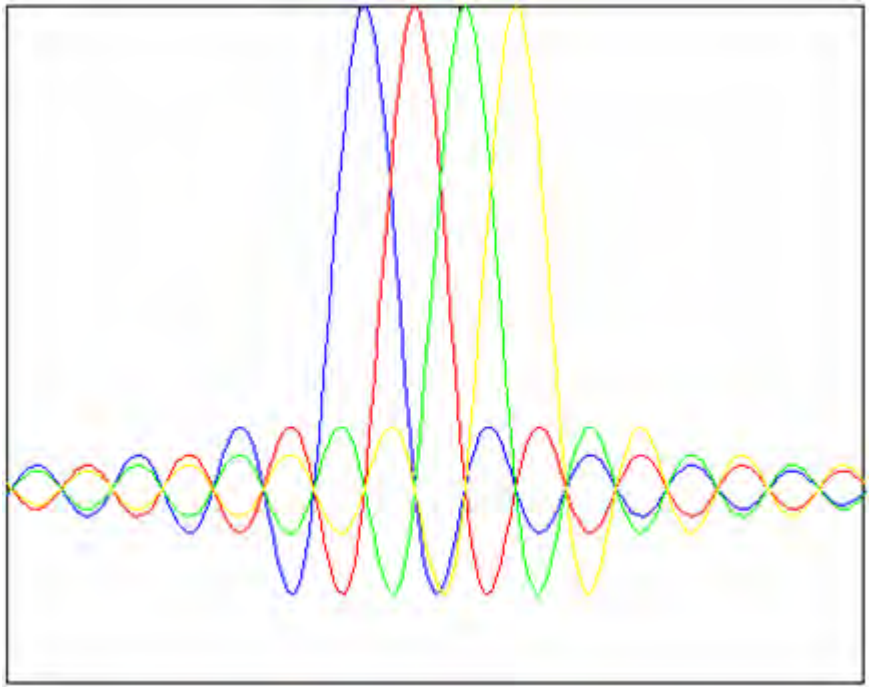
Orthogonal Frequency Division Multiplexing for 4G Mobile Communication Systems

BALTAGIANNI, Evagelia



Advisor
Dr. Costas Chaikalis

This dissertation is submitted in partial fulfilment of the requirements of Staffordshire University for the award of M.Sc. in Computer Science

January 2010

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Abstract

Evolution in mobile communications demands development of ideal mechanisms in order to approach a new generation (4th Generation). Such mechanism is Orthogonal Frequency Division Multiplexing (OFDM), which is a multi-carrier modulation technique for achieving the high data rate and spectral efficiency requirements for wireless communication systems of the near future. In the present thesis, an OFDM system is enhanced by adding an advanced error correction technique widely used in the communications industry, Turbo coding. A comparison of graphics of BER/SNR with and without OFDM and with and without Turbo coding is made. Turbo coding uses log-Map as decoding algorithm. Also BPSK modulation is used. The graphics that come up as results show that the use of Turbo coding and OFDM make the system produce a lower BER.

Keywords: OFDM, ISI, Turbo coding, BPSK, BER, SNR, log-Map

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Acknowledgements

I would like to express my thanks to my supervisor, Dr. Costas Chaikalis for his scholarly advice. My family supports me during my effort. Without them I couldn’t succeed.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Lists of contents

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Chapter 3 – Turbo codes

Chapter 4 – Simulation Results and Analysis

Chapter 5 –Conclusions and Future work

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Lists of figures

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Lists of tables

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Chapter 1 – Introduction

1.1 Introduction

The 4G mobile communication systems are going to deal with problems that 3G systems weren't able to solve and also to provide a wide variety of new services, from high-quality voice to high-definition video to high-data-rate wireless channels. The term 4G is used broadly to include several types of broadband wireless access communication systems. One of the terms used to describe 4G is MAGIC—Mobile multimedia, Anytime anywhere, Global mobility support, Integrated wireless solution, and Customized personal service [1]. The aim is to reach communication ubiquity (every time, everywhere) and to provide users with a new set of services [2].

With 4G infrastructures we can have access information anywhere, anytime, with a seamless connection to a wide range of information and services, and also we can receive a large volume of information, data, pictures, video, etc, with premium quality and high security. 4G will offer all types of services at an affordable cost. The key concept is integrating the 4G capabilities with all of the existing mobile technologies through advanced technologies [1].

With the advent of 4th Generation, the need for high speed data transmission has increased. Orthogonal Frequency Division Multiplexing is especially suitable for high speed communication due to its resistance to Intersymbol Interference (ISI), caused by multipath channels. ISI occurs when a transmission interferes with itself and the receiver cannot decode the transmission correctly [3]. OFDM is a special case of multi-carrier modulation. Multi-carrier modulation is the concept of splitting a signal into a number of signals, modulating each of these new signals to several frequency channels, and combining the data received on the multiple channels at the receiver [4].

1.2 Project background

OFDM is a frequency-division multiplexing technique, which is used to transmit large amounts of data on a radio signal. Basically, a 'big' radio signal is subdivided into

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

smaller signals and then transmitted to the receiver using different frequencies. The three most important requirements of 4G mobile networks: higher coverage and capacity, with desired QoS at minimum cost, are expected to be fulfilled with OFDM. [5].

In OFDM, the multiple frequency channels, known as sub-carriers, are orthogonal to each other. The use of orthogonal subcarriers would allow the subcarriers spectra to overlap, thus increasing the spectral efficiency. As long as orthogonality is maintained, it is still possible to recover the individual subcarriers signals despite their overlapping spectrums [6].

OFDM can be implemented easily, is extremely robust to harsh wireless channel environments and is well-suited to selectively populate areas of spectrum to avoid interfering (ISI) with primary users. The advantages of this modulation are the reason for its increasing usage.

1.3 Aim(s) and objectives of research

The goal of this thesis will be the enhancement of OFDM simulation model in an already existed transceiver mobile communication simulation, which uses Turbo coding and BPSK modulation. Graphics of BER (Bit error rate) versus SNR (Signal to noise ratio) will come up as results of the enhancement and will be studied and evaluated. BER improvements are expected when OFDM with Turbo coding is used, compared to without OFDM and / or without Turbo coding. Random Interleaver, Log-Map as Turbo decoding Algorithm and code rate 1/2 or 1/3 is used.

1.4 Structure of dissertation

In this section, an overview of how the main body of dissertation is organised is given. Also the indicative content for each of the subsequent chapters is outlined. This thesis presents the usage of Orthogonal Frequency Division Multiplexing for 4G Mobile Communication Systems.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

In Chapter 2 discussion about the origins of OFDM development is made. The basic principle of OFDM is introduced. OFDM benefits and drawbacks are presented. Examples of OFDM use in current Systems are given.

In Chapter 3 Turbo codes are introduced. Iterative decoding, which is the ‘turbo’ principle is presented. Recursive systematic and non systematic convolutional codes, Turbo encoder end decoder, Interleaving, Turbo decoding algorithms such as MAP Algorithm are discussed.

In Chapter 4 Simulation Results are given and an Analysis on them is made. Graphics of BER (Bit error rate) versus SNR (Signal to noise ratio) are studied and evaluated. BER improvements are observed when OFDM with Turbo coding is used, compared to without OFDM and / or without Turbo coding. Random Interleaver, Log-Map as Turbo decoding Algorithm and code rate 1/2 or 1/3 is used.

Chapter 5 consists in conclusions of this thesis and a few suggestions, for a future work, are made in order to succeed much more BER improvements.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Chapter 2 – OFDM

2.1 Introduction to OFDM

The name ‘OFDM’ is derived from the fact that the digital data is sent using many carriers, each of a different frequency (Frequency Division Multiplexing) and these carriers are orthogonal to each other, hence Orthogonal Frequency Division Multiplexing [7].

Digital data can be transmitted with efficiency and reliability over a radio channel, even in multipath environments. In OFDM a wideband signal is converted into a large number of narrow bandwidth carriers [8]. These carriers form a block of spectrum placed side-by-side in the frequency domain. The frequency spacing and time synchronization of the carriers are chosen in such a way that the carriers are orthogonal, meaning that they do not cause interference with each other, although the carriers overlap with each other in the frequency domain [7].

When broadcasting across a specific bandwidth, OFDM subcarriers can be selectively disabled to prevent interfering with other users. This technique is known as non-contiguous orthogonal frequency multiplexing (NC-OFDM) [9].The spectrum of a four subcarrier OFDM system is shown in Figure 2.1. [10].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

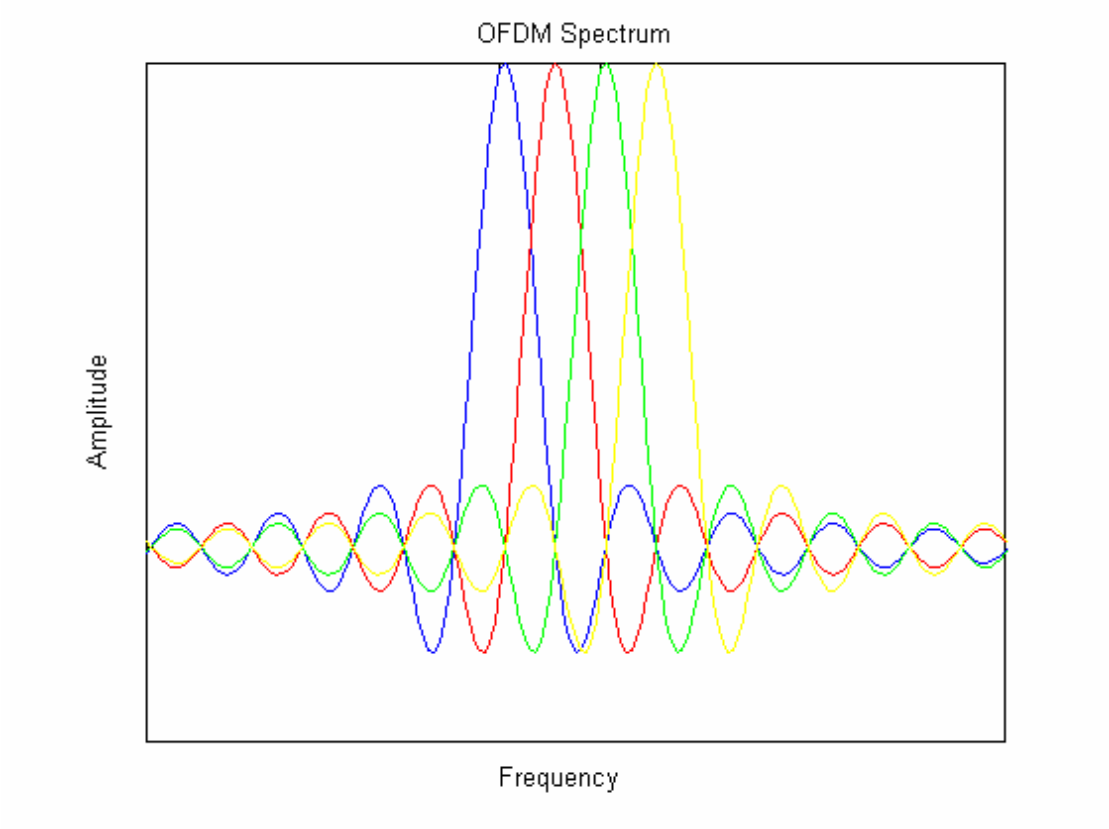


Figure 2.1. Four subcarrier OFDM spectrum [10]

2.2 Basic principle of OFDM

An OFDM signal is a sum of sub carriers that are individually modulated by using phase shift keying (PSK) or quadrature amplitude modulation (QAM) [7]. The basic principle of OFDM is to split a high-rate data stream into a number of lower rate streams that are transmitted simultaneously over a number of sub carriers as shown in Figure 2.2. The symbol duration increases for lower rate parallel sub carriers, so the relative amount of dispersion in time caused by multipath delay spread is decreased. Guard time in every OFDM symbol is introduced in order to eliminate completely Intersymbol Interference. This means that in the guard time, the OFDM symbol is cyclically extended to avoid intercarrier interference [11].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

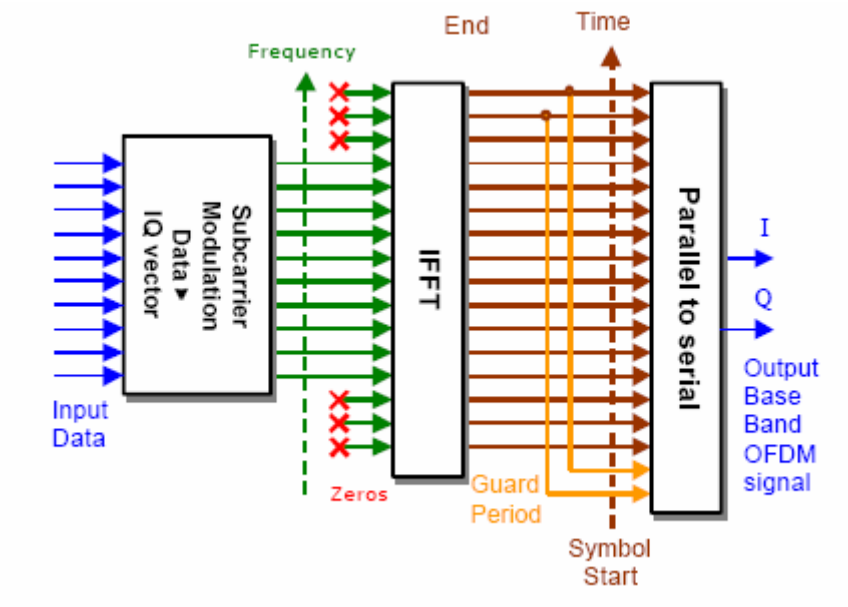


Figure 2.2. Generation of OFDM System [11]

OFDM can be thought of as a combination of multi-carrier modulation and frequency shift keying modulation (FSK) that converts a wideband signal into a series of independent narrowband signals placed side-by-side in the frequency domain [12]. Orthogonality amongst the carriers is achieved by separating the carriers by an integer multiple of the inverse of symbol duration of the parallel bit stream. The entire allocated channel is occupied through the aggregated sum of the narrow orthogonal sub-bands. In order for the carriers to not interfere with each other, the spectral peak of each carrier must coincide with the zero crossing of all the other carriers as depicted in Figure 2.3 [7]. A "zero-crossing" is a point where the sign of a function changes (e.g. from positive to negative) [13].

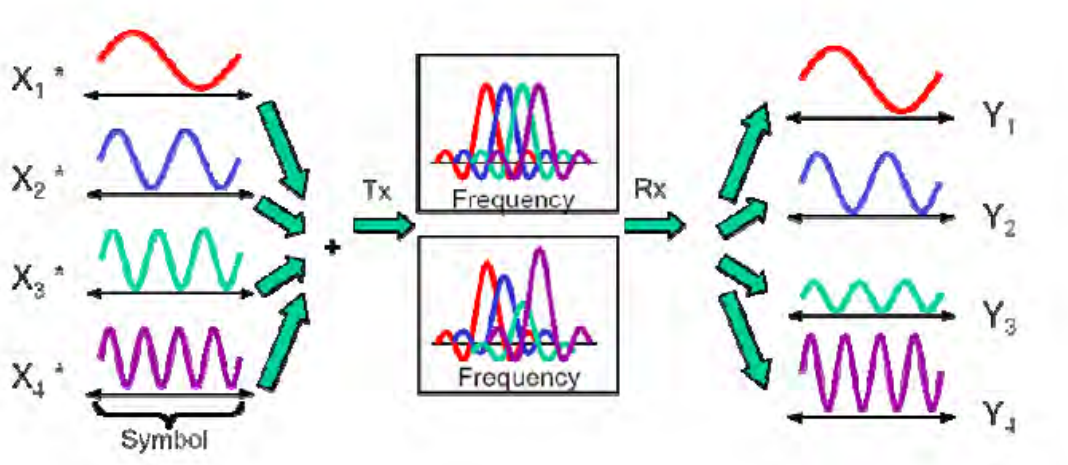


Figure 2.3. Overlapping orthogonal subcarriers viewed from the frequency domain [7]

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

2.3 OFDM history

Orthogonal Frequency Division Multiplexing (OFDM) is a form of multi-carrier transmission. Multi-carrier methods have been in use since 1957. The first multi-carrier radios used a bank of filters to separate the signals but Fast Fourier Transforms (FFTs) have been in use for this purpose since 1971 [14].

What is modulation? "Modulation refers to the process of transforming data to analog signals that are sent over the air. This is the technique by which information is wirelessly transmitted by varying one or more of the signal's basic characteristics - frequency, amplitude and/or phase." The choice of modulation techniques affects the cost, scalability, and reliability of wireless transmissions [3].

A single carrier system modulates information onto one carrier using frequency, phase, or amplitude adjustment of the carrier. For digital signals, the information is in the form of bits, or collections of bits called symbols, that are modulated onto the carrier. As the bandwidth used by a single carrier system increases, the susceptibility to interference from other continuous signal sources becomes greater [4].

Frequency division multiplexing (FDM) extends the concept of single carrier modulation by using multiple subcarriers within the same single channel. The total data rate to be sent in the channel is divided between the various subcarriers. FDM systems usually require a guard band between modulated subcarriers to prevent the spectrum of one subcarrier from interfering with another. If the FDM system had been able to use a set of subcarriers that were orthogonal to each other, a higher level of spectral efficiency could have been achieved. The use of orthogonal subcarriers would allow the subcarriers spectra to overlap, thus increasing the spectral efficiency [4].

2.3.1 FDM - OFDM

OFDM is similar to Frequency Division Multiplexing (FDM) in that it multiplexes carriers across frequency, but with two important differences.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

First, FDM is the traditional method to separate signals intended for different radios. When it is used to allow multiple users to share the same channel it is called frequency-division multiple access (FDMA). OFDM is often used for multiple access as well, but the primary motivation for using OFDM is to increase performance over using a single carrier modulation.

Secondly, OFDM differs from traditional FDM in its subcarrier spacing. In OFDM, the carriers overlap to a great degree. Each carrier is ideally represented mathematically by a sin(x)/x pulse, which have nulls at a spacing of 1/Ts where Ts is the symbol time of each subcarrier. In OFDM, the carrier spacing is 1/Ts, which is precisely the location of nulls in a sin(x)/x pulse and thus, ideally, there is zero inter-carrier interference. OFDM is more spectrally efficient than FDM. The spectrum and power spectral density of OFDM and FDM are depicted in Figure 2.4 [10].

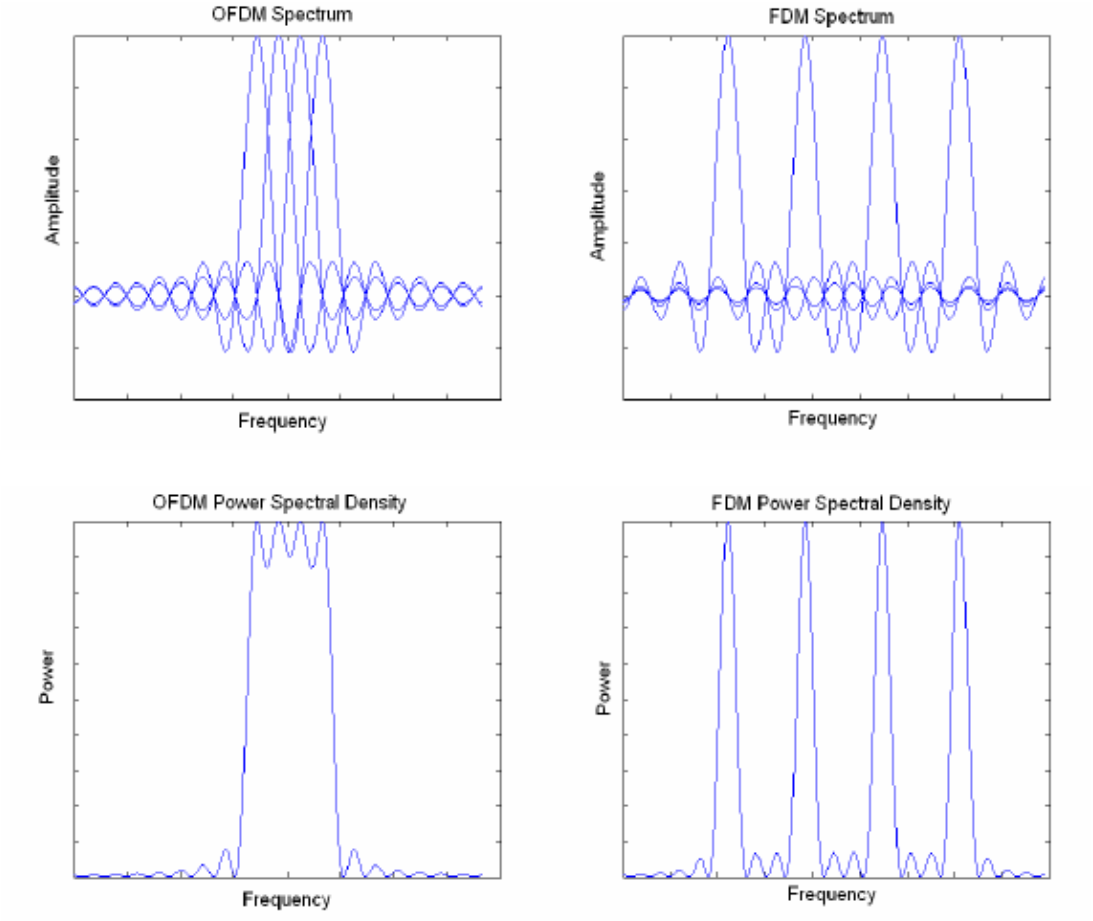


Figure 2.4. Spectrum and power spectral density of OFDM and FDM transmissions [10]

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

OFDM allows the spectrum of each tone to overlap. As the tones are orthogonal, they do not interfere with each other. The overall amount of spectrum required is reduced, by allowing the tones to overlap. OFDM enables user data to be modulated onto the tones. The information is modulated onto a tone by adjusting the tones phase, amplitude, or both [10].

OFDM can also be considered a multiple access technique, because an individual tone or groups of tones can be assigned to different users. Multiple users share a given bandwidth in this manner, yielding the system called OFDMA. OFDMA is a derivative radio-technology, which can be used for shared access. Each user can be assigned a predetermined number of tones when they have information to send, or alternatively, a user can be assigned a variable number of tones based on the amount of information that they have to send [10].

2.3.2 The origins of OFDM development

The origins of OFDM development started in the late 1950's [15] with the introduction of Frequency Division Multiplexing (FDM) for data communications. In 1966 Chang patented the structure of OFDM [15] and published the concept of using orthogonal overlapping multi-tone signals for data communications. OFDM was patented at Bell Labs in 1966. [14] In 1971 Weinstein [15] introduced the idea of using a Discrete Fourier Transform (DFT) for implementation of the generation and reception of OFDM signals, eliminating the requirement for banks of analogue sub carrier oscillators. This presented an opportunity for an easy implementation of OFDM, especially with the use of Fast Fourier Transforms (FFT), which are an efficient implementation of the DFT. This suggested that the easiest implementation of OFDM is with the use of Digital Signal Processing (DSP), which can implement FFT algorithms. It is only recently that the advances in integrated circuit technology have made the implementation of OFDM cost effective. The reliance on DSP prevented the wide spread use of OFDM during the early development of OFDM [7].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

2.4 Multipath Propagation

Wireless communications systems at the physical layer level must deal with a potentially hostile channel environment. Among these is additive white Gaussian noise (AWGN), multi-path propagation, where the radio signal arrives at the receiver via two or more paths [10].

2.4.1 Fading

Fading due to multi-path delay spread is due to different paths arriving at different times at the antenna. If all the significant paths (in terms of power) arrive within the time of a single symbol period, the net effect is a random complex attenuation of the symbol. This type of distortion is known as *flat fading.* Each path effectively acts as a tap in an FIR filter, which smears the signal in the time domain and filters it in the frequency domain. This is known *as frequency selective fading* [10].

The phenomenon of multiple signal paths arriving and interfering at the antenna is generally known as *small-scale fading* [16]. This is opposed to *large-scale fading*, which occurs when large obtrusive objects (or simply large distances between radios) drastically reduce received signal power [16]. Large scale fading is typically compensated for by varying the transmit power accordingly. Small-scale fading can be broken down into roughly two concepts: (a) fading due to Doppler spread, and (b) fading due to multi-path delay spread. The effect of the multipath channel can be fully characterized by Equation (1), where y is the received signal, x is the transmitted signal, and h is the channel transfer function [16].

$$y(t) = x(t) \otimes h(d,t) \quad (2.1)$$

The channel impulse response varies both as a function of delay, d, and time, t. Signal paths will arrive at the antenna at different delays and the magnitude and phase of these paths will change over time. Doppler spread refers to the dispersion in frequency caused by the motion of one or both radios while communicating with each other. The motion of the radios causes a Doppler shift, the degree of which can be characterized by a Doppler bandwidth. The greater the Doppler bandwidth, the greater the variations of h(d, t) with time [10].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

2.4.2 Inter Symbol Interference ISI

A common problem found in high speed communication is Inter Symbol Interference (ISI). If the significant paths arrive at time intervals greater than a symbol period, in addition to the random complex attenuation, there is *inter-symbol interference* (ISI) in the time domain. ISI occurs when a transmission interferes with itself and the receiver cannot decode the transmission correctly. For example, in a wireless communication system such as that shown in Figure 2.5, the same transmission is sent in all directions. Because the signal reflects from large objects such as mountains or buildings, the receiver sees more than one copy of the signal. Since the indirect paths take more time to travel to the receiver, the delayed copies of the signal interfere with the direct signal, causing ISI [17].

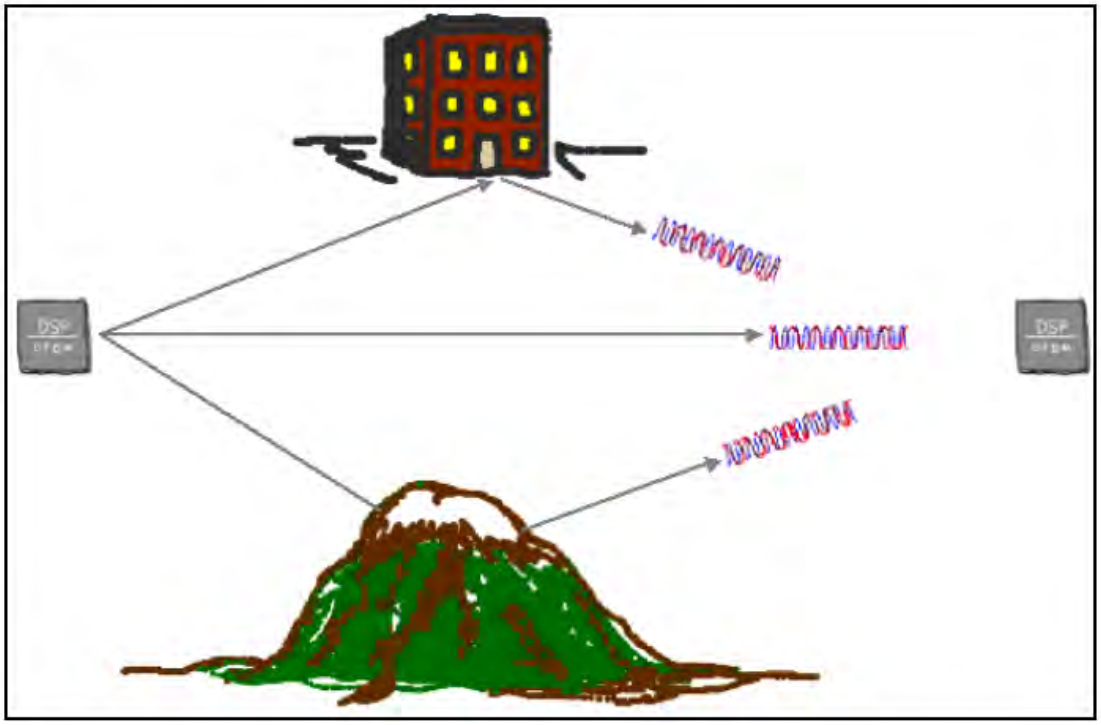


Figure 2.5. Multipath Demonstration [17]

OFDM is especially suitable for high speed communication due to its resistance to ISI. As communication systems increase their information transfer speed, the time for each transmission necessarily becomes shorter. Since the delay time caused by multipath remains constant, ISI becomes a limitation in high-data-rate communication [18]. OFDM avoids this problem by sending many low speed transmissions simultaneously.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

For example, Figure 2.6 shows two ways to transmit the same four pieces of binary data. Suppose that this transmission takes four seconds. Then, each piece of data in the left picture has a duration of one second. On the other hand, OFDM would send the four pieces simultaneously as shown on the right. In this case, each piece of data has a duration of four seconds. This longer duration leads to less problems with ISI [17] [19].

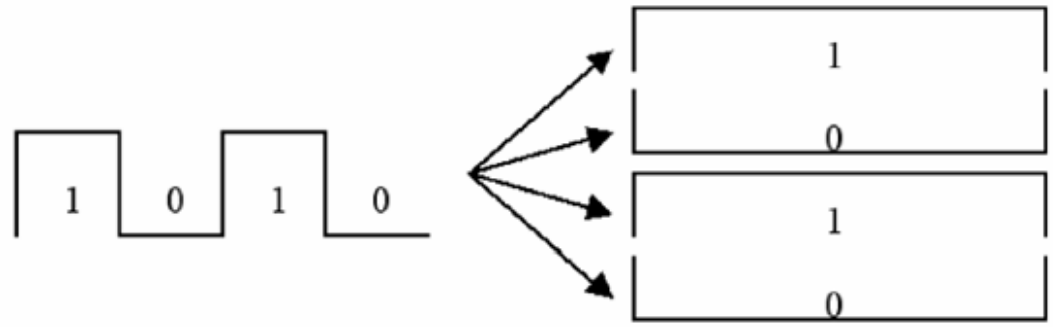


Figure 2.6. Traditional vs. OFDM Communication [17]

ISI and frequency-selective fading are the crucial bottlenecks for very high data-rate systems using single carrier modulation. An OFDM system can alleviate the ISI and frequency selective fading problem without the need for complex equalization [10].

2.5 Cyclic Prefix

In an OFDM system, ISI causes severe that is difficult to recover from [10]. By inserting a guard interval between OFDM symbols in the time domain that is designed to be longer than the maximum delay spread of the channel, elimination of the possibility of ISI is achieved. However, there is a trade off between the guard interval and the data throughput.

Cyclic prefix is the most common form of guard interval, where a certain number of samples at the end of the time-domain OFDM symbol are copied to the beginning. Cyclic prefix is used to combat the inter-symbol-interference and inter-channel-interference introduced by the multi-path channel through which the signal is propagated [20].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

The basic idea is to replicate part of the OFDM time-domain waveform from the back to the front to create a guard period. This is depicted in Figure 2.7. The duration of the guard period *Tg* should be longer than the worst-case delay spread of the target multi-path environment. At the receiver, certain position within the cyclic prefix is chosen as the sampling starting point, which satisfies the criteria [20]

$$\tau_{max} < Tx < Tg \quad (2.2)$$

where τ_{max} is the worst-case multi-path spread. As illustrated in the Figure 2.7, once the above condition is satisfied, there is no ISI since the previous symbol will only have effect over samples within [0, τ_{max}] . Sampling period starting from T_x will encompass the contribution from all the multi-path components so that all the samples experience the same channel and there is no ICI [20].

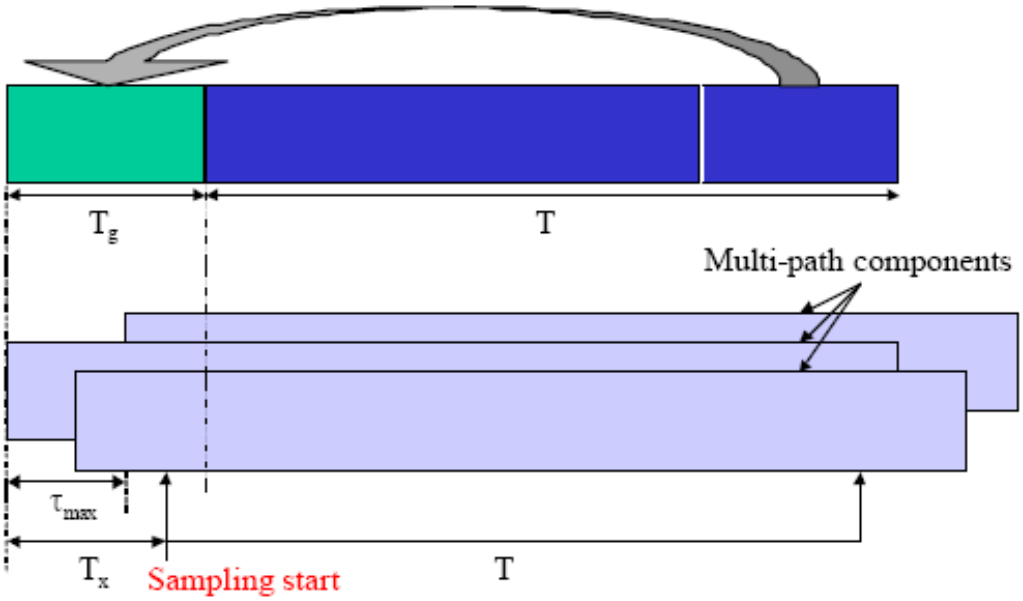


Figure 2.7. Cyclic prefix [20]

The cyclic prefix is used in a variety of ways in different OFDM system implementations, including timing synchronization, frequency synchronization, and carrier equalization [10].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

2.6 Signal representation

In an OFDM system, data is carried on narrow-band sub-carriers in frequency domain. Data is transformed into time-domain using IFFT at the transmitter and transformed back to frequency-domain using FFT at the receiver. The total number of sub-carriers translates into the number of points of the IFFT/FFT [20]. Suppose the data set to be transmitted is

$$U(-N/2), U(-N/2+1), \ldots, U(N/2-1) \qquad (2.3)$$

where N is the total number of sub-carriers. The discrete-time representation of the signal after IFFT is

$$u(n) = \frac{1}{\sqrt{N}} \sum_{k=-N/2}^{N/2-1} U(k) e^{j2\pi \frac{k}{N} n} \qquad (2.4)$$

where $n \in [-N/2, N/2)$. At the receiver side, the data is recovered by performing FFT on the received signal, i.e.

$$U(k) = \frac{1}{\sqrt{N}} \sum_{n=-N/2}^{N/2-1} u(n) e^{-j2\pi \frac{n}{N} k} \qquad (2.5)$$

where $k \in [-N/2, N/2)$. Most literature uses the continuous-time representation of the signal, i.e.

$$\frac{1}{\sqrt{N}} \sum_{k=-N/2}^{N/2-1} U(k) e^{j2\pi \frac{k}{T}(t-T/2)} \qquad (2.6)$$

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

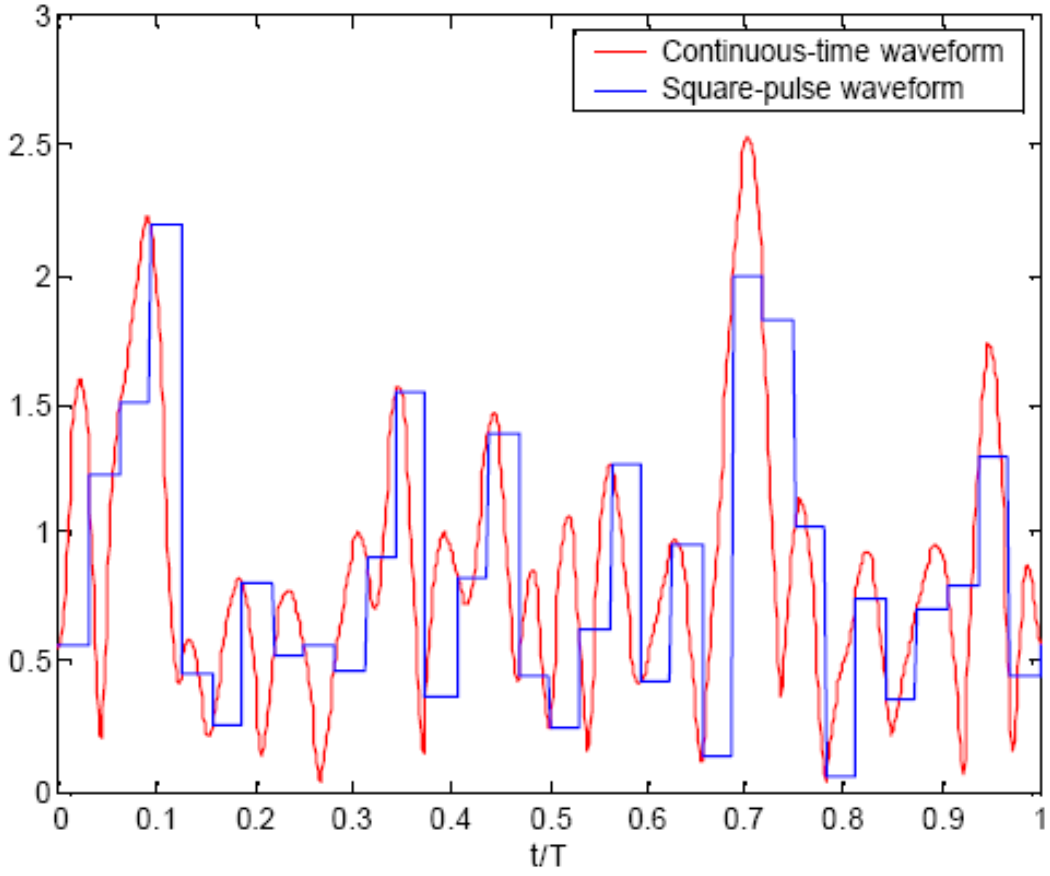


Figure 2.8. OFDM signal waveform [20]

where $t \in [0,T]$ and T is the symbol period. The kth datum $U(k)$ is carried on the kth narrowband carrier

$$e^{j2\pi\frac{k}{T}t} \qquad (2.7)$$

The samples of the continuous-time signal at

$$0, \frac{T}{N}, \frac{2T}{N}, \ldots, \frac{(N-1)T}{N} \qquad (2.8)$$

are the IFFT of the data set $U(k)$. In practice, however, square pulses of amplitude $u(n)$ and duration T / N are transmitted rather than the continuous multi-carrier signal as expressed above. Figure 2.8 shows the time domain waveform of a typical OFDM symbol [20].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

2.7 Synchronization

Synchronization is a big hurdle in OFDM. Synchronization usually consists of three parts:

1. Frame detection
2. Carrier frequency offset estimation and correction
3. Sampling error correction.

Frame detection is used to determine the symbol boundary so that correct samples for a symbol frame can be taken. Due to the carrier frequency difference of the transmitter and receiver, each signal sample at time *t* contains an unknown phase factor

$$e^{j2\pi\Delta f_c t} \quad (2.9)$$

where Δfc is the unknown carrier frequency offset. This unknown phase factor must be estimated and compensated for each sample before FFT at the receiver since otherwise the orthogonality between subcarriers is lost. For example, when the carrier is at 5GHz, an 100ppm crystal offset corresponding to a frequency offset of 50kHz. For a symbol period of $T = 3.2\ \mu s$, $\Delta fcT = 1.6$ [20].

Because the sampling clock difference between the transmitter and receiver, each signal sample is off from its correct sampling time by a small amount which is linearly increasing with the index of the sample. For example, for 100ppm crystal offset, it will be off by 1 sample after 10000 samples. If a symbol contains 100 samples, then within each symbol the maximum offset will be 1% of a sample [20].

Although this may cause the orthogonality degration between the sub-carriers, it can usually be ignored. If sampling error must be corrected, then interpolation filter must be used to construct the signal at correct sampling time [20].

2.8 Mathematical Representation

At baseband, an OFDM signal can be represented by a sum of modulated complex exponentials,

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

$$S(t) = \sum_{k=0}^{N-1} X_k \exp(j2\pi k\Delta f t) \quad (2.10)$$

where Xk is a complex number representing a BPSK, QPSK, or QAM baseband symbol modulating the k^{th} subcarrier and Δf is the subcarrier spacing. If this signal is sampled as in Equation (2.11),

$$S(nT_s) = \sum_{k=0}^{N-1} X_k \exp(j2\pi k\Delta f n T_s) \quad (2.11)$$

then the sampled signal is exactly equivalent to an inverse N-point *discrete Fourier transform* (DFT), taking the Xk as the frequency bin arguments [9]. The DFT and the inverse DFT are given in Equations (2.12) and (2.13).

$$X_l = \sum_{l=0}^{N-1} x_n \exp\left(\frac{-j2\pi nl}{N}\right) \quad l = 0,\ldots,N-1 \quad (2.12)$$

$$x_n = \frac{1}{N}\sum_{n=0}^{N-1} X_l \exp\left(\frac{j2\pi nl}{N}\right) \quad n = 0,\ldots,N-1 \quad (2.13)$$

The *Fast Fourier Transform* (FFT) is simply a computationally efficient implementation of the DFT. The IFFT and FFT are the core modulation and demodulation operations used in OFDM [10].

2.9 Where does OFDM find use?

The next generation wireless systems will be fully or partially OFDM-based. Implementations of OFDM rely on very high speed digital signal processing. Orthogonal Frequency Division Multiplexing (OFDM) transceivers are widely used in wireless applications including Digital Audio Broadcasting (DAB) systems, Digital Video Broadcasting (DVB) systems [8] and the terrestrial digital television systems

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

(DVB-T) [21]. OFDM is also now being used in Digital Subscriber Line (DSL) standards, ADSL and VDSL broadband access via telephone network copper wires.

Due to implementation complexity, OFDM applications have been scarce until recently with the advances in DSP technology. Standards such as HIPERLAN2 (High Performance Local Area Network) and IEEE 802.11a and IEEE 802.11g have emerged to support IP-based services [4] [8]. It has also been proposed for wireless broadband access standards such as IEEE Std. 802.16 (WiMAX) [22], American IEEE Std. 802.11 (WiFi) and its European equivalent (High Performance Local Area Network) HIPRLAN/2, 802.20 (proposed PHY) [14].

2.10 OFDM benefits

OFDM belongs to the most popular techniques that can cope with the key requirements of the future mobile communications. OFDM is easy to be implemented in the digital domain because of the use of DFT (Disctrete Fourier Transform). In addition, OFDM is a flexible modulation scheme and it can be made adaptive.

OFDM is insusceptible to most forms of impulse noise, since the transmission is parallel. OFDM has the capability of handling very strong echoes (multipath fading) thanks to the use of a cyclic prefix.

OFDM has high bandwidth efficiency, since the parallel subcarriers are overlapping, but as they are orthogonal to each other, they do not cause interference. With this way OFDM solves the problem of (ISI).

OFDM provides high flexibility in resource allocation, since it splits the broadband channel into a number of parallel subchannels. Thus various resources such as power and data rate can be dynamically allocated to different subchannels [21] [23].

The main benefit of OFDM is that the subcarriers in the frequency band can actually overlap one-another. The data to be transmitted is split into *n* parallel data streams, each of which modulates a subcarrier as shown in Figure 2.9 [8].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

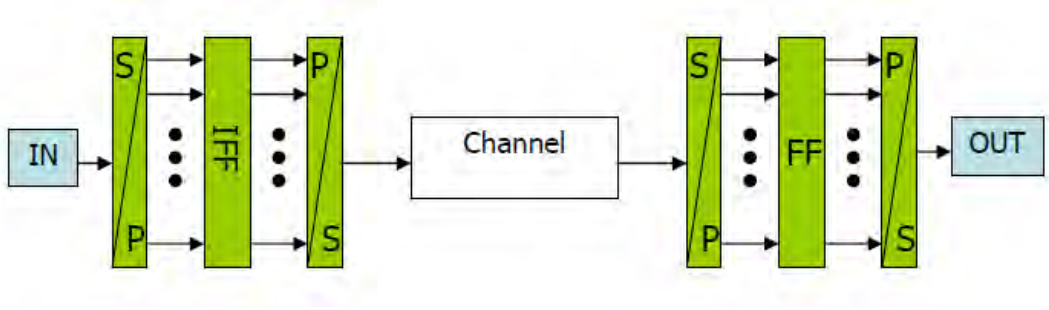


Figure 2.9. Abstract view of OFDM sender and receiver modulation [8]

OFDM communication systems naturally alleviate the problem of multipath propagation with its low data rate per subcarrier. The data rate per subcarrier is only a fraction of conventional single carrier systems having the same throughput. This is one of the biggest advantages of OFDM modulation [8].

By using multiple distinct subcarriers, a frequency selective fading channel can be transformed into multiple approximately flat-fading channels. This principle is best understood graphically by Figures 2.10 and 2.11. In each figure, the transmitted signal is filtered by the transfer function of channel, leading to distortion of the signal. Clearly, the distortion imparted to the OFDM signal is much less severe [10].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

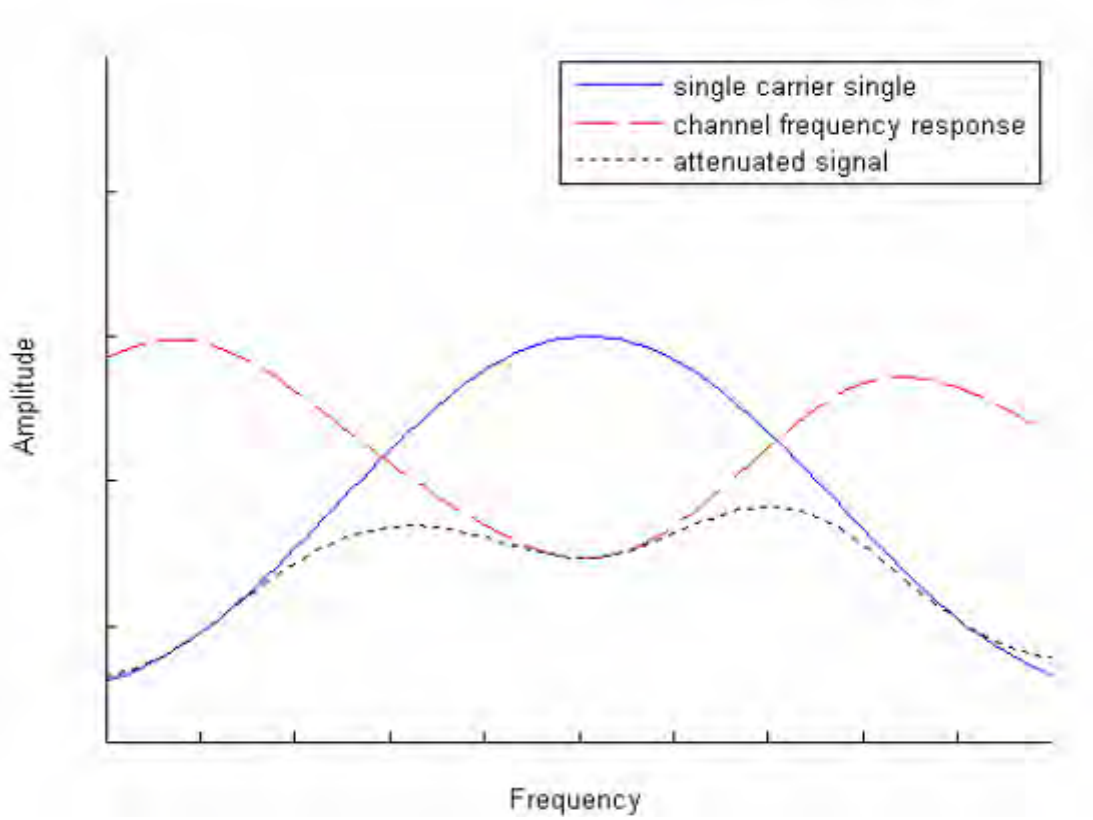


Figure 2.10. Single carrier signal undergoing frequency selective fading [10]

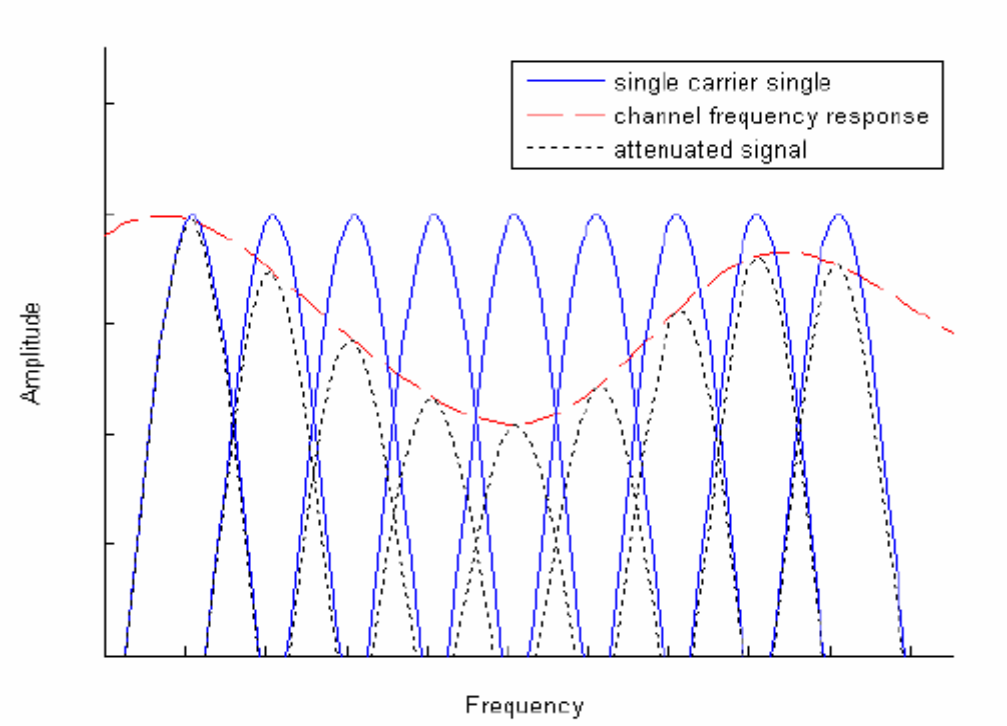


Figure 2.11. Approximately flat fading sub-channels in a frequency selective channel [10]

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Figure 2.10 demonstrates a situation where the channel frequency response due to multi-path is varying in frequency more quickly than the signal is. The bandwidth over which the magnitude response of the channel is basically flat is known as the *coherence bandwidth*. Obviously, the bandwidth of the signal in Figure 2.10 is greater than the coherence bandwidth of the channel. Conversely, in the OFDM signal in Figure 2.11, each subcarrier has a bandwidth smaller than the coherence bandwidth of the channel. If the bandwidth of each signal passing over the channel is smaller than the coherence bandwidth, the signal will undergo flat fading [10].

When designing an OFDM system, the individual subcarrier bandwidth is set to be significantly smaller than the coherence bandwidth. OFDM is essentially a solution to severe frequency-selective fading, which is one of the most significant challenges for single-carrier systems attempting to achieve higher data rates [10].

2.11 OFDM drawbacks

High Peak-to-Average Power Ratio (PAPR) has been recognized as one of the major practical problem involving OFDM modulation. High PAPR results from the nature of the modulation itself, where multiple sub-carriers/sinusoids are added together to form the signal to be transmitted [10].

Single carrier systems using BPSK, QPSK, or QAM have known envelope signals, and thus known output power levels as well. In contrast OFDM can exhibit a widely varying signal envelope, as it is a sum of modulated subcarriers. The maximum peak-to-average power ratio (PAPR) of an OFDM system is approximately equal to the number of subcarriers, N [24]. During the actual IFFT and FFT calculations, the output of the transform requires more bits to represent the sample than those of the input samples. If these output samples are truncated to the same number of bits as the input samples, there is a loss of precision which leads to a degradation in the signal-to-noise ratio [10].

The second more commonly referenced problem is that the RF amplifiers used to transmit the OFDM signal must either have really large dynamic range, or must be

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

operated with a large back-off, which will also lead to a degradation of the signal-to-noise ratio [10].

A number of techniques to achieve PAPR reduction at the expense of transmit signal power increase, bit error rate (BER) increase, data rate loss, computational complexity increase, have been proposed [10]. These techniques include amplitude clipping and filtering [25], coding [26], Active Constellation Extension (ACE) [27], and multiple signal representation techniques such as Selected Mapping (SLM) [28] etc.

2.12 Examples of OFDM use in current Systems

2.12.1 IEEE 802.16-2004

The following block diagrams describe the proposed structure of the OFDM transmitter and receiver. These designs serve as template which aides in the design of the individual modules. These diagrams are meant to be a generic framework for any packet-based OFDM system, in addition to IEEE 802.16-2004 [10].

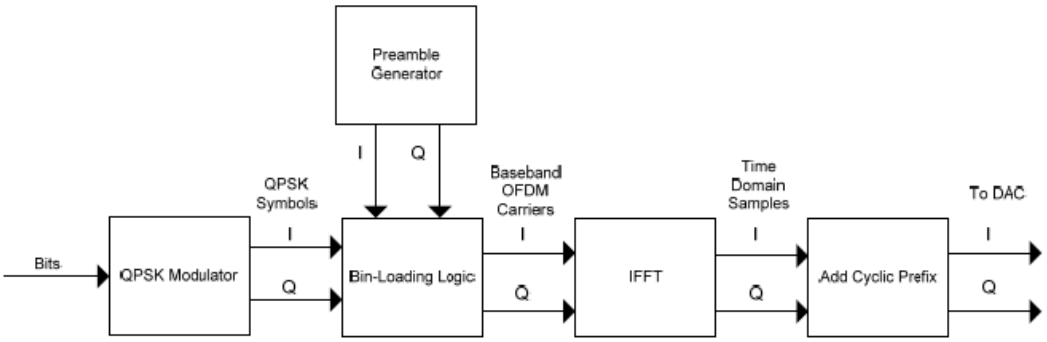


Figure 2.12. OFDM Transmitter Block Diagram [10]

The transmitter, shown in Figure 2.12, operates as follows. At the beginning of every frame, the preamble generator generates the two preamble symbols which the bin-loader maps directly to the IFFT. After the preamble, the QPSK modulator begins mapping every 2 bits to QPSK symbols. These symbols are passed to the bin-loading module that maps the QPSK symbols to data carriers, as well as producing the pilot carriers and guard carriers. These carriers are loaded into the IFFT, which converts the carriers into an equal number of time domain samples. The final step is to add the

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

cyclic prefix to form the OFDM symbol. The cyclic prefix is formed by taking the last 32 time domain samples of each OFDM symbol and copying them to the front of the symbol [10].

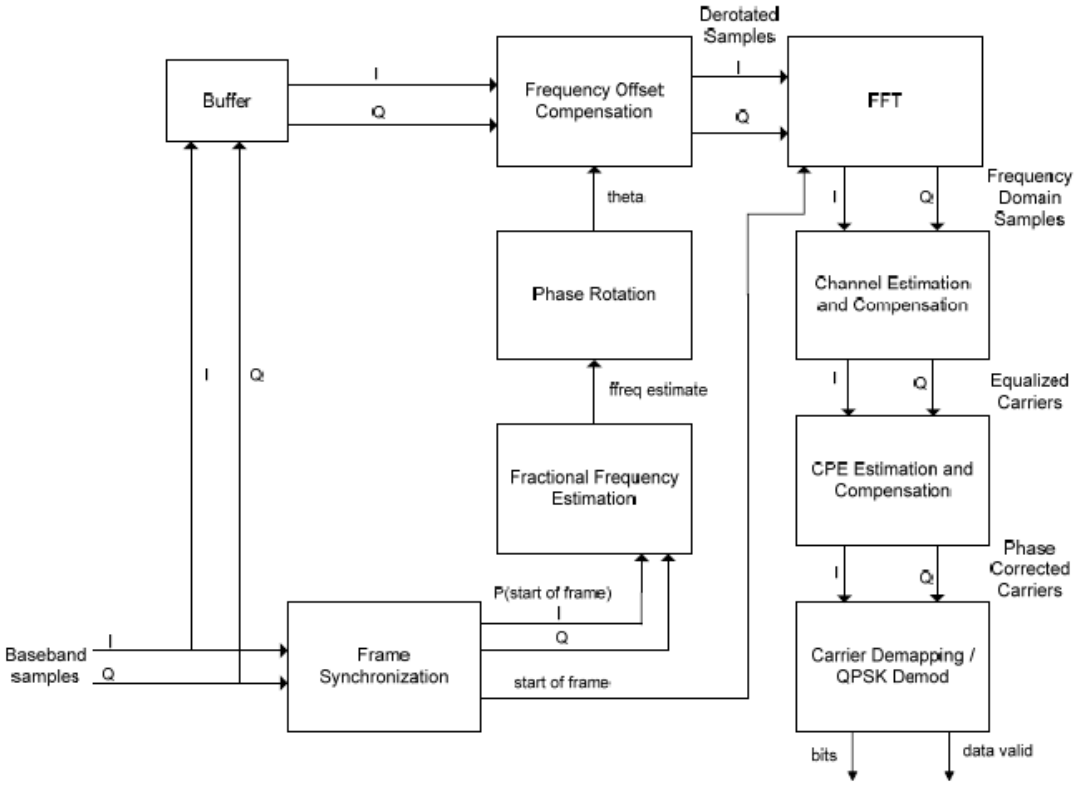


Figure 2.13. OFDM Receiver Block Diagram [10]

Figure 2.13 illustrates the functionality of the OFDM receiver. Time domain baseband samples are constantly being digitized by the ADCs and passed to the OFDM receiver. The frame synchronization module processes this data until it detects the beginning of a frame. Once it detects the beginning of a frame it asserts the 'start of frame' signal and passes the correlation metric 'P' to the fractional frequency estimation block. Using this correlation information, the fractional frequency estimation module calculates the frequency offset estimate and passes this to the phase rotation module. The phase rotation module uses the estimate to produce the proper angle in which to de-rotate the packet. A frequency offset causes an *evolving* phase rotation that changes every sample. The phase rotation block calculates this evolving phase rotation. The frequency offset compensation block uses this phase rotation calculation to de-rotate the time domain samples. The purpose of the buffer block is to store the time domain data while the previously mentioned blocks are

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

processing the data. This ensures that the frequency offset compensation will begin rotating the packet from the very beginning, instead of somewhere in the middle [10].

Once the time domain data has been compensated for carrier frequency offsets, it is passed to the FFT block, which converts it to frequency domain subcarriers. The FFT module includes a counter and begins to increment when the start of frame signal is asserted, so it will know when to begin processing the first OFDM symbol. The cyclic prefix does not need to be explicitly removed since the FFT can be turned on and off at regular intervals to 'skip over' the cyclic prefix. The channel estimationand compensation block performs two functions. First, it uses the second preamble and internally stored data to calculate the carrier equalizer taps. Since each carrier is undergoing flat fading, at worst, only one tap per carrier is required. After this is completed, it uses these taps to equalize all the data symbols in the packet [10].

Once the carriers are equalized, they are then passed to the CPE estimation and compensation module. This module uses the pilot carriers in each data symbol to estimate the rotation of the carriers due to phase noise and residual frequency offset, and then de-rotates the carriers appropriately. Finally, these carriers are passed to the carrier de-mapping / QPSK demodulator which filters out all the non-data carriers and demodulates the data carriers into a bit stream and asserts a 'data valid' flag, which serves as a write enable to any logic reading the bits out of the system [10].

2.12.2 IEEE 802.11a

Each sub-carrier in an OFDM system is modulated in amplitude and phase by the data bits. Depending on the kind of modulation technique used one or more bits are used to modulate each sub-carrier. Modulation techniques typically used are BPSK, QPSK, 16QAM, 64QAM etc. The process of combining different sub-carriers to form a composite time-domain signal is achieved using Fast Fourier transform. Different coding schemes like block coding, convolutional coding or both are used to achieve better performance in low SNR conditions. Interleaving is done which involves assigning adjacent data bits to non-adjacent bits to avoid burst errors under highly selective fading. Block diagram of an OFDM transceiver is shown in Figure 2.14 [30].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

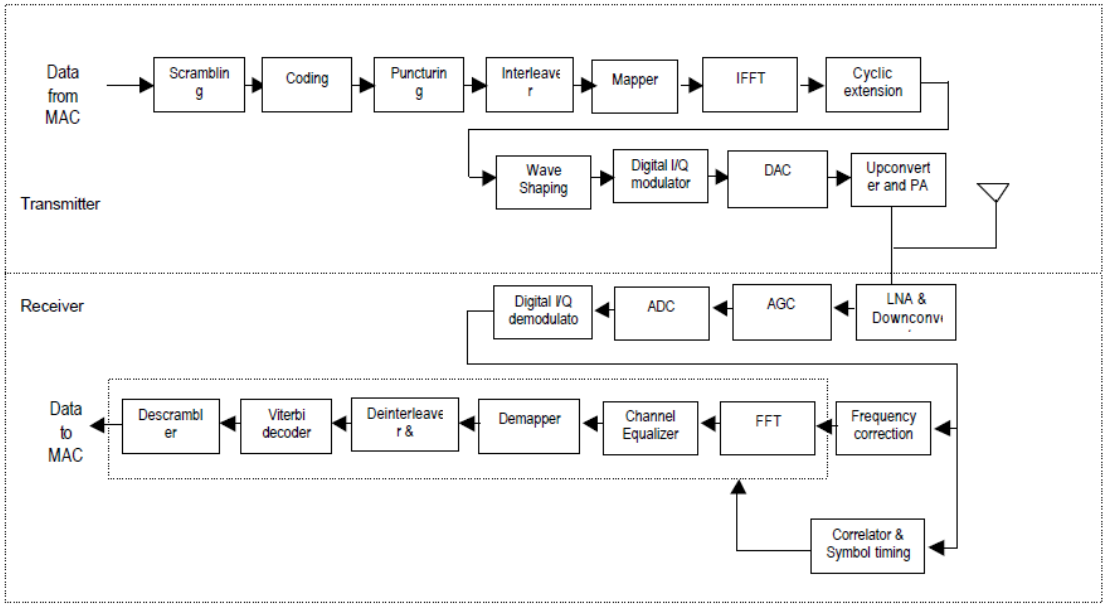


Figure 2.14. Block diagram of the 802.11a OFDM transceiver [30]

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Chapter 3 – Turbo codes

3.1 Introduction to Turbo codes

Turbo codes were first presented at the International Conference on Communications in 1993. Until then, it was widely believed that to achieve near Shannon's bound performance (the theoretical maximum information transfer rate of the channel "0.7 dB"), a decoder with infinite complexity or close implementation was needed [30]. Shannon show in principle how to achieve capacity. The incoming data should be split into blocks containing as many bits as possible. Each possible data block is then mapped to another block of *n* code symbols, called a *codeword*, which is transmitted over the channel.

The set of codewords, and their mapping to data blocks, is called a *code*, or more specifically a *forward error correcting* (FEC) code. A decoder, which is at the receiver, must find the codeword that most closely resembles the word it receives, including the effects of noise and interference on the channel. The power of the code to correct errors and overcome noise and interference depends on the degree of resemblance, since the decoder is more likely to confuse codewords that resemble one another more closely. This is characterised in terms of the minimum number of places in which any two codewords differ, called the *Hamming distance* [31].

Turbo coding is the most recent (early 1990s) development in error correction that combines two or more relatively simple convolutional codes and an interleaver to produce a block code that can perform within a fraction of a decibel of the Shannon limit [13]. Turbo coding is an advanced error correction technique widely used in the communications industry. The basis of turbo coding is to introduce redundancy in the data to be transmitted through a channel. The redundant data helps to recover original data from the received data. In data transmission, turbo coding helps achieve near Shannon limit performance [36].

Parallel-concatenated codes, as they are also known, can be implemented by using either block codes (PCBC) or convolutional codes (PCCC) [30].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

3.2 Convolutional codes

Block codes are the older and simpler codes, where a group (block) of bits is processed as a whole in order to generate a new (longer) coded block for transmission. In order to give extra information to the decoder about the initial frame sent, these codes are using check bits ("parity bits") attached to end of the signal. The performance of the code is relatively poor, due to the matter that the information sequence is segmented into blocks that are decoded independently to form a succession of codewords, so there is no feedback about the previous data frames (blocks) sent. This problem is solved with the introduction of the convolutional codes.

Convolutional codes differ from block codes in the sense that they do not break the message stream into fixed-size blocks [32]. Convolutional codes work on bit or symbol streams of arbitrary length. If desired, a convolutional code can be turned into a block code.

3.2.1 Recursive systematic and non systematic convolutional codes

The recursive systematic convolutional (RSC) encoder is obtained from the nonrecursive nonsystematic (conventional) convolutional encoder by feeding back one of its encoded outputs to its input [33].

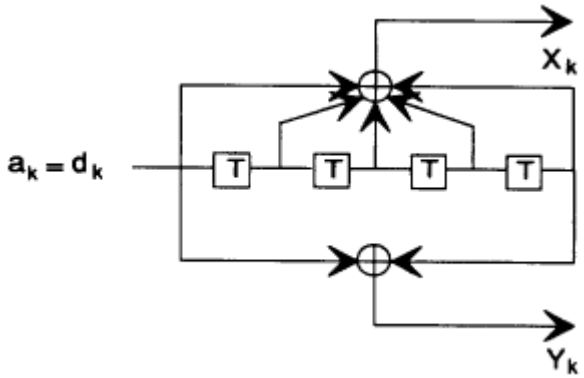


Figure 3.1. Nonrecursive Convolutional Encoder [33]

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

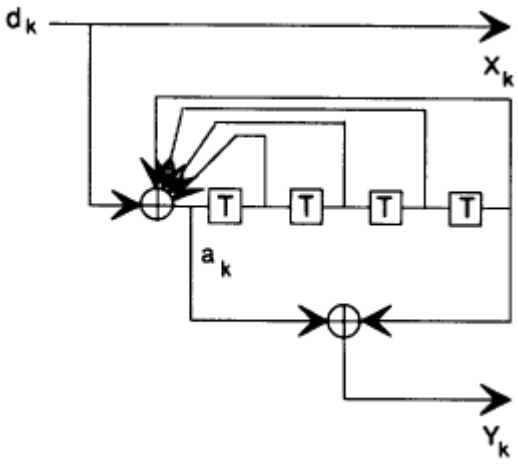


Figure 3.2. Recursive Convolutional Encoder [34]

Consider a binary rate R= 1/2 convolutional encoder with constraint length K and memory M=K-1. The input to the encoder at time k is a bit d_k and the corresponding codeword C_k is the binary couple (X_k, Y_k) with

$$X_k = \sum_{i=0}^{K-1} g_{1i} d_{k-i} \quad mod.2 \quad g_{1i} = 0,1 \tag{3.1}$$

$$Y_k = \sum_{i=0}^{K-1} g_{2i} d_{k-i} \quad mod.2 \quad g_{2i} = 0,1 \tag{3.2}$$

where G1 : $\{g_{1i}\}$, G2: $\{g_{2i}\}$ are the two encoder generators, generally expressed in octal form.

The BER of a classical Non Systematic Convolutional (NSC) code is lower than that of a classical Systematic code with the same memory M at large SNR. At low SNR, it is general the other way round [34].

A binary rate R=1/2 RSC code is obtained from a NSC code by using a feedback loop and setting one of the two outputs X_k or Y_k equal to the input bit d_k. For an RSC code, the shift register (memory) input is no longer the bit d_k but is a new binary variable a_k. If X_k= d_k (respectively Y_k= a_k), the output Y_k (respectively X_k) is equal to equation (3.2) (respectively 3.1) by substituting a_k for d_k and the variable a_k is recursively calculated as

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

$$a_k = d_k + \sum_{i=1}^{K-1} \gamma_i a_{k-i} \qquad mod.2 \qquad (3.3)$$

Where γ_i is respectively equal to g_{1i} if $X_k = d_k$ and to g_{2i} if $Y_k = a_k$
Equation 3.3 can be rewritten as

$$d_k = \sum_{i=0}^{K-1} \gamma_i a_{k-i} \qquad mod.2. \qquad (3.4)$$

Generally, we assume that the input bit d_k takes values 0 or 1 with the same probability. From equation (3.3), we can show that variable a_k exhibits the same statistical property

$$P_r\{a_k = 0 / a_1 = \varepsilon_1, \ldots a_{k-1} = \varepsilon_{k-1}\} = P_r\{d_k = \varepsilon\} = 1/2 \qquad (3.5)$$

With ε is equal to

$$\varepsilon = \sum_{l=1}^{K-1} \gamma_i \varepsilon_i \qquad mod.2 \qquad \varepsilon = 0,1. \qquad (3.6)$$

Thus the trellis structure is identical for the RSC code and the NSC code and these two codes have the same free distance d_f. However, the two output sequences $\{X_k\}$ and $\{Y_k\}$do not correspond to the same input sequence $\{d_k\}$ for RSC and NSC codes. This is the main difference between the two codes.

When punctured code is considered, some output bits Xk or Yk are deleted according to a chosen puncturing pattern defined by a matrix P [34]. For instance, starting from a rate R=1/2 code, the matrix P of rate 2/3 punctured code is

$$P = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \qquad (3.7)$$

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

3.3 Puncturing - Code Rate

Puncturing, in coding theory, is the process of removing some of the parity bits after encoding with an error-correction code. This has the same effect as encoding with an error-correction code with a higher rate, or less redundancy. Puncturing considerably increases the flexibility of the system without significantly increasing its complexity, because the same decoder can be used regardless of how many bits have been punctured. In some cases, a pre-defined pattern of puncturing is used in an encoder and the inverse operation, known as depuncturing, is implemented by the decoder. Puncturing is used in UMTS, Wifi, GPRS, EDGE, as well as in the DVB-T and DRM Standards [13].

A convolutional code is a type of error-correcting code which is described by three integers, n, k and K [4]. Error detection and correction are techniques to ensure that data is transmitted without errors, even across unreliable media or networks [13]. It is defined as *n* the number of output bits, *k* is the number of input bits and K is the number of memory registers. The quantity *k/n* is called the code rate. It is a measure of the efficiency of the code [35].

In the case that the code rate is *k/n*, for every *k* bits of useful information, the coder generates totally *n* bits of data, which *n-k* of them are redundant. As it has for block codes thus and for convolutional codes, it has the same code significance (information per coded bit). However, *n* does not define a block or codeword length as it does for codes. Since a convolutional encoder produces two output bits for every input bit, its rate is 1/2.

The number of memory registers *K* is a parameter known as the constraint length. *K* represents the number of *k* – tuple stages in the encoding shift register. A different and important characteristic of convolutional codes, contrary to block codes, is that the encoder has memory, the *n* – tuple sent out by the convolutional encoding procedure. This is not only a function of an input *k* – tuple. It is also a function of the previous *K* – 1 input *k* – tuples. In practice, *K* is transform in order to control the capacity and complexity of the code and *n* and *k* are small integers [36].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

3.4 Turbo encoder

Turbo encoders and decoders are key elements in today's communication systems to achieve the best possible data reception with the fewest possible errors [37]. Using recursive systematic convolutional codes (RSC), a turbo encoding scheme can be constructed using two of them. The two recursive systematic convolutional encoders receive the same data, but the second encoder receives the data after being permuted by an inner interleaver [38].

A turbo code is formed from the parallel concatenation of two codes separated by an interleaver [32]. From the two encoders, which compose the mechanism, only one of them is used. This is on account of the systematic output from the other component encoder is just a permuted version of the chosen systematic output [39]. The two encoders do not have to be identical [40]. Since the inner interleaver has a fixed structure and it works on a frame (or block) basis, turbo codes can be considered as block codes. The purpose of this interleaver is to 'cut' the burst errors such that after correction of the errors from the first decoder, the remaining errors become correctable error patterns in the second decoder [38]. Figure 3.3 shows a generic turbo code encoder.

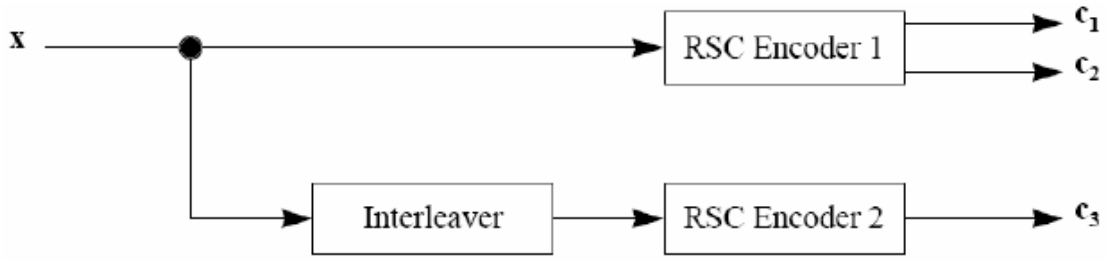


Figure 3.3. Turbo encoding scheme [38]

Figure 3.4 shows the setup of a turbo encoder. This encoder transforms the user bits *uk* into the output bits *cn* that are transmitted. It consists of a parallel concatenation of relatively simple convolutional component codes (ENC1 and ENC2) separated by a pseudo random interleaver of size *N* [41].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

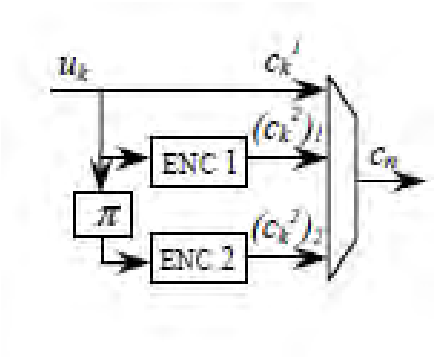


Figure 3.4. Turbo encoder [41]

Due to this interleaver, the turbo code operates as a block code. At the beginning of the block, the encoder starts in a known state. In order to force the encoder in a known state at the end of the block, typically some extra tail bits are added [42]. Traditionally, this is only done for the first component encoder ENC1.

Some more tail bits after the interleaver are added, to also have a known end state for the second encoder. These extra bits are encoded only by ENC2 and then sent through the channel. Another option to force both encoders in a known end state is to use precursor bits as in [43]. Both schemes reduce the throughput only very slightly, but improve the decoding performance and allow an interesting decoder module optimization [41].

3.5 Turbo decoder [32]

Although, a pure maximum likelihood decoder for a turbo code is too complex to implement, an increasingly good approximation is obtained with the iterative decoding scheme presented in Figure 3.5. Each decoder module (DEC1 and DEC2) performs a half iteration. It generates a correction term (called extrinsic information λ_k^{ext}), which the next module uses as a priori information (denoted as intrinsic information λ_k^{int}) after appropriate (de)interleaving [41].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

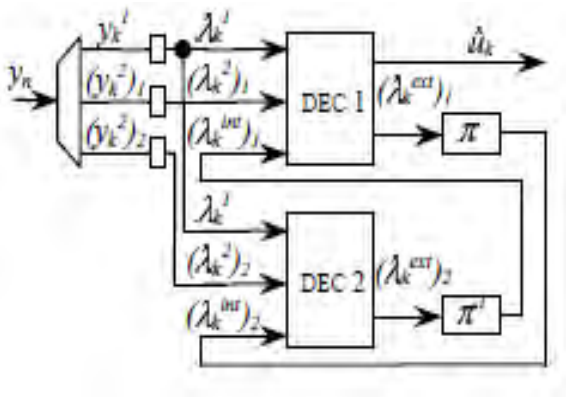


Figure 3.5. Turbo decoder [41]

The decoder operates iteratively, and in the first iteration the first component decoder produces a soft output as its estimate of the data bits. The soft output from the first encoder is then used as additional information for the second decoder, which uses this information along with the channel outputs to calculate its estimate of the data bits. The second iteration can begin, and the first decoder decodes the channel outputs again, but now with additional information about the value of the input bits provided by the output of the second decoder in the first iteration. This cycle is repeated and with every iteration, the Bit Error Rate (BER) of the decoded bits tends to fall. However the improvement in performance obtained with increasing numbers of iterations decreases as the number of iterations increases [44].

3.6 Interleaving

If a decoding error occurs in a codeword, it usually results in a number of data errors. When these are passed on to the next decoder they may overwhelm the ability of that code to correct the errors. The performance of the outer decoder might be improved if these errors were distributed between a number of separate codewords. This can be achieved using an / de-interleaver. Interleaver is a device that rearranges the position of bits in a stream (increase the distance between the errors) [31]. The size of the interleaver determines the length of the codeword.

There are several types of interleavers such as,

a) Block interleaver,

b) Row – Column interleaver,

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

c) Odd-even block interleaver,

d) Pseudo-random interleaver,

e) Convolutional interleaver,

f) Helical interleaver,

g) Uniform interleaver,

h) Cyclic shift interleaver and

i) Code matched interleaver.

The simplest type of interleaver (known as a *rectangular* or *block interleaver*) is illustrated in Figure 3.6.

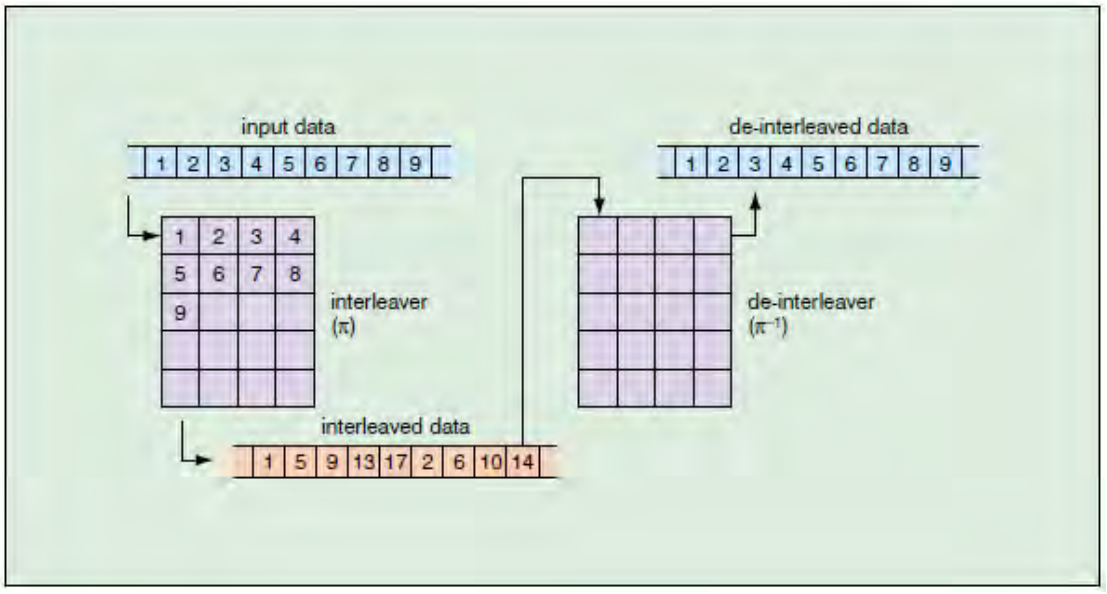


Figure 3.6. Operation of interleaver and de-interleaver [31]

It consists of a two-dimensional array, into which the data is read along its rows. Once the array is full, the data is read out by columns, thus permuting the order of the data. (Because it performs a permutation, an interleaver is commonly denoted by the Greek letter π, and its corresponding de-interleaver by π^{-1}.) The original order can then be restored by a corresponding de-interleaver: an array of the same dimensions in which the data is read in by columns and read out by rows [31].

The type of interleaver used in a turbo encoder can seriously hinder the performance of the constructed turbo code [45].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

3.7 Iterative decoding: the ‘turbo’ principle

The conventional decoding technique for array codes is that shown in Figure 3.7: the inner code is decoded first, then the outer.

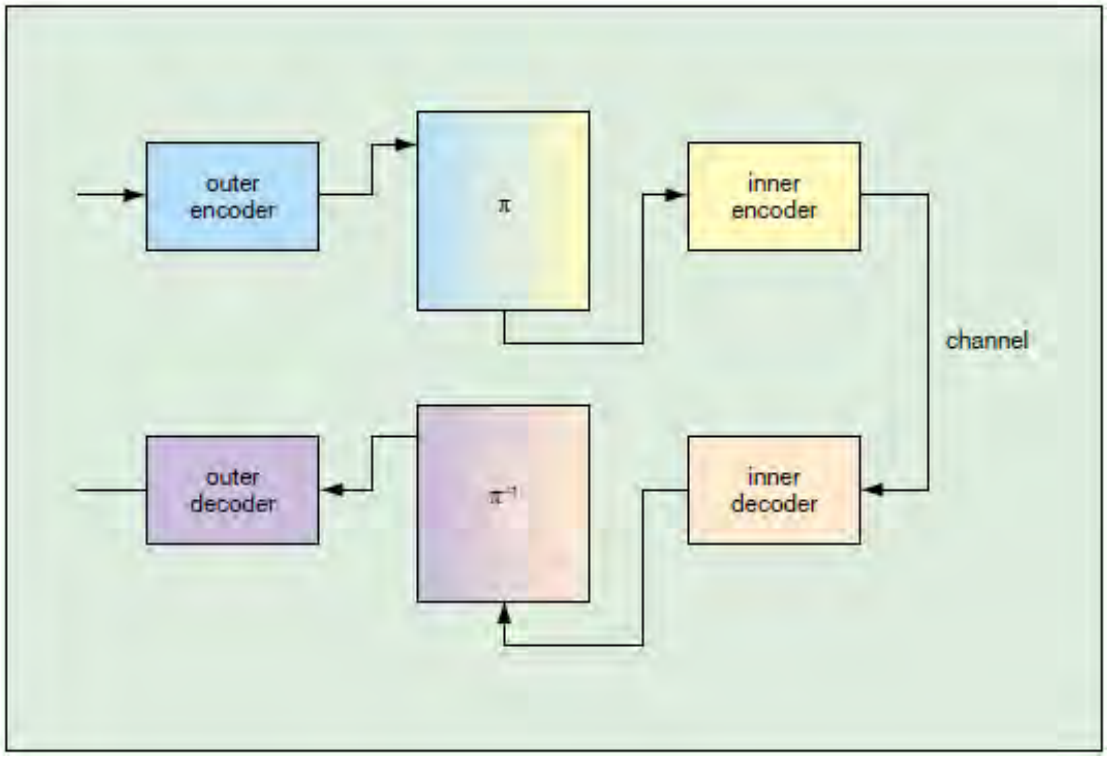


Figure 3.7. Concatenated encoder and decoder with interleaver [31]

If the output of the outer decoder were reapplied to the inner decoder it would detect that some errors remained, since the columns would not be codewords of the inner code. (A codeword of a single error correcting code must contain either no errors or at least three.) This in fact is the basis of the iterative decoder: to reapply the decoded word not just to the inner code, but also to the outer, and repeat as many times as necessary.

One further ingredient is required for the iterative decoder. That ingredient is *soft-in, soft-out* (SISO) decoding. If the received signal is close to a decision threshold (say between 0 and 1) in the demodulator, then that decision has low reliability, and the decoder should be able to change it when searching for the most probable codeword. Making use of this information in a conventional decoder, called *soft decision decoding*, leads to a performance improvement of around 2dB in most cases [31].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Soft information usually takes the form of a *log-likelihood ratio* for each data bit. The likelihood ratio is the ratio of the probability that a given bit is ‘1’ to the probability that it is ‘0’. If we take the logarithm of this, then its sign corresponds to the most probable hard decision on the bit (if it is positive, ‘1’ is most likely; if negative, then ‘0’). The absolute magnitude is a measure of our certainty about this decision [31].

The log-likelihood ratio exactly mirrors Shannon’s quantitative measure of information content, in which the information content of a symbol is measured by the logarithm of its probability. Thus we can regard the log-likelihood ratio as a measure of the total information we have about a particular bit. In fact this information comes from several separate sources. Some comes from the received data bit itself: this is known as the *intrinsic information*.

Information is also extracted by the two decoders from the other received bits of the row and the column codeword. When decoding one of these codes, the information from the other code is regarded as *extrinsic information*. It is this information that needs to be passed between decoders, since the intrinsic information is already available to the next decoder and to pass it on would only dilute the extrinsic information [31].

Hence the iterative decoder has the structure shown in Figure 3.8, in which the intrinsic information has been separated from the extrinsic, so that the output of each decoder contains only extrinsic information to pass on to the next decoder. After the outer code has been decoded for the first time both the extrinsic information and the received data are passed back to the first decoder, reinterleaved back to the appropriate order for this decoder, and the whole process iterated again. It is this feedback that has given rise to the term ‘turbo-code’, since the original inventors likened the process to a turbo-charged engine, in which part of the power at the output is fed back to the input to boost the performance of the whole system. Thus the term ‘turbo’ should really be applied to the decoder structure rather than the codes themselves [31].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

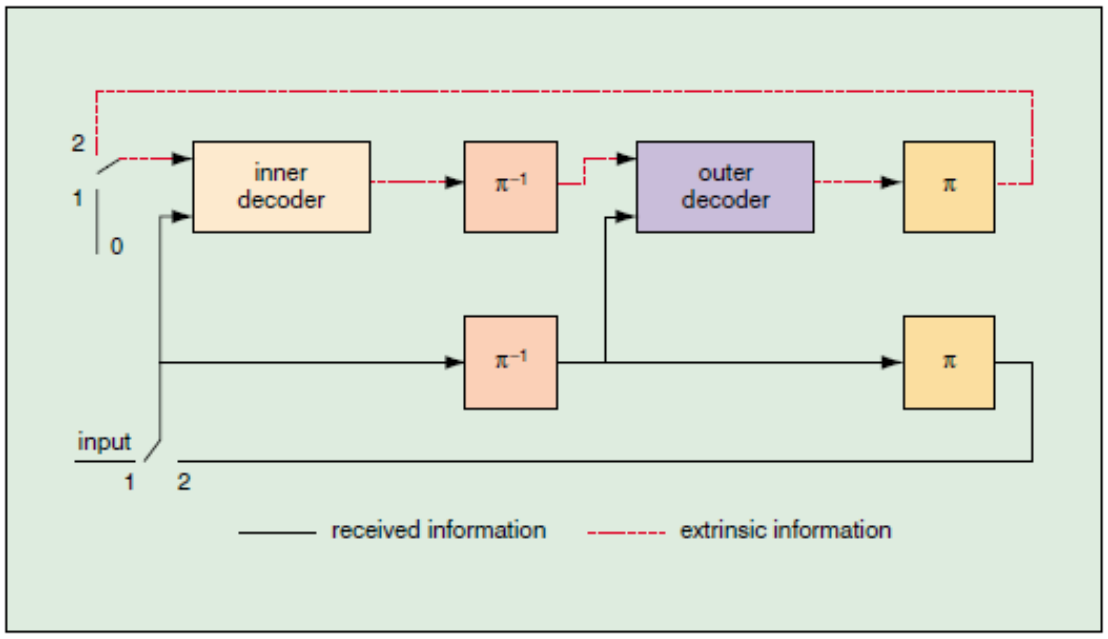


Figure 3.8. Iterative decoder. The switches are in position 1 for the first iteration and in position 2 for subsequent iterations [31]

3.8 Turbo decoding algorithms

The two main types of decoder are Maximum A Posteriori (MAP) and the Soft Output Viterbi Algorithm (SOVA). MAP looks for the most likely symbol received, SOVA looks for the most likely sequence. Both MAP and SOVA perform similarly at high Eb/No. At low Eb/No MAP has a distinct advantage, gained at the cost of added complexity.

3.8.1 MAP Algorithm

Turbo decoding is based on the principle of comparing the probability of a received soft input data being a ‘1’ and ‘0’. Turbo decoder uses a decoding scheme called the MAP (Maximum A Posteriori Probability) algorithm [37].

MAP was first proposed by Bahl et al and was selected by Berrou et al as the optimal decoder for turbo codes. MAP looks for the most probable value for each received bit by calculating the conditional probability of the transition from the previous bit, given the probability of the received bit. The algorithm determines the probability of whether each received data symbol is a ‘1’ as well as a ‘0’. This is done with the help

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

of the data, parity symbols, and the decoder knowledge of the encoder trellis. A trellis is a form of a state transition table, of the encoder input/output.

Based on the data and parity information, the MAP decoder computes the probability of the encoder being in a particular state. Depending on the soft data, parity value and the weight from the previous state, the probability that the data is a '1' or '0' can be computed. The MAP decoder computes the weight for each data symbol in a given block for both the forward and reverse directions. This results in the computation of forward and reverse metrics. Using these two values the probabilities are computed. After the probabilities are determined they are compared and a decision is made. The Turbo Decoder IP core uses the logarithm of the probability to reduce computation; this is known as Log Likelihood Ratio (LLR). The computation of the probabilities is done iteratively to obtain a reliable result. Once the result is considered reliable, one can make a final decision as to whether the data symbol is a '1' or '0'. Turbo decoder can implement both the Log-Map and Max-log-Map algorithm. The Log-Map algorithm gives a slightly better performance than the Max-Log- Map but utilizes more resources and runs at a slower frequency [37].

The Log Likelihood Ratio is the probability that the received data bit is a '0' divided by the probability that the received data bit is a '1' [37].

$$L(D) = \log \frac{P(D=0)}{P(D=1)} \quad (3.8)$$

Thus, taking the logarithm we will have a positive value if P(D=1) > P(D=0), and negative value for the opposite. A positive value means the data value is a '1', otherwise a '0'. For one complete cycle of iteration, one needs to compute the LLR using parity for non-interleaved as well as interleaved data [37].

3.8.2 Viterbi algorithm (VA)

The Viterbi algorithm is a dynamic programming algorithm for finding the most likely sequence of hidden states – called the Viterbi path – that results in a sequence of observed events [46].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

The Viterbi algorithm was conceived by Andrew Viterbi in 1967 as an error-correction scheme for noisy digital communication links, finding universal application in decoding the convolutional codes used in both CDMA and GSM digital cellular, dial-up modems, satellite, deep-space communications, and 802.11 wireless LANs. It is now also commonly used in speech recognition, keyword spotting, computational linguistics, and bioinformatics. For example, in speech-to-text (speech recognition), the acoustic signal is treated as the observed sequence of events, and a string of text is considered to be the "hidden cause" of the acoustic signal. The Viterbi algorithm finds the most likely string of text given the acoustic signal [46].

3.8.2.1 SOVA

The SOVA is a modification of the Viterbi algorithm (VA) in order to compute a symbol-by-symbol soft-output values which give a reliability information [47].

The SOVA is a maximum-likelihood (ML) sequence optimization with a symbol – by – symbol soft – outputs. But in the SOVA, the ML path is considered by both of survivor and competition paths. The SOVA decoders evaluate the a-priori value of information sequences u, L (u) and weighted signal, $L_c y$. The a-priori value is obtained from previous decoder. If there is no previous a-priori value then L (u) is set to zero. The output from SOVA decoder is estimated value (u') and extrinsic value, L (u') that will be iterated to the other SOVA decoder. The structure of SOVA decoder is depicted on Figure 3.9 [48].

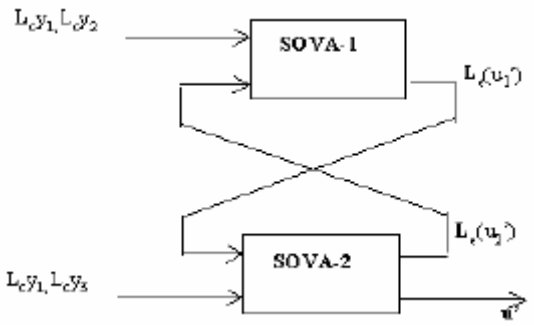


Figure 3.9. Structure of iterative SOVA decoder [48]

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

The main advantage of the SOVA is its sequence optimization which can deliver a sequence solution belonging to the codebook. The main disadvantages of the SOVA symbol-by-symbol softoutputs are its non-optimality neither in the ML sequence optimization sense nor in the symbol optimization sense, and it's no easy possibility to be identified to an exact mathematical expression [47].

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Chapter 4 – Simulation Results and Analysis

4.1 Simulation Model Analysis

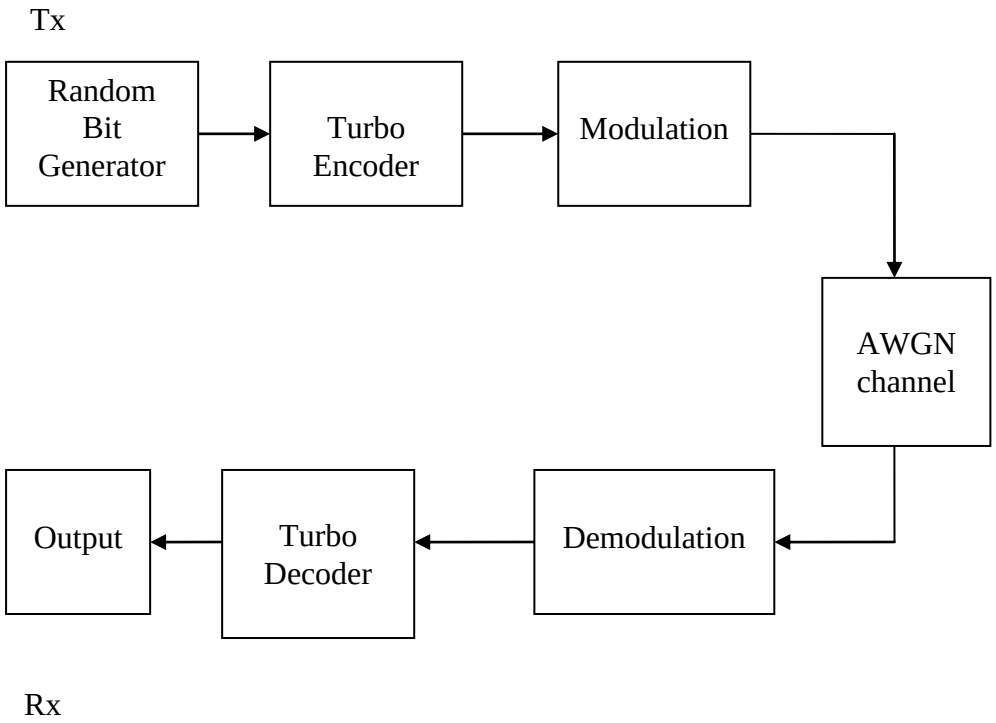


Figure 4.1. Simulation Model Analysis

Random Bit Generator: Frame size (information bits + tail bits) = 1500 bits

Number of iterations for each frame = 5

Encoder: Interleaving = Random interleaver

Code rate = 1/2 or 1/3

Constraint length (:Number of input bits) k = free distance - memory

free distance d : The minimal Hamming distance between different encoding sequences.

Constraint length K = m + 1

Modulation: OFDM, BPSK simulation using a carrier cosine wave with ISI. Create modulated BPSK signal. First expand the bit stream. The modulation is repeateted 6 times for our 6 channels in OFDM. The IFFT of each of these signals is taken. A conversion from serial to parallel is made.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

AWGN channel: Fading amplitude a = 1

Demodulation: OFDM, BPSK. First multiply received bitstream by cosine wave with carrier frequency. Decision is made with a threshold of zero.

Turbo Decoder: Turbo decoding Algorithm = Log-Map
Iterations = 5

Output: The final result of the simulation

Matlab was used in order to get simulation results.
For the results that are presented in the following sections, the code in APPENDIX was used. The code in APPENDIX can be found in [49] [50], which has been modified in order to get the simulation results for the needs of this thesis.
In comparison to already published results to reference [30], similar behavior of the system systems is observed.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.2 Simulation Results – Without OFDM – Without Turbo

Table 4.1. Simulation Results – SNR (db) vs BER –

Without OFDM – Without Turbo – Frame size = 1500 bits

SNR(db)	BER
0.00	8.400000e+000
0.25	7.000000e+000
0.50	6.966667e+000
0.75	6.800000e+000
1.00	5.700000e+000
1.25	4.933333e+000
1.50	4.866667e+000
1.75	4.433333e+000
2.00	4.433333e+000
2.25	3.700000e+000
2.50	3.633333e+000
2.75	2.600000e+000
3.00	2.300000e+000
3.25	2.166667e+000
3.50	1.766667e+000
3.75	1.466667e+000
4.00	1.466667e+000
4.25	1.066667e+000
4.50	8.666667e-001
4.75	7.666667e-001
5.00	7.000000e-001
5.25	6.000000e-001
5.50	4.666667e-001
5.75	4.666667e-001
6.00	2.333333e-001
6.25	2.000000e-001
6.50	1.333333e-001
6.75	1.000000e-001
7.00	1.000000e-001
7.25	1.000000e-001
7.50	1.000000e-001

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

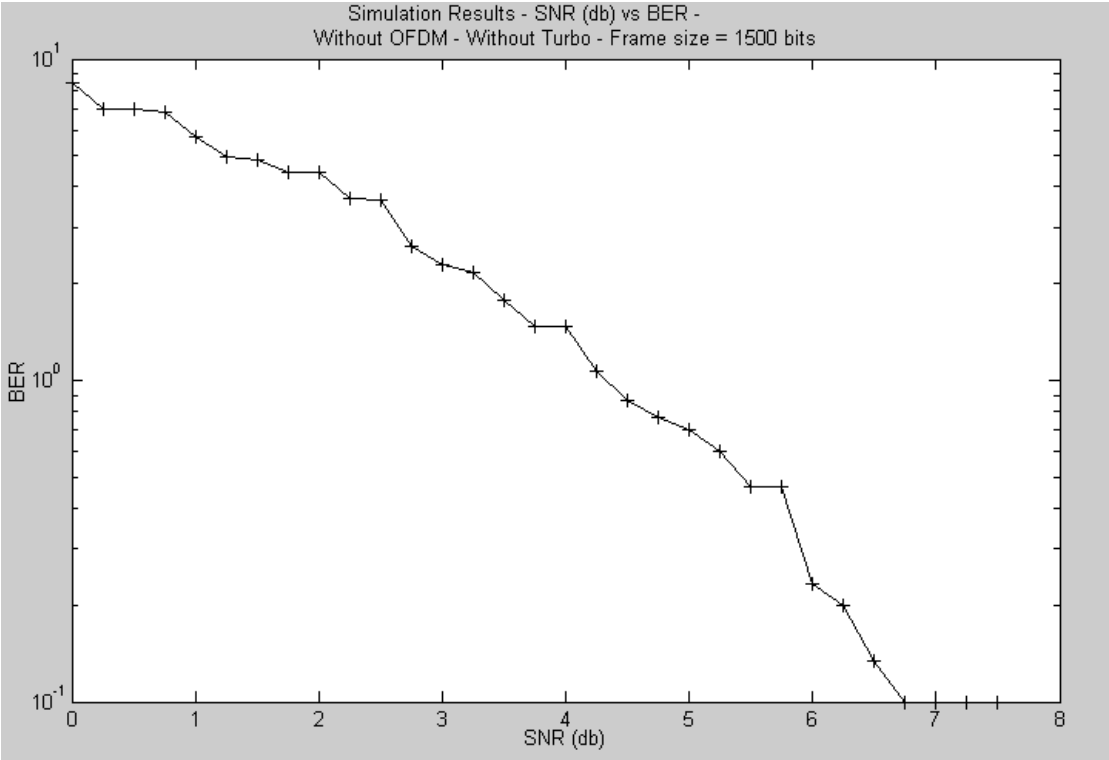


Figure 4.2. Simulation Results – SNR (db) vs BER – Without OFDM – Without Turbo – Frame size = 1500 bits

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.3 Simulation Results – Without OFDM – With Turbo – Code rate: 1/2

Table 4.2. Simulation Results – SNR (db) vs BER – Without OFDM – With Turbo – Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/2 – Frame size = 1500 bits

SNR	BER
0.00	6.766667e+000
0.25	6.516667e+000
0.50	6
0.75	5.077778e+000
1.00	4.872222e+000
1.25	4.322222e+000
1.50	4.072222e+000
1.75	3.583333e+000
2.00	3.094444e+000
2.25	2.694444e+000
2.50	2.305556e+000
2.75	2.094444e+000
3.00	1.888889e+000
3.25	1.450000e+000
3.50	1.338889e+000
3.75	1.183333e+000
4.00	1.055556e+000
4.25	8.666667e-001
4.50	7.388889e-001
4.75	6.444444e-001
5.00	4.833333e-001
5.25	3.944444e-001
5.50	3.000000e-001
5.75	2.666667e-001
6.00	2.000000e-001
6.25	1.388889e-001
6.50	9.444444e-002
6.75	7.222222e-002
7.00	6.666667e-002
7.25	6.111111e-002
7.50	2.222222e-002

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

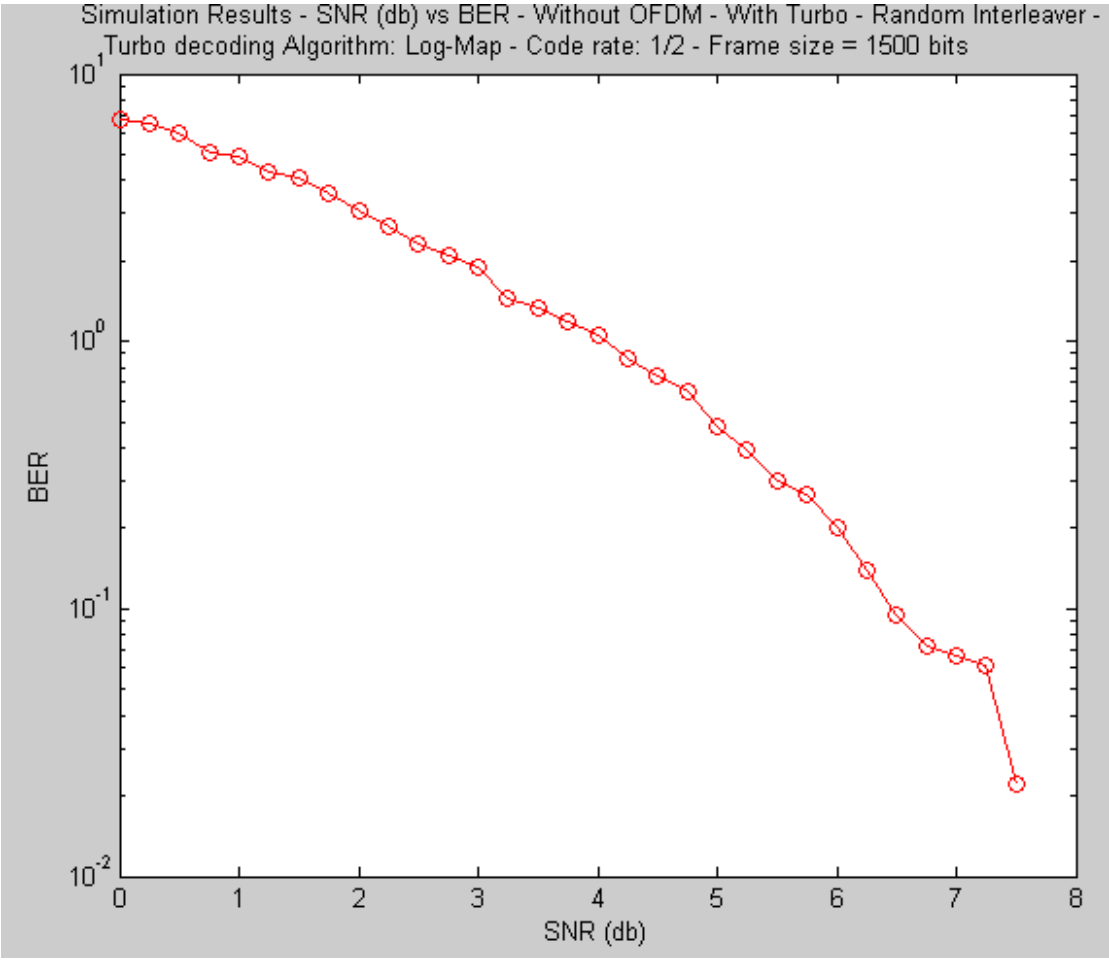


Figure 4.3. Simulation Results – SNR (db) vs BER – Without OFDM – With Turbo – Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/2 – Frame size = 1500 bits

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.4 Simulation Results – Without OFDM – With Turbo – Code rate: 1/3

Table 4.3. Simulation Results – SNR (db) vs BER – Without OFDM – With Turbo – Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/3 – Frame size = 1500 bits

SNR	BER
0.00	5.577778e+000
0.25	5.211111e+000
0.50	4.716667e+000
0.75	4.266667e+000
1.00	3.872222e+000
1.25	3.461111e+000
1.50	3.088889e+000
1.75	2.900000e+000
2.00	2.483333e+000
2.25	2.272222e+000
2.50	1.894444e+000
2.75	1.527778e+000
3.00	1.511111e+000
3.25	1.233333e+000
3.50	1.033333e+000
3.75	8.833333e-001
4.00	7.888889e-001
4.25	7.611111e-001
4.50	5.222222e-001
4.75	4.333333e-001
5.00	4.166667e-001
5.25	3.777778e-001
5.50	2.166667e-001
5.75	1.722222e-001
6.00	1.500000e-001
6.25	1.111111e-001
6.50	8.333333e-002
6.75	7.777778e-002
7.00	5.555556e-002
7.25	2.777778e-002
7.50	2.777778e-002

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

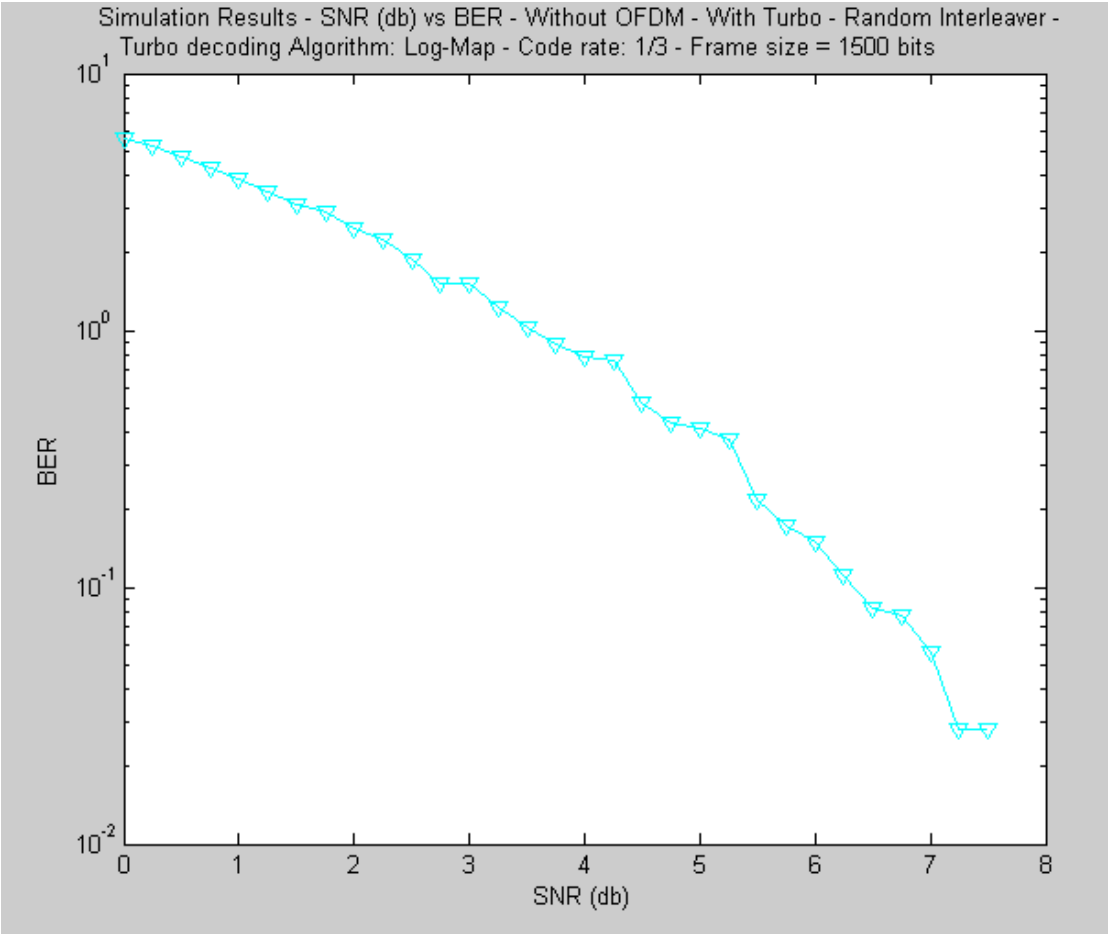


Figure 4.4. Simulation Results – SNR (db) vs BER – Without OFDM – With Turbo – Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/3 – Frame size = 1500 bits

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.5 Simulation Results – Without OFDM – Without Turbo vs With Turbo

Table 4.4. Simulation Results – SNR (db) vs BER – Without OFDM Simulation Results – SNR (db) vs BER – Without OFDM – Without Turbo vs With Turbo (Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/2 vs 1/3– Frame size = 1500 bits)

SNR	BER		
	without Turbo	with Turbo	
		punctured code rate 1/2	unpunctured code rate 1/3
0.00	8.400000e+000	6.766667e+000	5.577778e+000
0.25	7.000000e+000	6.516667e+000	5.211111e+000
0.50	6.966667e+000	6	4.716667e+000
0.75	6.800000e+000	5.077778e+000	4.266667e+000
1.00	5.700000e+000	4.872222e+000	3.872222e+000
1.25	4.933333e+000	4.322222e+000	3.461111e+000
1.50	4.866667e+000	4.072222e+000	3.088889e+000
1.75	4.433333e+000	3.583333e+000	2.900000e+000
2.00	4.433333e+000	3.094444e+000	2.483333e+000
2.25	3.700000e+000	2.694444e+000	2.272222e+000
2.50	3.633333e+000	2.305556e+000	1.894444e+000
2.75	2.600000e+000	2.094444e+000	1.527778e+000
3.00	2.300000e+000	1.888889e+000	1.511111e+000
3.25	2.166667e+000	1.450000e+000	1.233333e+000
3.50	1.766667e+000	1.338889e+000	1.033333e+000
3.75	1.466667e+000	1.183333e+000	8.833333e-001
4.00	1.466667e+000	1.055556e+000	7.888889e-001
4.25	1.066667e+000	8.666667e-001	7.611111e-001
4.50	8.666667e-001	7.388889e-001	5.222222e-001
4.75	7.666667e-001	6.444444e-001	4.333333e-001
5.00	7.000000e-001	4.833333e-001	4.166667e-001
5.25	6.000000e-001	3.944444e-001	3.777778e-001
5.50	4.666667e-001	3.000000e-001	2.166667e-001
5.75	4.666667e-001	2.666667e-001	1.722222e-001
6.00	2.333333e-001	2.000000e-001	1.500000e-001
6.25	2.000000e-001	1.388889e-001	1.111111e-001
6.50	1.333333e-001	9.444444e-002	8.333333e-002
6.75	1.000000e-001	7.222222e-002	7.777778e-002
7.00	1.000000e-001	6.666667e-002	5.555556e-002
7.25	1.000000e-001	6.111111e-002	2.777778e-002
7.50	1.000000e-001	2.222222e-002	2.777778e-002

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

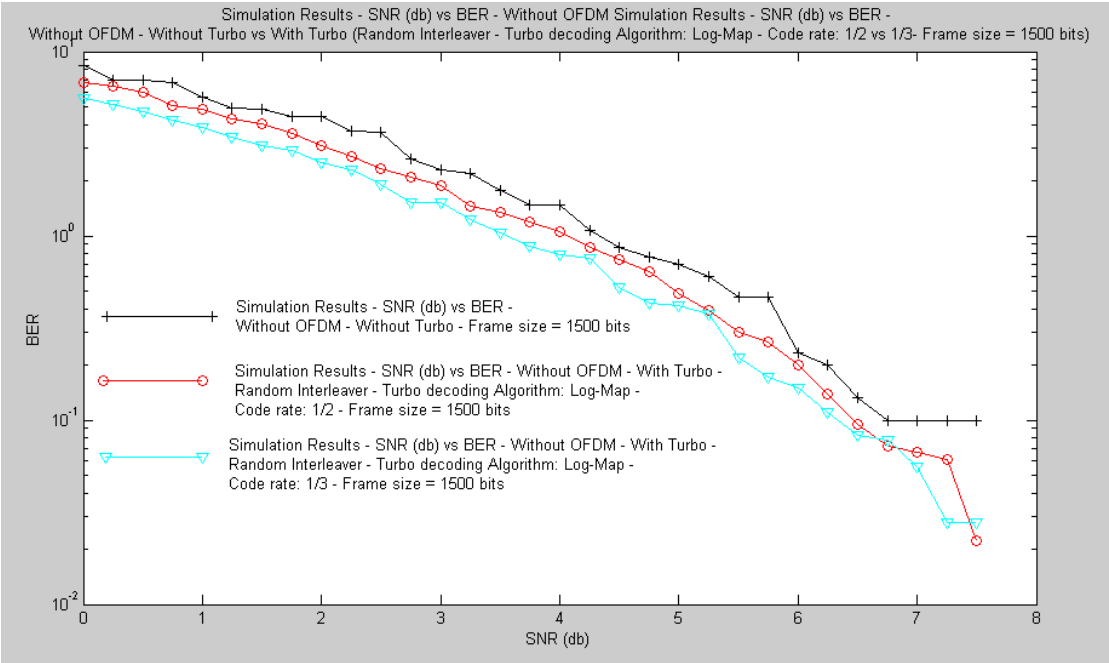


Figure 4.5. Simulation Results – SNR (db) vs BER – Without OFDM Simulation Results – SNR (db) vs BER – Without OFDM – Without Turbo vs With Turbo (Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/2 vs 1/3– Frame size = 1500 bits)

At a BER 10e-01, a SNR improvement of 0,3 dbs is observed from without Turbo to with Turbo and with code rate = 1/2.

At a BER 10e-01, a SNR improvement of 0,2 dbs is observed from with Turbo and with code rate = 1/2, to with Turbo and with code rate = 1/3.

At a BER 10e-01, a SNR improvement of 0,5 dbs is observed from without Turbo to with Turbo and with code rate = 1/3.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.6 Simulation Results – With OFDM – Without Turbo

Table 4.5. Simulation Results – SNR (db) vs BER

– With OFDM – Without Turbo – Frame size = 1500 bits

SNR	BER
0.00	8.066667e+000
0.25	6.733333e+000
0.50	6.633333e+000
0.75	6.533333e+000
1.00	5.366667e+000
1.25	4.433333e+000
1.50	4.366667e+000
1.75	4.300000e+000
2.00	4.266667e+000
2.25	3.466667e+000
2.50	3
2.75	2.566667e+000
3.00	2.100000e+000
3.25	1.766667e+000
3.50	1.666667e+000
3.75	1.233333e+000
4.00	1.233333e+000
4.25	9.333333e-001
4.50	7.333333e-001
4.75	6.000000e-001
5.00	5.666667e-001
5.25	4.666667e-001
5.50	4.666667e-001
5.75	4.000000e-001
6.00	2.000000e-001
6.25	1.333333e-001
6.50	1.000000e-001
6.75	6.666667e-002
7.00	6.666667e-002
7.25	6.666667e-002
7.50	3.333333e-002

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

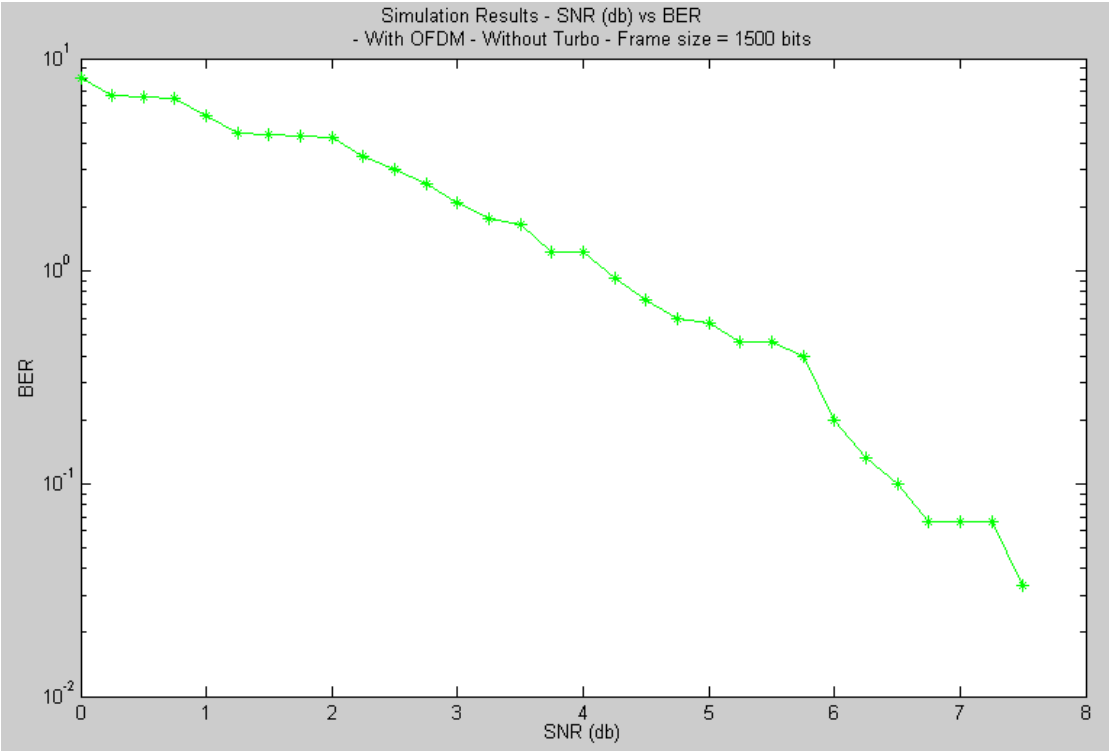


Figure 4.6. Simulation Results – SNR (db) vs BER

– With OFDM – Without Turbo – Frame size = 1500 bits

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.7 Simulation Results – With OFDM – With Turbo – Code rate: 1/2

Table 4.6. Simulation Results – SNR (db) vs BER – With OFDM – With Turbo – Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/2 – Frame size = 1500 bits

SNR	BER
0.00	6.444444e+000
0.25	6.194444e+000
0.50	5.888889e+000
0.75	4.922222e+000
1.00	4.638889e+000
1.25	4.194444e+000
1.50	3.777778e+000
1.75	3.355556e+000
2.00	3.005556e+000
2.25	2.627778e+000
2.50	2.266667e+000
2.75	2.077778e+000
3.00	1.877778e+000
3.25	1.433333e+000
3.50	1.277778e+000
3.75	1.155556e+000
4.00	1.038889e+000
4.25	8.111111e-001
4.50	7.111111e-001
4.75	6.277778e-001
5.00	4.111111e-001
5.25	3.222222e-001
5.50	2.888889e-001
5.75	2.444444e-001
6.00	1.555556e-001
6.25	1.388889e-001
6.50	8.888889e-002
6.75	6.666667e-002
7.00	5.555556e-002
7.25	2.222222e-002
7.50	1.666667e-002

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

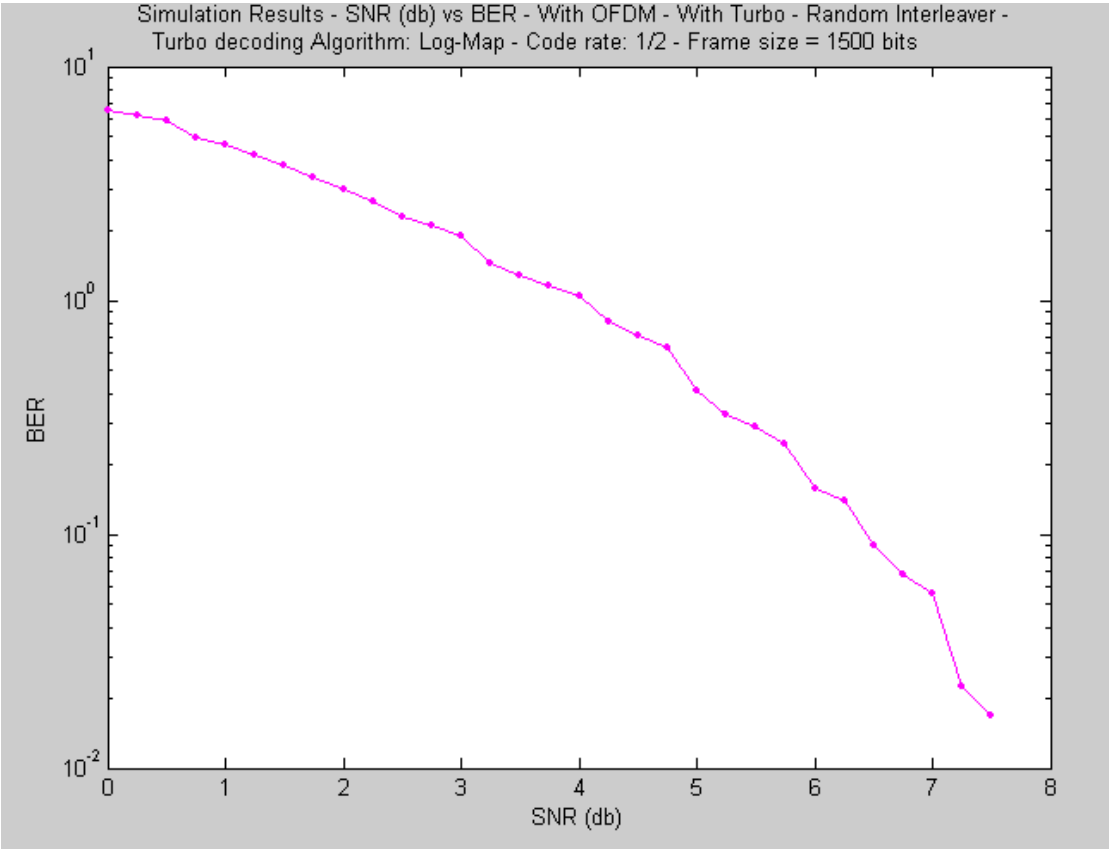


Figure 4.7. Simulation Results – SNR (db) vs BER – With OFDM – With Turbo – Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/2 – Frame size = 1500 bits

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.8 Simulation Results – With OFDM – With Turbo – Code rate: 1/3

Table 4.7. Simulation Results – SNR (db) vs BER – With OFDM – With Turbo – Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/3 – Frame size = 1500 bits

SNR	BER
0.00	5.283333e+000
0.25	5.033333e+000
0.50	4.688889e+000
0.75	4.266667e+000
1.00	3.794444e+000
1.25	3.305556e+000
1.50	2.905556e+000
1.75	2.777778e+000
2.00	2.472222e+000
2.25	2.194444e+000
2.50	1.866667e+000
2.75	1.694444e+000
3.00	1.444444e+000
3.25	1.116667e+000
3.50	9.666667e-001
3.75	8.277778e-001
4.00	6.777778e-001
4.25	6.277778e-001
4.50	4.944444e-001
4.75	3.777778e-001
5.00	3.388889e-001
5.25	2.666667e-001
5.50	2.111111e-001
5.75	1.500000e-001
6.00	1.444444e-001
6.25	8.333333e-002
6.50	8.333333e-002
6.75	6.111111e-002
7.00	5.000000e-002
7.25	5.000000e-002
7.50	1.666667e-002

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

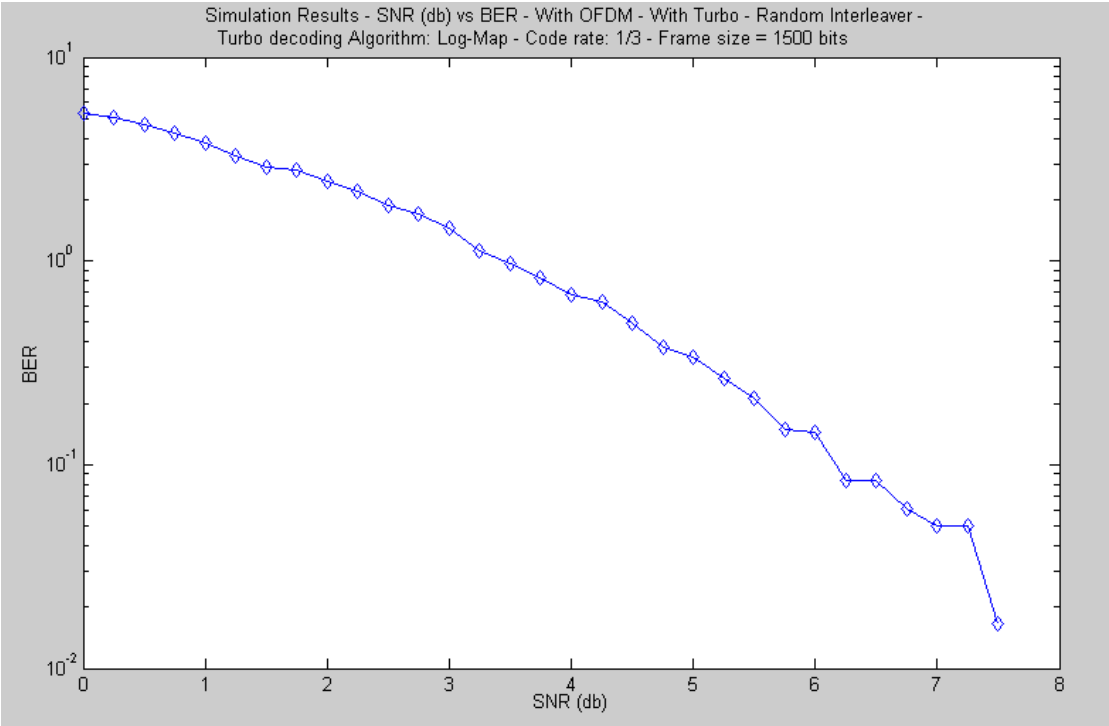


Figure 4.8. Simulation Results – SNR (db) vs BER – With OFDM – With Turbo – Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/3 – Frame size = 1500 bits

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.9 Simulation Results –With OFDM – Without Turbo

Table 4.8. Simulation Results – SNR (db) vs BER – With OFDM – Without Turbo vs With Turbo (Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/2 vs 1/3 – Frame size = 1500 bits)

SNR	BER		
	without Turbo	with Turbo	
		punctured code rate 1/2	unpunctured code rate 1/3
0.00	8.066667e+000	6.444444e+000	5.283333e+000
0.25	6.733333e+000	6.194444e+000	5.033333e+000
0.50	6.633333e+000	5.888889e+000	4.688889e+000
0.75	6.533333e+000	4.922222e+000	4.266667e+000
1.00	5.366667e+000	4.638889e+000	3.794444e+000
1.25	4.433333e+000	4.194444e+000	3.305556e+000
1.50	4.366667e+000	3.777778e+000	2.905556e+000
1.75	4.300000e+000	3.355556e+000	2.777778e+000
2.00	4.266667e+000	3.005556e+000	2.472222e+000
2.25	3.466667e+000	2.627778e+000	2.194444e+000
2.50	3	2.266667e+000	1.866667e+000
2.75	2.566667e+000	2.077778e+000	1.694444e+000
3.00	2.100000e+000	1.877778e+000	1.444444e+000
3.25	1.766667e+000	1.433333e+000	1.116667e+000
3.50	1.666667e+000	1.277778e+000	9.666667e-001
3.75	1.233333e+000	1.155556e+000	8.277778e-001
4.00	1.233333e+000	1.038889e+000	6.777778e-001
4.25	9.333333e-001	8.111111e-001	6.277778e-001
4.50	7.333333e-001	7.111111e-001	4.944444e-001
4.75	6.000000e-001	6.277778e-001	3.777778e-001
5.00	5.666667e-001	4.111111e-001	3.388889e-001
5.25	4.666667e-001	3.222222e-001	2.666667e-001
5.50	4.666667e-001	2.888889e-001	2.111111e-001
5.75	4.000000e-001	2.444444e-001	1.500000e-001
6.00	2.000000e-001	1.555556e-001	1.444444e-001
6.25	1.333333e-001	1.388889e-001	8.333333e-002
6.50	1.000000e-001	8.888889e-002	8.333333e-002
6.75	6.666667e-002	6.666667e-002	6.111111e-002
7.00	6.666667e-002	5.555556e-002	5.000000e-002
7.25	6.666667e-002	2.222222e-002	5.000000e-002
7.50	3.333333e-002	1.666667e-002	1.666667e-002

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

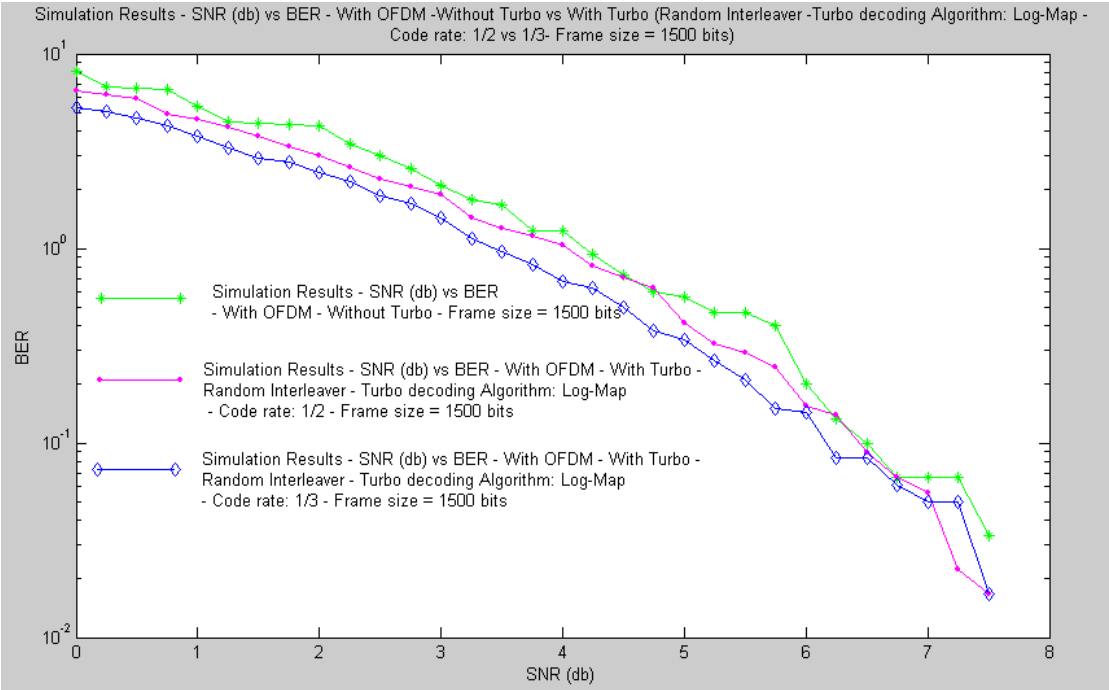


Figure 4.9. Simulation Results – SNR (db) vs BER – With OFDM – Without Turbo vs With Turbo (Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/2 vs 1/3– Frame size = 1500 bits)

At a BER 10e+00, a SNR improvement of 0,1 dbs is observed from without Turbo to with Turbo and with code rate = 1/2.

At a BER 10e+00, a SNR improvement of 0,6 dbs is observed from with Turbo and with code rate = 1/2, to with Turbo and with code rate = 1/3.

At a BER 10e+00, a SNR improvement of 0,7 dbs is observed from without Turbo to with Turbo and with code rate = 1/3.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.10 Simulation Results – With OFDM vs Without OFDM - Without Turbo

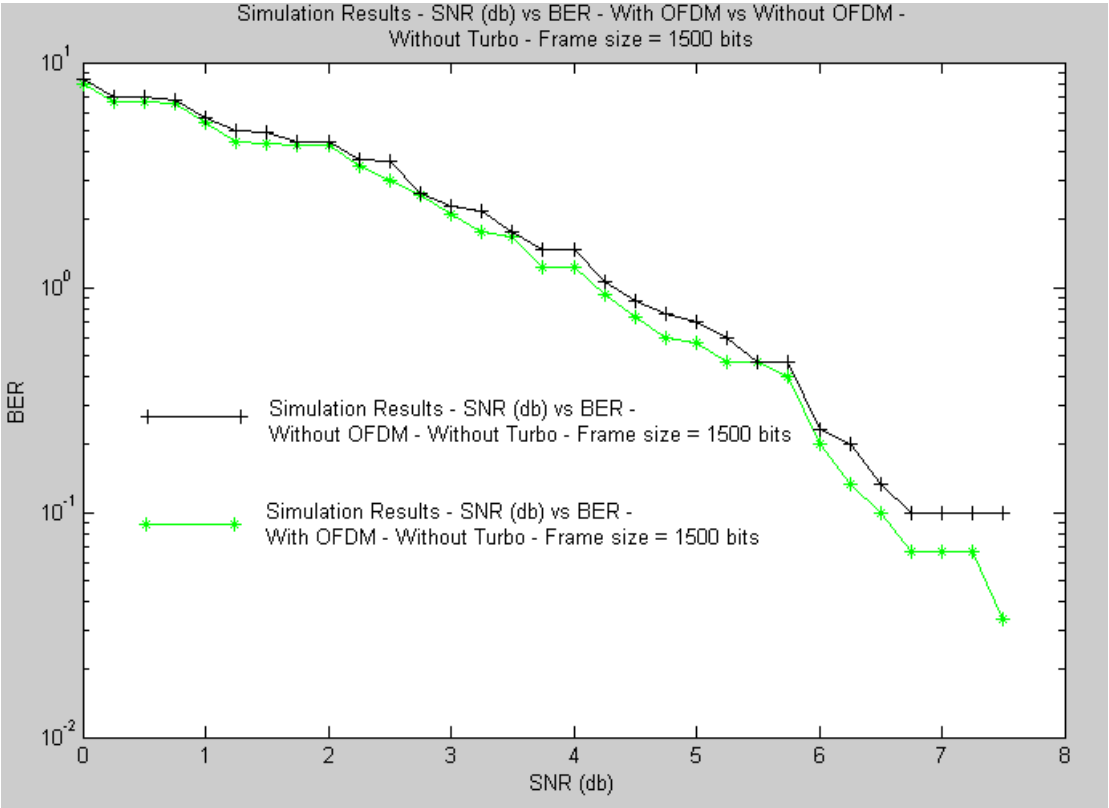


Figure 4.10. Simulation Results – SNR (db) vs BER
– With OFDM vs Without OFDM –
Without Turbo – Frame size = 1500 bits

At a BER 10e+00, a SNR improvement of 0,1 dbs is observed from without OFDM to with OFDM.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.11 Simulation Results – With OFDM vs Without OFDM – With Turbo – Code rate: 1/2

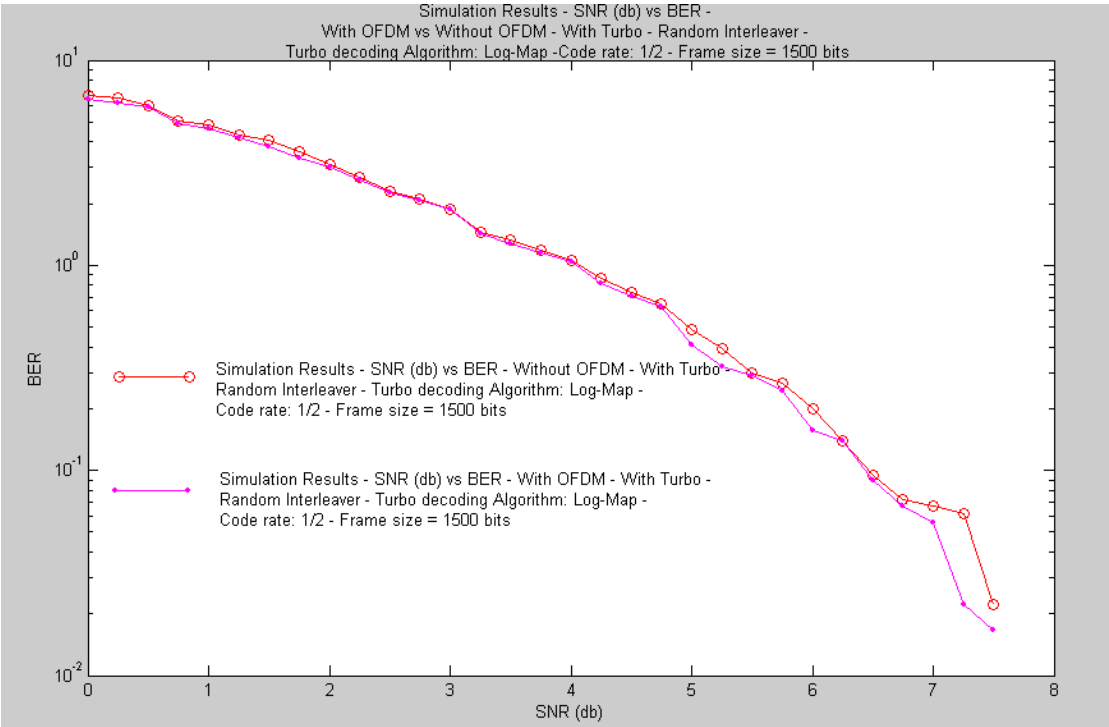


Figure 4.11. Simulation Results – SNR (db) vs BER – With OFDM vs Without OFDM – With Turbo – Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/2 – Frame size = 1500 bits

Although the above graphs seems to be similar, at some BERs, a SNR improvement of 0,2 dbs is observed from without OFDM to with OFDM.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.12 Simulation Results –With OFDM vs Without OFDM – With Turbo – Code rate: 1/3

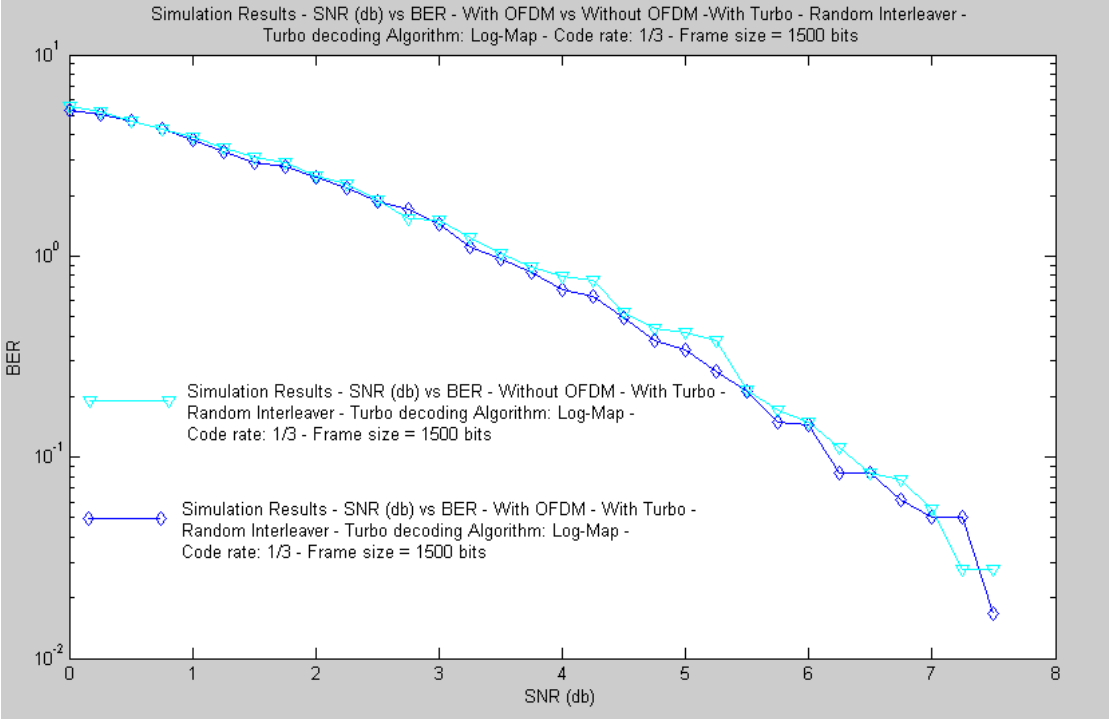


Figure 4.12. Simulation Results – SNR (db) vs BER –
With OFDM vs Without OFDM –With Turbo – Random Interleaver –
Turbo decoding Algorithm: Log-Map
– Code rate: 1/3 – Frame size = 1500 bits

At a BER = 10e-01, a SNR improvement of 0,1 dbs is observed from without OFDM to with OFDM.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

4.13 Simulation Results – With OFDM vs without OFDM – Without Turbo vs With Turbo

Table 4.9. Simulation Results – SNR (db) vs BER – With OFDM vs Without OFDM – Without Turbo vs With Turbo (Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/2 vs 1/3– Frame size = 1500 bits)

SNR	without OFDM			with OFDM		
	BER					
	without Turbo	with Turbo		without Turbo	with Turbo	
		punctured code rate 1/2	unpunctured code rate 1/3		punctured code rate 1/2	unpunctured code rate 1/3
0.00	8.400000e+000	6.766667e+000	5.577778e+000	8.066667e+000	6.444444e+000	5.283333e+000
0.25	7.000000e+000	6.516667e+000	5.211111e+000	6.733333e+000	6.194444e+000	5.033333e+000
0.50	6.966667e+000	6	4.716667e+000	6.633333e+000	5.888889e+000	4.688889e+000
0.75	6.800000e+000	5.077778e+000	4.266667e+000	6.533333e+000	4.922222e+000	4.266667e+000
1.00	5.700000e+000	4.872222e+000	3.872222e+000	5.366667e+000	4.638889e+000	3.794444e+000
1.25	4.933333e+000	4.322222e+000	3.461111e+000	4.433333e+000	4.194444e+000	3.305556e+000
1.50	4.866667e+000	4.072222e+000	3.088889e+000	4.366667e+000	3.777778e+000	2.905556e+000
1.75	4.433333e+000	3.583333e+000	2.900000e+000	4.300000e+000	3.355556e+000	2.777778e+000
2.00	4.433333e+000	3.094444e+000	2.483333e+000	4.266667e+000	3.005556e+000	2.472222e+000
2.25	3.700000e+000	2.694444e+000	2.272222e+000	3.466667e+000	2.627778e+000	2.194444e+000
2.50	3.633333e+000	2.305556e+000	1.894444e+000	3	2.266667e+000	1.866667e+000
2.75	2.600000e+000	2.094444e+000	1.527778e+000	2.566667e+000	2.077778e+000	1.694444e+000
3.00	2.300000e+000	1.888889e+000	1.511111e+000	2.100000e+000	1.877778e+000	1.444444e+000
3.25	2.166667e+000	1.450000e+000	1.233333e+000	1.766667e+000	1.433333e+000	1.116667e+000
3.50	1.766667e+000	1.338889e+000	1.033333e+000	1.666667e+000	1.277778e+000	9.666667e-001
3.75	1.466667e+000	1.183333e+000	8.833333e-001	1.233333e+000	1.155556e+000	8.277778e-001
4.00	1.466667e+000	1.055556e+000	7.888889e-001	1.233333e+000	1.038889e+000	6.777778e-001
4.25	1.066667e+000	8.666667e-001	7.611111e-001	9.333333e-001	8.111111e-001	6.277778e-001
4.50	8.666667e-001	7.388889e-001	5.222222e-001	7.333333e-001	7.111111e-001	4.944444e-001
4.75	7.666667e-001	6.444444e-001	4.333333e-001	6.000000e-001	6.277778e-001	3.777778e-001
5.00	7.000000e-001	4.833333e-001	4.166667e-001	5.666667e-001	4.111111e-001	3.388889e-001
5.25	6.000000e-001	3.944444e-001	3.777778e-001	4.666667e-001	3.222222e-001	2.666667e-001
5.50	4.666667e-001	3.000000e-001	2.166667e-001	4.666667e-001	2.888889e-001	2.111111e-001
5.75	4.666667e-001	2.666667e-001	1.722222e-001	4.000000e-001	2.444444e-001	1.500000e-001
6.00	2.333333e-001	2.000000e-001	1.500000e-001	2.000000e-001	1.555556e-001	1.444444e-001
6.25	2.000000e-001	1.388889e-001	1.111111e-001	1.333333e-001	1.388889e-001	8.333333e-002
6.50	1.333333e-001	9.444444e-002	8.333333e-002	1.000000e-001	8.888889e-002	8.333333e-002
6.75	1.000000e-001	7.222222e-002	7.777778e-002	6.666667e-002	6.666667e-002	6.111111e-002
7.00	1.000000e-001	6.666667e-002	5.555556e-002	6.666667e-002	5.555556e-002	5.000000e-002
7.25	1.000000e-001	6.111111e-002	2.777778e-002	6.666667e-002	2.222222e-002	5.000000e-002
7.50	1.000000e-001	2.222222e-002	2.777778e-002	3.333333e-002	1.666667e-002	1.666667e-002

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

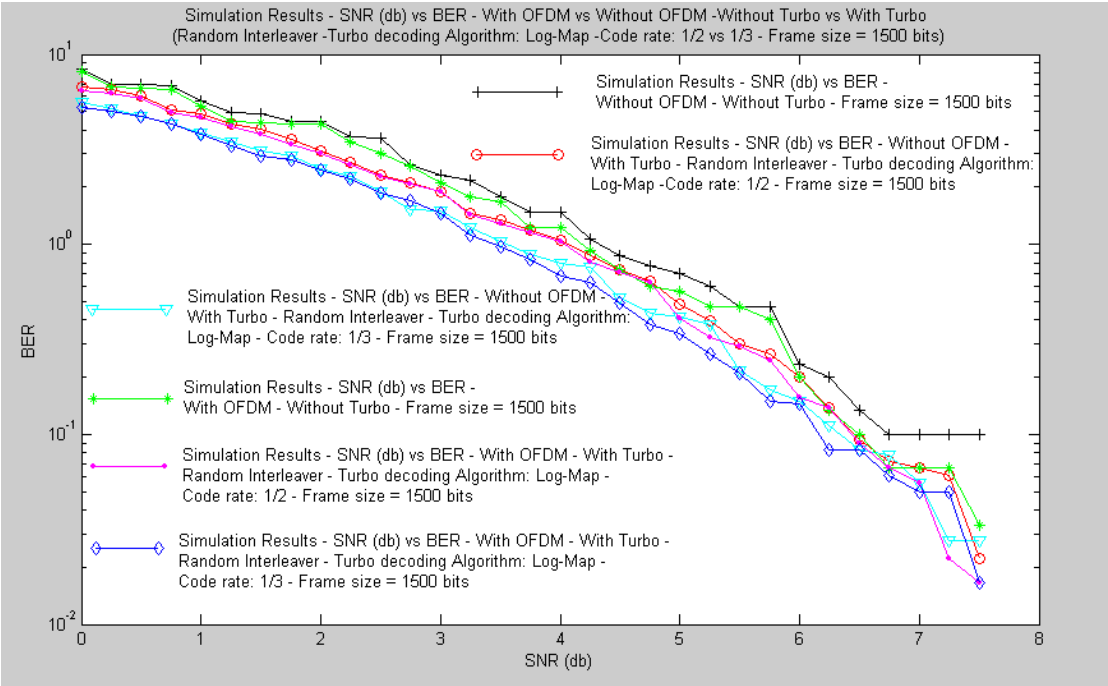


Figure 4.13. Simulation Results – SNR (db) vs BER – With OFDM vs Without OFDM – Without Turbo vs With Turbo (Random Interleaver – Turbo decoding Algorithm: Log-Map – Code rate: 1/2 vs 1/3 – Frame size = 1500 bits)

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Chapter 5 – Conclusions and Future work

5.1 Conclusions

OFDM appears to be a suitable technique as a modulation technique for high performance wireless telecommunications.

At a BER 10e-01, a SNR improvement of 0,3 dbs is observed from without OFDM – without Turbo to without OFDM – with Turbo and with code rate = 1/2.

At a BER 10e-01, a SNR improvement of 0,2 dbs is observed from without OFDM – with Turbo and with code rate = 1/2, to without OFDM – with Turbo and with code rate = 1/3.

At a BER 10e-01, a SNR improvement of 0,5 dbs is observed from without OFDM – without Turbo to without OFDM – with Turbo and with code rate = 1/3.

At a BER 10e+00, a SNR improvement of 0,1 dbs is observed from with OFDM – without Turbo to with OFDM – with Turbo and with code rate = 1/2.

At a BER 10e+00, a SNR improvement of 0,6 dbs is observed from with OFDM – with Turbo and with code rate = 1/2, to with OFDM – with Turbo and with code rate = 1/3.

At a BER 10e+00, a SNR improvement of 0,7 dbs is observed from with OFDM – without Turbo to with OFDM – with Turbo and with code rate = 1/3.

At a BER 10e-01, a SNR improvement of 0,2 dbs is observed from without OFDM – without Turbo to with OFDM – without Turbo.

At a BER = 10e-01, a SNR improvement of 0,1 dbs is observed from without OFDM – with Turbo with code rate = 1/3 to with OFDM– with Turbo with code rate = 1/3.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

5.2 Future work

OFDM promises to be a suitable modulation technique for high capacity wireless communications and will become increasing important in the future as wireless networks become more relied on. Future work areas involve the improvement of BER with not only the use of OFDM and BPSK modulation, but a combination of OFDM with other modulation techniques such as QPSK, QAM, FSK etc. Also different codes rates in Turbo coding such as 2/3 may be used in order to achieve BER improvements.

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

References

[1] Jawad Ibrahim, "4G Features", December 2002

[2] Pereira,V., Sousa,T., "Evolution of Mobile Communications: from 1G to 4G"

[3] Brooks,A., Hoeler,S., "Design and Implementation of OFDM Signaling", March 2001

[4] Pappaport,T., "Inter-Carrier Interference Cancellation for OFDM Systems", Spring 2003

[5] Mishra, A., "*Fundamentals of Cellular Network Planning & Optimisation*", © 2004 John Wiley & Sons, Ltd. ISBN: 0-470-86267-X

[6] Litwin,L.,Pugel,M., "The principles of OFDM", January 2001

[7] Mohamedian,M., Elrahman,I., Osman,I., "PERFORMANCE OF ORTHOGONAL FREQUENCY DIVISION MULTIPLEXING (OFDM) UNDER THE EFFECT OF WIRELESS TRANSMISSION SYSTEM DRAWBACKS", September 2007

[8] Yeung,K., "Detailed OFDM Modeling in Network Simulation of Mobile Ad Hoc Networks", 2003

[9] Rakesh Rajbanshi, Alexander M. Wyglinski, and Gary J. Minden, "An Efficient Implementation of NC-OFDM Transceivers for Cognitive Radios," in Proceedings of the 1st International Conference on Cognitive Radio Oriented Wireless Networks and Communications (Mykonos Island, Greece), June 2006

[10] Guffey,J., "OFDM Physical Layer Implementation for the Kansas University Agile Radio", February 2008

[11] Lawrey,E., "Adaptive Techniques for Multi-users OFDM", James cooks University, December 2001

[12] Yeung,G., Takai,M., Bagrodia,R., Mehrnia,A., Daneshrad,B., "Detailed OFDM Modeling in Network Simulation of Mobile Ad Hoc Networks", 2004

[13] Wikipedia, an online encyclopedia available at www.wikipedia.org

[14] Techfocus Media, Inc, "Implementation of an OFDM Wireless Transceiver using IP Cores on an FPGA", FPGA and Programmable Logic Journal, 2005

[15] Anibal Luis Intinid, "Orthogonal Frequency Division Multiplexing for Wireless Networks", University of California, December 2000, available online at http://www.engineering. Ucsb.edu

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

[16] Rappaport,T., "Wireless Communications: Principles and Practice", Pearson Education, Inc., 2nd edition

[17] Brooks,A., Hoelzer,S., "Design and Implementation of Orthogonal Frequency Division Multiplexing (OFDM) Signaling", March 2001

[18] Keller, Thomas, and Lajos Hanzo. "Adaptive Multicarrier Modulation: A Convenient Framework for Time-Frequency Processing in Wireless Communications" *IEEE Proceedings of the IEEE* 88 (May, 2000): 609-640

[19] Wang, Zhengdao, and Georgios B. Giannakis. "Wireless Multicarrier Communications" *IEEE Signal Processing Magazine* (May, 2000): 29-48

[20] Chiu,Y., Markovic,D., Tang, H., Zhang, N., "OFDM Receiver Design", December 2000

[21] Zhang,W., Xia,X., Letaief,K., "SPACE-TIME/FREQUENCY CODING FOR MIMO-OFDM IN NEXT GENERATION BROADBAND WIRELESS SYSTEMS", 2007

[22] Shen,Y., Martinez,E., "Channel Estimation in OFDM Systems", January 2006

[23] Taha,A., "Performance Analysis of ICC technique for OFDM PAPR reduction and its Application over BTC", 2006

[24] Hanzo,L., Munster,M., Choi,B., Keller,T., "OFDM and MC-CDMA for Broadband Multi-User Communications, WLANs and Broadcasting", IEEE Press, Wiley, 2003

[25] Pyndiah,R., Glavieux,A., Picart,A. Jacq,S., "near optimum decoding of product codes", Global Telecommunications Conference, 1994. GLOBECOM '94. 'Communications: The Global Bridge'., IEEE 28 Nov.-2 Dec. 1994 Page(s):339 - 343 vol.1

[26] Jones,A., Wilkinson,T., Barton,S., "Block coding for reduction of Peak to mean envelope power ratio of multicarrier transmission schemes", Electronic Letters, vol. 30. no. 25, pp. 2098-2099, December 1994

[27] Ramesh Mehendra pyndiah, "near optimum decoding of product codes: Block turbo codes", Communications, IEEE Transactions on Volume 46, Issue 8, Aug. 1998 Page(s):1003 -1010

[28] Slimane,S., "Peak-to-average power ratio reduction of OFDM signals using pulse shaping", IEEE Globecom, Vol 3, Page(s):1412 1416, 27 Nov.-1 Dec. 2000

[29] Pandey,A., Agrawalla,S., Manivannan,S., “VLSI implementation of OFDM modem”, 2001

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

[30] Lou I. Ilunga, "Adaptive, Turbo-coded OFDM", June 30, 2005

[31] Burr,A., "Turbo-codes: the ultimate error control codes?", ELECTRONICS & COMMUNICATION ENGINEERING JOURNAL, AUGUST 2001

[32] Kasper,E., "Turbo Codes", Laboratory for Theoretical Computer Science, Helsinki University of Technology, Finland

[33] Fu-hua Huang, "Evaluation of Soft Output Decoding for Turbo Codes", Blacksburg Virginia, May 1997

[34] Berrou,C., Glaviex,A., Thitimajshima,P., "NEAR SHANNON LIMIT ERROR – CORRECTING CODING AND DECODING: TURBO CODES"

[35] Langton,C., "Coding and Decoding with Convolutional Codes", Copyright July1999, available at www.complextoreal.com

[36] Sklar,B., "Digital Communications – Fundamental and Applications", Second Edition (Prentice-Hall, ISBN 0-13-084788-7), Communications Engineering Services, Tarzana, California and University of California, Los Angeles, 2001

[37] Lattice Semiconductor Corp, "Turbo Decoder", April 2003

[38] Chaikalis,C., "SOVA/LOG-MAP: ANALYSIS AND 3G IMPLEMENTATION FOR UNCORRELATED RAYLEIGH FADING", Department of Electrical Engineering, Technological Educational Institute of Lamia, Greece, January 2007

[39] Huang,F., "Evaluation of Soft Output Decoding for Turbo Codes", Faculty of Virginia Polytechnic Institute and State University, Blacksburg, Virginia, May 1997

[40] Barbulescu,S., Pietrobon,S., "Turbo Codes: a tutorial on a new class of powerful error correcting coding schemes, Part I: Code Structures and Interleaver Design", Institute for Telecommunications Research University of South Australia, October 1998

[41] Schurgers,C., Catthoor,F., Engels,M., "OPTIMIZED MAP TURBO DECODER"

[42] Berrou,C., Glavieux,A., Thitimajshima,P., "Near Shannon limit error-correcting coding and decoding: Turbo-codes" *Proc. International Conference on Communications (ICC'93)*, Geneva, Switzerland, May 1993, pp. 1064-1070

[43] Barbulescu,A., Pietrobon,S., "Interleaver design for turbo codes", *Electronics Letters*, Vol.30, No.25, December 1994, pp. 2107-2108

[44] Sami Ahmed Haider, Khalida Noori, "ADAPTIVE TURBO CODED OFDM", Journal of Digital Information Management, Volume 5, Number 6, December 2007

[45] Schlegel,C., Perez,L., "Trellis and Turbo Coding" IEEE Press, Hoboken, NJ: John Wiley & Sons, INC

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

[46] http://www.answers.com/topic/viterbi-algorithm

[47] Nguyen,H., Duhamel,P., "Optimal Soft-Output Viterbi Algorithm"

[48] Miftahul Huda, Achmad Affandi, Suwadi, Tri Budi Santoso, Arifin, Aries P., "Performance of Iterative Soft Output Viterbi Algorithm (SOVA) Decoder on Turbo Coding under Fading Channel", 2002

[49] http://www.pudn.com/downloads37/sourcecode/app/detail118916.html

[50] http://en.pudn.com/downloads62/sourcecode/comm/detail213909_en.html

Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

Appendix

aploawgn.m

```
% Copyright January 2010, E. Baltagianni
% for academic use only

% This script simulates a system without a coding scheme.
% Random information bits are modulated, and transmitted through an AWGN
channel.

clear all

% Frame size
L_total = input(' Please enter the frame size (= info + tail, default: 15)   ');
if isempty(L_total)
   L_total = 15;     % infomation bits plus tail bits
end     % if

% Number of frames
nframe=200;
% nframe=1500; in turbo_awgn.m

% Fading amplitude; a=1 in AWGN channel
a = 1;

EbN0db= input(' Please enter EbN0db : default [2.0]    ');
if isempty(EbN0db)
     EbN0db = [2.0];
end     % if

num(1:nframe,1:length(EbN0db))=zeros(nframe,length(EbN0db));

fprintf('\n\n-----------------------------------------------------\n');
fprintf(' Frame size = %6d\n',L_total);
fprintf(' EbN0db = ');

for i = 1:length(EbN0db)
    fprintf('%10.2f',EbN0db(i));
end     % for

fprintf('\n-----------------------------------------------------\n\n');

fprintf('+ + + + Please be patient. Wait a while to get the result. + + + +\n');

%********************** start **********************

frcount=0;     % frame counter
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
while frcount<nframe
   frcount = frcount + 1;
   x = round(rand(1, L_total));     % info bits

%************************ Process and RF Transmission ************************
   fs=1/21e-6;     % sampling frequency
   fc=12500;      % center frequency
   fd=fs/100;      % symbol frequency
   % fd=fs/70; in turbo_awgn.m
   rf_sima=dmod(x,fc,fd,fs,'psk',2);     % PSK Transmission
   Ps=pwr(rf_sima);       %Signal power

   for nEN = 1:length(EbN0db)

%************************ noisy channel ************************

       en = 10^(EbN0db(nEN)/10);      % convert Eb/N0 from unit db to normal
numbers
       sigma = sqrt(Ps/(2*en/(fs/fd)));        % standard deviation of AWGN noise
       rf_noisy = rf_sima+sigma*randn(1,L_total*(fs/fd)); % received bits
       % rf_noisy = rf_sima+sigma*randn(1,L_total*(2+puncture)*(fs/fd)); in
turbo_awgn.m

%************************ receiver ************************
      zn=bpskdemod(rf_noisy,fc,fd,fs,'hard');     % Digital demodulation of
received signal
      % zn=bpskdemod(rf_noisy,fc,fd,fs,'soft'); in turbo_awgn.m
      % method = hard  for hard decision
      % method = soft  for soft decision
      [num(frcount,nEN),rcode(nEN)]=biterr(x,zn)     % Calculation of the
number of ennoreous bits and the Bit Error Rate
      % [num(frcount,nEN),rcode(nEN)]=biterr(x,xhat(1:length(x) in
turbo_awgn.m
   end     % for

end     % while

for nEN=1:length(EbN0db)

  finalBER(nEN)=sum(num(1:frcount,nEN))/(nframe*(L_total));
  % finalBER(nEN)=sum(num(1:frcount,nEN))/(nframe*(L_total-m)); in
turbo_awgn.m
  fprintf('FINAL BIT ERROR RATE=%d%% for
EbN0db=%5.2f',100*finalBER(nEN), EbN0db(nEN));
  fprintf('\n');
end     % for

plot(EbN0db,finalBER,'-.r*')
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

bin_state.m

```
function bin_state = bin_state( int_state, m )
% converts an vector of integer into a matrix; the i-th row is the binary form
% of m bits for the i-th integer
for j = 1:length( int_state )
   for i = m:-1:1
      state(j,m-i+1) = fix( int_state(j)/ (2^(i-1)) );
      int_state(j) = int_state(j) - state(j,m-i+1)*2^(i-1);
   end
end
bin_state = state;
```

Bpskdemod.m

```
function z=bpskdemod(y,fc,fd,fs,method);
% Copyright January 2010 E. Baltagianni
% for academic use only

% bpsk   demodulator
% z=bpskdemod(y,fc,fd,fs,method)
% method = hard  for hard decision
% method = soft  for soft decision

Tb=fs/fd; %bit duration in samples
nbit=length(y)/Tb; %number of bits


x0=zeros(1,nbit);
x1=ones(1,nbit);
y0=dmod(x0,fc,fd,fs,'psk',2);%  bpsk transmission
y1=dmod(x1,fc,fd,fs,'psk',2);% bpsk transmission

z0=y0.*y;
z1=y1.*y;



for i=1:nbit
   ze0=sum(z0((i-1)*Tb+1:i*Tb));
   ze1=sum(z1((i-1)*Tb+1:i*Tb));
   if findstr(method, '/hard')
      if ze1>ze0
         z(i)=1;
      else
         z(i)=0;
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
        end
    elseif findstr(method, '/soft')
        z(i)=ze1-ze0;
    else
        warning ('Invalid method')
        z(i)=nan;
    end
end


% BPSK simulation using a carrier cosine wave with ISI
%%%%%%%%%%%%%%%%%%%%%%%%%%%%clc;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%5close all;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%clear all;
% figure(1)
n=160;
% n=320; in increaseber.m
% n=324; in ofdm_ber.m
for i=1:n
    data(i)= 2*round(rand)-1;
end
% create modulated BPSK signal
% first expand the bit stream
exdata=[];
for i=1:length(data)
  for rep=1:5
  exdata= [exdata data(i)];
  end
end
ts=.1;
t=1:ts:80.9;
carrier=cos(pi*t);
% multiply expanded bitstream by cosine wave with carrier frequency
% this is the BPSK that is to be transmitted over the channel
bpsk=carrier.*exdata;
% bpsk=[bpsk(length(bpsk)-1) bpsk(length(bpsk)) bpsk];
% plot(bpsk)
% generating the noise
% p=rand(1,800)*2*pi;
p=rand*2*pi;
snr=10;
r=sqrt(-1*(1/snr*log(1 - rand)));
% no = 5*(r.* exp(j*p));
no = (r.* exp(j*p));
% value of alpha
al=rand+j*rand;
% al=1;
% Spreading channel with the alpha as the variable
for k=5:5:795
    for l = 1:5
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
    % al=round(rand)+j*round(rand)
     rec(k+l)=bpsk(k+l)+al*bpsk(k-5+l);
    end
end
rxdata=rec+ no ;
% begin demodulation
% first multiply recieved bitstream by cosine wave with carrier frequency
% figure(2)
uncarry=rxdata.*carrier;
% plot(uncarry)
% demodulate by integrating
dec1=[];
for inc=1:5:length(uncarry)
  dec=trapz(inc:inc+4,uncarry(inc:inc+4));
  dec1=[dec1 dec];
end
% make decision with a threshold of zero
demod=[];
for i=1:length(dec1)
    if dec1(i)>0
        demod=[demod 1];
    else
        demod=[demod -1];
    end
end
% stem(demod)
% calculate errors
error=0;
for i=1:length(demod)
    if data(i)~=demod(i)
        error=error+1;
    end
end
error
ber=error/n
figure(3)
title('Comparing the Bits at transmitter and receiver')
stem(data)
hold
stem(demod,'rx')
```

demultiplex.m

```
function subr = demultiplex(r, alpha, puncture);
% At receiver end, serial to parallel demultiplex to get the code word of each
% encoder
% r: received bits
% alpha: interleaver mapping
% puncture = 0: use puncturing to increase rate to 1/2;
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
% puncture = 1; unpunctured, rate 1/3;
% Frame size, which includes info. bits and tail bits
L_total = length(r)/(2+puncture);
% Extract the parity bits for both decoders
if puncture == 1 % unpunctured
   for i = 1:L_total
      x_sys(i) = r(3*(i-1)+1);
      for j = 1:2
         subr(j,2*i) = r(3*(i-1)+1+j);
      end
   end
else % punctured
   for i = 1:L_total
      x_sys(i) = r(2*(i-1)+1);
      for j = 1:2
         subr(j,2*i) = 0;
      end
      if rem(i,2)>0
         subr(1,2*i) = r(2*i);
      else
         subr(2,2*i) = r(2*i);
      end
   end
end
% Extract the systematic bits for both decoders
for j = 1:L_total
   % For decoder one
   subr(1,2*(j-1)+1) = x_sys(j);
   % For decoder two: interleave the systematic bits
   subr(2,2*(j-1)+1) = x_sys(alpha(j));
end
```

turbo_awgn.m

```
% Copyright January 2010, E. Baltagianni
% for academic use only

% This script simulates the classical turbo encoding-decoding system.
% It simulates parallel concatenated convolutional codes.
% Two component rate 1/2 RSC (Recursive Systematic Convolutional)
component encoders are assumed.
% First encoder is terminated with tails bits. (Info + tail) bits are scrambled
and passed to
% the second encoder, while second encoder is left open without tail bits of
itself.
%
% Random information bits are modulated into +1/-1, and transmitted through
a AWGN channel.
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
% Interleavers are randomly generated for each frame.
%
% Log-MAP algorithm without quantization or approximation is used.
% By making use of ln(e^x+e^y) = max(x,y) + ln(1+e^(-abs(x-y))),
% the Log-MAP can be simplified with a look-up table for the correction
function.
% If use approximation ln(e^x+e^y) = max(x,y), it becomes MAX-Log-MAP.
%
clear all

% Choose decoding algorithm
dec_alg = input(' Please enter the decoding algorithm. (0:Log-MAP, 1:SOVA)
default 0    ');
if isempty(dec_alg)
   dec_alg = 0;
end     %if

% Frame size
L_total = input(' Please enter the frame size (= info + tail, default: 15)   ');
if isempty(L_total)
   L_total = 15;     % infomation bits plus tail bits
end     %if

% Code generator
g = input(' Please enter code generator: ( default: g = [1 0 1 1; 0 1 0 1 ] )      ');
if isempty(g)
   g = [ 1 0 1 1;
         0 1 0 1 ];
end     %if

% Number of frames
nframe=1500;
% nframe=200; in aplo_awgn.m

[n,K] = size(g);
m = K - 1;
nstates = 2^m;

% puncture = 0, puncturing into rate 1/2;
% puncture = 1, no puncturing
puncture = input(' Please choose punctured / unpunctured (0/1): default 0     ');
if isempty(puncture)
    puncture = 0;
end     %if

% Code rate
rate = 1/(2+puncture);

% Fading amplitude; a=1 in AWGN channel
a = 1;
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
% Number of iterations
niter = input(' Please enter number of iterations for each frame: default 5
');
if isempty(niter)
   niter = 5;
end     %if

EbN0db= input(' Please enter EbN0db: default [2.0]    ');
if isempty(EbN0db)
     EbN0db = [2.0];
end     %if

num(1:nframe,1:length(EbN0db))=zeros(nframe,length(EbN0db));


fprintf('\n\n-----------------------------------------------------\n');
if dec_alg == 0
   fprintf(' === Log-MAP decoder === \n');
else
   fprintf(' === SOVA decoder === \n');
end     %if
fprintf(' Frame size = %6d\n',L_total);
fprintf(' code generator: \n');
for i = 1:n
    for j = 1:K
        fprintf( '%6d', g(i,j));
    end
    fprintf('\n');
end     % for
if puncture==0
   fprintf(' Punctured, code rate = 1/2 \n');
else
   fprintf(' Unpunctured, code rate = 1/3 \n');
end     %if
fprintf(' iteration number =  %6d\n', niter);
fprintf(' EbN0db = ');
for i = 1:length(EbN0db)
    fprintf('%10.2f',EbN0db(i));
end     % for
fprintf('\n-----------------------------------------------------\n\n');

fprintf('+ + + + Please be patient. Wait a while to get the result. + + + +\n');
%************************* start *************************
frcount=0;     % frame counter

while frcount<nframe
    frcount = frcount + 1;

    x = round(rand(1, L_total-m));     % info bits
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
   [temp, alpha] = sort(rand(1,L_total));     % random interleaver mapping
   en_output = encoderm( x, g, alpha, puncture ) ;     % encoder output (+1/-1)
   sa=find(en_output==-1);
   en_output(sa)=zeros(size(sa));     % convert -1 to 0

%************************    Process and RF Transmission
************************
   fs=1/21e-6;     % sampling frequency
   fc=12500;     % center frequency
   fd=fs/70;     % symbol frequency
   % fd=fs/100; in aplo_awgn.m
   rf_sima=dmod(en_output,fc,fd,fs,'psk',2);     % PSK Transmission
   Ps=pwr(rf_sima);     % Signal power

  for nEN = 1:length(EbN0db)

%************************  noisy channel ************************
      en = 10^(EbN0db(nEN)/10);     % convert Eb/N0 from unit db to normal
numbers
     sigma = sqrt(Ps/(2*en/(fs/fd)));     % standard deviation of AWGN noise
     rf_noisy = rf_sima+sigma*randn(1,L_total*(2+puncture)*(fs/fd));     %
received bits
     % rf_noisy = rf_sima+sigma*randn(1,L_total*(fs/fd)); in aplo_awgn.m
     L_c = 4*a*en*rate;     % reliability value of the channel

%************************   receiver   ************************

      zn=bpskdemod(rf_noisy,fc,fd,fs,'soft');     % Digital Demodulation of
received signal
      % zn=bpskdemod(rf_noisy,fc,fd,fs,'hard'); in aplo_awgn.m
      % method = hard  for hard decision
      % method = soft  for soft decision
      zn0=1e-003*zn;     %scaling of demodulated bits

      % xhat: estimated bits
      % r: received bits
      % g: generator matrix
      % niter: number of interations
      % puncture: puncture = 0, puncturing into rate 1/2 - puncture = 1, no
puncturing
      % alpha: interleaver mapping
      % L_c: reliability value of the channel
      % dec_alg: 0:Log-MAP, 1:SOVA
      xhat=tdecoder(zn0, g, niter, puncture,alpha, L_c,dec_alg);
      [num(frcount,nEN),rcode(nEN)]=biterr(x,xhat(1:length(x)))     %
Calculation of the number of erroneous bits and the Bit Error rate
      % num(frcount,nEN),rcode(nEN)]=biterr(x,zn)in aplo_awgn.m
   end     % for

end     % while
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
for nEN=1:length(EbN0db)

  finalBER(nEN)=sum(num(1:frcount,nEN))/(nframe*(L_total-m));
  % finalBER(nEN)=sum(num(1:frcount,nEN))/(nframe*(L_total)); in
aplo_awgn.m
  fprintf('FINAL BIT ERROR RATE=%d%% for
EbN0db=%5.2f',100*finalBER(nEN), EbN0db(nEN));
  fprintf('\n');
end     % for

 plot(EbN0db,finalBER,'-.r*')
```

Trellis.m

```
function [next_out, next_state, last_out, last_state] = trellis(g)
% set up the trellis given code generator g
% g given in binary matrix form. e.g. g = [ 1 1 1; 1 0 1 ];
% next_out(i,1:2): trellis next_out (systematic bit; parity bit) when input = 0,
state = i; next_out(i,j) = -1 or 1
% next_out(i,3:4): trellis next_out (systematic bit; parity bit) when input = 1,
state = i;
% next_state(i,1): next state when input = 0, state = i;next_state(i,i) = 1,...2^m
    % next_state(i,2): next state when input = 1, state = i;
% last_out(i,1:2): trellis last_out (systematic bit; parity bit) when input = 0,
state = i; last_out(i,j) = -1 or 1
% last_out(i,3:4): trellis last_out (systematic bit; parity bit) when input = 1,
state = i;
% last_state(i,1): previous state that comes to state i when info. bit = 0;
% last_state(i,2): previous state that comes to state i when info. bit = 1;
[n,K] = size(g);
m = K - 1;
max_state = 2^m;
% set up next_out and next_state matrices for systematic code
for state=1:max_state
    state_vector = bin_state( state-1, m );
    % when receive a 0
    d_k = 0;
    a_k = rem( g(1,:)*[0 state_vector]', 2 );
    [out_0, state_0] = encode_bit(g, a_k, state_vector);
    out_0(1) = 0;
    % when receive a 1
    d_k = 1;
    a_k = rem( g(1,:)*[1 state_vector]', 2 );
    [out_1, state_1] = encode_bit(g, a_k, state_vector);
    out_1(1) = 1;
    next_out(state,:) = 2*[out_0 out_1]-1;
    next_state(state,:) = [(int_state(state_0)+1),(int_state(state_1)+1)];
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
end
% find out which two previous states can come to present state
last_state = zeros(max_state,2);
for bit=0:1
   for state=1:max_state
      last_state(next_state(state,bit+1), bit+1)=state;
      last_out(next_state(state, bit+1), bit*2+1:bit*2+2) ...
         = next_out(state, bit*2+1:bit*2+2);
   end
end
```

tdecoder.m

```
function xhat=tdecoder(r, g, niter, puncture,alpha, L_c,dec_alg)

% xhat=tdecoder(r, g, niter, puncture,alpha, dec_alg)
% xhat: estimated bits
% r: received bits
% g: generator matrix
% niter: number of interations
% puncture: puncture = 0, puncturing into rate 1/2 - puncture = 1, no
puncturing
% alpha: interleaver mapping
% L_c: reliability value of the channel
% dec_alg: 0:Log-MAP, 1:SOVA

%********** **********  receiver   ****************************************
% Code rate
rate = 1/(2+puncture);

L_total=length(r)/(2+puncture);
yk = demultiplex(r,alpha,puncture); % demultiplex to get input for decoder 1
and 2

% Scale the received bits
     rec_s = 0.5*1e-004*L_c*yk;

% Initialize extrinsic information
     L_e(1:L_total) = zeros(1,L_total);

%********** iterations **********
     for iter = 1:niter
% Decoder one
        L_a(alpha) = L_e;  % a priori info.
        if dec_alg == 0
          L_all = logmapo(rec_s(1,:), g, L_a, 1);  % complete info.
        else
          L_all = sova(rec_s(1,:), g, L_a, 1);  % complete info.
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
        end
        L_e = L_all - 2*rec_s(1,1:2:2*L_total) - L_a;  % extrinsic info.

% Decoder two
        L_a = L_e(alpha);  % a priori info.
        if dec_alg == 0
          L_all = logmapo(rec_s(2,:), g, L_a, 2);  % complete info.
        else
          L_all = sova(rec_s(2,:), g, L_a, 2);  % complete info.
        end
        L_e = L_all - 2*rec_s(2,1:2:2*L_total) - L_a;  % extrinsic info.

    end         %iter
%************************ end of iterations *************************
% Estimate the info. bits
        xhat(alpha) = (sign(L_all)+1)/2;
```

Rsc_encode.m

```
function y = rsc_encode(g, x, terminated)
% encodes a block of data x (0/1)with a recursive systematic
% convolutional code with generator vectors in g, and
% returns the output in y (0/1).
% if terminated>0, the trellis is perfectly terminated
% if terminated<0, it is left unterminated;
% determine the constraint length (K), memory (m), and rate (1/n)
% and number of information bits.
[n,K] = size(g);
m = K - 1;
if terminated>0
   L_info = length(x);
   L_total = L_info + m;
else
   L_total = length(x);
   L_info = L_total - m;
end
% initialize the state vector
state = zeros(1,m);
% generate the codeword
for i = 1:L_total
   if terminated<0 | (terminated>0 & i<=L_info)
      d_k = x(1,i);
   elseif terminated>0 & i>L_info
      % terminate the trellis
      d_k = rem( g(1,2:K)*state', 2 );
   end
   a_k = rem( g(1,:)*[d_k state]', 2 );
   [output_bits, state] = encode_bit(g, a_k, state);
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
    % since systematic, first output is input bit
    output_bits(1,1) = d_k;
    y(n*(i-1)+1:n*i) = output_bits;
end
```

pwr.m

```
function p=pwr(x)
% Copyright January 2010 E. Baltagianni
% for academic use only

% Power of an array
%    p=pwr(x)

p=sum(x.^2)/length(x);
```

ofdm_ber.m

```
% Vinay Mohan Jonnakuti
% Wireless Communication:
% BPSK modulation and demodulation with increased bit rate
% BPSK simulation using a carrier cosine wave with increased bit rate
clc;
close all;
clear all;

% no of sub carrier channels
channels=6;
c=6;
% bits per channels
bits=54
% total no of bits to be transmitted
n=324;
% n=320; in increaseber.m
% n=160; in BPSKmodanddemod.m
for i=1:n
    data(i)= 2*round(rand)-1;
end

% Converting the series into parallel for the channels
s = reshape(data,c,bits);

% BPSK modulation for a single channel
signal1 = s(1,:);
% first expand the bit stream
exdata1=[];
exdata2=[];
exdata3=[];
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
exdata4=[];
exdata5=[];
exdata6=[];
for i=1:length(signal1)
  for rep=1:2
  exdata1= [exdata1 signal1(i)];
  end
end

signal2 = s(2,:);
% first expand the bit stream
exdata=[];
for i=1:length(signal2)
  for rep=1:2
  exdata2= [exdata2 signal2(i)];
  end
end

signal3= s(3,:);
% first expand the bit stream
exdata=[];
for i=1:length(signal3)
  for rep=1:2
  exdata3= [exdata3 signal3(i)];
  end
end

signal4= s(4,:);
% first expand the bit stream
exdata=[];
for i=1:length(signal4)
  for rep=1:2
  exdata4= [exdata4 signal4(i)];
  end
end

signal5= s(5,:);
% first expand the bit stream
exdata=[];
for i=1:length(signal5)
  for rep=1:2
  exdata5= [exdata5 signal5(i)];
  end
end

signal6= s(6,:);
% first expand the bit stream
exdata=[];
for i=1:length(signal6)
  for rep=1:2
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
  exdata6= [exdata6 signal6(i)];
  end
end

% Bpsk modulation
m=10*n;
% Generating the carrier signal
ts=.1;
tp=1:ts:11.79;

% Generating the modulated signal 1
carrier1=cos(2*pi*tp);
bpsk_sig1=exdata1.*carrier1;
% Generating the modulated signal 2
carrier2=cos(4*pi*tp);
bpsk_sig2=exdata2.*carrier2;
% Generating the modulated signal 3
carrier3=cos(6*pi*tp);
bpsk_sig3=exdata3.*carrier3;
% Generating the modulated signal 4
carrier4=cos(8*pi*tp);
bpsk_sig4=exdata4.*carrier4;
% Generating the modulated signal 5
carrier5=cos(10*pi*tp);
bpsk_sig5=exdata5.*carrier5;
% Generating the modulated signal 6
carrier6=cos(12*pi*tp);
bpsk_sig6=exdata6.*carrier6;


% taking the iFFT of each of these signals
if_sig1=ifft(bpsk_sig1);
if_sig2=ifft(bpsk_sig2);
if_sig3=ifft(bpsk_sig3);
if_sig4=ifft(bpsk_sig4);
if_sig5=ifft(bpsk_sig5);
if_sig6=ifft(bpsk_sig6);

fin(1,:)=if_sig1;
fin(2,:)=if_sig2;
fin(3,:)=if_sig3;
fin(4,:)=if_sig4;
fin(5,:)=if_sig5;
fin(6,:)=if_sig6;

transmit=reshape(fin,1,648);


% generating the noise
% p=rand(1,800)*2*pi;
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
p=rand*2*pi;
snr=10;
r=sqrt(-1*(1/snr*log(1 - rand)));
% no = 5*(r.* exp(j*p));
no = (r.* exp(j*p));
% value of alpha
al=rand+j*rand;
% al=1;
% Spreading channel with the alpha as the variable

for k=2:2:646
   for l = 1:2
   % al=round(rand)+j*round(rand)
    rec(k+l)=transmit(k+l)+al*transmit(k-2+l);
   end
end

rxdata=rec+ no ;

% Converting from serial to parallel
myrec=reshape(rxdata,6,108)

rxdata1=fft(myrec(1,:));
rxdata2=fft(myrec(2,:));
rxdata3=fft(myrec(3,:));
rxdata4=fft(myrec(4,:));
rxdata5=fft(myrec(5,:));
rxdata6=fft(myrec(6,:));

% taking the FFT


% begin demodulation
% first multiply recieved bitstream by cosine wave with carrier frequency


uncarry1=rxdata1.*carrier1;
uncarry2=rxdata2.*carrier2;
uncarry3=rxdata3.*carrier3;
uncarry4=rxdata4.*carrier4;
uncarry5=rxdata5.*carrier5;
uncarry6=rxdata6.*carrier6;

% plot(uncarry)
% demodulate by integrating
dec1=[];
dec2=[];
dec3=[];
dec4=[];
dec5=[];
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
dec6=[];
% 1
for inc=1:2:length(uncarry1)
  dec=trapz(inc:inc+1,uncarry1(inc:inc+1));
  dec1=[dec1 dec];
end
% 2
for inc=1:2:length(uncarry2)
  dec=trapz(inc:inc+1,uncarry2(inc:inc+1));
  dec2=[dec2 dec];
end
% 3
for inc=1:2:length(uncarry3)
  dec=trapz(inc:inc+1,uncarry3(inc:inc+1));
  dec3=[dec3 dec];
end
% 4
for inc=1:2:length(uncarry4)
  dec=trapz(inc:inc+1,uncarry4(inc:inc+1));
  dec4=[dec4 dec];
end
% 5
for inc=1:2:length(uncarry5)
  dec=trapz(inc:inc+1,uncarry5(inc:inc+1));
  dec5=[dec5 dec];
end
% 6
for inc=1:2:length(uncarry6)
  dec=trapz(inc:inc+1,uncarry6(inc:inc+1));
  dec6=[dec6 dec];
end
final_rec(1,:)=dec1;
final_rec(2,:)=dec2;
final_rec(3,:)=dec3;
final_rec(4,:)=dec4;
final_rec(5,:)=dec5;
final_rec(6,:)=dec6;

fin_rec_parallel=reshape(final_rec,1,324)
% make decision with a threshold of zero
demod=[];
for i=1:length(fin_rec_parallel)
   if fin_rec_parallel(i)>0
      demod=[demod 1];
   else
      demod=[demod -1];
   end
end

% stem(demod)
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
% calculate errors
error=0;
for i=1:length(demod)
   if data(i)~=demod(i)
      error=error+1;
   end
end
error;

ber=error/324
figure(3)
stem(data)
hold
stem(demod,'rx')
```

logmapo.m

```
function L_all = logmapo(rec_s,g,L_a,ind_dec)
% Log_MAP algorithm using straightforward method to compute branch
metrics
% no approximation is used.
% Can be simplified to Max-Log-MAP by using approximation ln(e^x+e^y) =
max(x,y).
% Input: rec_s: scaled received bits.
% rec_s = 0.5 * L_c * yk = ( 2 * a * rate * Eb/N0 ) * yk
% g: code generator for the component RSC code, in binary matrix form.
% L_a: a priori info. for the current decoder,
% scrambled version of extrinsic Inftyo. of the previous decoder.
% ind_dec: index of decoder. Either 1 or 2.
% Encoder 1 is assumed to be terminated, while encoder 2 is open.
%
% Output: L_all: log-likelihood ratio of the symbols. Complete information.
% Total number of bits: Inftyo. + tail
L_total = length(rec_s)/2;
[n,K] = size(g);
m = K - 1;
nstates = 2^m; % number of states in the trellis
% Set up the trellis
[next_out, next_state, last_out, last_state] = trellis(g);
Infty = 1e10;
% Initialization of Alpha
Alpha(1,1) = 0;
Alpha(1,2:nstates) = -Infty*ones(1,nstates-1);
% Initialization of Beta
if ind_dec==1
   Beta(L_total,1) = 0;
   Beta(L_total,2:nstates) = -Infty*ones(1,nstates-1);
elseif ind_dec==2
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
    Beta(L_total,1:nstates) = zeros(1,nstates);
else
    fprintf('ind_dec is limited to 1 and 2!\n');
end
% Trace forward, compute Alpha
for k = 2:L_total+1
    for state2 = 1:nstates
        gamma = -Infty*ones(1,nstates);
        gamma(last_state(state2,1)) = (-rec_s(2*k-3)+rec_s(2*k-
2)*last_out(state2,2))....
            -log(1+exp(L_a(k-1)));
        gamma(last_state(state2,2)) = (rec_s(2*k-3)+rec_s(2*k-
2)*last_out(state2,4))....
            +L_a(k-1)-log(1+exp(L_a(k-1)));
        if(sum(exp(gamma+Alpha(k-1,:)))<1e-300)
            Alpha(k,state2)=-Infty;
        else
            Alpha(k,state2) = log( sum( exp( gamma+Alpha(k-1,:) ) ) );
        end
    end
    tempmax(k) = max(Alpha(k,:));
    Alpha(k,:) = Alpha(k,:) - tempmax(k);
end
% Trace backward, compute Beta
for k = L_total-1:-1:1
    for state1 = 1:nstates
        gamma = -Infty*ones(1,nstates);
        gamma(next_state(state1,1)) = (-
rec_s(2*k+1)+rec_s(2*k+2)*next_out(state1,2))....
            -log(1+exp(L_a(k+1)));
        gamma(next_state(state1,2)) =
(rec_s(2*k+1)+rec_s(2*k+2)*next_out(state1,4))....
            +L_a(k+1)-log(1+exp(L_a(k+1)));
        if(sum(exp(gamma+Beta(k+1,:)))<1e-300)
            Beta(k,state1)=-Infty;
        else
            Beta(k,state1) = log(sum(exp(gamma+Beta(k+1,:))));
        end
    end
    Beta(k,:) = Beta(k,:) - tempmax(k+1);
end
% Compute the soft output, log-likelihood ratio of symbols in the frame
for k = 1:L_total
    for state2 = 1:nstates
        gamma0 = (-rec_s(2*k-1)+rec_s(2*k)*last_out(state2,2))....
            -log(1+exp(L_a(k)));
        gamma1 = (rec_s(2*k-1)+rec_s(2*k)*last_out(state2,4))...
            +L_a(k)-log(1+exp(L_a(k)));
        temp0(state2) = exp(gamma0 + Alpha(k,last_state(state2,1)) +
Beta(k,state2));
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
        temp1(state2) = exp(gamma1 + Alpha(k,last_state(state2,2)) +
Beta(k,state2));
    end
    L_all(k) = log(sum(temp1)) - log(sum(temp0));
End
```

int_state.m

```
function int_state = int_state( state )
% converts a row vector of m bits into a integer (base 10)
[dummy, m] = size( state );
for i = 1:m
    vect(i) = 2^(m-i);
end
int_state = state*vect';
```

increseber.m

```
% Vinay Mohan Jonnakuti
% Wireless Communication:
% BPSK modulation and demodulation with increased bit rate
% BPSK simulation using a carrier cosine wave with increased bit rate
clc;
close all;
clear all;
figure(1)
n=320;
% n=324; in ofdm_ber.m
% n=160; in BPSKmodanddemod.m
for i=1:n
    data(i)= 2*round(rand)-1;
end
% create modulated BPSK signal
% first expand the bit stream
exdata=[];     % exdata1=[]; exdata2=[]; exdata3=[]; .... in ofdm_ber.m
for i=1:length(data)     % for i=1:length(signal1 signal2 ....) in ofdm_ber.m
  for rep=1:2
  % for rep=1:5 in BPSKmodanddemod.m
  exdata= [exdata data(i)];     % exdata1 exdata2 ....=[exdata1 exdata2
....(data(i)] in ofdm_ber.m
  end
end

exdata=data;
ts=.1;
t=1:ts:32.9;
% tp=1:ts:11.79; in ofdm_ber.m
% t=1:ts:80.9; in BPSKmodanddemod.m
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
carrier=cos(pi*t);
% carrier1=cos(2*pi*tp); carrier2=cos(4*pi*tp); ..... in ofdm_ber.m

% multiply expanded bitstream by cosine wave with carrier frequency
% this is the BPSK that is to be transmitted over the channel
bpsk=carrier.*exdata;
% bpsk_sig1=exdata1.*carrier1; bpsk_sig2=exdata2.*carrier2; ..... in
ofdm_ber.m

% bpsk=[bpsk(length(bpsk)-1) bpsk(length(bpsk)) bpsk];
subplot(313)
plot(bpsk)


% generating the noise
% p=rand(1,800)*2*pi;
p=rand*2*pi;
snr=10;
r=sqrt(-1*(1/snr*log(1 - rand)));
% no = 5*(r.* exp(j*p));
no = (r.* exp(j*p));
% value of alpha
al=rand+j*rand;
% al=1;
% Spreading channel with the alpha as the variable

for k=5:5:315
   for l = 1:5
   % al=round(rand)+j*round(rand)
    rec(k+l)=bpsk(k+l)+al*bpsk(k-2+l);
   end
end

rxdata=rec+ no ;

% begin demodulation
% first multiply recieved bitstream by cosine wave with carrier frequency
figure(2)

uncarry=rxdata.*carrier;

% plot(uncarry)
% demodulate by integrating
dec1=[];
for inc=1:2:length(uncarry)
  dec=trapz(inc:inc+1,uncarry(inc:inc+1));
  dec1=[dec1 dec];
end

% make decision with a threshold of zero
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
demod=[];
for i=1:length(dec1)
   if dec1(i)>0
      demod=[demod 1];
   else
      demod=[demod -1];
   end
end

% stem(demod)

% calculate errors
error=0;
for i=1:length(demod)
   if data(i)~=demod(i)
      error=error+1;
   end
end
error;

data1=data(1:160)
demod4=demod(1:160)
ber=error/n
figure(3)
stem(data1)
hold
stem(demod4,'rx')
```

encoderm.m

```
unction en_output = encoderm( x, g, alpha, puncture )
% uses interleaver map 'alpha'
% if puncture = 1, unpunctured, produces a rate 1/3 output of fixed length
% if puncture = 0, punctured, produces a rate 1/2 output
% multiplexer chooses odd check bits from RSC1
% and even check bits from RSC2
% determine the constraint length (K), memory (m)
% and number of information bits plus tail bits.

% n encoded data bits, where 1/n is the code rate.

[n,K] = size(g);
m = K - 1;
L_info = length(x);
L_total = L_info + m;
% generate the codeword corresponding to the 1st RSC coder
% end = 1, perfectly terminated;
input = x;
output1 = rsc_encode(g,input,1);
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
% make a matrix with first row corresponing to info sequence
% second row corresponsing to RSC #1's check bits.
% third row corresponsing to RSC #2's check bits.
y(1,:) = output1(1:2:2*L_total);
y(2,:) = output1(2:2:2*L_total);
% interleave input to second encoder
for i = 1:L_total
   input1(1,i) = y(1,alpha(i));
end
output2 = rsc_encode(g, input1(1,1:L_total), -1 );
y(3,:) = output2(2:2:2*L_total);
% paralell to serial multiplex to get output vector
% puncture = 0: rate increase from 1/3 to 1/2;
% puncture = 1; unpunctured, rate = 1/3;
if puncture > 0 % unpunctured
   for i = 1:L_total
      for j = 1:3
         en_output(1,3*(i-1)+j) = y(j,i);
      end
   end
else % punctured into rate 1/2
   for i=1:L_total
      en_output(1,n*(i-1)+1) = y(1,i);
      if rem(i,2)
         % odd check bits from RSC1
         en_output(1,n*i) = y(2,i);
      else
         % even check bits from RSC2
         en_output(1,n*i) = y(3,i);
      end
   end
end
% antipodal modulation: +1/-1
% en_output = 2 * en_output - ones(size(en_output));
```

Encode_bit.m

```
function [output, state] = encode_bit(g, input, state)
% This function takes as an input a single bit to be encoded,
% as well as the coeficients of the generator polynomials and
% the current state vector.
% It returns as output n encoded data bits, where 1/n is the
% code rate.
% the rate is 1/n
% k is the constraint length
% m is the amount of memory
[n,k] = size(g);
m = k-1;
% determine the next output bit
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14

```
for i=1:n
    output(i) = g(i,1)*input;
    for j = 2:k
        output(i) = xor(output(i),g(i,j)*state(j-1));
    end;
end
state = [input, state(1:m-1)];
```


Institutional Repository - Library & Information Centre - University of Thessaly
12/09/2026 16:17:19 EEST - 216.73.216.14