ratio (E_b / N_0) and the normalized Carrier Frequency Offset (ΔFT).

5.1.1 Received Signal with Carrier Frequency Offset and White Gaussian Noise

We shall initially consider a system without a cyclic extension. In this case, the discrete time transmitted signal for a single OFDM symbol is given by (3.7) as

$$z(n) = \frac{1}{N} \sum_{k=0}^{N-1} a_k \exp\left(j \frac{2\pi kn}{N} \right) \tag{5.1}$$

The received signal, from (3.10) and (3.12), and subject to both Carrier Frequency Offset and Additive White Gaussian Noise, is then

$$r(n) = \exp j\left(\frac{2\pi \Delta FTn}{N} + \theta_0 \right)\left(\frac{1}{N} \sum_{k=0}^{N-1} a_k \exp\left(j \frac{2\pi kn}{N} \right) + z_N(n) \right), \tag{5.2}$$

where ΔFT is the Carrier Frequency Offset, normalized relative to the subcarrier spacing ($1/T$), θ_0 is the initial phase offset between the carrier and the receiver; and $z_N(n)$ is the Additive Gaussian White Noise sample sequence.

Performing a DFT on the received samples gives the demodulated signal in the mth subchannel as

$$
\begin{aligned}
r_m &= \sum_{n=0}^{N-1} \exp\left(-j\frac{2\pi mn}{N}\right) r(n) \\
&= \sum_{n=0}^{N-1} \exp\left(-j\frac{2\pi mn}{N}\right) \exp j\left(\frac{2\pi\Delta F T n}{N} + \theta_0\right)\left(\frac{1}{N}\sum_{k=0}^{N-1} a_k \exp\left(j\frac{2\pi kn}{N}\right) + z_N(n)\right) \qquad (5.3) \\
&= \exp(j\theta_0)\left(\sum_{k=0}^{N-1}\frac{a_k}{N}\sum_{n=0}^{N-1}\exp\left(j\frac{2\pi n(k-m+\Delta FT)}{N}\right)\right) + N_m
\end{aligned}
$$

where

$$N_m \doteq \exp(j\theta_0)\sum_{n=0}^{N-1}\exp\left(j\frac{2\pi n(\Delta FT - m)}{N}\right) z_N(n) \qquad (5.4)$$

is due to the demodulated Gaussian noise in subchannel m.

The inner sum of (5.3) is recognized as being a geometric series, which can be summed using the well-known formula $\sum_{k=0}^{N-1} r^k = \frac{1-r^N}{1-r}$

to give

$$
\begin{aligned}
\sum_{n=0}^{N-1} \exp\left(j\frac{2\pi n(k-m+\Delta FT)}{N} \right)
&= \frac{1-\exp\left(j\frac{2\pi N(k-m+\Delta FT)}{N} \right)}{1-\exp\left(j\frac{2\pi (k-m+\Delta FT)}{N} \right)} \\
&= \frac{\exp\left(-j\frac{\pi N(k-m+\Delta FT)}{N} \right) - \exp\left(j\frac{\pi N(k-m+\Delta FT)}{N} \right)}{\exp\left(-j\frac{\pi (k-m+\Delta FT)}{N} \right) - \exp\left(j\frac{\pi (k-m+\Delta FT)}{N} \right)} \\
&\qquad \times \frac{\exp\left(j\frac{\pi N(k-m+\Delta FT)}{N} \right)}{\exp\left(j\frac{\pi (k-m+\Delta FT)}{N} \right)} \\
&= \frac{\sin(\pi(k-m+\Delta FT))}{\sin\left(\frac{\pi(k-m+\Delta FT)}{N} \right)} \exp\left(j\frac{(N-1)(k-m+\Delta FT)}{N} \right)
\end{aligned}
\tag{5.5}
$$

Equations (5.3) and (5.5) together then give

$$
\begin{aligned}
r_m &= \exp(j\theta_0)\left(\sum_{k=0}^{N-1} \frac{a_k}{N} \frac{\sin(\pi(k-m+\Delta FT))}{\sin\left(\frac{\pi(k-m+\Delta FT)}{N} \right)} \exp\left(j\frac{(N-1)(k-m+\Delta FT)}{N} \right) \right) + N_m \\
&= \exp(j\theta_0)\left(a_m c_0 + \sum_{\substack{k=0 \\ k\neq m}}^{N-1} a_k c_{k-m} \right) + N_m,
\end{aligned}
\tag{5.6}
$$

where

$$
c_{k-m} \doteq \frac{1}{N} \frac{\sin(\pi(k-m+\Delta FT))}{\sin\left(\frac{\pi(k-m+\Delta FT)}{N} \right)} \exp\left(j\frac{(N-1)(k-m+\Delta FT)}{N} \right)
\tag{5.7}
$$

In expression (5.6) for r_m, the useful signal component is given by the term $a_m c_0$; the term N_m arises from the demodulated Gaussian noise; and the remaining terms arise from Inter Channel Interference (ICI). The expression reduces to (3.14) in the case of no phase or frequency offsets.

It is assumed in this analysis that the receiver is able to keep track of the carrier frequency and phase offsets of the received signal. To compensate for them, the receiver applies a phase rotation of $\exp(-j(\theta_0 + \arg(c_0))$ to each of the subchannel received signals to get

$$\begin{aligned} r'_m &= \exp(-j(\theta_0 + \arg(c_0)).\left(\exp(j\theta_0)\left(a_m c_0 + \sum_{\substack{k=0 \\ k \neq m}}^{N-1} a_k c_{k-m} \right) + N_m \right) \\ &= a_m \left| c_0 \right| + \exp\left(-j\arg(c_0)\right) \sum_{\substack{k=0 \\ k \neq m}}^{N-1} a_k c_{k-m} + N_{1m}, \end{aligned} \tag{5.8}$$

where

$$N_{1m} \doteq \exp(-j(\theta_0 + \arg(c_0))N_m, \tag{5.9}$$

and feeds the resulting subchannel signals to their respective decision devices. In expression (5.8), the first term is due to the desired signal component, the second is the ICI term due to the CFO, and the last is due to the AWGN.

5.1.2 Variance of Signal

After the phase rotation, the useful signal component in the mth subchannel, $a_m \left| c_0 \right|$, is attenuated by a factor of $\left| c_0 \right|$, which from (5.7) is

$$\left| c_0 \right| = \frac{1}{N} \frac{\sin(\pi \Delta F T)}{\sin\left(\frac{\pi \Delta F T}{N} \right)} \tag{5.10}$$

Hence the variance of the signal is

$$
\begin{aligned}
(V_s)_m &= E\left[|a_m|^2 |c_0|^2\right] \\
&= |c_0|^2 E_s \\
&= 2|c_0|^2 E_b
\end{aligned} \tag{5.11}
$$

where the variance of the complex data symbols in each subchannel is defined by

$$
E_s \doteq E\left[|a_m|^2\right], \tag{5.12}
$$

and for the QPSK symbols we are considering we also have that

$$
E_s = 2E_b. \tag{5.13}
$$

Consequently, from (5.11), the variance of the received signal is the same for each subchannel.

We also note that for a large number of subchannels (5.10) becomes

$$
\begin{aligned}
|c_0| &\simeq \frac{1}{N}\frac{\sin(\pi\Delta FT)}{\left(\dfrac{\pi\Delta FT}{N}\right)} \\
&= \frac{\sin(\pi\Delta FT)}{\pi\Delta FT} \\
&= \text{sinc}(\Delta FT)
\end{aligned} \tag{5.14}
$$

5.1.3 Noise and ICI Variance

The variance of the Gaussian noise sequence, $N_{1m} = \exp(-j(\theta_0 + \arg(c_0))N_m$, is unaffected by the phase rotation. Therefore the noise variance at the output of subchannel m is

$$
\begin{aligned}
(V_n)_m &= E\left[|N_{1m}|^2\right] \\
&= E\left[|N_m|^2\right] \\
&= N_0
\end{aligned} \tag{5.15}
$$

Additionally, there is ICI, with variance, from (5.8), of

$$
\begin{aligned}
(V_{ICI})_m &= E\left[\left|\exp(-j\arg(c_0))\sum_{\substack{k=0\\k\neq m}}^{N-1} a_k c_{k-m}\right|^2\right] \\
&= E\left[\left|\sum_{\substack{k=0\\k\neq m}}^{N-1} a_k c_{k-m}\right|^2\right] \\
&= E_s \sum_{\substack{k=0\\k\neq m}}^{N-1} \left|c_{k-m}\right|^2 \\
&= \frac{E_s}{N^2}\sum_{\substack{k=0\\k\neq m}}^{N-1} \frac{\sin^2(\pi(k-m+\Delta FT))}{\sin^2\left(\dfrac{\pi(k-m+\Delta FT)}{N}\right)} \\
&= \frac{2E_b}{N^2}\sum_{\substack{k=0\\k\neq m}}^{N-1} \frac{\sin^2(\pi(k-m+\Delta FT))}{\sin^2\left(\dfrac{\pi(k-m+\Delta FT)}{N}\right)},
\end{aligned}
\tag{5.16}
$$

where the third line follows because the data symbols, a_k, are uncorrelated (see Appendix A.3), and the fourth from equation (5.7).

We will now simplify the last line of (5.16). To begin with, we note that the numerators of each term in the sum are periodic, and by adding or subtracting suitable multiples of π to the argument of each sine function, each numerator may be written as $\sin^2(\pi\Delta FT)$. Similarly, we note that the denominators are periodic, and it is easy to check that by adding or subtracting appropriate multiples of π/N to the arguments of the sine functions, the denominators are just the values

$$
\sin^2\left(\frac{\pi(1+\Delta FT)}{N}\right),\ \sin^2\left(\frac{\pi(2+\Delta FT)}{N}\right),\ \ldots,\ \sin^2\left(\frac{\pi(N-1+\Delta FT)}{N}\right) \tag{5.17}
$$

in some order.

Hence equation (5.16) reduces (for all values of m) to

$$\left(V_{ICI}\right)_m = \frac{2E_b}{N^2}\sum_{k=1}^{N-1}\frac{\sin^2(\pi\Delta FT)}{\sin^2\left(\frac{\pi(k+\Delta FT)}{N}\right)} \tag{5.18}$$

We see that the variances of the Additive White Gaussian Noise, the variance of the Inter Channel Interference, and the variance of the signal are all independent of *m*. Consequently, from now on we drop the *m* subscript, for convenience.

In the literature, many authors assume that the ICI in subchannels at the band edges is different from the ICI for subchannels in the middle of the band, but we see from (5.18) that this is not the case.

To further simplify expression (5.18) we note, using (5.5) and (5.7), that the sum of all the quantities $|c_k|^2$ is

$$\begin{aligned}\sum_{k=0}^{N-1}|c_k|^2 &= \frac{1}{N^2}\left(\sum_{k=0}^{N-1}\sum_{n=0}^{N-1}\exp\left(j\frac{2\pi n(k+\Delta FT)}{N}\right)\right)\\ &\quad\times\left(\sum_{k_1=0}^{N-1}\sum_{n_1=0}^{N-1}\exp\left(j\frac{-2\pi n_1(k_1+\Delta FT)}{N}\right)\right)\\ &= \frac{1}{N^2}\sum_{n=0}^{N-1}\sum_{n_1=0}^{N-1}\left(\sum_{k=0}^{N-1}\exp\left(j\frac{2\pi nk}{N}\right)\right)\left(\sum_{k_1=0}^{N-1}\exp\left(j\frac{-2\pi n_1k_1}{N}\right)\right)\\ &\quad\times\exp\left(j\frac{2\pi(n-n_1)\Delta FT}{N}\right),\end{aligned} \tag{5.19}$$

where we use the case *m=0,* since, in accordance with the previous remarks, the sum is independent of *m*.

In this quadruple sum, if either $n\neq 0$ or $n_1\neq 0$, then one or the other of the sums over k or k_1 is equal to zero (using the well-known result that $\sum_{k=0}^{N-1}\omega^k=0$ if $\omega^N=1$, but $\omega\neq 1$), and their contribution to the overall sum is zero. This leaves the case $n = n_1 = 0$, when the sum reduces to

$$\frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{k_1=0}^{N-1}(1\times1\times1) = 1 \tag{5.20}$$

Hence we get that the overall sum is

$$\sum_{k=0}^{N-1}\left|c_k\right|^2 = 1, \tag{5.21}$$

(showing that the total power in the signal is unaffected by phase rotations of the individual frequency components), and so

$$\sum_{k=1}^{N-1}\left|c_k\right|^2 = 1-\left|c_0\right|^2. \tag{5.22}$$

Substituting this into equation (5.18) gives the final simplified expression for the variance of the ICI in any subchannel as

$$V_{ICI} = 2\left(1-\left|c_0\right|^2\right)E_b \tag{5.23}$$

This, too, is a new result, and applies to OFDM systems with any number of subchannels.

5.1.4 Expression for Bit Error Rate

We recall [51] that the Bit Error Rate of a QPSK signal having symbol energy E_s and subject to Additive White Gaussian Noise (in the absence of Carrier Frequency Offset) with zero mean is given by

$$BER = \frac{1}{2}erfc\left(\sqrt{\frac{1}{2}\frac{E_s}{N_0}}\right). \tag{5.24}$$

From (5.11), we have that the symbol energy in a system with Carrier Frequency Offset is attenuated by a factor of $|c_0|^2$; so in (5.24) we replace E_s by

$$E_s \rightarrow |c_0|^2 E_s \tag{5.25}$$

We would like to similarly replace the Gaussian Noise variance, N_0, by a new variance (computed from N_0 and the variance of the ICI) from a Gaussian distribution with zero mean. If the ICI were also from a Gaussian distribution, it would be legitimate to simply add the two variances to form a combined noise and ICI variance from a Gaussian distribution. However, although the ICI has zero mean, it is not Gaussian even for large N, since by (5.18) the ICI contributions from subchannels having a large value of k are negligible, and hence the Central Limit Theorem does not apply. Nevertheless, for small values of ΔFT, the ICI is small, and so the combined Gaussian noise and ICI distribution is dominated by the Gaussian noise term, and is hence approximately Gaussian. Therefore, using (5.23), we replace N_0 in (5.24) by

$$N_0 \rightarrow N_0 + \left(1 - |c_0|^2\right) E_s \tag{5.26}$$

The expression for the Bit Error Rate of an OFDM system subject to Additive White Gaussian Noise and a small Carrier Frequency Offset then becomes

$$BER = \frac{1}{2} erfc\left(\sqrt{\frac{1}{2}\frac{|c_0|^2 E_s}{N_0 + \left(1-|c_0|^2\right)E_s}}\right)$$

$$= \frac{1}{2} erfc\left(\sqrt{\frac{|c_0|^2 E_b}{N_0 + 2\left(1-|c_0|^2\right)E_b}}\right) \quad (5.27)$$

$$= \frac{1}{2} erfc\left(\sqrt{\frac{|c_0|^2 (E_b/N_0)}{1 + 2\left(1-|c_0|^2\right)(E_b/N_0)}}\right)$$

This last expression may be further simplified if the number of subchannels, N, is large using (5.14) to get

$$BER = \frac{1}{2} erfc\left(\sqrt{\frac{\text{sinc}^2(\Delta FT)(E_b/N_0)}{1 + 2\left(1-\text{sinc}^2(\Delta FT)\right)(E_b/N_0)}}\right), \quad (5.28)$$

which is independent of N. We see that it depends only on the two dimensionless quantities ΔFT and E_b/N_0.

For large values of E_b/N_0, the ICI is the dominant noise term and this expression tends to the constant value.

$$BER = \frac{1}{2} erfc\left(\sqrt{\frac{\text{sinc}^2(\Delta FT)}{2\left(1-\text{sinc}^2(\Delta FT)\right)}}\right) \quad (5.29)$$

In other words, for any particular value of Carrier Frequency Offset there exists an error floor below which the Bit Error Rate cannot fall, even by increasing the Signal to Noise Ratio. This derivation of the asymptotic Bit Error Rate of large E_b/N_0 calculated in (5.29) is not strictly correct, since it uses equations (5.27) and (5.28), which were themselves derived under the assumption that E_b/N_0 is, in fact, small. Nevertheless, the qualitative conclusion that for a fixed value of Carrier Frequency Offset the Bit Error Rate tends to a constant for large E_b must be correct, since in this case the Bit Error Rate is just a function of the ratio of E_b to $V_{ICI} = 2\left(1-|c_0|^2\right)E_b$, and this ratio is a constant independent of E_b. However, (5.29) is overly pessimistic about the BER at the error floor.

5.1.5 Bit Error Rate of System with Cyclic Extension

In a system with a cyclic extension that is neither windowed nor filtered, the transmitted signal may be written

$$z(n) = \frac{1}{N}\sqrt{\frac{N}{N+v_1+v_2}}\sum_{k=0}^{N-1} a_k \exp\left(j\frac{2\pi kn}{N}\right) \tag{5.30}$$

where v_1 is the number of prefix samples, v_2 is the number of postfix samples, and the data symbols a_k have variance given, as before, by (5.12). In this expression, n ranges from $-v_1$ to $N-1+v_2$.

The scaling of the signal is chosen so that the average power and energy per bit, E_b is the same as the case of a system without a cyclic extension, as in (5.1). However, the useful part of the received signal (after removal of the cyclic extension) is attenuated by the same scaling factor, and so in expression (5.27) we replace the variance E_s by

$$E_s \to \left(\frac{N}{N+v_1+v_2}\right)E_s \tag{5.31}$$

Hence equation (5.27) for the Bit Error Rate becomes

$$BER = \frac{1}{2} erfc\left(\sqrt{\frac{|c_0|^2 \left(\frac{N}{N+\nu_1+\nu_2}\right)\left(\frac{E_b}{N_0}\right)}{1+2\left(1-|c_0|^2\right)\left(\frac{N}{N+\nu_1+\nu_2}\right)\left(\frac{E_b}{N_0}\right)}}\right) \tag{5.32}$$

Similarly, equation (5.28) for a system with a large number of subchannels becomes

$$BER = \frac{1}{2} erfc\left(\sqrt{\frac{\text{sinc}^2(\Delta FT) \left(\frac{N}{N+\nu_1+\nu_2}\right)\left(\frac{E_b}{N_0}\right)}{1+2\left(1-\text{sinc}^2(\Delta FT)\right)\left(\frac{N}{N+\nu_1+\nu_2}\right)\left(\frac{E_b}{N_0}\right)}}\right) \tag{5.33}$$

The extension of these ideas to the case of a systems that use different signal constellations or a filtered of windowed cyclic extension is straightforward.

5.2 BER Degradation

Previous authors [42 - 44] have analyzed the effect of Carrier Frequency Offsets in terms of the BER Degradation, D, which may be defined [43] as "the increase of signal-to-noise ratio (SNR) required at the input of the decision device in order to compensate for the decrease in SNR caused by the carrier frequency offset".

From (5.27) the SNR at the input to the decision device when there is a CFO is

$$SNR = \frac{|c_0|^2 E_b}{N_0 + 2(1-|c_0|^2)E_b} \tag{5.34}$$

Comparing this with the SNR in the absence of CFO, E_b/N_0, and expressing the result in dB, gives

$$D = -10\log\left(\frac{|c_0|^2 E_b}{N_0 + 2(1-|c_0|^2)E_b} \cdot \frac{N_0}{E_b}\right)$$

$$= -10\log\left(\frac{|c_0|^2}{1 + 2(1-|c_0|^2)\dfrac{E_b}{N_0}}\right) \qquad (5.35)$$

$$= -10\log\left(|c_0|^2\right) + 10\log\left(1 + 2(1-|c_0|^2)\frac{E_b}{N_0}\right)$$

For large values of N this reduces to

$$D = -10\log\left(\text{sinc}^2(\Delta FT)\right) + 10\log\left(1 + 2(1-\text{sinc}^2(\Delta FT))\frac{E_b}{N_0}\right) \qquad (5.36)$$

This expression is a new result. For small values of ΔFT it may be written, using Taylor series expansions for sin() and ln(), as

$$D = -\frac{10}{\ln(10)}\ln\left(\left(\frac{\sin(\pi\Delta FT)}{\pi\Delta FT}\right)^2\right) + \frac{10}{\ln(10)}\ln\left(1 + 2\left(1 - \left(\frac{\sin(\pi\Delta FT)}{\pi\Delta FT}\right)^2\right)\frac{E_b}{N_0}\right)$$

$$\simeq -\frac{10}{\ln(10)}\ln\left(1 - \frac{1}{3}(\pi\Delta FT)^2\right) + \frac{10}{\ln(10)}\ln\left(1 + 2\left(1 - \left(1 - \frac{1}{3}(\pi\Delta FT)^2\right)\right)\frac{E_b}{N_0}\right) \qquad (5.37)$$

$$\simeq \frac{10}{\ln(10)}\frac{1}{3}(\pi\Delta FT)^2 + \frac{10}{\ln(10)}\frac{1}{3}(\pi\Delta FT)^2\frac{2E_b}{N_0}$$

$$= \frac{10}{\ln(10)}\frac{1}{3}(\pi\Delta FT)^2\left(1 + \frac{2E_b}{N_0}\right)$$

Similarly, for systems using a cyclic extension, we get from (5.33)

$$D = \frac{10}{\ln(10)}\frac{1}{3}(\pi\Delta FT)^2\left(1 + \frac{2E_b}{N_0}\frac{N}{N + \nu_1 + \nu_2}\right) \qquad (5.38)$$

Apart for a slight change in nomenclature, (5.38) is the expression derived for the degradation by Pollet and Moeneclaey in [43].

Chapter 6

6 Results

6.1 Comparison of Theoretical BER Formula with Simulations

6.1.1 Dependence of BER Formula on Number of Subchannels

From (5.10) and (5.23) we note that the dependence of the formula (5.27) for the BER on the number of subchannels, N, is all contained in the quantity $|c_0|$, defined in (5.10). However, from Figure 6.1 we see that for $N \geq 8$ there is essentially no difference in the value of $|c_0|$ for different values of N. Consequently, for $N \geq 8$ we may use the simplified formula for the BER given in (5.28), which is independent of N. In practice, the number of subchannels in an OFDM system will be much larger than 8.

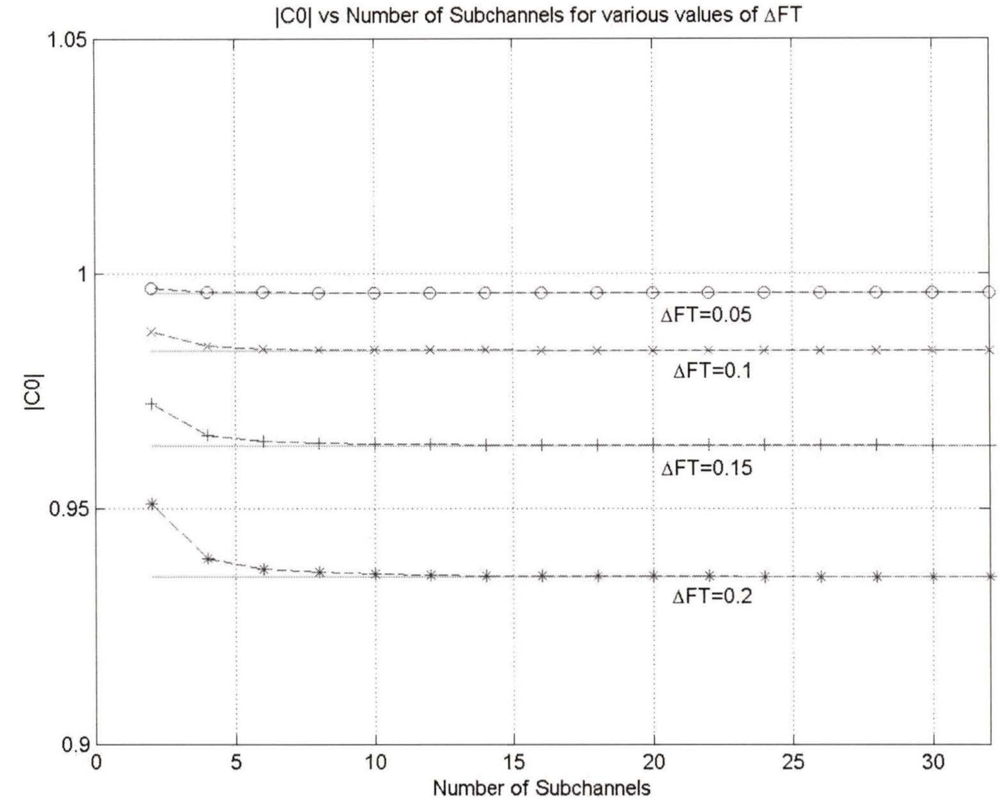

Figure 6.1 $|c_0|$ **vs Number of Subchannels for various values of** ΔFT

6.1.2 Comparison with 8 Subchannels

Figure 6.2, Figure 6.3 and Figure 6.4 compare formula (5.28) for the Bit Error Rate with simulations, using a number of different subchannels and no cyclic extension. We see that the results show a very close agreement (less than 0.2 dB) with the theory for values of ΔFT less than 0.1, and good agreement (less than 1 dB) for $0.1 \le \Delta FT \le 0.2$. In practice, ΔFT will usually be less than 0.1 [53]. We see that the formula deviates slightly from the simulations for large SNR, as expected, giving overly pessimistic estimates of the BER.

We also see that the simulations for 8, 16 and 32 subchannels are nearly identical, reiterating the fact that the Bit Error Rate is independent of the number of subchannels for $N \geq 8$.

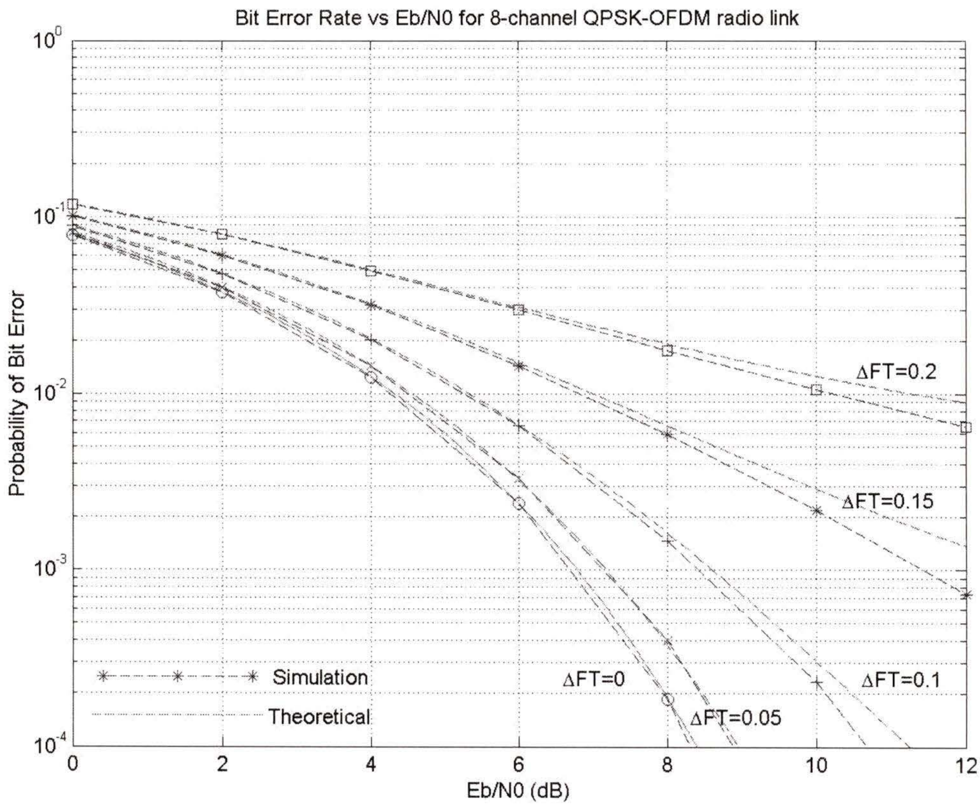

Figure 6.2 Probability of Bit Error vs E_b/N_0 for 8 channel OFDM System

6.1.3 Comparison with 16 Subchannels

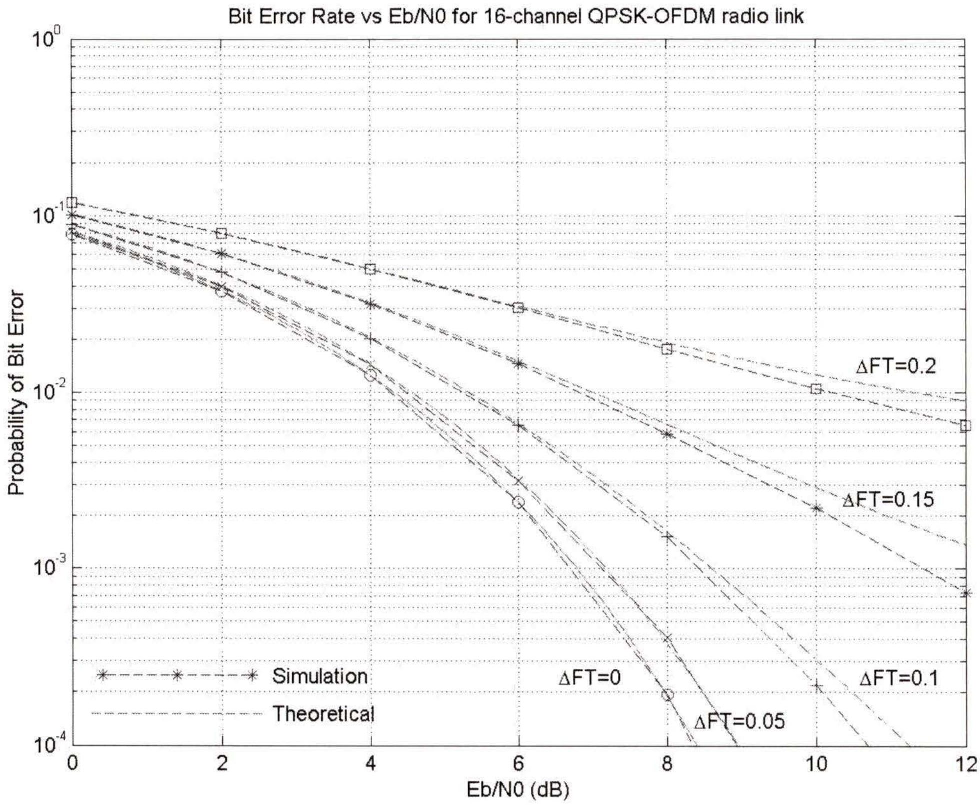

Figure 6.3 Probability of Bit Error vs E_b/N_0 for 16 channel OFDM System

6.1.4 Comparison with 32 Subchannels

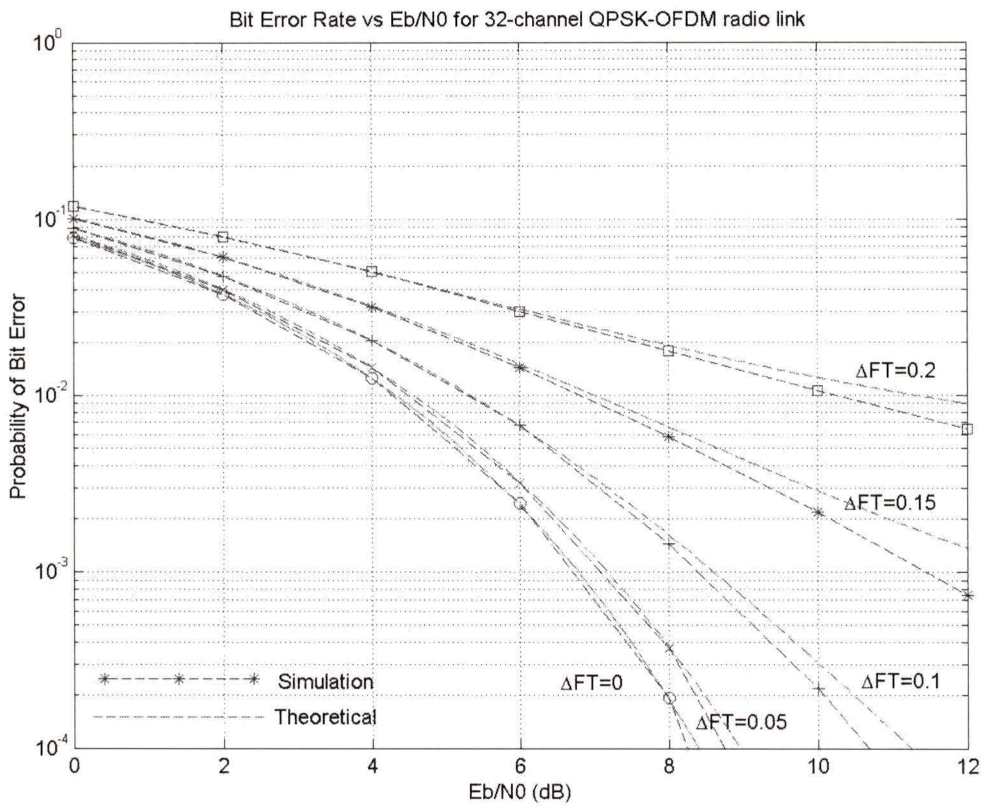

Figure 6.4 Probability of Bit Error vs E_b/N_0 for 32 channel OFDM System

6.1.5 System with Cyclic Prefix

Figure 6.5 compares formula (5.33) with a simulation of a system having 16 subchannels and a cyclic extension of 0.25. Also shown is the theoretical curve for a system with no cyclic extension and no CFO, for reference. Once again, there is good agreement (less than 1 dB for $\Delta FT < 0.1$) between the formula and the simulation for small values of CFO and SNR. However, the agreement is not quite as good as before, although for the values of CFO and SNR that are found in practice, it is adequate. As before, the formula is overly pessimistic.

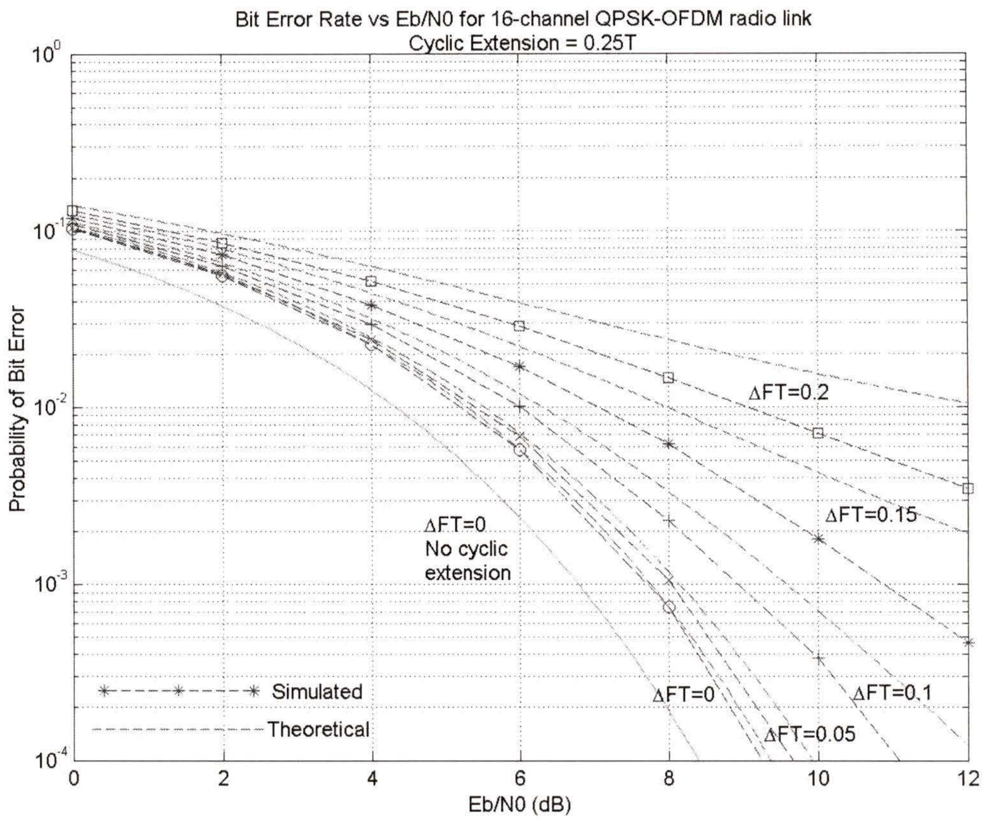

Figure 6.5 Probability of Bit Error vs E_b/N_0 for System with Cyclic Extension

6.2 Comparison of BER Degradation Formulas

In this section we compare the Degradation (D) taken from the simulation of Section 6.1.4 (having 32 subchannels) with the theoretical formulas for the Degradation given in (5.36) and (5.37). Figure 6.6 compares the simulated Degradation with the new formula (5.36), while Figure 6.7 is the comparison with formula (5.37) from the literature [43]. We see that the new formula is much more accurate than that previously published, particularly for large values of CFO and SNR.

6.2.1 Comparison with Equation (5.36)

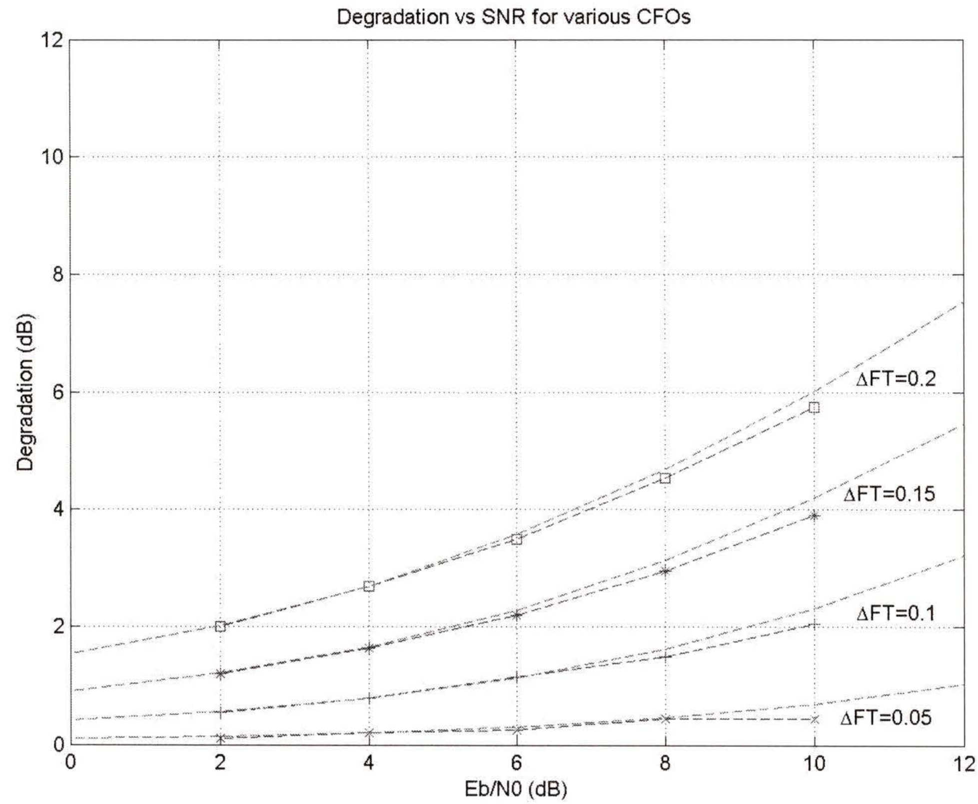

Figure 6.6 Degradation vs E_b/N_0 for various values of CFO

6.2.2 Comparison with Equation (5.37)

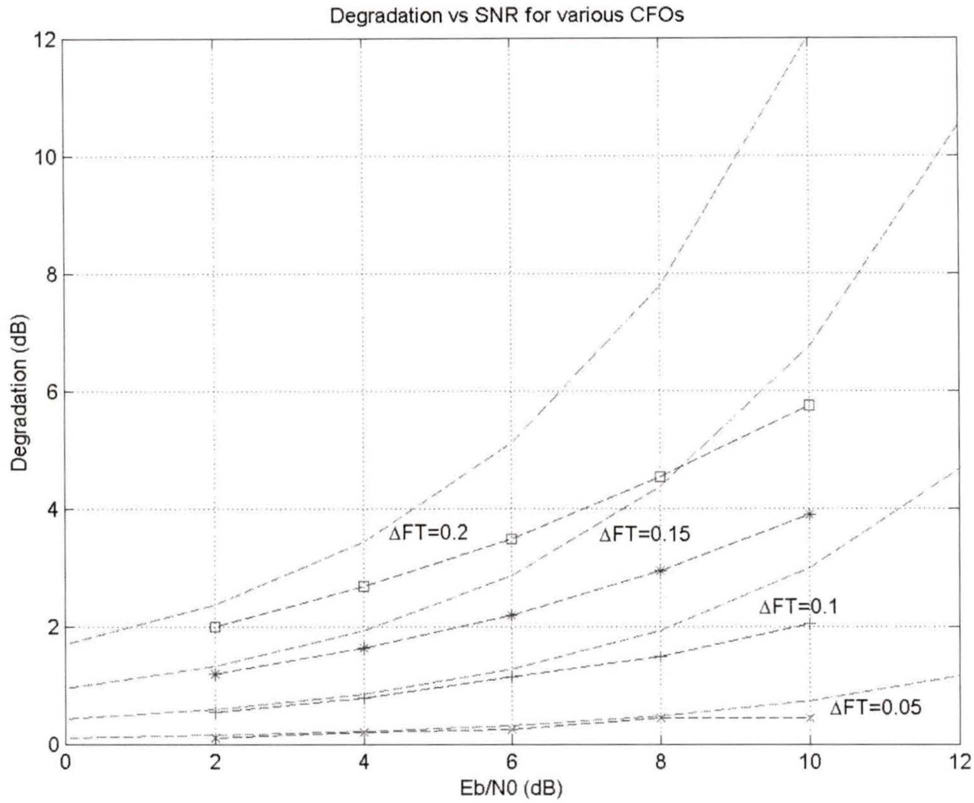

Figure 6.7 Degradation vs E_b/N_0 for various values of CFO

6.2.3 Comparison of Degradation Formulas

We may compare the accuracy of the new formula (5.36) with that of the old formula (5.37) by choosing a particular value of E_b/N_0 and plotting the difference between the theoretical Degradation and the simulated Degradation for different values of ΔFT. When this is done for $E_b/N_0 = 10$ dB we get Figure 6.8. We see that even at this large value of E_b/N_0 the difference between the new formula and the simulated results is just

0.25 dB for values of $\Delta FT \leq 0.2$, whereas the old formula differs by over 6 dB at $\Delta FT = 0.2$. Hence the new formula is very much better than the old.

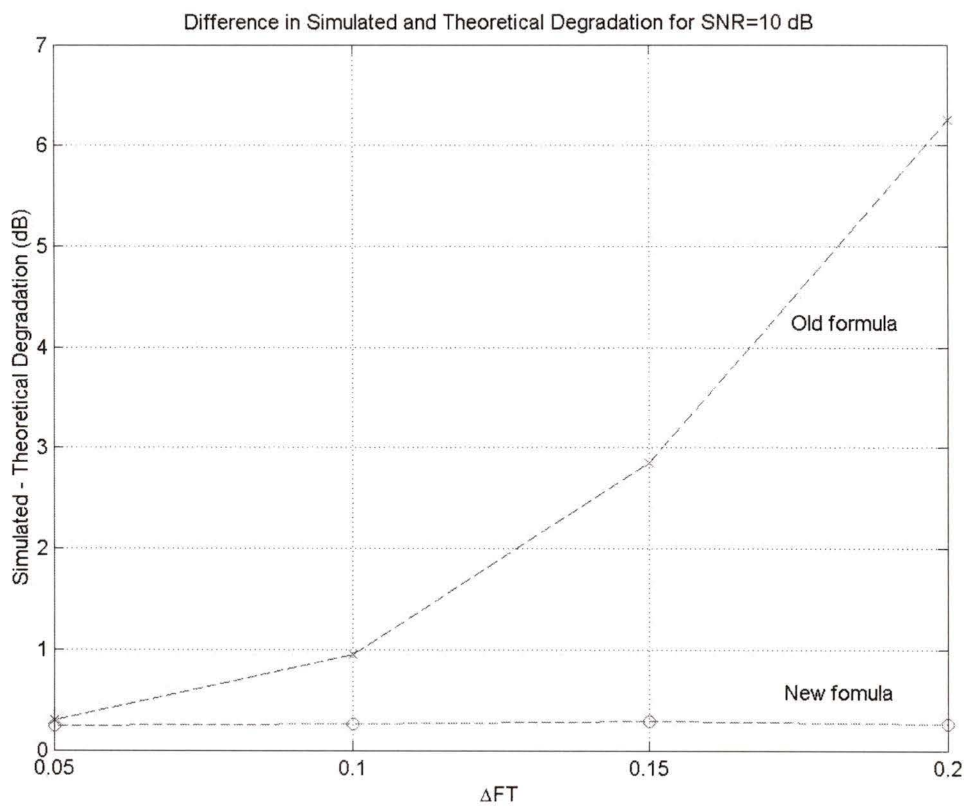

Figure 6.8 **Comparison of Degradation Formulas**

6.3 Recommendations of W-OFDM.

United States Patent 5,282,222 [53], the basis of the Wideband-OFDM (W-OFDM) system, gives (using a slightly different nomenclature) the following recommendations for choosing a maximum value of ΔFT such that the effect of combined Doppler shift and frequency offset does not substantially effect the BER:

When $0.2 \leq \beta \leq 0.3$, $\Delta FT \leq 0.075$

When $0.3 \leq \beta \leq 0.4$, $\Delta FT \leq 0.10$

When $0.4 \le \beta \le 0.5$, $\Delta FT \le 0.125$

When $0.5 \le \beta \le 0.6$, $\Delta FT \le 0.15$

Here, β is the roll-off of the band-limiting filter.

From equation (5.29), for the limiting case when E_b/N_0 is large so we are essentially considering the effect of frequency offsets alone, we get that the BERs for these values of ΔFT are:

For $\Delta FT = 0.075$, $BER = 1.3 \times 10^{-13}$

For $\Delta FT = 0.10$, $BER = 2.4 \times 10^{-8}$

For $\Delta FT = 0.125$, $BER = 7.0 \times 10^{-6}$

For $\Delta FT = 0.15$, $BER = 1.6 \times 10^{-4}$

We see that for all cases except the last, the BER arising from frequency offsets is negligibly small. For the last case, the BER arising from a frequency offset *may* not be negligible for a channel in which the Gaussian noise alone gives rise to BERs of less than about 10^{-3}. Hence we can concur with the recommendations of W-OFDM.

Chapter 7

7 Conclusion

We have presented an overview of OFDM and discussed the pros and cons of it as a signaling method. Several slightly different implementations have been discussed, and we have gone into some detail with the mathematics of OFDM itself and the techniques used to simulate OFDM systems.

In particular, we have developed a simple formula that predicts the BER for an OFDM system in the presence of Gaussian noise and Carrier Frequency Offsets. We have compared the predictions of this formula with the results of simulations and found the results match for small (less than 0.1) values of CFO. During the development of this formula, we have shown that the average ICI power in each subchannel is a constant independent of the subchannel position, contrary to previous beliefs. And using the formula, we have shown that there is an error floor below which the BER cannot drop, even by increasing the signal power. We have also derived a new formula for the Degradation caused by CFOs and found it significantly more accurate (less than 0.3 dB difference between the formula and simulations for CFOs up to 0.2) than a similar formula found in the literature.

For future work, we would like to modify the formulas for BER and Degradation to include OFDM systems in which the subcarriers are modulated by QAM or PSK symbols, rather than just QPSK. It would also be interesting to simulate the effects of timing offsets as well Carrier Frequency Offsets on the reception of OFDM signals, and compare these results with results from real systems.

Bibliography

[1] R. R. Mosier and R. G. Clabaugh, "Kiniplex, a Bandwidth Efficient Binary Transmission System," *AIEE Transactions*, Vol. 76, pp. 723 – 728, Jan. 1958.

[2] T. M. Cover and J. A. Thomas, "Elements of Information Theory," *John Wiley & Sons, Inc,* 1991.

[3] R. W. Chang, "Synthesis of Band Limited and Orthogonal Signals for Multichannel Data Transmission," *Bell Systems Technical Journal*, Vol. 45, pp. 1775 – 1796, Dec. 1966.

[4] B. R. Salzberg, "Performance of an Efficient Parallel Data Transmission System," *IEEE Transactions on Communications*, Vol. COM-15, pp. 805 – 813, Dec., 1967.

[5] S. B. Weinstein and P. M. Ebert, "Data Transmission by Frequency-Division Multiplexing Using the Discrete Fourier Transform," *IEEE Transactions on Communications Technology*, Vol. COM-19, No.5, pp.628--634, Oct. 1971.

[6] B. Hirosaki, "An Orthogonal-Multiplexed QAM System Using the Discrete Fourier Transform," *IEEE Transactions on Communications Technology,* Vol. COM-29, No.7, pp.982 - 989, July 1981.

[7] B. Hirosaki, S. Hasegawa, and A. Sabato, "Advanced Groupband Data Modem Using Orthogonally Multiplexed QAM Technique," *IEEE Transactions on Communications*, Vol. COM-34, No 6, pp 587 – 592, June, 1986.

[8] L. J. Cimini, "Analysis and Simulation of a Digital Mobile Channel Using Orthogonal Frequency Division Multiplexing," *IEEE Transactions on Communications*, Vol. COM - 33, No.7, pp. 665 - 675, July 1985.

[9] J. A. C. Bingham, "Multicarrier Modulation for Data Transmission: An Idea Whose Time Has Come," *IEEE Communications Magazine*, Vol. 28, No. 5, pp. 5 - 14, May 1990.

[10] S. Boyd, "Multitone Signals with Low Crest Factor," *IEEE Transactions on Circuits and Systems,* Vol. CAS – 33, No. 10, October 1986.

[11] S. Narahashi and T. Nojima, "New phasing scheme of N-multiple carriers for reducing peak-to-average power ratio," *Electronics Letters,* Vol. 30, No. 17, 18th August 1994.

[12] M. Friese, “Multitone Signals with Low Crest Factor,” *IEEE Transactions on Communications,* Vol. 45, pp. 1338 – 1344, October 1997

[13] D. Wulich “Reduction of peak to mean ratio of multicarrier modulation using cyclic coding,” *Electronics Letters,* Vol. 32, No. 5, 28th February 1996.

[14] S. Shepherd, J. Orriss and S. Barton, “Asymptotic Limits in Peak Power Reduction by Redundant Coding in Orthogonal Frequency-Division Multiplex Modulation,” *IEEE Transactions on Communications*, Vol. 46, No. 1, January 1998.

[15] A. E. Jones, T. A. Wilkinson and S. K. Barton, "Block Coding Scheme for Reduction of Peak to Mean Envelope Power Ratio of Multicarrier Transmission Schemes," *Electronics Letters,* Vol. 30, No. 5, pp. 2098 - 2099, 8th December, 1994.

[16] S. J. Shepherd, P. W. J. Van Eetwald, C. W. Wyatt-Millington and S. K. Barton, “Simple coding scheme to reduce peak factor in QPSK multicarrier modulation,” *Electronics Letters,* Vol. 31, No. 14, pp. 1131 - 1132, 6th July, 1994.

[17] R. D. J. Van Nee, “OFDM Codes for Peak-to-Average Power Reduction and Error Correction,” *IEEE Global Telecommunications Conference,* London, pp. 740 – 744, November 18 – 22 1996.

[18] P. W. J. Van Eetwald, S. J. Shepherd and S. K. Barton, “The Distribution of Peak Factor in QPSK Multi-Carrier Modulation,” *Wireless Personal Communications*, Vol. 2, pp. 87 – 96, November 1995.

[19] J. Armstrong, “New OFDM peak-to-average power reduction scheme”, *Vehicular Technology Conference, 2001. VTC 2001 Spring*. IEEE VTS 53rd , Vol. 1, pp. 756 –760, 2001.

[20] P. Van Eetvelt, G. Wade and M Tomlinson, “Peak to average power reduction for OFDM schemes by selective scrambling,” *Electronics Letters,* Vol. 32, No. 21, pp. 1963 - 1964, 10th October, 1996.

[21] R. W. Bauml, R. F. H. Fischer and J. B. Huber, “Reducing the peak-to-average power ratio of multicarrier modulation by selected mapping,” *Electronics Letters,* Vol. 32, No. 22, pp. 2056 - 2057, 24th October, 1996.

[22] S. Muller and J. Huber, “OFDM with reduced peak- to average power ratio by optimum combination of partial transmit sequences,” *Electronics Letters,* Vol. 33, No. 5, pp. 368 - 369, 27th February, 1997.

[23] S. H. Muller and J. B. Huber, "A comparison of peak power reduction schemes for OFDM," *Global Telecommunications Conference, 1997. GLOBECOM '97.,* IEEE, Vol. 1, pp. 1 –5, 1997

[24] L.J. Cimini and N. R. Sollenberger, "Peak-to Average Power Ratio Reduction of an OFDM Signal Using Partial Transmit Sequences," *IEEE Communications Letters,* Vol. 4, No. 3, March 2000.

[25] X. Li and L. J. Cimini, "Effects of Clipping and Filtering on the Performance of OFDM," *IEEE Communications Letters,* Vol. 2, No. 5, May 1998.

[26] D. Wulich and L. Goldfield, "Reduction of Peak Factor in Orthogonal Multicarrier Modulation by Amplitude Limiting and Coding," *IEEE Transactions on Communications*, Vol. 47, No. 1, January 1999.

[27] X. Wang, T. T. Tjhung and C. S. Ng, "Reduction of Peak-to-Average Power Ratio of OFDM System Using a Companding Technique," *IEEE Transactions on Broadcasting*, Vol. 45, No. 3, September 1999.

[28] A. Mattson, G. Mendenhall, T. Dittmer Harris, "Comments on: "Reduction of Peak-to-Average Power Ratio of OFDM System Using a Companding Technique," IEEE Transactions on Broadcasting, Vol. 45, No. 3, September 1999." *IEEE Transactions on Broadcasting*, Vol. 45, No. 4, December 1999.

[29] X. Wang, T. T. Tjhung and C. S. Ng, "Reply to the Comments on: "Reduction of Peak-to-Average Power Ratio of OFDM System Using a Companding Technique," IEEE Transactions on Broadcasting, Vol. 45, No. 3, September 1999." *IEEE Transactions on Broadcasting*, Vol. 45, No. 4, December 1999.

[30] W. Henkel and B. Wagner, "Another Application for Trellis Shaping: PAR Reduction for DMT (OFDM)," *IEEE Transactions on Communications*, Vol. 48, No. 9, September 2000.

[31] D. Wulich, "Comment on the Peak Factor of Sampled and Continuous Signals," *IEEE Communications Letters,* Vol. 4, No. 7, May 2000.

[32] P. H. Moose, "A Technique for Orthogonal Frequency Division Multiplexing Frequency Offset Correction," *IEEE Transactions on Communications,* Vol. 42, No. 10, pp 2908 – 2914, October 1994.

[33] M. Sandell, J.-J. van de Beek, P. O. Borjesson, "Timing and Frequency Synchronisation in OFDM Systems using the Cyclic Prefix," *International Symposium on synchronisation*, Essen, Germany, pp. 16 – 19, December 14 – 15 1995.

[34] J.-J. van de Beek, M. Sandell and P.O. Borjesson, "ML Estimation of Time and Frequency Offsets in OFDM Systems," *IEEE Transactions on Signal Processing,* Vol. 45, pp. 1800 – 1805, July 1997.

[35] Y. Li, L. J. Cimini and N. Sollenberger, "Robust Channel Estimation for OFDM Systems with Rapid Dispersive Fading Channels," *IEEE Transactions on Communications,* Vol. 46, No. 7, pp 902 – 915, July 1998.

[36] L. Hazy and M. El-Tanany, "Synchronisation of OFDM Systems over Frequency Selective Fading Channels," *Proceedings of the 47th Vehicular Technology Conference*, Phoenix, AZ, pp. 2094 – 2098, IEEE May 1997.

[37] P. F. Driessen, "DPLL bit synchronizer with rapid acquisition using adaptive Kalman filtering techniques," *IEEE Transactions on Communications,* Vol. 42, No. 9, September 1994.

[38] T. Keller, L. Piazzo, P. Mandarini and L. Hanzo, "Orthogonal frequency division multiplex synchronization techniques for frequency-selective fading channels," *IEEE Journal on Selected Areas in Communications*, Vol. 19, No. 6, pp. 999 –1008, June 2001

[39] N. Pothecary, "Feedforward Linear Power Amplifiers," *Artech House*, 1999.

[40] J. K. Cavers, "Optimum indexing in predistorting amplifier linearizers," *Vehicular Technology Conference, 1997,* IEEE 47th, Vol. 2, pp. 676 - 680, 1997.

[41] S. I. Mann, M. A. Beach, K. A. Morris, "Digital baseband Cartesian loop transmitter," *Electronics Letters*, Vol. 37, No. 22, pp. 1360 – 1361, 25 October 2001.

[42] T. Pollet, M. Van Bladel and M. Moeneclaey, "BER sensitivity of OFDM systems to carrier frequency offset and Wiener phase noise," *IEEE Transactions on Communications,* Vol. 43 No. 2 Part 3, pp. 191 –193 Feb.-March-April 1995

[43] T. Pollet and M. Moeneclaey, "The effect of carrier frequency offset on the performance of band limited single carrier and OFDM signals," *Global Telecommunications Conference, 1996. GLOBECOM '96. 'Communications: The Key to Global Prosperity*, Vol. 1 , pp. 719 -723 1996.

[44] P. K. Remvik and N. Holte, "Carrier frequency offset robustness for OFDM systems with different pulse shaping filters," *Global Telecommunications Conference, 1997. GLOBECOM '97.,* IEEE , Vol. 1, pp. 11 – 15, 1997 .

[45] J. Armstrong, "Analysis of new and existing methods of reducing intercarrier interference due to carrier frequency offset in OFDM," *IEEE Transactions on Communications,* Vol. 47, No. 3, pp. 365 –369, March 1999.

[46] Y. Li and L. J. Cimini, "Bounds on the interchannel interference of OFDM in time-varying impairments," *IEEE Transactions on Communications*, Vol. 49, No. 3, pp. 401 – 404, March 2001.

[47] K. Sathananthan and C. Tellambura, "Probability of error calculation of OFDM systems with frequency offset," *IEEE Transactions on Communications*, Vol. 49, No. 11, pp. 1884 -1888 November 2001.

[48] R. Van Nee and R. Prasad, "OFDM for Wireless Multimedia Communications," *Artech House*, 2000.

[49] J.A.C. Bingham, "ADSL, VDSL, and Multicarrier Modulation," *John Wiley and Sons, Inc.*, 2000.

[50] F.M. Gardner and J.D.Baker, "Simulation Techniques," *John Wiley and Sons, Inc.,* 1997.

[51] B. Sklar, "Digital Communications," *Prentice-Hall, Inc.,* 1988.

[52] S. G. Wilson, "Digital Modulation and Coding," *Prentice-Hall, Inc.,* 1999

[53] M Fattouche and H Zagloul, "Method and Apparatus for Multiple Access between Tranceivers in Wireless Communications using OFDM Spread Spectrum," *United States Patent 5,282,222,* 1994

[54] http://www.ofdm-forum.com

[55] R. L. Kirlin, *Private communication*, 2002

Appendix

A.1 Power and Noise Spectrums

A.1.1 Power Spectrum of a Composite Multichannel System

Here we show that the power spectral density (PSD) of any multicarrier signal, (not just an OFDM signal), is equal to the sum of the PSDs of the individual subcarriers, provided the crosscorrelation between any two different subcarriers is zero. (Also see section A.3).

The multicarrier signal $z(t)$ may be written as

$$z(t) = \sum_{k=0}^{N-1} z_k(t), \tag{A.1}$$

where z_k is the kth subcarrier.

The autocorrelation function, $R(t,t+\tau)$, of $z(t)$ is

$$\begin{aligned} R(t,t+\tau) &= E\left[z(t).z^*(t+\tau)\right] \\ &= E\left[\sum_{k=0}^{N-1} z_k(t).\sum_{m=0}^{N-1} z_m^*(t+\tau)\right] \\ &= E\left[\sum_{k=0}^{N-1} z_k(t).z_k^*(t+\tau)\right] + E\left[\sum_{\substack{k,m \\ k\neq m}}^{N-1} z_k(t) z_m^*(t+\tau)\right] \\ &= \sum_{k=0}^{N-1} E\left[z_k(t).z_k^*(t+\tau)\right] + \sum_{\substack{k,m \\ k\neq m}}^{N-1} E\left[z_k(t) z_m^*(t+\tau)\right]. \end{aligned} \tag{A.2}$$

The assumption that the crosscorrelation between each pair of subcarriers is zero means that each term in the second sum on the right hand side of A.2 is zero, so the equation reduces to

$$R(t,t+\tau)=\sum_{k=0}^{N-1}R_k(t,t+\tau) \tag{A.3}$$

where $R_k(t,t+\tau)$ is the autocorrelation of the kth subcarrier.

Taking the Fourier Transform of each side, and using the linearity of the transform together with the fact that the Power Spectral Density is the Fourier Transform of the autocorrelation function, we arrive at the desired result

$$P(f)=\sum_{k=0}^{N-1}P_k(f) \tag{A.4}$$

A.1.2 Noise Power Spectrum

We now show that the noise sequence $z_N(n)$, generated in section 3.3.3.4 above, is white across the bandwidth B.

The autocorrelation of the noise sequence $z_N(n)$ (having variance σ^2) is, by definition,

$$\begin{aligned} R_N(n,n+\eta) &= E\left[z_N(n).z_N^*(n+\eta)\right] \\ &= \sigma^2\delta(\eta). \end{aligned} \tag{A.5}$$

The PSD of the noise sequence is then obtained from the FFT of the autocorrelation function as

$$\begin{aligned} P_N(k) &= \sum_{\eta=0}^{N-1}R_N(n,n+\eta)\exp\left(-j\frac{2\pi k\eta}{N}\right) \\ &= \sum_{\eta=0}^{N-1}\sigma^2\delta(\eta)\exp\left(-j\frac{2\pi k\eta}{N}\right) \\ &= \sigma^2 \end{aligned} \tag{A.6}$$

which is constant for each discrete frequency k. Hence the noise sequence $z_N(n)$ is white.

A.2 Evaluation of SNR

In this section, we show that the SNR, defined in (3.29) for a single subchannel, at the output of the correlator or matched filter can be evaluated as

$$SNR = \frac{E\left[|A|^2\right]}{\sigma^2} = \frac{2E_s}{N_0} \tag{A.7}$$

We show this for the case of a correlator only, since the calculation for a matched filter is very similar.

For QAM or PSK based systems, the signal in a single subchannel due to a single symbol can be written as

$$s(t) = a\phi(t) \tag{A.8}$$

where a is the complex modulating symbol and $\phi(t) = \exp(j2\pi f_m t)$.

The output of the of the correlator at strobe time due to the signal component is

$$\begin{aligned} A &= \int_0^T s(t)\varphi^*(t)dt \\ &= a\int_0^T \phi^2(t)dt \end{aligned} \tag{A.9}$$

Hence

$$\begin{aligned} E\left[|A|^2\right] &= E\left[\left|a\int_0^T \phi^2(t)dt\right|^2\right] \\ &= E\left[|a|^2\right]\left|\int_0^T \phi^2(t)dt\right|^2 \end{aligned} \tag{A.10}$$

The output of the correlator due to the noise component is

$$z_n(T) = \int_0^T n(t)\varphi^*(t)dt \tag{A.11}$$

The variance of this quantity, assuming the noise has zero mean and power spectral density $N_0/2$, is

$$\begin{aligned}
\sigma^2 &= E\left[\left|z_n(T)\right|^2\right] \\
&= E\left[\left|\int_0^T n(t)\varphi^*(t)dt\right|^2\right] \\
&= E\left[\left(\int_0^T n(t)\varphi^*(t)dt\right)\left(\int_0^T n^*(s)\varphi(s)ds\right)\right] \\
&= \int_0^T\int_0^T E\left[n(t)n^*(s)\right]\varphi^*(t)\varphi(s)dsdt \\
&= \int_0^T\int_0^T \frac{N_0}{2}\delta(t-s)\varphi^*(t)\varphi(s)dsdt \\
&= \frac{N_0}{2}\int_0^T \varphi^2(t)dt
\end{aligned} \tag{A.12}$$

Hence

$$\begin{aligned}
\frac{E\left[\left|A\right|^2\right]}{\sigma^2} &= \frac{E\left[\left|a\right|^2\right]\left|\int_0^T \phi^2(t)dt\right|^2}{\frac{N_0}{2}\int_0^T \varphi^2(t)dt} \\
&= \frac{E\left[\left|a\right|^2\right]\int_0^T \phi^2(t)dt}{N_0/2}
\end{aligned} \tag{A.13}$$

However, the average symbol energy, E_s, is just

$$E_s = E\left[\int_0^T |a\phi(t)|^2 \, dt\right] = E\left[|a|^2\right]\int_0^T \phi^2(t)dt \tag{A.14}$$

so (A.13) becomes

$$\frac{E\left[|A|^2\right]}{\sigma^2} = \frac{E_s}{N_0/2} = \frac{2E_s}{N_0} \tag{A.15}$$

A.3 Definitions

The following definitions [55] are used in the text.

A.3.1 Uncorrelated

Random variables x and y are said to be ***uncorrelated if their covariance = 0***:

$$E\left[\left(x-\mu_x\right)\left(y-\mu_y\right)\right]=0 \text{ or } \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left(x-\mu_x\right)\left(y-\mu_y\right)p(x,y)dxdy=0$$

A.3.2 Orthogonal

Random variables are ***orthogonal if their expected product = 0***:

$$E[xy]=0 \text{ or } \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}xyp(x,y)dxdy=0$$

So if two random variables are uncorrelated AND have zero mean, they are orthogonal.

A.3.3 Independent

Random variables are ***independent if their joint density is factorable***:

$$f(x,y)=f(x)f(y\,|\,x)=f(x)f(y)$$

The density is a function of all the moments and not just the second moment.

Jointly Gaussian random variables are functions of only their first and second moments, so we can say that:

Jointly Gaussian random variables that are uncorrelated are also independent, because their densities will factor. This is not true in general for other densities.

A.4 Simulation Code

A.4.1 Main Routine

A.4.1.1 Main.m

```
%       Name:           Main.m
%       Purpose:        Main routine for OFDM simulation.  Performs the following tasks:
%                       1. Sets up simulation parameters (e.g. number of bits, signaling
%                          scheme, etc).
%                       2. Performs simulations of the system at different signal to
%                          noise ratios.
%                       3. Calculates theoretical ideal results at different signal to noise
%                          ratios.
%                       4. Displays the simulated and theoretical results.
%       Description:    The user enters the desired input parameters for the simulation and
%                       then runs the code.  The software calculates a number of derived
%                       parameters and initializes the global variables.  A large number of
%                       simulated transmitter frames are generated and the average
%                       transmitter power calculated.  Then, for a number values of signal
%                       to noise ratio (SNR), a simulation of the transmission and
%                       reception of data through a communication channel is performed.
%                       The bit error rate (BER) for each simulation is calculated and
%                       stored.  The theoretical BER is calculated.  Finally, the results of
%                       each simulation are plotted along with the theoretical results.
%       Input:          User's input parameters.
%       Output:         Plot of simulated and theoretical bit error rates versus signal to
%                       noise ratio.
% Start of Main.m code.
clear all               % Clear memory of previous variables and functions.
close all               % Close previous figure windows.
hold off                % Default plot mode.
disp(' ');
disp( ['INITIALIZING'] );
% Global simulation parameters and variables.
% Simulation parameters.
global nBits;           % Total number of data bits in each simulation.
global BitRate;         % Overall bit rate.
global Modulation;      % Subchannel symbol modulation type (e.g. QPSK, BPSK, QAM).
global nChannels;       % Number of OFDM subchannels.
global OfdmType;        % FFT or OQAM.
global nDataSamples;    % Number of data samples in one OFDM frame.
global r;               % Rolloff of Root Raised Cosine filter in the case of OQAM,
                        % or Raised Cosine window in the case of FFT.
global FilterType;      % Root raised cosine or raised cosine.
global Delay;           % Number of symbols periods the filter delays the data.
global DeltaF;          % Receiver Carrier Frequency Offset (CFO) normalized
                        % relative to the subchannel spacing.
global ChannelFilter;   % Frequency response of channel.
global nPowerFrames;    % Number of OFDM frames used in signal power calculation.
% SNR parameters.
global SimEbOverN0dBTable;   % Table of SNR values (in dB) for simulation.
```

```
global ThEbOverN0dBTable;      % Table of SNR values (in dB) for theoretical
                               % calculation.
% Derived parameters.
global BitsPerSymbol;          % Number of data bits per subchannel symbol.
global BitsPerFrame;           % Number of data bits per OFDM frame.
global nFrames;                % Number of OFDM frames in each simulation.
global FrameRate;              % Overall frame rate.
global ChannelFilterLength;    % Length of channel filter.
global TxSignalPower;          % Power of transmitted signal.
% Set up simulation parameters.
nBits=16384;                   % Total number of data bits in each simulation.
BitRate=9600;                  % Overall bit rate.
Modulation='QPSK';             % Subchannel symbol modulation type is QPSK.
nChannels=2;                   % Number of OFDM subchannels.
OfdmType='OQAM';               % OFDM type is OQAM.
nDataSamples=8*nChannels;      % Number of data samples in a single symbol.
r=1/4;                         % Rolloff.
FilterType='fir/sqrt';         % Root raised cosine filter implemented as a FIR filter.
DeltaF=1/32768;                % Normalized receiver carrier frequency offset (CFO).
ChannelFilter=[1];             % Flat fading channel.
nPowerFrames=nBits/nChannels;% Number of transmitter frames in calculation of
                               % power.
MaxdB=10;                      % Maximum Eb/N0 value in dB.
MinBer=10^(-5);                % Minimum value of plotted Bit Error Rate.
SimEbOverN0dBTable=[2:5:MaxdB];      % Eb/N0 values for simulation in dB.
ThEbOverN0dBTable=[0:0.05:MaxdB];    % Eb/N0 values for theoretical calculation
                                     % in dB.
% Calculate derived simulation parameters
% Get simulation parameters dependent on symbol modulation.
feval( Modulation, 'Init', [] );
% Get simulation parameters dependent on OFDM type.
feval( [OfdmType 'Simulate'], 'Init', [] );
BitsPerFrame=BitsPerSymbol*nChannels;        % Number of data bits per OFDM frame.
if ( rem( nBits, BitsPerFrame) )             % Check no bits are left over.
   error('Frame Error')
end
nFrames=nBits/BitsPerFrame;          % Number of OFDM frames in each simulation.
FrameRate=BitRate/BitsPerFrame;      % Overall frame rate.
ChannelFilterLength=length(ChannelFilter);   % Calculate length of channel filter.
% Power calculation;
disp(' ');
disp( ['CALCULATING SIGNAL POWER'] );
TxSignalPower=SignalPower;           % Calculate signal power.
disp( ['SIGNAL POWER = ' num2str( TxSignalPower )] ); % Display power.
% Simulation
for m=1:length( SimEbOverN0dBTable ) % For each value of SNR (Eb/N0)...
   disp(' ');
   disp( ['SIMULATION with Eb/N0 = ' num2str(SimEbOverN0dBTable(m)) ' dB
      ...STARTED'] );
                                     % ...get simulated BER.
   [SimErr(m)]=feval( [OfdmType 'Simulate'], 'Sim', SimEbOverN0dBTable(m) );
   disp(['SIMULATION with Eb/N0 = ' num2str(SimEbOverN0dBTable(m)) ' dB
      ...FINISHED']);
   disp(' ');
end;
```

```
% Theoretical calculations
for m=1:length(ThEbOverN0dBTable)     % For each value of SNR (Eb/N0) in dB...
                                      % ... calculate theoretical BER.
   [ThErr(m)]=feval( Modulation, 'Theoretical', 10^( ThEbOverN0dBTable(m)/10 ) );
end
%Comparison and plot
semilogy( SimEbOverN0dBTable, SimErr, '*b', ThEbOverN0dBTable, ThErr, 'g' )
axis( [ThEbOverN0dBTable(1)  ThEbOverN0dBTable( length( ThEbOverN0dBTable )
   ...MinBer  1] )
ylabel( 'Probability of Bit Error' )
xlabel( 'Eb/N0 (dB)' )
if nChannels==1
   ModulationString=[' ' Modulation];
else
   ModulationString=[OfdmType '-OFDM'];
end
title( ['Bit Error Rate v Eb/N0 for ' ModulationString ' radio link'] )
grid
```

A.4.2 FFT Routines

A.4.2.1 FFTSimulate.m

```
%       Name:           FFTSimulate.m
%       Purpose:        Module for simulating the transmission and subsequent reception of a
%                       digital bit stream through an FFT-OFDM communication channel for a
%                       particular value of signal to noise ratio(SNR).
%       Description:    The module is initialized by calling the function with the first
%       argument being the string 'Init'.  This operation is performed before any simulation
%        takes place.  During the initialization phase, the cyclic prefix and postfix
%        windows and other parameters specific to FFT-OFDM communications
%        are calculated.
%        With the first argument as 'Sim', the module simulates a communication
%        link using the value of SNR in the second argument.  The various
%        parameters used by the simulation are stored in global variables.  The
%        following operations are performed:
%        A random sequence of bits long enough to form a single OFDM frame is
%        generated.  The bit sequence is converted to a symbol sequence using the
%        chosen modulation scheme.  A series to parallel operation on the symbol
%        sequence is followed by subchannel modulation and addition of
%        subchannels.  Finally, a cyclic prefix and postfix are generated, and the
%        prefix is combined with the previous frames postfix and prepended to
%        the signal to form a complex baseband signal corresponding to a single
%        transmitted OFDM frame.
%        The transmitted signal may optionally be filtered to simulate a channel
%        frequency response, after which samples of Additive White Gaussian
%        Noise (AWGN) are added.
%        Before receiver operationsare simulated, the received signal may
%        optionally be modulated by a small carrier frequency offset (CFO).
%       Then subchannel demodulation, prefix and postfix stripping, symbol
%       detection, and parallel to serial conversion take place, followed by
%                       symbol to bit demapping.
%                       Finally the received bitstream is compared with a copy of the transmitted
```

```
%                       bitstream and the number of bit errors logged.
%                       The steps above are repeated for subsequent frames until transmission
%                       and reception of the desired number of bits have been simulated.  The
%                       module then returns with the overall bit error rate (BER).
%       Input:          Command:
%                           'Init' - Initializes the module.
%                           'Sim' - Performs a simulation for one value of SNR (Eb/N0).
%                       EbOverN0dB:      The signal to noise ratio in dB.
%       Output:BER:     Bit error rate of simulation.
function [BER]=FFTSimulate( Command, EbOverN0dB );
global nBits;               % Total number of data bits in each simulation.
global Modulation;          % Subchannel symbol modulation type (e.g. QPSK, BPSK, QAM).
global nChannels;           % Number of OFDM subchannels.
global nDataSamples;        % Number of data samples in one OFDM frame.
global r;                   % Rolloff of Root Raised Cosine filter in the case of OQAM,
                            % or Raised Cosine window in the case of FFT.
global DeltaF;              % Normalized receiver frequency offset.
global Ms;                  % Total number of samples in each OFDM frame.
global SymbolRate;          % Overall symbol rate.
global BitsPerFrame;        % Number of data bits per OFDM frame.
global nFrames;             % Number of OFDM frames in each simulation.
global nPrefixSamples;  % Number of cyclic prefix samples in one FFT OFDM frame.
global nPostfixSamples; % Number of cyclic postfix samples in one FFT OFDM frame.
global PrefixWindow;        % Window function for the cyclic prefix.
global PostfixWindow;       % Window function for the cyclic postfix.
global PhaseOffset;         % Normalized phase offset for receiver due to cyclic prefix
                            % and postfix.
global TxPostfix;           % Current cyclic postfix
global ChannelZi;           % Matrix of initial conditions for channel filter.
global ChannelFilterLength; % Length of channel filter.
global TxSignalPower;   % Signal power of transmitted signal.
global Sigma;               % Standard deviation of AWGN noise samples appropriate
                            % to the chosen SNR.
global FrameCount;          % Number of received OFDM frames.
% Start of FFTSimulate.m code.
switch Command
        case 'Init'         % Initialize the module.
    nPrefixSamples=nDataSamples*r;  % Number of prefix samples computed from
                                    % rolloff.
   nPostfixSamples=nPrefixSamples;  % Same number of postfix samples.
    % Number of data samples in an FFT-OFDM frame.
    Ms=nPrefixSamples/2+nDataSamples+nPostfixSamples/2;
    % Raised cosine prefix window with rolloff r.
    PrefixWindow=( 1+cos( pi*(1+(0:nPrefixSamples-1)/nPrefixSamples) ) )/2;
    PostfixWindow=fliplr( PrefixWindow );% Postfix window is mirror image of prefix.
    % Normalized receiver phase offset due to prefix and postfix with CFO.
    PhaseOffset=(1+2*r)/(2*(1+r));
  case 'Sim'                        % Begin the simulation.
    BitErrors=0;                    % Initialize bit error accumulator.
    ChannelZi=zeros(ChannelFilterLength-1,1); % Channel filter initial conditions.
    TxPostfix=zeros(1,nPostfixSamples);         % Initialize first postfix.
    EbOverN0=10^(EbOverN0dB/10);              % Calculate SNR.  (This is defined as the
                                              % ratio of Eb, energy per transmitted bit, to
                                              % N0, the noise spectral density).
    % Calculate noise sample standard deviation for the chosen value of SNR.
```

```
      Sigma=sqrt( Ms*TxSignalPower/(2*BitsPerFrame*EbOverN0) );
      FrameCount=0;      % Initialize frame count.
      for n=0:nFrames-1  % For desired number of frames...
         TxBits=round( rand( 1, BitsPerFrame ) ); % Generate random bit sequence.
         % Convert bits to symbols.
         TxSymbols=feval( Modulation, 'Bits2Symbols', TxBits );
         TxSignal=FFTTransmit( TxSymbols );      % Form FFT-OFDM signal frame.
         RxSignal=Channel( TxSignal );            % Add channel distortion and
                                                  % noise (AGWN).
         RxSignal=FrequencyOffset( RxSignal );    % Add receiver carrier frequency
                                                  % offset(CFO).
         RxSymbols=FFTReceive( RxSignal );        % Receive and demodulate signal.
         % Convert symbols to bits.
         RxBits=feval( Modulation, 'Symbols2Bits', RxSymbols );
         BitErrors=BitErrors+nnz( TxBits-RxBits );          % Number of bit errors.
      end                   % end for
      BER=BitErrors/nBits;          % Calculate bit error rate for simulation.
end    % end switch
```

A.4.2.2 FFTTransmit.m

```
%     Name:        FFTTransmit.m
%     Purpose:     Forms a complex baseband OFDM signal frame using a Inverse
%     Fast Fourier Transform applied to the subchannel symbols.
%     Description:
%                  An Inverse Fast Fourier Transform (IFFT) is applied to the input
%                  vector of subchannel symbols to form the data sequence for the current
%                  frame.
%                  A cyclic prefix and postfix are generated and windowing is applied.  The
%                  prefix is combined with the previous cyclic postfix, and the data
%                  sequence is added.  The cyclic postfix is saved for combination with the
%                  next frame.
%     Input:       FrameSymbols:   A vector of subchannel symbols for a single OFDM
%                                  frame.
%     Output:      TxSignal: The complex baseband transmitted signal sample
%                     sequence.
function TxSignal=FFTTransmit( FrameSymbols )
global nDataSamples;          % Number of data samples in one OFDM frame.
global nPrefixSamples;        % Number of cyclic prefix samples in one FFT
                              % OFDM frame.
global nPostfixSamples;       % Number of cyclic postfix samples in one FFT
                              % OFDM frame.
global PrefixWindow;          % Window function for the cyclic prefix.
global PostfixWindow;         % Window function for the cyclic postfix.
global PrefixWindow;          % Window function for the cyclic prefix.
global PostfixWindow;         % Window function for the cyclic postfix.
global TxPostfix;             % Current cyclic postfix.
% Code for FFTTransmit.m
TxData=ifft( FrameSymbols, nDataSamples ).';  % Get data samples using IFFT
                                              % operation.
% Get cyclic prefix, window, and add to previous postfix.
TxPrefix=(.TxData( nDataSamples-nPrefixSamples+1:nDataSamples ).*PrefixWindow )
         ...+TxPostfix;
TxPostfix=TxData( 1:nPostfixSamples ).*PostfixWindow; % Save current windowed
```

```
                                    % postfix.
TxSignal=[TxPrefix TxData];         % Form OFDM frame from cyclic prefix and data
                                    % samples.
```

A.4.2.3 FFTReceive.m

```
%   Name:        FFTReceive.m
%   Purpose:     Receives and demodulates the subchannel symbols from a transmitted
%                OFDM signal frame using the Fast Fourier Transform (FFT) operation.
%   Description: An OFDM frame is received and the data samples stripped from
%                the cyclic prefix. A Fast Fourier Transform (FFT) operation is
%                applied to
%                recover the received subchannel symbols. Compensation is applied to
%                correct for the Carrier Frequency Offset (CFO), and a decision as to
%                which symbols were transmitted is made.
%   Input:       RxSignal: A frame of received samples.
%   Output:      RxSymbols:A vector of received and demodulated subchannel symbols.
function RxSymbols=FFTReceive( RxSignal )
global Modulation;      % Subchannel symbol modulation type (e.g. QPSK, BPSK, QAM).
global nChannels;       % Number of OFDM subchannels.
global nDataSamples;    % Number of data samples in one OFDM frame.
global DeltaF;          % Receiver Carrier Frequency Offset(CFO) normalized relative to
                        % the subchannel spacing.
global Ms;              % Total number of samples in each OFDM frame.
global nPrefixSamples; % Number of cyclic prefix samples in one FFT OFDM frame.
global PhaseOffset;     % Normalized phase offset for receiver due to cyclic prefix
                        % and postfix.
global FrameCount;      % Number of received OFDM frames.
% Code for FFTReceive.m
% Demodulate using FFT.
RxFrame=( fft( RxSignal( nPrefixSamples+1:nPrefixSamples+nDataSamples ) ) ).';
ReceivedSymbols=RxFrame( 1:nChannels );      % Get subchannel symbols.
% Apply phase correction proportional to the number of symbols and the CFO, plus a
% fixed phase offset.
ReceivedSymbols=ReceivedSymbols*exp(-j*2*pi*DeltaF*(FrameCount+PhaseOffset) );
FrameCount=FrameCount+1;                % Increment frame counter.
% Make decision about which symbols were transmitted.
RxSymbols=feval( Modulation, 'Decide', ReceivedSymbols );
```

A.4.3 OQAM Routines

A.4.3.1 OQAMSimulate.m

```
%   Name:        OQAMSimulate.m
%   Purpose:     Module for simulating the transmission and subsequent reception of a
%                digital bit stream through an OQAM-OFDM communication channel for
%                a particular value of signal to noise ratio(SNR).
%   Description: The module is initialized by calling the function with the first
%                argument being the string 'Init'. This operation is performed before
%                any simulation takes place. During the initialization phase, the transmitter
%                and receiver filter coefficients and other parameters specific to
%                OQAM-OFDM communications are calculated.
%   With the first argument as 'Sim', the module simulates a communication
```

```
%       link using the value of SNR in the second argument.  The various
%       parameters used by the simulation are stored in global variables.  The
%       following operations are performed:
%       A random sequence of bits long enough to form a single OFDM frame is
%       generated.  The bit sequence is converted to a symbol sequence using the
%       chosen modulation scheme.  A series to parallel operation on the symbol
%       sequence is followed by subchannel modulation and filtering.  Finally,
%       the addition of subchannels forms a complex baseband signal
%       corresponding to a single transmitted OFDM frame.
%       The transmitted signal may optionally be filtered to simulate a channel
%       frequency response, after which samples of Additive White Gaussian
%       Noise (AWGN) are added.
%       Before receiver operations are simulated, the received signal may
%       optionally be modulated by a small carrier frequency offset (CFO).
%       Then subchannel demodulation, filtering, symbol detection, and parallel
%       to serial conversion take place, followed by symbol to bit demapping.
%       Finally the received bitstream is compared with a copy of the transmitted
%       bitstream (appropriately delayed to compensate for the delays introduced
%       by the transmit, receive and channel filters), and the number of bit errors
%       is logged.
%       The steps above are repeated for subsequent frames until transmission
%       and reception of the desired number of bits have been simulated.  The
%       module then returns with the overall bit error rate (BER).
%      Input:          Command:
%                      'Init' - Initializes the module.
%                         'Sim' - Performs a simulation for one value of SNR (Eb/N0).
%                      EbOverN0dB:      The signal to noise ratio in dB.
%       Output:        BER:             Bit error rate of simulation.
function [BER]=OQAMSimulate( Command, EbOverN0dB );
global nBits;            % Total number of data bits in each simulation.
global Modulation;       % Subchannel symbol modulation type (e.g. QPSK, BPSK, QAM).
global nChannels;        % Number of OFDM subchannels.
global nDataSamples;     % Number of data samples in one OFDM frame.
global r;                % Rolloff of Root Raised Cosine filter in the case of OQAM,
                         % or Raised Cosine window in the case of FFT.
global FilterType;       % Root raised cosine or raised cosine.
global Delay;            % Number of symbols periods the filter delays the data.
global DeltaF;           % Normalized receiver frequency offset.
global Ms;               % Total number of samples in each OFDM frame.
global SymbolRate;       % Overall symbol rate.
global BitsPerFrame;     % Number of data bits per OFDM frame.
global nFrames;          % Number of OFDM frames in each simulation.
global RealTxFilter;     % Subchannel transmit filter for real part of OQAM symbol.
global ImagTxFilter;     % Subchannel transmit filter for imaginary part of OQAM symbol.
global RealRxFilter;     % Subchannel receive filter for real part of OQAM symbol.
global ImagRxFilter;     % Subchannel receive filter for imaginary part of OQAM symbol.
global TxFilterLength;   % Length of transmit filters.
global RxFilterLength;   % Length of receive filters.
global TxZiReal;         % Matrix of initial conditions for real symbol transmit filters.
global TxZiImag;         % Matrix of initial conditions for imaginary symbol transmit
                         % filters.
global ChannelZi;        % Matrix of initial conditions for channel filter.
global RxZiReal;         % Matrix of initial conditions for real symbol receive filters.
global RxZiImag;         % Matrix of initial conditions for imaginary symbol transmit
                         % filters.
```

```
global ChannelFilterLength;% Length of channel filter.
global TxSignalPower;   % Signal power of transmitted signal.
global Sigma;           % Standard deviation of AWGN noise samples appropriate to the
                        % chosen SNR.
global FrameCount;      % Number of received OFDM frames.
% Start of OQAMSimulate.m code.
switch Command
     case 'Init'                 % Initialize the module.
   Ms=nDataSamples;              % Number of data samples in an OQAM-OFDM frame.
   fd=SymbolRate/nChannels; % Symbol rate per subchannel.
   fs=fd*Ms;                     % Sample rate.
   Delay=3;                      % Number of symbol delays caused by filter.
   % Calculate raised cosine filter coefficients.
   RaisedCosineFilter=rcosine( fd, fs, FilterType, r, Delay );
   % Introduce half symbol delay between real and imaginary parts of transmitted
   % signal.
   % Add delay of half a symbol to imaginary symbol transmit filter.
   ImagTxFilter=[zeros( 1, Ms/2 ) RaisedCosineFilter];
   % Make real transmit filter the same length as imaginary filter.
   RealTxFilter=[RaisedCosineFilter zeros( 1, Ms/2 )];
   % Add delays to resynchronize the real and imaginary parts of the received signal.
   RealRxFilter=ImagTxFilter; % Add delay of half a symbol to real symbol
                                 % receive filter
   ImagRxFilter=RealTxFilter; % Make imaginary receive filter the same length as
                                 % real filter.
   TxFilterLength=length( RaisedCosineFilter )+Ms/2;  % Calculate filter lengths.
   RxFilterLength=TxFilterLength;
     case 'Sim'        % Begin the simulation.
   BitErrors=0;        % Initialize bit error accumulator.
   % Initial conditions are zero for all filters.
   TxZiReal=zeros( TxFilterLength-1, nChannels );
   TxZiImag=TxZiReal;
   ChannelZi=zeros( ChannelFilterLength-1, 1 );
   RxZiReal=zeros( RxFilterLength-1, nChannels );
   RxZiImag=RxZiReal;
   % Calculate SNR, defined as the ratio of Eb, energy per transmitted bit, to N0, noise
   % spectral density, from SNR in dB.
   EbOverN0=10^( EbOverN0dB/10 );
   % Calculate noise sample standard deviation for the chosen value of SNR.
   Sigma=sqrt( Ms*TxSignalPower/(2*BitsPerFrame*EbOverN0) );
   FrameCount=0;                 % Initialize frame count.
   FrameDelay=(2*Delay+1);       % Received frames will be delayed this amount
                                 % by transmit and receive filters.
   BitDelay=FrameDelay*BitsPerFrame;% Received bit delay.
   % Initialize vector that will offset the transmitted bits by the received bit delay, for
   % bit error comparison.
    DelayedTxBits=zeros( 1, BitDelay );
   % For desired number of frames, plus frames to flush filters...
   for n=0:nFrames-1+(2*Delay)
      TxBits=round( rand( 1, BitsPerFrame ) ); % Generate random bit sequence.
       % Introduce delay to help later comparison with received bits.
      DelayedTxBits=[DelayedTxBits(BitsPerFrame+1:BitDelay) TxBits];
       % Convert bits to symbols.
     TxSymbols=feval( Modulation, 'Bits2Symbols', TxBits );
     TxSignal=OQAMTransmit( TxSymbols );  % Form OQAM-OFDM signal frame.
```

```
        RxSignal=Channel( TxSignal );  % Add channel distortion and noise (AGWN).
        % Add receiver carrier frequency offset(CFO).
        RxSignal=FrequencyOffset( RxSignal );
        RxSymbols=OQAMReceive( RxSignal );  % Receive and demodulate signal.
        % Convert symbols to bits.
        RxBits=feval( Modulation, 'Symbols2Bits', RxSymbols );
        % Calculate errors
        if (n>=2*Delay)  % After delays due to filters...
          % ...compare received bits to transmitted bits.
          BitErrors=BitErrors+nnz( DelayedTxBits(1:BitsPerFrame)-RxBits );
        end            % end if
      end              % end for
      BER=BitErrors/nBits;                                              %
Calculate bit error rate for simulation.
end                    % end switch
```

A.4.3.2 OQAMTransmit.m

```
%     Name:        OQAMTransmit.m
%     Purpose:     Transmits an OFDM frame using the Offset Quadrature Amplitude
%                  Modulation -     Orthogonal Frequency Division Multiplexing
%                  ( OQAM-OFDM ) technique.
%     Description: The real and imaginary parts of each subchannel symbol are applied to
%                  individual subchannel filters, with delays appropriate to OQAM.  The
%                  outputs of each pair of filters are added and modulated onto
%                  subchannels.  The subchannels signals are then summed to form a full
%                  OQAM-OFDM frame.
%     Input:       FrameSymbols:    A vector of complex subchannel symbols.
%     Output:      TxSignal:  The transmitted OQAM-OFDM signal sample sequence.
function TxSignal=OQAMTransmit( FrameSymbols )
global nChannels;    % Number of OFDM subchannels.
global Ms;           % Total number of samples in each OFDM frame.
global RealTxFilter; % Subchannel transmit filter for real part of OQAM symbol.
global ImagTxFilter; % Subchannel transmit filter for imaginary part of OQAM symbol.
                     % Note that this filter also delays the signal by one half of a
                     % symbol time relative to the real transmit filter.
global TxZiReal;     % Matrix of initial conditions for real symbol transmit filters.
global TxZiImag;     % Matrix of initial conditions for imaginary symbol transmit
                     % filters.
% Start of OQAMTransmit code.
% The following lines construct sequences of impulses proportional to the symbol values
% to input each of the transmit filters.  The impulses occur at the frame rate.
RealMatrix=zeros( nChannels, Ms );     % Initialize sequences to all zeros.
ImagMatrix=RealMatrix;
RealMatrix(:,1)=real( FrameSymbols );  % Form impulse.
ImagMatrix(:,1)=imag( FrameSymbols ); % Form impulse.
% Filter real sequence.
[FilteredRealMatrix TxZiReal]=filter( RealTxFilter, 1, RealMatrix, TxZiReal, 2 );
% Filter and delay imaginary sequence.
[FilteredImagMatrix TxZiImag]=filter( ImagTxFilter, 1, ImagMatrix, TxZiImag, 2);
% Add real and imaginary signals from each subchannel.
TxMatrix=FilteredRealMatrix+j*FilteredImagMatrix;
TxSignal=zeros( 1, Ms );          % Initialize output sequence.
for k=0:nChannels-1               % For all subchannels...
```

```
  % ...Modulate and sum the  subchannels.
  TxSignal=TxSignal+TxMatrix(k+1,:).*exp( j*2*pi*k*( (0:Ms-1)/Ms+1/4 ) );
end
```

A.4.3.3 OQAMReceive.m

```
%       Name:         OQAMReceive.m
%       Purpose:      Receives and demodulates the subchannel symbols from a transmitted
%                     OQAM-OFDM signal frame.
%       Description:  Each subchannel signal is individually demodulated and split into
%                      real and imaginary parts.  Each part is then applied to an individual
%                     receive filter and delays appropriate to OQAM introduced.
%                     The real and imaginary parts of each symbol are combined and a
%                      phase correction to compensate for the Carrier Frequency Offset
%                     (CFO) is applied.  Finally, a decision as to which symbols were
%                     transmitted is made.
%       Input:        RxSignal:  The received OQAM-OFDM signal sample sequence.
%       Output:       RxSymbols:         A vector of received subchannel symbols.
function RxSymbols=OQAMReceive( RxSignal )
global Modulation;    % Subchannel symbol modulation type (e.g. QPSK, BPSK, QAM).
global nChannels;     % Number of OFDM subchannels.
global Delay;         % Number of symbols periods the filter delays the data.
global DeltaF;        % Receiver Carrier Frequency Offset (CFO) normalized relative to
                      % the subchannel spacing.
global Ms;            % Total number of samples in each OFDM frame.
global RealRxFilter;  % Subchannel receive filter for real part of OQAM symbol.
global ImagRxFilter;  % Subchannel receive filter for imaginary part of OQAM symbol.
global RxZiReal;      % Matrix of initial conditions for real symbol receive filters.
global RxZiImag;      % Matrix of initial conditions for imaginary symbol transmit
                      % filters.
global FrameCount;    % Number of received OFDM frames.
% Start of OQAMReceive.m code.
RxMatrix=zeros( nChannels, Ms );          % Initialize matrix of complex subchannel signals.
for k=0:nChannels-1       % For each subchannel...
  % ...Demodulate subchannel.
  RxMatrix(k+1,:)=RxSignal.*exp( -j*2*pi*k*( (0:Ms-1)/Ms+1/4 ) );
end
RealMatrix=real( RxMatrix );              % Split subchannels into real and imaginary parts.
ImagMatrix=imag( RxMatrix );
% Filter and delay real signals.
[FilteredRealMatrix RxZiReal]=filter( RealRxFilter, 1, RealMatrix, RxZiReal, 2 );
% Filter imaginary signals.
[FilteredImagMatrix RxZiImag]=filter( ImagRxFilter, 1, ImagMatrix, RxZiImag, 2 );
% Construct complex received symbols.
ReceivedSymbols=FilteredRealMatrix(:,Ms/2+1)+j*FilteredImagMatrix(:,Ms/2+1);
% Apply phase correction proportional to the number of symbols and the CFO.
ReceivedSymbols=ReceivedSymbols*exp( -j*2*pi*DeltaF*(FrameCount-Delay+0.25) );
FrameCount=FrameCount+1;                  % Increment frame counter.
% Make decision about which symbols were transmitted.
RxSymbols=feval( Modulation, 'Decide', ReceivedSymbols );
```

A.4.4 Symbol Modulation

A.4.4.1 QPSK.m

```
%       Name:    QPSK.m
%       Purpose: Module that performs the transmission, reception and decoding
%                functions specific to QPSK modulation.
%       Description:    The module is initialized by calling the function with the first
%                (Command) argument being the string 'Init'.  This operation is performed
%                before any simulation takes place.  During the initialization phase
%                various parameters,   such as the number of bits per QPSK symbol, are
%                calculated.
%                   Further calls invoke the following functions:
%                      Bits2Symbols:  Converts the vector of bits stored in the second
%                                     (Data)   argument into a vector of QPSK symbols.
%                      Decide: Makes a decision on which of the received QPSK
%                                     symbols in the 'Data' vector was actually
%                                     transmitted.
%                      Symbols2Bits:  Converts the vector of QPSK symbols stored in the
%                                     second (Data) argument into a vector of bits.
%                      Theoretical:   Calculates the theoretical bit error rate
%                                     (BER) expected for QPSK modulation at the
%                                     value of SNR found in the second argument.
%       Input:      Command:
%                      'Init' - Initializes the module.
%                      'Bits2Symbols' - Converts bits to QPSK symbols.
%                      'Decide' - Decides which QPSK symbol was transmitted.
%                      'Symbols2Bits' - Converts QPSK symbols to bits.
%                      'Theoretical' - Calculates theoretical BER.
%                   Data:       Input data vector appropriate to the function called in the
%                               Command argument.
%       Output:     Out:        Output data vector appropriate to the function called in the
%                               Command argument.
function Out=QPSK( Command, Data )
global nBits;            % Total number of data bits in each simulation.
global BitRate;          % Overall bit rate.
global BitsPerSymbol;    % Number of data bits per symbol.
global nSymbols;         % Total number of symbols in each simulation.
global SymbolRate;       % Overall symbol rate.
global Bit2SymbolTable;% Lookup table for bit to QPSK symbol conversion.
global Symbol2BitTable;% Lookup table for QPSK symbol to bit conversion.
% Start of QPSK.m code.
switch Command
  case 'Init'                        % Initialize the QPSK module before a simulation.
    BitsPerSymbol=2;  % Number of data bits in a QPSK symbol.
    if (rem( nBits, BitsPerSymbol ))      % Check no bits are left over.
      error( 'Symbol Error' )
    end
    nSymbols=nBits/BitsPerSymbol;     % Number of QPSK symbols in the simulation.
    SymbolRate=BitRate/BitsPerSymbol;% Symbol rate per subchannel.
    Bit2SymbolTable=[-1-j -1+j;         % Define a lookup table for bit to symbol
                      1-j 1+j];         % conversion.
```

```
    Symbol2BitTable=[0 0;                          % Define a lookup table for symbol to bit
                     0 1;                          % conversion.
                     1 0;
                     1 1];
  case 'Bits2Symbols'                       % Convert a bit vector to a QPSK symbol vector
                                            % by table lookup.
    Out=zeros( length( Data )/2, 1 );       % Initialize the output vector.
    for n=0:(length( Data )/2)-1            % For each pair of bits in the input...
      % ... convert to a QPSK symbol.
      Out(n+1)=Bit2SymbolTable(Data(2*n+1)+1,Data(2*n+2)+1);
    end
  case 'Decide'          % Decides from a vector of received symbols which symbol was
                         % transmitted.
    Out=zeros( length( Data ), 1 );                % Initialize the output vector.
    for k=1:length( Data )                         % For each received symbol...
       if real( Data(k) )>0                        % ...decide on the real part.
          Out(k)=1;
      else
         Out(k)=-1;
      end
      if imag( Data(k) )>0                         % ...decide on the imaginary part.
         Out(k)=Out(k)+j;
      else
         Out(k)=Out(k)-j;
     end
    end                  % end for
  case 'Symbols2Bits'                       % Convert a QPSK symbol vector to a bit vector
                                            % by table lookup.
    Out=zeros( 1, 2*length( Data ) );       % Initialize the output vector.
    for n=0:length( Data )-1                       % For each QPSK symbol...
      p=(real( Data(n+1) )+1)+(imag( Data(n+1) )+1)/2; %...Form index into table
      Out(2*n+1:2*n+2)=Symbol2BitTable(p+1,:);         % ...generate two output bits.
   end
  case 'Theoretical'     % Calculate the theoretical bit error rate (BER) for QPSK
                                % modulation at a particular value of Eb/No (SNR).
    Out=0.5*erfc( sqrt( Data ) );%p389, p393
end     % end switch
```

A.4.5 Calculation of Signal Power

A.4.5.1 SignalPower.m

```
%       Name:       SignalPower.m
%       Purpose:    Estimates the signal power for a given modulation and OFDM type.
%       Description:The module generates 'nPowerFrames' worth of random transmitted
%                   OFDM frames and keeps an accumulating count of their energy. The
%                   power is then calculated from the total energy and the number of
%                   samples.
%       Input:      None.
%       Output:     Power:    The estimated signal power.
function Power=SignalPower()
global Modulation;    % Subchannel symbol modulation type (e.g. QPSK, BPSK, QAM).
global nChannels;     % Number of OFDM subchannels.
```

```
global OfdmType;          % FFT or OQAM.
global nPowerFrames;      % Number of OFDM frames used in signal power calculation.
global BitsPerFrame;      % Number of data bits per OFDM frame.
global Ms;                % Total number of samples in each OFDM frame.
global nPostfixSamples;   % Number of cyclic postfix samples in one FFT OFDM frame.
global TxPostfix;         % Current cyclic postfix
global Delay;             % Number of symbols periods the filter delays the data.
global TxFilterLength;    % Length of transmit filters.
global TxZiReal;          % Matrix of initial conditions for real symbol transmit filters.
global TxZiImag;          % Matrix of initial conditions for imaginary symbol transmit
                          % filters.
% Start of SignalPower.m code.
switch OfdmType
  case 'FFT'                        % FFT-OFDM
    TxPostfix=zeros( 1, nPostfixSamples );      % Initialize first postfix.
     % Generate next postfix.
    TxBits=round( rand( 1, BitsPerFrame ) );    % Generate random bit sequence.
    TxSymbols=feval( Modulation, 'Bits2Symbols', TxBits );% Convert bits to symbols.
    TxSignal=FFTTransmit( TxSymbols );          % Form FFT-OFDM signal frame.
  case 'OQAM'                                   % OQAM-OFDM
    TxZiReal=zeros( TxFilterLength-1, nChannels );    % Initial conditions are zero for
    TxZiImag=TxZiReal;                                % transmit filters.
    for n=0:2*Delay      % For enough frames to flush filters (2*Delay+1)...
      TxBits=round( rand( 1, BitsPerFrame ) ); % Generate random bit sequence.
      % Convert bits to symbols.
      TxSymbols=feval( Modulation, 'Bits2Symbols', TxBits );
      TxSignal=OQAMTransmit( TxSymbols ); % Form OQAM-OFDM signal frame.
    end                  % end for
end                      % end switch
% Accumulate total energy.
Energy=0;                % Initialize accumulator.
for n=0:nPowerFrames-1                          % For desired number of frames...
  TxBits=round( rand( 1, BitsPerFrame ) );      % Generate random bit sequence.
  TxSymbols=feval( Modulation, 'Bits2Symbols', TxBits );% Convert bits to symbols.
  TxSignal=feval( [OfdmType 'Transmit'], TxSymbols );  % Form OFDM signal frame.
  Energy=Energy+TxSignal*TxSignal';             % Accumulate energy.
end                      % end for
Power=Energy/(nPowerFrames*Ms);                 % Calculate power.
```

A.4.6 Channel Response and Noise

A.4.6.1 Channel.m

```
%     Name:     Channel.m
%     Purpose:  Simulates the effects of channel distortion and noise on the transmitted
%               signal.
%     Description:The transmitted OFDM signal is first passed through a channel filter
%           to simulate the effect of the channel frequency response.  Subsequently a
%           vector of random Gaussian noise samples is added, with variance chosen
%           according to the desired value of Signal to Noise Ratio.
%     Input:    SignalIn:     The sequence of samples of the transmitted signal.
%     Output:   SignalOut:    Channel distorted signal + noise (AWGN).
function SignalOut=Channel( SignalIn );
```

```
global ChannelFilter;    % Coefficients of frequency response of channel.
global ChannelZi;        % Matrix of initial conditions for channel filter.
global Ms;               % Total number of samples in each OFDM frame.
global Sigma;            % Standard deviation of AWGN noise samples appropriate to the
                         % chosen SNR.
% Start of Channel.m code.
% Filter signal to simulate the effect of the channel frequency response.
[ChannelSignal ChannelZi]=filter( ChannelFilter, 1, SignalIn, ChannelZi );
% Add noise (AWGN).
NoiseVector=(randn( 1, Ms )+j*randn( 1, Ms ))*Sigma;% Form complex Gaussian noise
%                                              % vector having variance Sigma^2
                                               % and zero mean.
SignalOut=ChannelSignal+NoiseVector;           % Add Gaussian noise (AWGN).
```

A.4.7 Carrier Frequency Offset

A.4.7.1 FrequencyOffset.m

```
%    Name:        FrequencyOffset.m
%    Purpose:     Simulates the effect of a receiver Carrier Frequency Offset (CFO) on the
%                 reception of an OFDM frame.
%    Description: Adds a steadily increasing phase offset to each input sample to simulate
%                 the effect of a constant Carrier Frequency Offset(CFO).
%    Input:       SignalIn:  Sequence of transmitted samples for the current frame.
%    Output:      SignalOut: Sequence of received samples with simulated CFO.
% Start of FrequencyOffset.m code.
function SignalOut=FrequencyOffset( SignalIn )
global DeltaF;           % Receiver Carrier Frequency Offset (CFO) normalized relative to
                         % the subchannel spacing.
global Ms;               % Total number of samples in each OFDM frame.
global FrameCount;       % Number of OFDM transmitted and received so far.
% Add phase offset to each sample proportional both to the total number of samples so
% far and to the desired CFO.
SignalOut=SignalIn.*exp( j*2*pi*DeltaF*(FrameCount*Ms:FrameCount*Ms+Ms-1)
                         .../Ms )
```

VITA

Surname: Burfoot | Given Names: Stephen David

Place of Birth: London, England

Educational Institutions Attended:

University of Victoria	1995 to 2003
University of Sheffield	1971 to 1974

Degrees Awarded:

B. Eng. (Honours)	University of Sheffield	1974

Publications:

S. Catreux, S. Burfoot, P.F. Driessen, R.L. Kirlin, "Propagation measurements using array signal processing," *Communications, Computers and Signal Processing, 1997. 10 Years PACRIM 1987-1997 - Networking the Pacific Rim. 1997 IEEE Pacific Rim Conference on* , Volume: 1 , 1997 Page(s): 379 -382 vol.1