Chapter 4
Principles of OFDM

4.1 Introduction

Orthogonal Frequency Division Multiplexing (OFDM) is a multicarrier transport technology for high data rate communication system. The OFDM concept is based on spreading the high speed data to be transmitted over a large number of low rate carriers. The carriers are orthogonal to each other and frequency spacing between them are created by using the Fast Fourier transform (FFT).\(^1\)

OFDM originates from Frequency Division Multiplexing (FDM), in which more than one low rate signal is carried over separate carrier frequencies (Table 4.1). In FDM, separation of signal at the receiver is achieved by placing the channels sufficiently far apart so that the signal spectra does not overlap. Of course, the resulting spectral efficiency is very low as compared with OFDM, where a comparison is depicted in Fig. 4.1. Also, Fig. 4.2 shows an analogy of OFDM against single carrier and FDM in terms of spectral efficiency.

FDM is first utilized to carry high-rate signals by converting the serial high-rate signal into parallel low bit streams. Such a parallel transmission scheme when compared with high-rate single carrier scheme is costly to build. On the other hand, high-rate single carrier scheme is more susceptible to intersymbol interference (ISI). This is due to the short duration of the signal and higher distortion by its wider frequency band as compared with the long duration signal and narrow bandwidth subchannels in the parallel system.

The major contribution to the FDM complexity problem was the application of the FFT to the modulation and demodulation processes. Fortunately, this occurred at the same time digital signal processing techniques were being introduced into the design of modems.

\(^1\) Originally Weinstein and Ebert introduced discrete Fourier transform (DFT) to create orthogonal waveforms. FFT is an efficient implementation of DFT and become defacto method with advanced very-large-scale integration (VLSI) technology. FFT reduces the number of multiplication from \(N^2\) to \(N/2\log N\) for radix-2 and \(3N/8\log_2(N - 2)\) for radix-4 schemes, where \(N\) is the number of orthogonal channels. Typically complexity of additions that is necessary is not significant compared with multiplication complexity.

Table 4.1 OFDM history (source: Wikipedia)

<table>
<thead>
<tr>
<th>Year</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1957</td>
<td>Kineplex, multicarrier HF modem</td>
</tr>
<tr>
<td>1966</td>
<td>Chang, Bell Labs: OFDM paper and US patent 3488445</td>
</tr>
<tr>
<td>1971</td>
<td>Weinstein and Ebert proposed use of FFT and guard interval</td>
</tr>
<tr>
<td>1985</td>
<td>Cimini described the use of OFDM for mobile communications</td>
</tr>
<tr>
<td>1985</td>
<td>Telebit Trailblazer Modem incorporates a 512-carrier Packet Ensemble Protocol</td>
</tr>
<tr>
<td>1987</td>
<td>Alard and Lasalle: Coded OFDM for digital broadcasting</td>
</tr>
<tr>
<td>1988</td>
<td>TH-CSF LER, first experimental Digital TV link in OFDM, Paris area</td>
</tr>
<tr>
<td>1989</td>
<td>OFDM international patent application PCT/FR 89/00546, filed in the name of THOMSON-CSF, Fouche, de Couasnon, Travert, Monnier and others</td>
</tr>
<tr>
<td>1990</td>
<td>TH-CSF LER, first OFDM equipment field test, 34 Mbps in a 8-MHz channel, experiments in Paris area</td>
</tr>
<tr>
<td>1990</td>
<td>TH-CSF LER, first OFDM test bed comparison with VSB in Princeton, USA</td>
</tr>
<tr>
<td>1992</td>
<td>TH-CSF LER, second generation equipment field test, 70 Mbit/s in a 8-MHz channel, twin polarizations, Wuppertal, Germany</td>
</tr>
<tr>
<td>1992</td>
<td>TH-CSF LER, second generation field test and test bed with BBC, near London, UK</td>
</tr>
<tr>
<td>1993</td>
<td>TH-CSF show in Montreux SW, 4 TV channel and one HDTV channel in a single 8-MHz channel</td>
</tr>
<tr>
<td>1993</td>
<td>Morris: Experimental 150 Mbit/s OFDM wireless LAN</td>
</tr>
<tr>
<td>1994</td>
<td>US patent 5282222, method and apparatus for multiple access between transceivers in wireless communications using OFDM spread spectrum</td>
</tr>
<tr>
<td>1995</td>
<td>ETSI Digital Audio Broadcasting standard Eureka: First OFDM-based standard</td>
</tr>
<tr>
<td>1997</td>
<td>ETSI DVB-T standard</td>
</tr>
<tr>
<td>1998</td>
<td>Magic WAND project demonstrates OFDM modems for wireless LAN</td>
</tr>
<tr>
<td>1999</td>
<td>IEEE 802.11a wireless LAN standard (Wi-Fi)</td>
</tr>
<tr>
<td>2000</td>
<td>Proprietary fixed wireless access (V-OFDM, Flash-OFDM, etc.)</td>
</tr>
<tr>
<td>2002</td>
<td>IEEE 802.11g standard for wireless LAN</td>
</tr>
<tr>
<td>2004</td>
<td>IEEE 802.16-2004 standard for wireless MAN (WiMAX)</td>
</tr>
<tr>
<td>2004</td>
<td>ETSI DVB-H standard</td>
</tr>
<tr>
<td>2004</td>
<td>Candidate for IEEE 802.15.3a standard for wireless PAN (MB-OFDM)</td>
</tr>
<tr>
<td>2004</td>
<td>Candidate for IEEE 802.11n standard for next-generation wireless LAN</td>
</tr>
<tr>
<td>2005</td>
<td>OFDMA is candidate for the 3GPP Long Term Evolution (LTE) air interface E-UTRA downlink</td>
</tr>
<tr>
<td>2007</td>
<td>The first complete LTE air interface implementation was demonstrated, including OFDM-MIMO, SC-FDMA and multi-user MIMO uplink</td>
</tr>
</tbody>
</table>

Fig. 4.1 Comparison of OFDM and FDM
Fig. 4.2 Comparison of OFDM over FDM and single-carrier systems. OFDM and FDM are resilient to interference, since flow of water can be easily stopped in single-carrier systems. OFDM is more spectral efficient than FDM, since it utilizes the surface effectively with adjacent tiny streams.

![Diagram](image)

Fig. 4.3 A very basic OFDM system

![Diagram](image)

Fig. 4.4 Spectrum of OFDM signal

The technique involved assembling the input information into blocks of \( N \) complex numbers, one for each sub-channel as seen in Fig. 4.3. An inverse FFT is performed on each block, and the resultant transmitted serially. At the receiver, the information is recovered by performing an FFT on the received block of signal samples.

The spectrum of the signal on the line is identical to that of \( N \) separate QAM signals as seen in Fig. 4.4, where \( N \) frequencies separated by the signalling rate.
Fig. 4.5 OFDM spectrum for each QAM signal

Each QAM signal carries one of the original input complex numbers. The spectrum of each QAM signal is of the form \( \frac{\sin(kf)}{f} \), with nulls at the center of the other subcarriers as seen in Fig. 4.5. This ensures orthogonality of subcarriers.

However, orthogonality is threatened by intersymbol interference (ISI), which is caused by leakage of symbols into another due to multipath interference. To combat for ISI, a guard time is introduced before the OFDM symbol. Guard time is selected longer than impulse response or multipath delay so as not to cause interference of multipath components of one symbol with the next symbol.

Orthogonality is also threatened by intercarrier interference (ICI), which is crosstalk between subcarriers, since now the multipath component of one subcarrier can disturb the another one. ICI in OFDM is prevented by cyclically extending the guard interval as seen in Fig. 4.6 to ensure integer number of cycles in the symbol time as long as the delay is smaller than the guard time.

Another issue is how to transmit the sequence of complex numbers from the output of the inverse FFT over the channel. The process is straightforward if the signal is to be further modulated by a modulator with I and Q inputs as in Fig. 4.7.

Otherwise, it is necessary to transmit real quantities. This can be accomplished by first appending the complex conjugate to the original input block. A 2N-point inverse FFT now yields 2N real numbers to be transmitted per block, which is equivalent to N complex numbers.

OFDM increases the robustness against frequency selective fading or narrowband interference due to narrowband flat fading subchannels. As compared with single carrier system, a single fade or interferer can cause the entire link to fail, but in
4.1 Introduction

Fig. 4.6 OFDM with cyclic shift

Fig. 4.7 Real and Imaginary components of an OFDM symbol: The superposition of several harmonics modulated by data symbols
OFDM, since there are several subcarriers, only small percentage of the subcarriers is affected. Error correction coding is used to correct the erroneous subcarriers.

OFDM on the other hand suffers from noise such as amplitude with a very large dynamic range; therefore, it requires RF power amplifiers with a high peak to average ratio. It is also more sensitive to carrier frequency offset than single carrier systems are due to leakage of the FFT.

OFDM has been particularly successful in numerous wireless applications, where its superior performance in multipath environments is desirable. Wireless receivers detect signals distorted by time and frequency selective fading. OFDM in conjunction with proper coding and interleaving is a powerful technique for combating the wireless channel impairments that a typical OFDM wireless system might face.

4.2 A simple OFDM system

Let us consider Fig. 4.8 as a simple OFDM system to understand the mechanics behind it. The incoming data is converted from serial to parallel and grouped into bits each to form a complex number $x$ after PSK or QAM modulation in order to be transmitted over $N$ low-rate data streams. Each low-rate data stream is associated with a subcarrier of the form

$$
\phi_k(t) = e^{j2\pi f_k t},
$$

(4.1)

where $f_k$ is the frequency of the $k$th subcarrier. Consequently, one baseband OFDM symbol with $N$ subcarrier is

![Fig. 4.8 Simplified OFDM system](image-url)
4.2 A simple OFDM system

Fig. 4.9 An example of four subcarriers in time and frequency with same modulation

\[ s(t) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} x_k \phi_k(t), \quad 0 < t < T, \]

where \( x_k \) is the \( k \)th complex data symbol and \( T \) is the length of the OFDM symbol.

Now, let us look at the constructed OFDM symbol in detail to analyze the orthogonality of subcarriers. Consider four subcarriers \( (N = 4) \), and first assume same modulation for each subcarrier. Figure 4.9 shows those four subcarriers in time and frequency. Also consider Fig. 4.10 for an example to highlight the effect of different modulation in each subcarrier.

Orthogonality of OFDM subcarriers in frequency domain is with Dirac pulses convolved with \( \text{sinc}(\pi fT) \). Since in time domain a subcarrier \( \phi_k \) is multiplied with a rectangle\((T)\), which is in frequency domain a convolution between \( \delta(f - f_k) \) and \( \text{sinc}(\pi fT) \). This is basically \( 1/T \) shifted version of \( \text{sinc}(f) \) for each \( f_k \) and \( \text{sinc}(\pi fT) \) has zeros for all frequencies that are integer multiple of \( 1/T \).

Notice that the orthogonality of OFDM subcarriers can be demonstrated in time domain as well. Within a symbol time \( (T) \), there are integer number of cycles in the symbol interval, and the number of cycles between adjacent subcarriers differs exactly by one (see first subfigures in Figs. 4.9 and 4.10). In the receiver when it is demodulated, down converted, and integrated with a frequency \( j/T \), then the \( x_j \) is received since any other subcarrier when it is down converted with a frequency \((i - j)/T \) produces zero after integration since \((i - j)/T \) produces integer number of cycles within the integration interval \( (T) \).

When signal is transmitted over a channel, channel dispersion destroys the orthogonality between subcarriers and causes ICI, and delay spread causes ISI between
successive OFDM symbols. As we mentioned before, cyclic prefix (CP) is used to preserve the orthogonality and avoid ISI. We will see that this makes equalization in the receiver very simple. If multipath exceeds the CP, then constellation points in the modulation is distorted. As can be seen from Fig. 4.11, when multipath delay exceeds the CP, the subcarriers are not guaranteed to be orthogonal anymore, since modulation points may fall into anywhere in the respective contour. As delay spread gets more severe, the radius of the contour enlarges and crosses the other contours. Hence, this causes error.

The CP is utilized in the guard period between successive blocks and constructed by the cyclic extension of the OFDM symbol over a period $\tau$:

$$s(t) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} x_k \phi_k(t), \quad -\tau < t < NT. \quad (4.3)$$
The required criteria is that $\tau$ is chosen bigger than channel length $\tau_h$ so as not to experience an ISI. The CP requires more transmit energy and reduces the bit rate to $(Nb/NT + \tau)$, where $b$ is the bits that a subcarrier can transmit.

The CP converts a discrete time linear convolution into a discrete time circular convolution. Thus, transmitted data can be modeled as a circular convolution between the channel impulse response and the transmitted data block, which in the frequency domain is a pointwise multiplication of DFT samples. Then received signal becomes

$$y(t) = s(t) * h(t) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} H_k x_k \phi_k(t), \quad 0 < t < NT,$$

where

$$H_k = \int_0^{\tau_h} h(t) e^{j2\pi f_k t} dt. \quad (4.5)$$

Hence, $k$th subcarrier now has a channel component $H_k$, which is the fourier transform of $h(t)$ at the frequency $f_k$.

The OFDM symbol is sampled ($t = nT$ and $f_k = k/NT$) in the receiver and demodulated with an FFT. Consequently, the received data has the following form

$$y_k = H_k x_k, \quad k = 0, \ldots, N - 1. \quad (4.6)$$

The received actual data can be retrieved with $N$ parallel one-tap equalizers. One-tap equalizer simply uses the estimated channel ($\hat{H}_k$) components and use it to retrieve estimated $\hat{x}_k$ as follows

$$\hat{x}_k = \frac{y_k}{\hat{H}_k} = \frac{H_k}{\hat{H}_k} x_k. \quad (4.7)$$

Also note that the spectrum of OFDM decays slowly. This causes spectrum leakage to neighboring bands. Pulse shaping is used to change the spectral shape by either commonly used raised cosine time window or passing through a filter.

An OFDM system design considers setting the guard interval ($\tau$) as well as the symbol time ($T$) and FFT size with respect to desired bit rate $B$ and given tolerable delay spread. The guard interval is selected according to delay spread, and typically it is 2–4 times the root-mean-squared delay spread with respect to chosen coding and modulation.

Symbol time is set with respect to guard time and it is desirable to select much larger than the guard time since the loss in SNR in the guard time is compensated. Symbol time as we know determines the subcarrier spacing ($f_b = 1/T$). Number of subcarriers $N$ is found with respect to desired bit rate, since total number of bits ($b_T$) to carry in one symbol is found with $B/(T + \tau)$ and selected coding and modulation determines the number of bits ($b$) in one subcarrier. Hence, the number of subcarriers is $N = b_T/b$. For instance, $b$ is two for 16QAM with rate 1/2. The required bandwidth ($W$) is then $N \times f_b$. Alternatively, this method is reversed to find out the symbol time starting from the given bandwidth.

This section described how the basic OFDM transceiver is formed. However, there are more components to build a complete OFDM transceiver. Figure 4.12
4.3 Coding

In the previous section, we gave a simple uncoded OFDM system and highlighted the key features. In a multipath environment, all subcarriers will experience different fading environment and all will arrive with different amplitudes. Some of them will experience a deep fade, which will cause error during detection, and the average probability of error will be the same as that for a flat fading single-carrier system with the same average geometric mean of the subcarriers’ SNRs. These errors may dominate the bit error rate. To avoid this domination, error correction coding is utilized. Coding used in OFDM systems is to correct the certain number of errors in order to enable high rates in the presence of fading and interference from other wireless channels.

2 “Why use error coding? Error coding may be selected to improve data reliability, reduce system costs, or increase the range. For instance, 3 dB coding gain can

- increase throughput 2-fold or
- increase range by 40% or
- reduce bandwidth by 50% or
- reduce antenna size by 30% or
- reduce transmitter power by half.”
Fig. 4.13 Two-dimensional coding for OFDM with respect to channel impulse response

This section starts with an introduction of block and convolutional coding. Then, we introduce concatenated coding as a way to combine different coding schemes to further reduce the error rate. A combination of block coding and convolutional coding along with proper time and frequency interleaving as seen in Fig. 4.13 constitutes such a concatenated coding strategy to achieve better frequency diversity.\(^3\)

In the past several years, convolutional coding with Viterbi decoding has begun to be supplemented in the geostationary satellite communication arena with Reed-Solomon coding. The two coding techniques are usually implemented serially as concatenated block and convolutional coding. Typically, the information to be transmitted is first encoded with the Reed-Solomon code, then with the convolutional code. On the receiving end, Viterbi decoding is performed first, followed by Reed-Solomon decoding. This is the technique that is used in most if not all of the direct-broadcast satellite (DBS) systems, and in cellular communication as well.

Later, we review some fundamental features of turbo coding and explain why it is a good potential candidate for OFDM systems. Turbo coding is a new parallel-concatenated convolutional coding technique. Initial hardware encoder and decoder implementations of turbo coding have already appeared on the market. This

\(^3\) But achievable frequency diversity is limited by number of resolvable independent paths in the channel impulse response. Intuitive explanation is as follows: if the channel is one-tap, the SNR on all subcarriers is the same since channel length is small compared with the OFDM symbol that makes the subcarries correlated. If the number of resolvable taps increases, the correlation between subcarriers decreases and diversity increases. But resolvable number of taps are limited, and therefore subcarriers cannot be independent.
technique achieves substantial improvements in performance over concatenated Viterbi and Reed-Solomon coding. Turbo coding has recently been used in many communication systems successfully. Random-like structure of turbo codes have resulted in outstanding performance by providing small error rates at information rates of close to theoretical channel capacity.

Finally, we conclude the section with Trellis coding, which is a coding technique that also leverages modulation and low-density parity check (LDPC) coding, which is becoming popular in cellular communication and included as a supported coding scheme in several standards.

4.3.1 Block Coding

Block coding is a way of mapping \( k \) symbols to \( n \) symbols with \( n > k \). We call the block of \( k \) symbols a messageword, and the block of \( n \) symbols a codeword. The process of mapping \( k \) message symbols to \( n \) code symbols is called encoding, and the reverse process is called decoding.

This mapping can be systematic (Fig. 4.14), wherein the block of \( n \) codeword symbols consists of the \( k \) messageword symbols plus \( (n - k) \) added redundant symbols, or nonsystematic, where the messageword symbols cannot be directly recovered from the codeword without decoding. Generally, any linear block code can be represented as an “equivalent” systematic code. A block code is called “linear” if the sum of two codewords is always a valid codeword, and a scalar multiple of any codeword is also a valid codeword.

Block codes, in particular the Reed-Solomon class, are used to combat for bursty errors. Burst errors occur in single-carrier systems when the impulsive noise has duration greater than a symbol period and coherence time of the channel is longer than symbol period.

In OFDM system, an impulsive noise with a wide-frequency content causes burst error to affect several adjacent subcarriers if coherence bandwidth of the channel is wider than the subcarrier spacing.

![Fig. 4.14 Construction of a systematic block code](image-url)
4.3 Coding

4.3.1.1 Interleaving

The performance of codes in the presence of bursts can be improved by the process of interleaving. To separate the correlated components, the code is made to operate on symbols with sufficient spacing so that the errors are more independent. At the receiver, the symbols are de-interleaved before decoding. The decoder therefore operates on symbols spaced some symbol periods apart as transmitted. Figure 4.15 shows a block interleaving method for transmitter and receiver. This interleaver writes by row and reads by column. Pseudo-random interleaver on the other hand reads the information bits to the encoder in a random but fixed order.

Interleaving can be in time, frequency, or both. For instance, for time interleaving, channel coherence time is on the order of 10–100 s of symbols, which makes the channel highly correlated across adjacent symbols. Interleaving makes sure that symbols especially adjacent symbols experience nearly independent fading gains.

4.3.1.2 Cyclic Redundancy Check

Cyclic redundancy check (CRC) is a simple form of block coding for error detection. Fixed number of check bits follows messageword. If the receiver detects an error, a retransmission is requested. Figure 4.16 shows a typical encoder implementation of CRC with an \( n \)-bit feedback shift register whose connection pattern is a primitive

\[
P(x) = x^{16} + x^{15} + x^2 + 1
\]

Fig. 4.15 Implementation of block interleaving

Fig. 4.16 CRC-16 implementation: \( P(x) = x^{16} + x^{15} + x^2 + 1 \)
polynomial. At the encoder, shift register starts with a predetermined pattern, and input data is fed both to the channel and to the feedback shift register. At the receiver, received data and output is concomitantly fed back to the register. A CRC of length \( n \) can detect any error pattern of length less than \( n \) with probability \( 1 - 2^{-n} \).

Let’s now look at how coding is used to correct errors. In brief, the idea behind error correction coding is to start with a “message” (i.e., the thing you want to encode) of length \( k \), and convert it to a “codeword” of longer length \( n \), in such a way that the additional information in the coded form allows one to recover the original message if parts of it are corrupted. To see how this works, we will need some additional definitions:

### 4.3.1.3 Hamming weight

The Hamming weight of a codeword is a metric that indicates the number of nonzero symbols in codeword. For example, if the codeword is 10010001, its weight would be 3. If the codeword is the nonbinary 23012001, then the weight would be 5. It works the same way regardless off the base field.

### 4.3.1.4 Hamming Distance

The Hamming distance is a comparison metric between two codewords by the number of places where the codewords differ. So, for example, given the two binary codewords 100111 and 110000, the Hamming distance between them would be 4.

### 4.3.1.5 Minimum distance of a code

The minimum distance of a code is another metric that typically gives a characteristic of the code by measuring the minimum distance between all the codewords in the code. This is achieved by taking the distance between each codeword and every other codeword in the code, and the minimum gives the minimum distance of the code. For linear codes, minimum distance equals the lowest hamming weight in the code. Error correction capability of the code is highly correlated with the minimum distance of a code. Consider a simple binary repetition code of length 4, where there are two codewords (1111) and (0000). The minimum distance is 4, since minimum hamming weight is 4. Suppose we send (1111):

<table>
<thead>
<tr>
<th>Transmitted</th>
<th>Received</th>
<th>Hamming distance to (1111)</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>1111</td>
<td>0111</td>
<td>1</td>
<td>1111</td>
</tr>
<tr>
<td>1111</td>
<td>1010</td>
<td>2</td>
<td>Fail</td>
</tr>
<tr>
<td>1111</td>
<td>0001</td>
<td>3</td>
<td>0000</td>
</tr>
</tbody>
</table>
If the received bits have equal hamming distance to two code words, then decoder cannot decode it and fails. Otherwise, decoder decides for the codeword that has the smallest distance away from the received word. This type of decoding can be generalized to much larger and more sophisticated codes, and for a minimum distance \( d \) we can correct up to distance \( \text{floor}(\frac{(d - 1)}{2}) \).

### 4.3.2 Reed-Solomon Coding

Reed-Solomon (RS) codes are a very popular block coding technique of today as compared with evolving capacity-approaching codes. RS is easy to decode and suits best for high-rate systems with small data packets.

#### 4.3.2.1 Cyclic Codes

RS codes are cyclic codes in addition to being linear block codes. A cyclic code preserves the property of being cyclic in the sense that when shifted the result is still a codeword. For instance, if codeword \((0111)\) is shifted one digit, the result \((1011)\) is also a codeword. Consequently, all circular shifts of any codeword in the code are also codewords in the code. In polynomial terms, multiplication of cyclic codeword \(c(x)\) with \(x\) results in circular shift of the codeword \(x \cdot c(x) \mod (x^n - 1)\) and the following manipulations are possible:

\[
a_{n-1}x_{n-1}c(x) + a_{n-2}x_{n-2}c(x) + \cdots + a_0c(x),
\]

since the sum of two codewords is always a codeword in linear codes, and multiplication of a codeword by a scalar always results in a codeword. Notice that all operations must be done using Galois field (GF) arithmetic.\(^4\) For example, a binary code uses two values, the binary numbers \(\{0,1\}\), as symbols. In general, though, a code uses \(q\)-ary symbols. \(Q\)-ary symbols use symbols taken from an alphabet \(A\) of \(q\) possible values. So, for example, 5-ary symbols would be symbols chosen from a set of five elements, such as \(\{0, 1, 2, 3, 4\}\). Practical RS codes use \(q = 256\), since it can be represented using 8-bit symbols per codeword.

There are two common definitions of Reed-Solomon codes: as polynomial codes over finite fields and as cyclic codes of length \(q - 1\) over \(GF(q)\).

---

\(^4\) “A field is a set of elements that can be added, subtracted, multiplied, and divided, with the important stipulation that the result of any of those operations is always still an element of the field. Additionally, we require that additive and multiplicative inverses and identity elements exist for each (non-zero) element of the field, and that the field elements obey the familiar commutative, associative, and distributive properties. The real numbers are a familiar example of a field: we can add, subtract, multiply and divide any two (non zero) real numbers, and the result is always another real number. Multiplicative and additive inverses can be found for any real number; the multiplicative identity element is ‘1’, the additive identity element is ‘0’. The real numbers are an infinite field. We can construct a field with a finite number of elements, if we follow certain rules for constructing such fields.”
4.3.2.2 Polynomial Codes over Certain Finite Fields

The idea is very simple, there is a message \( m(x) \)

\[
m(x) = m_{k-1}x^{k-1} + m_{k-2}x^{k-2} + \cdots + m_1x + m_0
\]  

(4.9)

in the form of a polynomial whose coefficients \((m_i)\) are taken from finite field \(GF(q)\). Codeword is found by evaluating the \( m(x) \) at \( n \) distinct elements of the finite field:

\[
(c_0, c_1, c_2, \ldots, c_{n-1}) = m(a_0), m(a_1), m(a_2), \ldots, m(a_{n-1}),
\]  

(4.10)

where \( n \) distinct elements of the field are \( a_0, a_1, \ldots, a_{n-1} \). A generalization of the above construction leads to the definition of generalized Reed-Solomon (GRS) codes:

\[
(c_0, c_1, c_2, \ldots, c_{n-1}) = v_0m(a_0), v_1m(a_1), v_2m(a_2), \ldots, v_{n-1}m(a_{n-1}),
\]  

(4.11)

where \( v_0, v_1, \ldots, v_{n-1} \) be \( n \) nonzero (but not necessarily distinct) elements of \( GF(q) \).

If the message has \( k \) symbols, and the length of the code is \( n = q - 1 \), then the code consists of \( n \) equations in \( k \) unknowns, which is overspecified when \( n > k \), hence the correct coefficients can be recovered even if some of them are corrupted.

4.3.2.3 Generator Polynomial Approach

Given the message \( m(x) \) in the form of a polynomial, as outlined earlier, whose \( k \) coefficients are taken from the finite field with \( q \) elements, we can construct RS codewords with \( c(x) = m(x)g(x) \) (or the equivalent systematic construction). All we need to do is specify the generator polynomial of the code.

The general form of the generator polynomial of a RS code is defined in such a way as to have its roots \( 2t \) consecutive powers of a primitive element. Thus we can write,

\[
g(x) = (x - ab)(x - ab + 1)(x - ab + 2)\cdots(x - ab + 2t - 1).
\]  

(4.12)

For convenience, the constant \( b \) is often chosen to be 0 or 1. Given the generator polynomial, RS codewords can now be constructed as \( c(x) = m(x)g(x) \), where \( g(x) = (x - a^t)(x - a^{t+1})(x - a^{t+2t-1}) \) and \( m(x) \) are the information element. This method is often used in practice, since polynomial multiplication is relatively easy to implement in hardware. Therefore, an RS code with \( 2t \) check symbols can correct up to floor\((2t + 1 - 1)/2\) = \( t \) errors or \( 2t \) erasures. An erasure occurs when the position of an error symbol is known.

RS codes are the best minimum distance obtainable codes. Recall that minimum distance is the most important property of an error correction code. Since they are

---

5 “Original definition used by Irving S. Reed and Gustave Solomon in their paper “Polynomial codes over certain finite fields” published in the Journal of the Society for Industrial and Applied Mathematics in 1960.”
linear, cyclic, and their generator polynomial has well-defined roots, RS codes are easy to encode and relatively easy to decode. The coding gain with RS dictates that the probability of error correction in the decoded data is higher than the probability of error correction without Reed-Solomon. Hence, the probability of error in the remaining decoded data is lower.

A popular Reed Solomon code is RS(255,233) with 8-bit symbols, where \( n = 255 \), \( k = 223 \), and \( s = 8 \) and \( 2t = 32 \). The decoder can correct any 16 symbol errors in the codeword. Larger value of \( t \) means that larger value of errors can be corrected, but it requires more computational power.

### 4.3.3 Convolutional Coding

Convolutional coding is another famous coding technique that operates on serial streams of symbols rather than blocks. A convolutional encoder is usually described by two parameters: the code rate and the constraint length. The code rate is \( \frac{k}{n} \), where \( k \) is the number of bits into the convolutional encoder and \( n \) is the number of channel symbols output by the convolutional encoder in a given encoder cycle. The constraint length parameter, \( K \), denotes the “length” of the convolutional encoder, which denotes the number of stages and the cycles an input bit retains in the convolutional encoder.

Viterbi\(^6\) decoding or sequential decoding are used for convolutional encoding. Sequential decoding performs well with long-constraint-length convolutional codes, but it has a variable decoding time. Viterbi decoding on the other hand has a fixed decoding time. In hardware implementation it is preferable, but complexity of the algorithm increases exponentially with a function of constraint length. Viterbi also permits soft decision decoding, where the minimum Euclidean distance\(^7\) between the received sequence and all allowed sequences, rather than Hamming Distance, is used to form decisions.

#### 4.3.3.1 Encoder

To perform convolutional encoding, shift register\(^8\) and module-two addition combinatorial logic is needed. The encoder shown in Fig. 4.17 encodes the \( K = 3 \), \((7,5)\)

\(^6\)“Viterbi decoding was developed by Andrew J. Viterbi, a founder of Qualcomm Corporation. The technique is presented in ‘Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm,’ published in IEEE Transactions on Information Theory, Volume IT-13, pages 260-269, in April, 1967.”

\(^7\)A straight line distance between any two points is called the Euclidean distance. Euclidean distance can be considered between sequences.

\(^8\)“A shift register is a chain of flip–flops wherein the output of the \( n \)th flip–flop is tied to the input of the \( (n+1) \)th flip–flop. At every active edge of the clock, output is considered input of the flip–flop, and thus the data are shifted over one stage.”
convolutional code where rate is $1/2$ and $m = 2$. The octal numbers 7 and 5 represent the code generator polynomials, which when read in binary (1112 and 1012) correspond to the shift register connections to the upper and lower modulo-two adders, respectively.

Each input bit has an effect on three successive pairs of output symbols. That gives the convolutional code its error-correcting power. The example encoder has two bits of memory, and so there are four possible states. In the state diagram shown in Fig. 4.18, the transitions from a current state to a next state are determined by the current input bit, and each transition produces an output of $n$ bits.

The code rate of convolutional coding can be increased by puncturing. Puncturing removes some of the bits after encoding. This gives the same code rate as if it is encoded with a higher rate code. But with puncturing, any code rate is achievable with the same decoder.
In addition, interleaving may be applied to a convolutional coding as in block coding, since Viterbi algorithm operates at optimum if the received inputs are independent. Interleaving eliminates the correlation in time and frequency and sufficiently provides independent received inputs by converting Rayleigh channel to an approximated Gaussian one. Figure 4.19 illustrates the performance of an interleaved convolutional code over a Rayleigh channel. Notice that interleaving provides a far lower error rate and steeper curve.

4.3.3.2 Viterbi Decoder

The Viterbi algorithm operates on the Trellis diagram. Figure 4.20 shows the Trellis diagram for rate 1/2 $K = 3$ convolutional encoder, for a 15-bit message. The operation consists of finding the path of states, where a state denotes the past history of the sequence over the constraint length as described earlier. This Viterbi decoder presented in the figure operates on hard decision and so considers only hamming distance. For soft decoding, the algorithm would find the transmitted sequence whose
Euclidean distance, or equivalently whose accumulated square distance, is closest to the received sequence.

Figure 4.20 shows the Trellis diagram with four states. Notice that encoder is flushed to the initial state with two memory flushing bits (two zeroes) appended to 15-bit message. If input is one, it is represented by solid lines and otherwise it is represented by dotted lines. Notice that arrows are in line with the state transition table presented in Fig. 4.18.

Viterbi decoder decides about the original bits according to accumulated error metric. At time instant $t$, the smallest accumulated error metric is selected according to the history of what states preceded the states. This method is called traceback method.\(^9\)

Accumulated error metric for each branch is shown in Fig. 4.21. It also shows the state transition diagram. Notice that state transition diagram shows the restricted transition between states, and solid lines are used for input one and dashed lines are for input zero. Viterbi decoder selects the state that has the smallest accumulated error metric and iteratively performs backward.

For each branch, there is accumulated error metric and minimum is selected. If accumulated error metric is equal, then the decoder decides by looking forward. For instance, at time $t = 3$ and $t = 12$, there are errors, and the decoder can end up with the same accumulated error. Next forward step shows in both cases the correct state.

### 4.3.4 Concatenated Coding

Concatenated coding has emerged from the need to achieve better error correcting capabilities with less-complex structures. The concatenated coding uses two or more codes after each other or, in parallel, usually with some kind of interleaving. The constituents of the codes are decoded with their own decoders.

\(^9\) "Traceback is not scalable with longer messages due to memory constraints and decoder delay. A traceback depth of $K \times 5$ is sufficient for Viterbi decoding. Source: C. Fleming, 'A Tutorial on Convolutional Coding with Viterbi Decoding,' Spectrum Applications, Copyright 1999-2006."
The idea of concatenated coding is illustrated in Fig. 4.22. Serial and parallel concatenated coding is illustrated. Notice that in serial concatenated coding, parity bits generated from the first encoding are also encoded in the second code. We do not see this parity of parity if codes are working in parallel.

Concatenated coding, which combines the block coding and convolutional coding, is illustrated in Fig. 4.23. This structure effectively combats for errors. The block coding is applied before convolutional coding and block decoding is applied after convolutional decoding. Interleaving is performed in between for superior performance with different interleaving patterns.

The inner convolutional code performs superior error correction with soft decision decoding, and if convolutional code makes an error, it causes a large burst, since Viterbi algorithm may pick a wrong sequence. In this case, we know that block coding, especially an interleaved Reed-Solomon coding, is superior correcting the bursty errors.
Concatenated coding provides means to constructing long codes, and it also confirms Shannon’s channel coding theorem by stating that if the code is long enough, any error can be corrected.

### 4.3.5 Trellis Coding

Trellis coding\(^{10}\) in simplest terms is a combination of coding and modulation. Coding and modulation is gelled together neatly to show that coded modulation schemes are capable of operation near Shannon capacity on bandlimited channels. Soft decision decoding is based on minimum Euclidean distance of sequences and is part of the demodulation procedure.

Trellis coding adds redundant constellation points rather than redundant bits or symbols. Consequently, bit rate increases but the symbol rate stays the same and it conserves bandwidth. Increasing the constellation size reduces Euclidean distance between the constellation points, but sequence coding offers a coding gain that overcomes the power disadvantage of going to higher constellation.

Figure 4.24 shows Trellis-coded modulation. The first step in designing a Trellis code is to form an expanded constellation and to partition it into subsets. The points within each subset are made far apart in Euclidean distance, and will correspond to uncoded bits. The remaining, or coded bits, determine the choice of subset. The code rate is different and each adds \( m \) extra bit to the symbol bit size.

Figure 4.25 shows Trellis coding for QAM modulation, which adds one extra bit and expands the constellation without increasing the signal energy. The signal energy is kept the same, since the distance between the symbols decreases. Although it sounds like a disadvantage in performance, advantage comes from the restriction on what transitions are allowed in the constellation. Those transitions are being

---

\(^{10}\) Invented by Gottfried Ungerboeck in 1982. This technique is used in telephone-line modems to squeeze high ratios of bits-per-second to Hertz out of 3 KHz-bandwidth analog telephone lines. The Viterbi decoding algorithm is also used in decoding Trellis-coded modulation as well.
4.3 Coding

Fig. 4.25  Trellis-coded modulation: BPSK: code rate 1/2, output QPSK; QPSK code rate 2/3, output 8PSK; 8PSK, code rate = 3/4, output 16QAM

Fig. 4.26  QPSK with and without Trellis coding

determined by a simple convolutional code so that only certain sequences of subsets are permitted. The sequence is best described by a state transition diagram or “Trellis.” Allowed sequences are kept apart, so that maximum distance separation is achieved between branches as seen in Fig. 4.26.

There are many ways to map the coded bits into symbols. Mapping by set partitioning is a technique introduced by Ungerboeck. The technique introduces subsets in the constellation and the Euclidean distance between sequences of signal points in different subsets is significantly increased as seen in Fig. 4.27.

At the receiver, a Viterbi algorithm is used for the combined demodulation and decoding. For each received symbol the distance to the nearest member of each subset is measured. The square of this value serves as the metric in extending the survivor states of the Viterbi algorithm. For each survivor state, not only must the coded bits and accumulated squared distance be stored, but also the uncoded bits corresponding to which member of the subset was the nearest point for each symbol.
In OFDM, typical Trellis coding is performed over the subcarriers of a symbol. As a result, at the beginning of each symbol the code is started in a known state, and memory is not extended over a greater interval of time.

### 4.3.6 Turbo Coding

A typical turbo coding\(^\text{11}\) includes parallel concatenated convolutional codes where the information bits are coded by two or more recursive systematic convolutional (RSC) codes, which are typically interleaved and optionally punctured as in Fig. 4.28. Let us first look at the encoding process. The components in encoding are:

- **The RSC encoder**: Why we use recursive format? Convolutional codes operate in feed-forward form such as \((G_1, G_2) = (1 + D^2, 1 + D + D^2)\). This structure produces codewords with very low weight, since for instance, a single 1 (...0001000...) gives a codeword equal to generator sequence, and it will propagate through any interleaver as a single 1. This produces larger number of codewords with very low weight. Recursive structure uses division as in \((1, G_2/G_2) = (1, (1 + D + D^2)/(1 + D^2))\), which does not change the encoding sequences but changes the mapping of input sequences to output sequences. As a result, a weight-one input gives a codeword of semi-infinite weight, since it diverges from the all-zero path, but never remerges and there will always exists a Trellis path that diverges and remerges later corresponding to a weight-two data sequence.
- **The interleaver**: takes each incoming block of \(N\) data bits and rearranges them in a pseudo-random fashion in order to give patterns that has high weight.
- **The puncturer**: periodically deletes the selected bits to reduce coding overhead. Deletion of parity bits is recommended.

---

\(^{11}\)“Turbo coding was introduced by Berrou, Glavieux, and Thitimajshima (from ENST Bretagne, France) in their title *Near Shannon Limit error-correcting coding and decoding: Turbo-codes* published in the Proceedings of IEEE International Communications Conference in 1993.”
4.3 Coding

The structure and complexity of turbo encoder design is restricted by decoding delay and complexity:

- **Decoding delay** is important for system performance, since significant delay in the system may degrade the system. Increase in the number of parallel coding structure increases the delay.
- **Coding gain** can be increased by designing the system with low SNR but the same BER. However, many other receiver functions such as synchronization and adaptive algorithms require a minimum SNR.

The standard decoding process is iterative as can be seen in Fig. 4.29. We explain two decoding schemes employed by each constituents decoder: Maximum a posteriori probability (MAP) (aka a posteriori probability (APP) or BCJR algorithm) technique and soft input soft output Viterbi algorithm (SOVA). Three different types of soft inputs are available for each decoder:

- The un-coded information symbols,
- The redundant information resulting from first RSC code,
- A priori (extrinsic) information, which is the estimate of the information sequence obtained from the first decoding.
In general, a symbol-by-symbol MAP algorithm is optimal for state estimation of a Markov process. MAP algorithms for turbo decoding calculate the logarithm of the ratio of APP of each information bit being one to the APP of the bit being zero, and the decoder decides $x_k = 1$ if $P(x_k = 1|Y) > P(x_k = 0|Y)$, and it decides $x_k = 0$, otherwise where $Y$ is the received codeword. The log of APP ratio is defined as

$$L(x_k) = \log \left( \frac{P(x_k = 1|Y)}{P(x_k = 0|Y)} \right),$$

which translates into

$$L(x_k) = \log \left( \frac{P(Y|x_k = 1)}{P(Y|x_k = 0)} \right) + \log \left( \frac{P(x_k = 1)}{P(x_k = 0)} \right),$$

with the second term representing the a priori information. Since $P(x_k=1)=P(x_k=0)$ typically, the a priori information is zero for conventional decoders but for iterative decoders, Decoder 1 receives extrinsic information for each $x_k$ from Decoder 2, which serves as a priori information. Similarly Decoder 2 receives extrinsic information from Decoder 1.

The MAP technique is complicated and requires nonlinear operations that make it less attractive for practical purposes. A simplification of MAP algorithm, namely, SOVA leads to a practical suboptimum technique. While performance is slightly inferior to an optimal MAP algorithm, the complexity is significantly less. MAP takes into account all paths by splitting them into two sets, namely, the path that has an information bit one at a particular step and paths that have bit zero at that step and returns the log likelihood ratio of the two.

SOVA considers only the survivor path of the Viterbi algorithm. Therefore, only the survived competing path, which joins the path chosen in the Viterbi algorithm is taken into account for reliability estimation.
While the encoders have a parallel structure, the decoders operate in serial, which results in an asymmetric structure. For example, in the first iteration the first decoder has no a priori information and the same iterative turbo decoding principle can be applied to serial and hybrid concatenated codes as well.

Turbo codes increase data rate without increasing the power of a transmission, or they can be used to decrease the amount of power used to transmit at a certain data rate. Turbo coding shows high error correction performance because of its structure based on interleaving in conjunction with concatenated coding and iterative decoding using (almost) uncorrelated extrinsic information. Turbo coding such as block turbo coding and convolutional turbo coding are included in IEEE 802.16 as supported coding schemes.

### 4.3.7 LDPC Coding

Turbo codes provide a performance that is very close to the maximum rate dictated by Shannon theorem over a noisy channel as compared with all coding schemes to date. Low-density parity check (LDPC)\(^\text{12}\) is an emerging new technique that gets even more closer to Shannon rate with long codewords.\(^\text{13}\) LDPC codes are linear block codes that show good block error correcting capability and linear decoding complexity in time.

An LDPC code operates on an \(H\) matrix containing a low count of ones – hence the name low-density parity-check codes. This is used in encoding in order to derive equations from the \(H\) matrix to generate parity check bits. Iterative decoding utilizes “soft inputs” along with these equations in order to generate estimates of sent values.

A \((n, k)\) LDPC encoder would have an \(H\) matrix, which is \(m \times n\) in size where \(m = n - k\). For instance, a \((8,4)\) LDPC encoder with a code rate of 4/8 might have the following \(H\) matrix as an example

\[
\begin{pmatrix}
01011001 \\
11100100 \\
00100111 \\
10011010
\end{pmatrix},
\]

where columns (1–4) are to represent the message and columns (4–8) are to represent the parity bits. It is low density because number of 1s in each row \(w_r\) is \(\ll m\) and number of 1s in each column \(w_c\) is \(\ll n\). Also LDPC is regular if \(w_c\) is constant for every column and \(w_r = w_c(n/m)\) is also constant for every row. Otherwise it

---


\(^{13}\) “In 1999, Richardson introduced an irregular LDPC code with code length 1 million. The code is shown to perform within 0.3 dB of the Shannon limit. In 2001, Chung introduced a closer design which is 0.0045 dB away from capacity.”
is irregular. There are several mechanisms introduced to construct LDPC codes by Gallager, MacKay,\textsuperscript{14} etc. In fact, randomly chosen codes are also possible.

LDPC encoding is similar to systematic block code in, which codeword \((c_0, \ldots, c_n)\) would consist of the message bits \((m_0, \ldots, m_k)\) and some parity check bits as we mentioned earlier. The solution is solving the parity check equations to calculate the missing values

\[
H c^T = 0, \quad (4.16)
\]

where this manipulation can be performed with a generator matrix \(G\). \(G\) is found from \(H\), which can be written as follows with Gaussian elimination

\[
H = [P^T : I] \quad (4.17)
\]

and \(G\) is

\[
G = [I : P]. \quad (4.18)
\]

Hence, \(c\) codeword is found for message word \(x\) as follows \(c = xG = [x : xP]\).

The graphical representation\textsuperscript{15} for the same LPDC is given in Fig. 4.30. Graphical representation utilizes variable nodes (v-nodes) and check nodes (c-nodes). The graph has \(m\) c-nodes and \(n\) v-nodes, where \(m\) stands for the number of parity bits. Check node \(f_i\) is connected to \(c_i\) if \(h_{ij}\) of \(H\) is a 1. This is important to understand the decoding. Decoding tries to solve the \((n-k)\) parity check equations of the \(H\) matrix.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig4.30.png}
\caption{Graphical representation of (8,4) LMDS}
\end{figure}

\textsuperscript{14} “David MacKay, Michael Luby, and others resurrected the LDPC in the mid-1990s.”

\textsuperscript{15} “Robert Tanner, in 1981 generalized LDPC codes and developed a graphical method of representing these codes, now called Tanner graphs or bipartite graphs...”
There are several algorithms defined to date and the most common ones are message passing algorithm, belief propagation algorithm, and sum–product algorithm.

LDPC decoding is an iterative process where in each round,

Step 1 v-nodes $c_i$ send a message to their c-nodes $f_j$. In the first step, $c_i$ only has the received bit $y_i$.

Step 2 c-nodes $f_j$ determine a response to its connected v-nodes. The $f_j$ considers all the messages correct except message $c_i$ and calculates a response to $c_i$. Here, LDPC decoder might find out that the received bits are correct and terminates the decoding if all equations are fulfilled.

Step 3 v-nodes receives these responses from c-nodes and uses this information along with the received bit in order to find out that the originally received bit is correct. Then, sends this information back to c-nodes.

Step 4 go to Step 2.

A simple decoder might use hard decision decoding and would do majority vote in Step 3. An example is depicted in Fig. 4.31 for a codeword $c = [10010101]$. From the figure one can see that the second iteration is enough to detect the correct codeword if bit $c_1$ is flipped to 1. As you can see in the second step, c-nodes come up with a response for each v-node. If there are even numbers of 1s in all v-nodes except $c_i$ then $f_j$ response to $c_i$ is 0, otherwise 1. Then, in step 3, v-nodes apply majority vote to determine their decision. For example if the originally received bit is 1 for $c_0$ and messages from check nodes are $f_1 \rightarrow 0$ and $f_3 \rightarrow 1$ then the decision is 1 as well.

Fig. 4.31 Hard decision decoding for LDPC
Soft decision decoding based on belief propagation introduced by Gallager is preferred since it yields better performance. Belief propagation introduces probabilities as messages that are being passed and these probabilities are used to update the confidence for the bits in the equations.

Let us denote $q_{ij}$ as the message to be passed from v-node $c_i$ to c-node $f_j$ and $r_{ji}$ is the message to be passed from the same c-node to the same v-node. $q_{ij}$ and $r_{ij}$ basically represent the amount of belief in $y_i$ whether it is a “0” or a “1.”

In Step 1, all v-nodes send $q_{ij}$ messages. At the first step, $q_{ij}(1) = P_i$ and $q_{ij}(0) = 1 - P_i$, where $P_i = \Pr(c_i = 1|y_i)$. In Step 2, c-nodes calculates the $r_{ji}$:

$$r_{ji}(0) = \frac{1}{2} + \frac{1}{2} \prod_{j' \in V_j/i} (1 - 2q_{j' j}(1)), \quad (4.19)$$

where $r_{ji}(1) = 1 - r_{ji}(0)$ and $V_j/i$ all v-nodes except $c_i$. This is basically the probability that there is an even number of 1s among $V_j/i$. In Step 3, v-nodes update their responses according to following

$$q_{ij}(0) = K_{ij}(1 - P_i) \prod_{j' \in C_j/i} r_{j' i}(0) \quad (4.20)$$
$$q_{ij}(1) = K_{ij} P_i \prod_{j' \in C_j/i} r_{j' i}(1), \quad (4.21)$$

where $C_j/i$ now stands for all c-nodes except $f_j$ and $K_{ij}$ are constants in order to ensure that $q_{ij}(1) + q_{ij}(0) = 1$.

Also at this step, v-nodes update the decision $\hat{c}_i$ with information from every c-node. If the estimated $\hat{c}$ satisfies $\mathbf{H}\hat{c}^T = 0$, then the algorithm terminates. The probabilities for 0 and 1 are found out to be with the following equations

$$Q_i(0) = K_i(1 - P_i) \prod_{j' \in C_i} r_{j' i}(0) \quad (4.22)$$
$$Q_i(1) = K_i P_i \prod_{j' \in C_i} r_{j' i}(1), \quad (4.23)$$

where $K_i$ is the constant that makes $Q_i(1) + Q_i(0) = 1$. Hence, $\hat{c}_i$ is 1 if $Q_i(1) > Q_i(0)$, otherwise it is 0. These equations can be modified for log domain as well to change the multiplications into additions.

Unlike turbo coding, LDPC codes can determine when a correct codeword is detected and LDPC decoding based on belief propagation can be simpler than turbo decoding. It can show gains of more than 0.5 dB from a low code rate turbo coding and up to 2 dB from other coding solutions.

### 4.4 Synchronization

Synchronization is the essential part of the receiver since oscillator impairments and clock differences along with phase noise during demodulation degrade the performance.
Let us look at a typical front end of an OFDM receiver, which is depicted in Fig. 4.32. The received signal is first down converted to an IF frequency then to baseband with IQ demodulator. Later, the waveform is converted to digital format by sampling. Receiver first synch with symbol boundary in time domain then it locks to subcarrier frequencies.

Hence, OFDM needs to employ time and frequency synchronization. Time synchronization is to decide for the symbol boundaries. Commonly, a sequence of known symbols-preamble are used to detect the symbol boundaries. It has less sensitivity to timing offset as compared with single-carrier systems, since timing offset does not violate the orthogonality of subcarriers in OFDM system, but causes ISI in single-carrier systems.

Unlike time synchronization, frequency synchronization, which is to estimate the frequency offset in the oscillators in order to align the oscillators in the transmitter and receiver, is essential otherwise ICI occurs, since subcarriers could be shifted from its original position and the receiver may experience nonorthogonal signals. Since the carriers are spaced closely in frequency domain, a small fraction of frequency offset is barely tolerable. Also, practically oscillators do not produce a carrier at exactly one frequency but rather a carrier with random phase noise. This phase noise in time domain corresponds to frequency deviation in frequency domain, thereby causing ICI.

### 4.4.1 Timing Offset

OFDM is insensitive to timing offset as long as offset is within the guard time. Consequently, no ISI and ICI is guaranteed. On the other hand, optimum symbol detection is important, since any lag in detection may increase the sensitivity to delay spread. Also, timing offset changes the phases of subcarriers but does not
violate orthogonality. These phase shifts are estimated during channel estimation if receiver employs coherent receiver. Assuming that there is no ISI, $y_k$, the received signal, is

$$y_k = \sum_{n=0}^{N-1} x_n e^{j \theta} e^{j 2\pi \frac{n}{N} f_s t} |_{t=\frac{k}{N}} ,$$

(4.24)

where $\theta$ is envelope delay distortion and $d$ is sampling time offset. And $\hat{x}_m$ after FFT is

$$\hat{x}_m = \frac{1}{N} \sum_{k=0}^{N-1} y_k e^{-j 2\pi \frac{m}{N} k} = \frac{1}{N} \sum_{n=0}^{N-1} x_n e^{j (\theta + 2\pi \frac{n}{N} d)} \sum_{k=0}^{N-1} e^{-j 2\pi \frac{k}{N} (n-m)} ,$$

(4.25)

from where we can see that introduced phase offset effects all subcarriers linearly. These phase rotations can be corrected in the channel estimation stage.

### 4.4.2 Frequency Offset

OFDM is sensitive to frequency offset since it causes ICI, which basically introduces interference from all other subcarriers. If $y_k$ is the received signal with no timing offset and $\delta_f$ is the frequency offset, then

$$y_k = \sum_{n=0}^{N-1} x_n e^{j 2\pi \left(\frac{n}{N} f_s + \delta_f\right)} |_{t=\frac{k}{N}}$$

(4.26)

and after FFT we get

$$\hat{x}_m = \Theta(\delta_f) x_m + \sum_{n=0}^{N-1} \sum_{k=0}^{N-1} x_n e^{j 2\pi \frac{k}{N} (n-m + n\delta_f)},$$

(4.27)

where $\hat{x}_m$ has an attenuation component ($\Theta(\delta_f)$) as well as interference component from other subcarriers.

### 4.4.3 Phase Noise

Phase noise occurs during intermediate stages of demodulation, such as between RF and IF. Phase noise is a zero mean random process of deviation of the oscillator’s phase. Power spectral density (PSD) of phase noise is normalized with the power of the sine wave, oscillator output. Typical phase noise PSD is shown in Fig. 4.33 and this PSD is high-pass filtered with phase locked loop (PLL) and oscillator is locked for intermediate modulation.
We can follow the same analysis that was used to examine frequency offset, and assume there are no other impairments. Then, the output of the receiver demodulator is

\[
\hat{x}_m = \frac{1}{N} \sum_{n=0}^{N-1} x_n \sum_{k=0}^{N-1} e^{j\phi(k)} e^{-j\frac{2\pi}{N} (n-m)}, \quad m = 0, \ldots, N - 1, \tag{4.28}
\]

where \(\phi(k)\) is the phase noise for \(k^{th}\) subcarrier and can be approximated to \(\approx 1 + j\phi(k)\), thereby

\[
\hat{x}_m = x_m - \frac{jx_m}{N} \sum_{k=0}^{N-1} \phi(k) - \frac{j}{N} \sum_{n=0, n \neq m}^{N-1} x_n \sum_{k=0}^{N-1} \phi(k) e^{-j\frac{2\pi}{N} (n-m)}, \tag{4.29}
\]

where we can see that the first term is desired output and the last two terms are two effects of the phase noise: Second term in the equation is a random phase disturbance, which occurs in each symbol. This is known as common phase error (CPE) and can readily be eliminated by measuring the phase variation of a pilot subcarrier and subtracting that rotation from all subcarriers. Third term in the equation is ICI and it is peculiar. The interference may be treated as a Gaussian noise if the different subcarriers are independent. Also decrease in the subcarrier spacing increases the interference.

Figure 4.34 shows the BER performance of phase noise with the following frequency shaping: \(-65\) or \(-75\) dB till 10 KHz offset, slope \(-20\) dB/dec, and \(-135\) dBc noise floor considering the phase-locked spectral shape in Fig. 4.33.

### 4.4.4 Pilot-Assisted Time/Frequency Synchronization

Time and frequency offset estimator mostly assumes that transmitted data are known at the receiver by transmitting pilot symbols. As a result, symbol timing and carrier frequency offset can be estimated at the receiver. There are also blind methods that uses cyclic prefix to estimate synchronization parameters with statistical redundancy.
Pilot-based methods suit better for high data rate communication, since synchronization should be as quick as possible since blind methods require averaging over a large number of OFDM symbols.

Initially a frame synchronization is performed to detect the start of the frame since typical channel may introduce unknown frequency offset and an unknown time delay. This is performed by correlating the incoming signal with a known preamble. Typical receiver that compensates for time and frequency offsets is shown in Fig. 4.35.

Frame synchronization is achieved by correlating the incoming signal with a known preamble. A matched filter can be used to correlate the input signal with
4.4.5 Blind Time-Frequency Synchronization

Blind synchronization is pilotless and based on maximum likelihood estimation. The parameters that need to be estimated require longer observation. As we know, OFDM symbol is cyclically extended, which is basically replicating the tail of the symbol in the guard interval. This cyclostationary property can be leveraged to estimate symbol timing $t_o$, frequency offset $f_o$, and symbol width $T$. The optimum estimator maximizes the following to find $t_o, f_o, \text{ and } T$:

$$
\Theta(t_o, f_o, T) = \text{Re} \left[ e^{j2\pi f_o T} \sum_{m=1}^{M} \int_{-T_G}^{0} y(t + t_o - mT_s) \times y^*(t + T + t_o - mT_s) dt \right],
$$

where $y$ is received signal, $T_G$ is guard interval length.

4.5 Detection and Channel Estimation

To estimate the transmitted bits at the receiver, channel knowledge is required in addition to the estimates of random phase shift and amplitude change, caused by carrier frequency offset and timing offset. The receiver applies either coherent detection or noncoherent detection to estimate the unknown phase and amplitude changes introduced by multipath fading channel.

The coherent detection of subscribers requires channel estimation. These are done by channel equalizer, which multiplies each subscriber in an OFDM symbol with a complex number.

Noncoherent detection on the other hand does not use any reference values but uses differential modulation where the information is transmitted in difference of the two successive symbols. The receiver uses two adjacent symbols in time or two adjacent subcarriers in frequency to compare one with another to acquire the transmitted symbol.

4.5.1 Coherent Detection

Channel estimation for coherent detection is performed with pilot symbols considering that each subscriber experiences flat fading (Fig. 4.36). This involves sparsely insertion of pilot symbols in a stream of data symbols and measuring the attenuation in these pilot symbols in order to find the channel impulse response for given
frequency of the pilot and estimating the channel impulse response by interpolating these results to find out the channel components in the data inserted subcarriers.

The coherent detection can be performed by either inserting pilot tones into all of the subcarriers of OFDM symbols with a period in time or inserting pilot tones into each OFDM symbol with a period in frequency. In the case of time-varying channels, the pilot signal should be repeated frequently. The spacing between pilot signals in time and frequency depends on coherence time and bandwidth of the channel. One can reduce the pilot signal overhead by using a pilot signal with a maximum distance of less than the coherence time and coherence bandwidths. Then, by using time and frequency interpolation, the impulse response and frequency response of the channel can be calculated.

Pilot spacing has to follow several requirements in order to strike a balance between channel estimation performance and SNR loss. Smaller pilot spacing in time and frequency would result in a good channel estimation, but the effective SNR for data symbols will be smaller. Channel variation in time and frequency are used
to determine the minimum pilot spacing in time and frequency. If \( B_D \) is Doppler spread and \( T_S \) is delay spread, a suitable choice for pilot spacing in time \( N_{tp} \), and in frequency \( N_{fp} \), is as follows:

\[
N_{tp} \approx \frac{1}{B_D T_S} \quad N_{fp} \approx \frac{1}{\Delta f T_S},
\]

where \( \Delta f \) is subcarrier bandwidth, \( T \) is symbol time.

These requirements lead to several pilot arrangement types in time and frequency. Figure 4.37 illustrates several pilot arrangements. The first one in the figure is a block-type pilot arrangement for slow fading channels. All subcarriers are used once and the estimated channel is used for the coming symbols since the channel is assumed to change very slowly for slow fading channels, and the channel is highly correlated for the consecutive symbols. For example, in wireless LAN systems, block-type pilot arrangement is used, since packet sizes are short enough to assume a constant channel for the duration of the packet.

The second approach is the comb-type pilot arrangement, which interleaves the pilots over time and frequency. Comb-type estimation uses several interpolation techniques to estimate the entire channel. Flat fading channels can be estimated by comb-type pilot arrangement and pilot frequency increases if the channel is frequency selective fading.

Besides interpolation, time and frequency correlation is used for channel estimation. If Doppler effects are kept small by keeping OFDM symbol shorter compared with coherence time of the channel, time correlation between OFDM symbol is high. Moreover, in an ideal OFDM system, if subcarrier spacing is small as compared with the coherence bandwidth of the channel, frequency correlation between the channel components of adjacent subcarriers is high.

Now, assume that the diagonal matrix \( \mathbf{X} \) contains the transmitted pilot symbols and vector \( \mathbf{y} \) contains the observed output of the FFT:

\[
\mathbf{y} = \mathbf{Xh} + \mathbf{n},
\]

**Fig. 4.37** Pilot positioning in time and frequency
where the channel estimation problem is to find the channel estimates \( \hat{h} \) as a linear combination of pilot estimates. The least-squares (LS) channel estimation is

\[ \hat{h}_{LS} = X^{-1}y, \]  

(4.33)

since LS minimizes \( \| y - X\hat{h} \| \) for all \( \hat{h} \).

LS operates with received and known transmitted pilot symbols. The frequency correlation can be further exploited with LS in order to minimize \( \| \hat{h} - h \|^2 \) for all possible linear estimators \( \hat{h} \). Then, the optimal linear minimum mean squared error (LMMSE) estimate is

\[ \hat{h}_{MMSE} = A\hat{h}_{LS}, \]  

(4.34)

where

\[ A = R_{hh_{LS}} R_{h_{LS}h_{LS}}^{-1} \]
\[ = R_{hh} (R_{hh} + \sigma_n^2 (XX^H)^{-1})^{-1}, \]  

(4.35)

and \( R_{hh} = E\{hh^H\} \) is the channel autocorrelation matrix. LMMSE estimator is complex and normally used as a base for designing new estimators.

### 4.6 Equalization

Equalization is used to combat for intersymbol interference (ISI) and works along with channel estimation. If symbol time is larger than delay spread, then the system suffers from ISI. In OFDM, typically symbol time is extended with guard interval to reduce the ISI. However, equalization is used in the frequency domain to remove amplitude and phase distortions caused by fading channel. Equalizer utilizes channel estimation and continually tracks the channel.

OFDM equalization typically employs a combination of CP with a standard equalization (e.g., linear equalization or decision feedback equalization). Equalization balances ISI mitigation with noise enhancement, since if channel is \( H(f) \) and equalizer selects \( H_e(f) \) as \( 1/H(f) \) then the frequency response of noise after equalization \( N'(f) \) becomes \( \frac{5N_o}{|H(f)|^2} \). Notice that for some frequency if there is a spectral null in the channel then noise power is greatly enhanced. In general, linear equalizers cause more noise enhancements than nonlinear equalizers. They can be both implemented using a transversal or lattice structure as seen in Fig. 4.38. The transversal structure is a filter with delay elements and tunable complex weights. The lattice filter has recursive structure, which is more complex than transversal structure but achieves better convergence, numerical stability, and scalability.

Figure 4.39 illustrates time and frequency domain equalization. Frequency domain equalization is used to compensate for channel complex gain at each subcarrier. Notice that ICI is absent in OFDM. Equalization after FFT is equivalent to a convolution of a FIR filter in time domain (residual equalization).

We first talk about time domain equalization methods: linear and nonlinear transversal structure equalizers. Then, we discuss frequency domain equalization.
Let us assume a system with a linear equalizers, where pulse shape is compensated with matched filter as seen in Fig. 4.40. If ISI channel is $f(t)$ and transmitted signal is $x(t)$ then received signal $y(t)$ is

$$y(t) = x(t) * f(t) + n(t) = \sum x_k f(t - kT_s) + n(t), \quad (4.36)$$

where $n(t)$ is white noise. When $y(t)$ is sampled with $T_s$, $y[n] = y(nT_s)$ and $v[n] = n(nT_s)$ become

$$y[n] = x_n f[0] + \sum_{k \neq n} x_k f[n - k] + v[n], \quad (4.37)$$
where the second term stands for ISI and ISI-free communication is achieved if $f[n-k] = 0$ for $k \neq n$.

If there is ISI then the equalizer $F_e(z)$ represented in $z$-domain is used to reduce the ISI. Linear equalizer is represented as below:

$$F_e(z) = \frac{i = -L}{L} \sum_{i} w_i z^{-i}, \quad (4.38)$$

where there are $N = 2L + 1$ taps and weights $w_i$, which are set to reduce the probability of error.

### 4.6.1 ZF: Zero Forcing Equalizer

ZF equalizer is a linear equalizer and sets $F_e(z)$ to $1/F(z)$, which cancels out the ISI but enhances the noise $N(z)$ by $\frac{1}{|F(z)|^2}$ as seen in Fig. 4.41. This significantly

\[ F_{\Sigma}(f) = \frac{1}{T_s} \sum_{n=-\infty}^{\infty} F(f + \frac{n}{T_s}) \]

and if $F_{\Sigma}(f) = f[0]$, then there is no ISI.
4.6 Equalization

Fig. 4.41 Zero forcing equalizer

increases the noise if there is an attenuation in some frequencies in the channel. Also note that, noise is no longer white but colored, which complicates the detector.

The ZF equalizer determines finite set of coefficients $w_i z_i$ according to $1/F(z)$, of course there are many ways, one technique is setting $w_i = c_i$ in

$$\frac{1}{F(z)} = \sum_{i=-\infty}^{\infty} c_i z^{-i} \quad (4.39)$$

in order to minimize

$$\left| \frac{1}{F(z)} - \sum_{i=-L}^{L} w_i z^{-i} \right|^2. \quad (4.40)$$

4.6.2 MMSE: Minimum Mean-Square Error Equalizer

MMSE equalizer is also a linear equalizer and provides a better balance between ISI and noise enhancement. Since it minimizes the expected mean-squared error between transmitted symbol $x_k$ and the symbol detected at the equalizer output $\hat{x}_k$:

$$E[x_k - \hat{x}_k]^2, \quad (4.41)$$

where

$$\hat{x}_k = \sum_{i=-L}^{L} w_i y[k - i] \quad (4.42)$$

and $F_e(z)$ is found to be

$$F_e(z) = \frac{1}{F(z) + N_o}. \quad (4.43)$$
Notice that noise is still colored; when noise is colored, then there is another filter before equalizer, which is called noise whitening filter, which whiten the noise in order to obtain a flat power spectrum.

One can see that if there is no noise then this equals to ZF equalizer, and when there is noise, it clearly shows a balance between reduction in ISI and enhancement in noise as seen in Fig. 4.42.

### 4.6.3 DFE: Decision Feedback Equalizers

As we see, linear equalizers when taking care of the ISI cause noise coloring. Decision feedback equalizer (DFE), a nonlinear equalizer, detects the symbols and removes the future ISI by subtracting it from the future signal.

DFE equalizer has two stages. Stage 1 is a forward filter to whiten the noise and to produce a response with postcursor ISI only. And stage 2 has the feedback filter to cancel that postcursor ISI as seen in Fig. 4.43.
The optimum forward filter for a zero-forcing DFE can be considered as a cascade of a matched filter followed by a linear equalizer and a causal whitening filter whose transfer function can be found by spectral factorization of the channel power spectrum.

Let $f(n)$ represent the channel impulse response, hence $f^*(-n)$ is the impulse response of the matched filter. Forward filter $c_f(n)$ performs ISI suppression and noise whitening. The backward filter $c_b(n)$ eliminates the interference of previous symbols, hence $1 + c_b(n)$ has to be a monic and causal filter. Therefore, its output should have postcursor intersymbol interference only, namely
\begin{equation}
1 + c_b(n) = f(n) * f^*(-n) * c_f(n).
\end{equation}

On the other hand, the variance of the noise at the output of the matched filter is
\begin{equation}
\sigma^2 \int_{-\pi/2}^{\pi/2} S_h(e^{j\omega t})d\omega,
\end{equation}
where $S_h$ is power spectrum of the channel impulse response. To minimize noise power at the input of the slicer, $c_f(n)$ should whiten the noise. Rewriting (4.44) in the frequency domain, the forward filter transfer function is
\begin{equation}
C_f(z) = \frac{1 + C_b(z)}{F(z)F^*-1(z^{-1})}.
\end{equation}
The forward filter may apply further linear transformation to the input signal to meet some optimality criteria such as minimizing square error or peak distortion. To whiten a stochastic process with spectrum of $\sigma^2 S_h(z)$, it should satisfy the following equation:
\begin{equation}
\sigma^2 S_h(z) \frac{1 + C_b(z)}{F(z)F^*-1(z^{-1})} \frac{1 + C_b^*(z^{-1})}{F^*(z)F^{-1}(z)} = \sigma^2
\end{equation}
since
\begin{equation}
S_h(z) = F(z)F^*-1(z^{-1})
\end{equation}
and from (4.47) we have:
\begin{equation}
S_h(z) = (1 + C_b(z))(1 + C_b^*(z^{-1})).
\end{equation}

Forcing ISI to be zero before the decision device will cause noise enhancement. Noise enhancement increases the probability of making error, which can easily propagate in such a system. This makes worse than before, since now the system starts to introduce ISI. Also colored noise still requires a complex decision device, which immediately also introduces delay to the system.

In an MSE decision feedback equalizer, the optimum forward filter minimizes the mean-square error before the detector and tolerates some ISI. The same procedure can be followed for the MSE equalizer to obtain
\[
S'_h(z) = (1 + C_h(z))(1 + C^*_h(z^{*-1})).
\] 

(4.50)

MSE-DFE reduces the noise enhancement as compared with the ZF-DFE. The performance is higher than other equalizers, but we still have the error propagation problem.

### 4.6.4 Adaptive Equalizers

Wireless channel is typically time varying \( f(t) = f(\tau, t) \). It is better for equalizer to train for the channel periodically to set the equalizer coefficients. Also, equalizer can track and adjust the coefficients with the use of detected data.

Training must be performed within coherence time \( T_C \) and depending on the length of training sequence \( \ell \) following equality must hold \((\ell + 1)T_s \geq T_C\).

Tracking is based on equalizer output bits \( \hat{x}_k \) and threshold detector output bits \( \hat{\hat{x}}_k \), since \( \hat{x}_k \) is round to the nearest constellation point via threshold detector. If there is an error, then that error is used to adjust the coefficients of \( H_e(z) \) to minimize \( \hat{x}_k \) and \( \hat{\hat{x}}_k \) via MMSE procedure.

There are available techniques for training and tracking. Performance metrics are number of multiplication for \( N \) tap, complexity, training convergence time, and tracking performance. MMSE (number of multiplication is \( N^2 \) to \( N^3 \)) is the most complex with fast training and tracking and least mean square (LMS) \((2N + 1)\) is least complex but with slow training and tracking. LMS updates the tap weight vector linearly with a step size \( (w_i(k + 1) = w_i(k) + \Delta e) \), where \( \Delta \) dictates the convergence speed and stability. Other techniques such as root least square (RLS) \((1.5N^2 + 4.5N)\) and fast Kalman DFE \((20N + 5)\) lie in between the two.

### 4.6.5 MLSE: Maximum Likelihood Sequence Estimation

Maximum likelihood sequence estimation (MLSE) is a nonlinear estimation technique that replaces the equalizing filter with a MLSE estimation as seen in Fig. 4.44. MLSE compares the received noisy sequence \( \{x_j\} \) with all possible noise-free received signal \( y(t) \) and select the closest one. MLSE avoids noise enhancement. MLSE is optimal but very complex to implement, since for length \( n \), there are \( 2^n \) different noise free sequences to compare.

### 4.6.6 Viterbi Equalizer

The Viterbi-equalizer seen in Fig. 4.44 reduces the complexity and minimizes the probability of detecting the wrong sequence of symbols. Fig. 4.45 depicts an
Fig. 4.44 Other equalizers

Fig. 4.45 An example for Viterbi equalizer

example for Viterbi equalizer. Notice that Viterbi equalizer operates with soft decision, where error metric is obtained by Euclidean distance. For instance at $t = 0$, Euclidean distance 1.39 for output one (from state $-1$ to state 1) is obtained by
|1.9 − 0.72|² and for output zero (from state −1 to state −1), 0.68 is obtained by |0.72 − (−0.1)|². In each branch, Euclidean distance is added to the previous accumulated error metric and at the end, depending on the minimum accumulated error metric, the equalizer traces back to find the correct path and output.

For transmitted sequences of length $n$ over a length $L + 1$ channel, it reduces the brute-force maximum-likelihood detection complexity of $2^n$ comparisons to $n$ stages of $2^L$ comparisons through elimination of Trellis paths where $L \ll n$. If the length of the channel is high, then its complexity increases as well.

### 4.6.7 Turbo Equalizer

Turbo equalizer seen in Fig. 4.44 is an iterative equalizer that utilizes a maximum a posteriori (MAP) equalizer and a decoder. The MAP equalizer considers a posteriori probability (APP) of the transmitted symbol with the past channel outputs. The decoder computes log likelihood ratio (LLR) with the transmitted symbol and past channel outputs. After some iteration, the turbo equalizer converges to estimate the transmitted symbol.

### 4.6.8 Equalization in OFDM

We know that OFDM systems use a cyclic prefix (CP) – guard interval – that is inserted at the beginning of each symbol. This transforms the linear convolution of data and channel into a circular one. As long as the CP is beyond delay spread, ISI is avoided. Otherwise, ISI is present and can degrade performance. As a result, unlike a single carrier system, which utilizes equalizer to minimize ISI, an equalizer can be utilized only to limit the length of ISI, since it is adequate to reduce it to a time span, which is less than the length of CP.

Of course, an equalizer is also utilized to reduce the required CP, since CP reduces the efficiency of the system. Reduction of the transmission efficiency is by a factor $N/(N+N_{CP})$ and more severe when the transmitted symbol rate is higher, because it requires longer CP. Effective channel impulse response (EIR) can be shorter than the selected CP duration with a time domain equalizer before FFT demodulation at the receiver.

Block diagram of a time domain equalizer for OFDM differentiates from DFE equalizer as seen in Fig. 4.46. The feedback filter is not in the loop and there is no requirement to be monic or causal, since truncating the impulse response does not require a monic feedback filter. The objective is to shorten the sampled CIR of length $N_h$, $h = [h_0, \ldots, h_{N_h-1}]^T$ to an EIR having significant samples for a length $N_f$, where $N_f < N_h$, with the use of a time domain equalizer of length $N_f$, $f = [f_0, \ldots, f_{N_f-1}]^T$. The error term is

$$e_k = f^T \ast r - b^T \ast x,$$  \hspace{1cm} (4.51)
Fig. 4.46 Time domain equalizer configuration

where \( x \) and \( r \) are vectors of training and received samples respectively. The squared error is given by

\[
E\{|e(k)|^2\} = f^T R_{rr} f^* + b^T R_{rx} b^* - f^T R_{rx} b^* - b^T R_{rx} f^*, \tag{4.52}
\]

where \( R_{rr}, R_{rx}, \) and \( R_{rx} \) are the corresponding correlation matrices of \( r \) and \( x \). The optimal \( f \) can be obtained by MMSE, which is given by

\[
d(E\{|e(k)|^2\})/df = 0 \tag{4.53}
\]

and this leads to

\[
f = R_{rr}^{-1} R_{rx} b. \tag{4.54}
\]

We can solve for \( b \) by substituting the above relation in (4.52) and

\[
E\{|e(k)|^2\} = b^T (R_{xx} - R_{rx}^T R_{rr}^{-1} R_{rx}) b^* = b^T \theta b^*, \tag{4.55}
\]

where \( b \) is then found as the eigenvector, which corresponds to the smallest eigenvalue of the matrix \( \theta \).

In general, we can use adaptive techniques to find near optimum equalizer coefficients. A large number of taps prevents us from using MMSE type of algorithms. On the other hand, a wide spread of eigenvalues of the input signal covariance matrix can slow down convergence of the equalizer. A different iterative technique for the equalizer tap adjustment is based on the steepest gradient methods such as LMS as explained below. Hence,

\[
f^{k+1} = f^k - \Delta_1 e(k)r^*, \quad b^{k+1} = b^k - \Delta_2 e(k)x^*, \tag{4.56}
\]

where \( \Delta \) are the LMS convergence control parameters and equalizer taps are adjusted during the training sequence where transmitted sequence is assumed to be known and the same tap values are preserved during data mode.

The estimation of channel impulse response can help in choosing the optimum window location. As explained previously, \( b \) is not necessarily causal; therefore,
delay parameter plays an important role in the performance of the equalizer. Other important parameters are the lengths of the two filters \( f \) and \( b \).

Typically, the channel impulse response is selected as an ARMA model of the form

\[
\frac{A(z)}{1 + B(z)}
\]

and classic decision feedback equalizers; an ARMA equalizer can have a feedback filter with noncausal transfer function, i.e., the general error term is:

\[
f * r_k - b * x_{k-d}
\]

the delay unit \( d \) has significant effect on overall performance. The number of taps \( M \) of filter \( b \) is approximately the same as the prefix length and number of taps for the forward filter is \( N \leq M \).

The delay unit should be chosen properly to capture the most significant window of the channel impulse response (CIR). Brute-force trial and error is one option. A more intelligent technique uses the estimate of the impulse response to pick a proper starting point where the energy of the impulse response (unit energy constraint UEC) is above a threshold. The threshold should be the ratio of the tap power to the overall window power. Since the length of the window function and the forward equalizer (FEQ) should be as short as possible, the proper choice of starting point is of significant importance.

Otherwise, AWGN is amplified by the forward equalizer, since FEQ tap coefficients are calculated as the inverse of the EIR. Consequently the subchannels that fall in the nulls are severely degraded because of the low SNR.

Besides robustness against ISI, OFDM is also robust against ICI. But if ICI is not completely avoided, then orthogonality of adjacent subcarriers are not preserved in the frequency domain.

Once ICI free transmission is assured that orthogonality of the subcarriers is maintained, possibly through the use of the cyclic prefix and time domain equalization as previously discussed, then the frequency domain equalization of an OFDM signal is an extremely simple process. This is certainly one of the key advantages of OFDM.

After demodulation, the subcarriers will be subjected to different losses and phase shifts, but there will be no interaction among them. Frequency domain equalization therefore consists solely of separate adjustments of subcarrier gain and phase, or equivalently of adjusting the individual decision regions. In the case where the constellations consist of equal amplitude points, as in PSK, this equalization becomes even simpler in that only phase needs to be corrected for each subcarrier, because amplitude has no effect on decisions.

A simplified picture of the place of frequency domain equalization is shown in Fig. 4.47, where the equalizer consists of the set of complex multipliers, \( \{A\} \), one for each subcarrier.

Here the linear channel transfer function \( F(f) \) includes the channel, the transmit and receive filters, and any time domain equalization if present. \( F(f) \) is assumed
4.6 Equalization

bandlimited to less than \(N/T\) for a complex channel. Cyclic extension is not shown although it is almost certain to be present. The following analysis assumes that both amplitude and phase need to be corrected, and that equalization consists of multiplying each demodulated component by a quantity such that a fixed set of decision regions may be used.

The signal presented to the demodulator is

\[
y(t) = \sum_{n=0}^{N-1} d_n h(t - n \frac{T}{N}).
\]

The \(k\)th output of the demodulator is then

\[
y_k(f) = x_k F_k, \quad k = 0, 1, \ldots, N - 1,
\]

where the \(F_k\) are samples of \(F(f)\)

\[
F_k = h \left( \frac{k}{T} \right).
\]

Thus each output is equal to its associated input data symbol multiplied by a complex quantity, which differs among the outputs, but are uncoupled. Equalization at its simplest then consists of setting the multipliers to \(1/F_k\) for each nonzero channel.

The above approach is optimum in every sense under high signal-to-noise conditions. It also produces minimum probability of error at any noise level, and is an unbiased estimator of the input data \(x_k\). However if the criterion to be optimized is minimum mean-square error particularly, then the optimum multipliers are modified to

\[
A_k = \frac{1}{F_k} \frac{1}{1 + \frac{\sigma_k^2}{|x_k F_k|^2}},
\]

where \(\sigma_k^2\) is the noise power in the demodulated subchannel. However, this value produces a biased estimator and does not minimize error probability.

As a practical implementation issue, for variable amplitude constellations, it is frequently desirable to have a fixed grid of decision regions. The \(A_k\)s can then be scaled in amplitude such that the separation of constellation points is constant, since a shift \(\tau\) in timing phase is equivalent to a phase shift.
$$F_k = e^{j2\pi \frac{k}{T} \tau}$$ (4.63)

and frequency domain equalization readily corrects for such timing shift.

In principle frequency domain equalization could be employed when orthogonality is lost because of interference among OFDM symbols. In this case, rather than a simple multiplier per subchannel, a matrix multiplication would be required. This approach is bound to require more computational load than the combination of time and frequency domain equalizers.

### 4.6.9 Time and Frequency Domain Equalization

During system initialization, any time domain equalizer must be adjusted before frequency domain equalization is performed. Then any periodic test signal with full frequency content, such as a repeated segment of a PN sequence without cyclic prefix, may be used to adapt the frequency domain equalizer.

An interesting interpretation of time domain and frequency domain equalizer can be obtained by studying their role in channel distortion compensation. As discussed earlier, a typical channel model for OFDM and channels with long impulse response is an ARMA model of the form in (4.58) where time domain equalization shortens the impulse response to a tolerable level for an OFDM system. Mathematically, it is equivalent to compensating the AR part of the channel impulse response \(1/(1 + B(z))\). So after successful time domain equalization, the equivalent impulse response of the channel is reduced to a FIR filter of short duration \(A(z)\). Since it does not violate orthogonality of subcarriers, we can remove its effect after the FFT by frequency domain equalization as seen in Fig. 4.48.

![Fig. 4.48 Time and frequency domain equalization](image-url)
4.7 Peak-to-Average Power Ratio and Clipping

One of the significant drawbacks of OFDM systems is the possibility to experience large peaks since the signal shows a random variable characteristic since it is the sum of $N$ independent complex random variables. These different carriers may all line up in phase at some instant and consequently produce a high peak, which is quantified by peak-to-average-power ratio (PAPR).

This distorts the transmitted signal if the transmitter contains nonlinear components such as power amplifiers (PAs). Since PA is forced to operate in the nonlinear region. The nonlinear effects may cause in-band or out-of-band distortion to signals such as spectral spreading, intermodulation, or change the signal constellation. Out-of-band distortion is detrimental even if the in-band distortion is tolerable. To have distortionless transmission, the PAs require a backoff, which is approximately equal to the PAPR. This decreases the efficiency for amplifiers and increases the cost. High PAPR also requires high range and precision for the analog-to-digital converter (ADC) and digital-to-analog converter (DAC), as a result, reducing the PAPR of practical interest.

4.7.1 What is PAPR?

Figure 4.49 depicts a PA. One can see the nonlinear behavior of the PA. It is desired to operate the PA in the linear region. To avoid the high peaks, average input power may be decreased. Operating region of the PA is called input back-off and the resultant signal is guaranteed to be in output back-off range. High input backoff reduces the power efficiency and would mandate the cost of the PA higher, since input backoff is usually greater than or equal to the PAPR of the signal. Ideally, the average and peak values should be as close as can be in order to maximize the efficiency of the PA. PAPR mitigation relaxes the PA backoff requirements as well as the high resolution requirements on ADC and DAC.

PAPR mitigation may fall into three categories: signal distortion, coding, and scrambling. Signal distortion basically distorts the signal around peaks with either clipping or peak windowing or peak cancelation. Coding utilizes forward error correction coding schemes to achieve signals with low PAPR. Scrambling also similar to coding utilizes scrambling sequences to achieve low PAPR. Let us first analyze the PAPR to get a better insight on the mitigation techniques, which we explain later in this section.

$\eta = \eta_{\text{max}} 10^{\frac{\text{PAPR}}{20}}$, 

(4.64)

where $\eta$ is power efficiency and $\eta_{\text{max}}$ is maximum power efficiency. $\eta_{\text{max}}$ is 50% and 78.5% for Class A and Class B power amplifiers.
Fig. 4.49 Power amplifier 1 dB compression point: It is desirable to make power amplifier remain linear over an amplitude range that includes the peak amplitudes. Parameters to describe the nonlinearities of the PAs include amplitude modulation/amplitude modulation (AM/AM) distortion, amplitude modulation/phase modulation (AM/PM) distortion, 1 dB compression point (P1dB), and 3rd order interception point (IP3).

If \( \mathbf{X} \) is data vector of length \( N \), time domain vector in the transmitter is \( \mathbf{x} = [x_0, \ldots, x_{N-1}] = \text{IDFT}(\mathbf{X}) \) and the PAPR is then defined to be

\[
\text{PAPR}(\mathbf{x}) = \frac{||\mathbf{x}||_\infty^2}{E(||\mathbf{x}||_2^2)/N},
\]

(4.65)

where \( E(\cdot) \) denotes expectation. \( ||\cdot||_\infty \) and \( ||\cdot||_2 \) represent the \( \infty \)-norm and 2-norm respectively. Therefore, \( ||\cdot||_2^2 = \sigma^2 \) denotes the average (RMS) power. When \( N \) is large, the output time vector converges to Gaussian distribution due to central limit theorem. Hence, the probability that the PAPR is above a threshold is written as

\[
\Pr\{\text{PAPR} > \lambda\} = 1 - (1 - e^{-\lambda})^N.
\]

(4.66)

This is plotted for different values of \( N \), and as it can be seen from Fig. 4.50, the system is more susceptible to PAPR when subcarrier size increases. For a baseband OFDM signal with \( N \) subcarriers, PAPR may be as large as \( N^2/N = N \) for PSK modulation if \( N \) subchannels add coherently. Reduction of subcarrier is one way to reduce PAPR but not efficient. Also from the figure, we can infer that high PAPR does not occur often. Considering these infrequent large peaks, a common approach is to perform clipping in order to mitigate the PAPR. These peaks are removed at a cost of self-interference and bandwidth regrowth. As long as these impairments are kept as small as can be, clipping is a powerful and simple technique to employ.
Also it has been proven that the absolute peak presented above is not a good measure to define the “peak” of the signal power. A good measure defines the “peak” as the level that the probability of crossings that level is negligible. As a result, clipping would occur whenever the signal exceeds this “peak,” which is typically defined as $m$-fold of average RMS power.

### 4.7.2 Clipping

Clipping is a nonlinear process and limits the amplitude at some desired maximum level. This simple mechanism introduces the following impairments: self-interference and out-of-band leakage. There are two prong ways to analyze the distortion caused by clipping: additive Gaussian noise or sporadic impulsive noise. They differ with respect to clip level since if clip level is low then clipping events are high, which tends to a Gaussian-like noise. If clip level is high then the clipping events are sporadic. Then the clipping forms a kind of impulsive noise rather than a continual background noise.

Let us define clipping system first as illustrated in Fig. 4.51. If input $x(t)$ is a multicarrier signal, output $y(t)$ after clipper is as follows

$$y = h(x) = \begin{cases} -l & x \leq l \\ x & |x| < l \\ l & x \geq l \end{cases}$$ (4.67)
where $h(.)$ is the nonlinear transfer function of clipping. Bussbang’s theorem may decompose the $y(t)$ in two uncorrelated signal components

$$y(t) = \kappa x(t) + c(t),$$  \hspace{0.5cm} \text{(4.68)}

where $\kappa \approx 1$ for $l \gg 1$. A clipping scenario is depicted in Fig. 4.52, where we can see that the clip level determines the frequency of occurrences. We first present the bit error rate analysis (BER) because of in-band distortion and then talk about the out-of-band distortion.

### 4.7.2.1 In-Band Distortion

We start with impulsive noise model and compare this with Gaussian model. The clip level crossings is elaborated as Poisson process in the literature. The rate of the Poisson process is determined by the power spectral density of the signal. Hence, the rate of the Poisson process is

$$\lambda = \frac{f_0}{\sqrt{3}} e^{-\frac{l^2}{2}},$$  \hspace{0.5cm} \text{(4.69)}

where $f_0 = N/T$ ($T$ is OFDM symbol duration) stands for the rectangular region in the power spectrum of the $x(t)$ and $P(C)$, probability of clip, is $2\lambda T$ for double-sided clipping. Also the length of the signal, which the signal stays above the clip level, is found to be (asymptotically) Rayleigh distributed where the expected value for the duration of a clip is given by
from where we can deduct that increase in the clip level \((l)\) decreases the crossing rate and duration. On the other hand, increase in the subcarriers \((N)\) increases the rate and duration. The BER analysis requires to find \(\text{Pr}(\text{error}|C)\), the error probability given that there is a clipping \(P(C)\). Then this is added to \(\text{Pr}(\text{error}|C^c)\), the error probability given that there is no clipping \(P(C^c)\). As a result, the error probability \(\text{Pr}(\text{error})\) is

\[
\text{Pr}(\text{error}) = \text{Pr}(\text{error}|C)\text{Pr}(C) + \text{Pr}(\text{error}|C^c)\text{Pr}(C^c).
\]  

\(\text{Pr}(\text{error}|C)\) is found to be

\[
4 \frac{(L-1)}{L} Q \left( \frac{3\pi l^2}{\sqrt{8(L^2-1)}} \right)^{1/3}
\]

with a square constellation of \(L^2\) points assumption.\(^{18}\) This is an upper bound and it is interesting to note that error due to clipping varies across subcarriers, the lower subcarriers dominate the overall error more than the others. The overall probability of symbol error is upper bounded by

\[
Q(z) = \frac{1}{\sqrt{2\pi}} \int_{z}^{\infty} e^{-\frac{u^2}{2}} du.
\]  

\(^{18}\)
Pr(error) = \frac{8N(L-1)}{\sqrt{3}L} e^{-\frac{i^2}{2}} Q\left(\frac{3\pi l^2}{\sqrt{8(L^2 - 1)}}\right)^{1/3}. \quad (4.73)

On the other hand, the BER for Gaussian model, which holds only if the clip level is low enough to permit more frequent occurrence per OFDM symbol is given by

Pr(error) = \frac{4N(L-1)}{L} Q\left(\frac{\sqrt{3}}{\sigma_c \sqrt{(L^2 - 1)}}\right), \quad (4.74)

where \( \sigma_c^2 \) is the power of the clipped portion and if signal power is normalized to unity, it is given by

\[ \sigma_c^2 = -\sqrt{\frac{2}{\pi}} l e^{-\frac{l^2}{2}} + 2(1 + l^2)Q(l). \quad (4.75) \]

Figure 4.53 plots these probabilities in addition to the BER performance in the receiver when AWGN and Rayleigh fading is present in the channel. One can see from the figure that if clipping level is very low, both models does not accurately model, since almost all signals are clipped below clip level 2.

Also around clip level 2, impulsive model is not that accurate as Gaussian model, since the impulses are considered to be concentrated not spread over time. Hence, this leads to greater probability of error as compared with real implementation. If clip level increases as can be seen that Gaussian noise only considers reduction
in the noise power but impulsive noise model considers spikes, which cause errors since small noise underestimates the error probability by several orders of magnitude. If we compare these with the BER performance in the receiver, illustrated in the same figure, we see that the performance is far worse than the one that obtained in the absence of fading, since now the channel introduces variable amplitude \( z \) and corresponding overall error probability is given by

\[
\Pr(\text{error}) = \frac{4\pi N(L - 1)}{L} \int_0^\infty z e^{-\frac{z^2}{2}} Q\left(\frac{3\pi l^2 z^2}{\sqrt{8(L^2 - 1)}}\right)^{1/3}.
\]

(4.76)

### 4.7.2.2 Out-of-Band Distortion

Out-of-band distortion (or spectral regrowth) caused by clipping widens the bandwidth. As can be seen from Fig. 4.54, spectral regrowth with decrease in clipping level directly increases adjacent channel interference and reduces the energy of in-band signal. This is important since spectrum leakage is subject to regulatory limits and directly determines the filtering requirements. In some applications, even if in-band distortion is acceptable or insignificant, out-of-band distortion is intolerable.

The PSD of spectral regrowth is given by

![Fig. 4.54 Spectrum](image-url)
for \( f > f_0 \), where we can see that if clip level is low, the spectral regrowth is high. Also, spectral regrowth decays with the frequency. This approach can be extended for impulsive nature, since bursty nature of clipping should be taken into account to get a realistic measure because instantaneous power spill may be averaged over and mislead system designers.

To remedy the out-of-band distortion, peak windowing may be applied, which multiplies large signal peak with a Gaussian shaped window. In practice, it convolves the original OFDM spectrum irrespective of number of subcarriers with the spectrum of the applied window. Cosine, Kaiser, and Hamming windows are suitable windows that have narrowband in frequency and not long in time domain. Peak windowing brings large reduction of PAPR regardless of number of subcarriers and has no effect to coding rate. As width of the window increases, the spectral regrowth decreases. BER performance on the other hand degrades with peak windowing, since it also distorts larger part of the unclipped signal.

This nonlinearity introduced to the system makes the system more vulnerable to errors. Hence, coding and scrambling techniques can be utilized along with clipping to increase the bit error rate performance and spectral efficiency.

### 4.7.3 Other Methods

Numerous techniques have been proposed to solve the PAPR problem. PAPR reduction techniques may fall into three categories: clipping, block coding, or peak cancelation where peak cancelation is an iterative way to subtract a known signal from the original signal to cancel the peaks. Block coding on the other hand adds redundancy to produce codes with low peaks. We review such existing techniques below.

- **Block coding** is one method to produce codes that achieves low PAPR. Of course, lower PAPR also limits the achievable code rate. Golay complementary sequences are a structured way to produce low PAPR codes with good FEC capabilities. Additional subcarriers are utilized in coding in order to achieve the error correction and PAPR reduction without distortion of the signal. If subcarriers have large amplitudes on the other hand, spectrum efficiency is poor and it requires large generation matrix.

- **Tone reservation (TR)** is one variant of block coding technique in which additional subcarriers carry no data but reserved for PAPR reduction. An effective cancelation signal in time domain can be found from only some number of additional subcarriers (aka reserved tones or null subcarriers) in the frequency domain

\[
\hat{x}[n] = x[n] + c[n] = \text{IFFT}(X_k + C_k),
\]  

\((4.78)\)
where $C_k$ stands for additional subcarriers. The new PAPR is

$$\text{PAPR}(\hat{x}) = \frac{||x + c||^2}{E(||x||^2)/N}, \quad (4.79)$$

since receiver ignores these additional subcarriers to recover the data. And one can see that $c[n]$ can be optimized to reduce the PAPR. Also PAPR reduction performance increases as the number of additional subcarriers increases, since the probability of constructing a cancelation signal increases. Notice also that distortion occurs only in the additional subcarriers, but not in the data carrying subcarriers. To construct the cancelation signal, TR employs a signal design algorithm, since signal must be designed in the frequency domain for its effect in the time domain. There are trial and error processes or computationally complex optimization procedures. The transmitter first checks for the peaks then for each peak TR method is performed. After peak cancelation, the composite signal is re-checked for secondary peaks that may appear during the peak cancelation.

- **Selective mapping (SLM)** utilizes redundant information and generates multiple instances of the same OFDM symbol to select the one with minimum PAPR. SLM reduces the PAPR by 2–3 dB, but requires side information at the receiver as well as multiple IFFT operation. The transmission of the side information is used to indicate the masking pattern, since the OFDM symbol is multiplied by $K$ ($K > 1$) different available phase vectors that has $N$ elements, each corresponds to each of the $N$ subcarriers. This generates $K$ statistically dependent OFDM symbols and the selected symbol is commonly referred to as the selected phase vector.

- **Partial transmit sequence (PTS)** creates $M$ subblocks, each of them are subjected to $L$-point IFFT and then multiplied by a phase vector to minimize the PAPR. At the end, subblocks are summed and transmitted. This way optimum phase can be created per subblock, but search complexity is exponential with the number of blocks and side information is required.

- **Dynamic range increase** is another PAPR mitigation method proposed to WiMAX-m (IEEE 802.16m) that increases the PA’s dynamic range to overcome high PAPR with low complexity. This is performed by envelope tracking so that VCC to the PA is raised to accommodate large peaks in the linear region.

- **Active constellation extension (ACE)** reduces the peak power by changing the signal constellation without affecting the BER performance, since minimum distance is preserved. For example in QPSK, there are four possible constellation points for each subcarrier where the transmitted bits could be mapped as seen in Fig. 4.55. These four points lie in each quadrant in the complex plane and are equidistant from the axis. A received data is assigned according to the quadrant in which the symbol is observed. Errors only occur if the received sample is mapped to one of the other three quadrants. Modification of the constellation points within the quarter plane is allowed in ACE, since this adds additional sinusoidal signals at the particular frequency to the transmitted signal. With correct
adjustment, these signals are used to cancel time-domain peaks in the transmitted OFDM signal.

In WiMAX, PAs must deliver more power, be more linear, and have the ability to handle a PAPR around 10 dB. This brings tight EVM\(^{19}\) requirement around −31 dB, based on 1% packet error rate. This enforces more linear component in the system and contributes WiMAX’s longer range with stringent receiver noise figure (7 dB maximum).

### 4.8 Application: IEEE 802.11a

OFDM came to prominence with IEEE 802.11a/g wireless local area networking standard.\(^{20}\)

IEEE 802.11a MAC is based on a random access scheme called CSMA/CA (carrier sense multiple access/collision avoidance\(^{21}\)) protocol, which only grants the

---

\(^{19}\) “The error vector magnitude or EVM is a measure used to quantify the deviation in the constellation points from the ideal locations due to various imperfections in the implementation such as carrier leakage, PA and D/A nonlinearity, phase noise, etc. These cause the actual constellation points to deviate from the ideal locations as seen in Fig. 4.55. Basically, EVM is a measure of how far the points are from the ideal locations.”

\(^{20}\) IEEE 802.11a-1999 (aka 802.11a) is an amendment to the IEEE 802.11 specification for operation in 5 GHz and IEEE 802.11g-2003 is another amendment to provide backward compatibility with IEEE 802.11b in 2.4 GHz.

\(^{21}\) Note that classic Ethernet uses CSMA/CD – collision detection.
channel by contention. In CSMA/CA, a wireless node that wants to transmit performs the following:

Step 1 Listen the channel.
Step 2 Transmit if channel is idle.
Step 3 If channel is busy, wait until transmission stops plus a contention period, which is a random period to ensure fairness. Contention period is quantified with a back-off counter where a node decrements the back-off counter if it detects channel idle for a fixed amount of time.
Step 4 Node transmits when back-off counter is zero.
Step 5 If the transmission is unsuccessful – no ACK, contention window is selected from a random interval, which is twice the previous random interval. The process is repeated until it gets a free channel.

Hidden node problem is solved with request-to-send (RTS) and clear-to-send (CTS) message exchange before transmitting the actual packet. If a station receives CTS packet from destination after sending the RTS packet, it reserves the channel for the duration of its packet plus a ACK. Packet size is upper bounded with the maximum packet size. Hence, if packet size is small, channel utilization is very low.

OFDM PHY transmits MAC protocol data units (MPDUs) as directed by the MAC layer. The OFDM PHY is composed of two elements: the physical layer convergence protocol (PLCP) and the physical medium dependent (PMD) sublayers. The PLCP prepares PLCP frame from MPDUs for transmission. The PLCP also delivers received frames from the air medium to the MAC layer. The PMD provides modulation and demodulation of the frame transmissions.

PLCP layer frame is illustrated in Fig. 4.56 and related key parameters are depicted in Table 4.2. There are two preamble sequences, each are two symbol length. First preamble contains ten short training sequences (STSs) and second preamble has two long training sequences (LTSs). First preamble is mostly for signal detection, automatic gain control, diversity selection, timing acquisition, and coarse frequency acquisition. Second preamble is used for channel estimation and fine frequency acquisition. Each symbol has a guard interval and first symbol after the second preamble contains information about rate, length, tail, service, etc., and always coded with a BPSK with coding rate of 1/2. Within each symbol also there are pilot subcarriers for frequency offset estimation and timing as seen in Fig. 4.57.

Table 4.3 shows the achievable physical layer data rates for IEEE 802.11a/g with convolutional coding. These numbers are raw rates and typically net throughput

Fig. 4.56 Format of an OFDM frame (© IEEE)
Table 4.2 IEEE 802.11a parameters

<table>
<thead>
<tr>
<th>IEEE 802.11a Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of data subcarriers $N_{SD}$</td>
<td>48</td>
</tr>
<tr>
<td>Number of pilot subcarriers $N_{SP}$</td>
<td>4</td>
</tr>
<tr>
<td>Number of subcarriers, total $N_{ST}$</td>
<td>52</td>
</tr>
<tr>
<td>Subcarrier frequency spacing $\Delta_f$</td>
<td>0.3125 MHz (=20 MHz/64)</td>
</tr>
<tr>
<td>IFFT/FFT period $T_{FFT}$</td>
<td>3.2 $\mu$s (1/$\Delta_f$)</td>
</tr>
<tr>
<td>PLCP preamble duration $T_{PREAMBLE}$</td>
<td>16$\mu$s ($T_{SHORT} + T_{LONG}$)</td>
</tr>
<tr>
<td>Duration of the SIGNAL $T_{SIGNAL}$</td>
<td>4.0$\mu$s ($T_{GI} + T_{FFT}$)</td>
</tr>
<tr>
<td>GI duration $T_{GI}$</td>
<td>0.8$\mu$s ($T_{FFT}$/4)</td>
</tr>
<tr>
<td>Training symbol GI duration $T_{GI2}$</td>
<td>1.6$\mu$s ($T_{FFT}$/2)</td>
</tr>
<tr>
<td>Symbol interval $T_{SYM}$</td>
<td>4$\mu$s ($T_{GI} + T_{FFT}$)</td>
</tr>
<tr>
<td>Short training sequence duration $T_{SHORT}$</td>
<td>8$\mu$s ($10 \times T_{FFT}$/4)</td>
</tr>
<tr>
<td>Long training sequence duration $T_{LONG}$</td>
<td>8$\mu$s ($T_{GI2} + 2 \times T_{FFT}$)</td>
</tr>
<tr>
<td>Signal Bandwidth $W$</td>
<td>16.66 MHz</td>
</tr>
</tbody>
</table>

![Fig. 4.57 OFDM subcarrier allocation for data and pilot](image)

Table 4.3 Achievable physical layer data rates with IEEE 802.11a

<table>
<thead>
<tr>
<th>Mode</th>
<th>Modulation</th>
<th>Code rate</th>
<th>Data rate (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BPSK</td>
<td>1/2</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>BPSK</td>
<td>3/4</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>QPSK</td>
<td>1/2</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>QPSK</td>
<td>3/4</td>
<td>18</td>
</tr>
<tr>
<td>5</td>
<td>16QAM</td>
<td>1/2</td>
<td>24</td>
</tr>
<tr>
<td>6</td>
<td>16QAM</td>
<td>3/4</td>
<td>36</td>
</tr>
<tr>
<td>7</td>
<td>64QAM</td>
<td>2/3</td>
<td>48</td>
</tr>
<tr>
<td>8</td>
<td>64QAM</td>
<td>3/4</td>
<td>54</td>
</tr>
</tbody>
</table>

is around 28 Mbps for 54 Mbps (with 54% inefficiency), which is achieved with 64QAM modulation and 3/4 coding rate in 20-MHz bandwidth.

### 4.9 Summary

In this chapter, we give principles of OFDM. We discuss OFDM theory and key components of OFDM transmission: coding, synchronization, channel estimation, equalization, and peak-to-average-power ratio. More detailed information about
OFDM and its applications can be found in *Multicarrier Digital Communications: Theory and Applications of OFDM*, published by Springer in 2004. Highlights about OFDM as a summary are:

- OFDM creates orthogonal spectral efficient low rate carriers in order to transmit high-rate signals.
- OFDM utilizes cyclic prefix in the guard interval in order to guarantee no ISI and ICI.
- OFDM is insensitive to timing offset but sensitive to frequency offset and phase noise.
- OFDM utilizes known preambles or pilot symbols for coherent detection and synchronization.
- OFDM does not need a time domain equalizer and needs only a simple frequency domain equalizer to correct amplitude and phase changes.
- OFDM may utilize a time domain equalizer to shorten the guard period.
- OFDM systems can utilize several coding schemes: Reed Solomon coding, convolutional coding, concatenated coding, Trellis coding, turbo coding, and LDPC coding.
- OFDM may show uncontrolled high peaks. Peak-to-average power ratio is a major problem and one way to alleviate is Clipping the high peaks: Clipping is a nonlinear process and it introduces distortion both inside and outside the given signal bandwidth.

**References**

88. Reed Solomon Codes. [http://komodo-industries.com/basic_reed-solomon_tutorial.html](http://komodo-industries.com/basic_reed-solomon_tutorial.html).