Complexity Efficient Decoder Design for Vehicular Communication

GWILHERM BAUDIC

KTH Electrical Engineering

Master's Degree Project
Stockholm, Sweden 2013

XR-EE-KT 2013:006

Abstract

Vehicular communication is currently seen as a key technology for enabling safer and more comfortable driving. In the general effort to reduce the number of casualties and improve the traffic flow despite an increasing number of vehicles, this field has a promising future. IEEE 802.11p has been chosen as the standard for the Physical Layer (PHY) design for wireless vehicular communication. However, the channels encountered in such situations pose several challenges for reliable communications. Time and frequency selectivity caused by dispersive environments and high mobility lead to doubly-selective channels. The systems are expected to conduct proper operation, in spite of these disturbances.

In this thesis, we focus on the design of receivers working on the PHY layer, with an emphasis on limited complexity. This poses high constraints on the algorithms, which already have to cope with the limited amount of information provided by the training sequences. The solutions considered all involve joint channel estimation and decoding, characterized by the use of an iterative structure. Such structures allow the channel estimation to benefit from the knowledge brought by the decoder, which ultimately decreases the error rate. Following a previous work, we use algorithms based on Minimum Mean Square Error (MMSE) or Maximum A Posteriori (MAP) estimation. These receivers were modified to operate on full frames instead of individual subcarriers, and various improvements were studied. We provide a detailed analysis of the complexity of the proposed designs, along with an evaluation of their decoding performance. The trade-offs between these two parameters are also discussed. A part of these analyses is used in [10]. Finally, we give an insight into some considerations which may arise when implementing the algorithms on testbeds.

Acknowledgments

First, I would like to express my great appreciation to Associate Prof. Tobias Oechtering from the Communication Theory department at KTH, who let me work on this topic and co-supervised the project. I am also grateful to Frédéric Gabry, Ph.D. student at KTH, for his advice and time taken to supervise my thesis. Their feedback helped to greatly improve this report. I would like to thank as well Olivier Goubet, student at KTH, who wrote the original algorithms this work aims to improve and extend. His help and suggestions, especially at the beginning of this thesis, were greatly appreciated.

At my home engineering school, ISAE-ENSICA, I would like to thank the International and the Careers services for allowing me to perform my master thesis at KTH. I am also particularly indebted to Prof. Jérôme Lacan who accepted to co-supervise my work.

Finally, I would like to thank my family for their constant support during my studies.

Notation

A'	Number of hypotheses for PDP support.
A	Number of hypotheses for DSD support.
D_a	Dimension of the dominant subspace of the submatrix representing hypothese a.
D	Dimension of the dominant subspace of the covariance matrix.
E_b	Energy per bit.
E_s	Energy per symbol.
F_s	Sampling frequency.
G_a	Probabilistic bound for the adaptive MMSE hypothesis test.
H	Channel coefficient.
$J_0(\cdot)$	Zero-th order Bessel function of the first kind.
K	Normalization factor for mapping.
L_{ch}	Number of paths.
L_{cp}	Length of the OFDM cyclic prefix.
L	Constraint length of the convolutional encoder.
M	Number of constellation symbols.
N_0	Noise spectral density.
N_a	Number of quantized amplitudes.
N_b	Number of data bits in the frame.
N_r	Number of shift registers.
N_s	Number of OFDM symbols.
N_{bpsc}	Number of coded bits per subcarrier.
N_{cbps}	Number of coded bits per OFDM symbol.
N_{csif}	Number of constellation symbols in the frame.
N_{it}	Number of iterations for the iterative algorithms.
N_{ph}	Number of quantized phases.
N_{scd}	Number of data subcarriers.
N_{scp}	Number of pilot subcarriers.
N_{scu}	Number of used subcarriers (for data or pilots).
N_{sc}	Number of OFDM subcarriers.
R_b	Bit rate.
R_s	Symbol rate.
R	Code rate.
S	State of the convolutional encoder.
T_s	Sampling interval.

T	OFDM symbol duration.
W_f	PDP support region.
W_t	DSD support region.
$\alpha_n(\cdot)$	Forward recursion metric in the BCJR algorithm.
$\beta_n(\cdot)$	Backward recursion metric in the BCJR algorithm.
$\boldsymbol{\sigma}$	Eigenvalue submatrix for the adaptive MMSE filter.
$\diamond$	Tracy-Singh product of column-wise partitioned matrices.
$\gamma_n(\cdot,\cdot)$	Branch metric in the BCJR algorithm.
$\hat{\mathbf{h}}$	Estimate of the vector $\mathbf{h}$.
$\lfloor\cdot\rfloor$	Floor function; that is, the largest integer not exceeding the parameter.
$\mathbf{R}$	Autocorrelation matrix for the Wiener filter.
$\mathbf{U}$	Eigenvector matrix or submatrix for the adaptive MMSE filter.
$\tilde{\mathbf{d}}$	Soft estimate of the vector $\mathbf{d}$.
$\mathbf{d}^H$	Hermitian transpose of the vector $\mathbf{d}$.
$\mathbf{d}^T$	Transpose of the vector $\mathbf{d}$.
$\mathbf{w}$	Wiener filter tap vector.
$\mathcal{S}'$	Transition set based on encoder output bits.
$\mathcal{S}$	Transition set based on encoder input bits.
$\mathcal{X}$	Symbol alphabet.
$\mathrm{diag}(\mathbf{d})$	Diagonal matrix with the elements of $\mathbf{d}$.
$\otimes$	Kronecker product.
σ^2	Noise variance.
*	Complex conjugation.
$a_n(\cdot)$	Forward recursion metric in the log BCJR algorithm.
$b_n(\cdot)$	Backward recursion metric in the log BCJR algorithm.
b	Information bit.
c	Coded bit.
f_c	Carrier frequency.
f_d	Maximum Doppler shift.
$g_n(\cdot,\cdot)$	Branch metric in the log BCJR algorithm.
m	Number of bits per constellation symbol.
q	Quantized channel coefficient for the MAP receiver.
x_a	Data error.
x	Symbol sent.
y	Received signal.
z_a	Reconstruction error.
z	Noise coefficient.

Acronyms

16-QAM	16 Quadrature Amplitude Modulation.
64-QAM	64 Quadrature Amplitude Modulation.
AWGN	Additive White Gaussian Noise.
BER	Bit Error Rate.
BPSK	Binary Phase-Shift Keying.
CLS	Comb Least Squares.
CP	Cyclic Prefix.
DFT	Discrete Fourier Transform.
DPSS	Discrete Prolate Spheroidal Sequences.
DS	Delay Spread.
DSD	Doppler power Spectral Density.
FDMA	Frequency Division Multiple Access.
FER	Frame Error Rate.
FFT	Fast Fourier Transform.
ICI	InterCarrier Interference.
IFFT	Inverse Fast Fourier Transform.
ISI	InterSymbol Interference.
ITS	Intelligent Transportation Systems.
ITU	International Telecommunication Union.
JCED	Joint Channel Estimation and Decoding.
LLR	Log-Likelihood Ratio.
LS	Least Squares.
MAP	Maximum A Posteriori.
MMSE	Minimum Mean Square Error.
OFDM	Orthogonal Frequency Division Multiplexing.
PAPR	Peak-to-Average Power Ratio.
PDF	Probability Density Function.
PDP	Power Delay Profile.

PHY	Physical Layer.
QPSK	Quadrature Phase-Shift Keying.
SISO	Soft-Input Soft-Output.
SNR	Signal-to-Noise Ratio.

Contents

Chapter 1

Introduction

This thesis deals with vehicular communication systems. In this chapter, we present some of the scenarios and specifications considered in this field. The current problems motivating this work will also be reviewed prior to introducing the structure and contents of this report.

1.1 An Overview of Vehicular Communication

As of today, vehicles are equipped with a whole range of systems and sensors aimed at improving the safety and comfort of their passengers. However, their field of action is mostly limited to the surrounding of the car. Distant events, such as crashes or traffic jams, can only be announced to the driver through road signs or specific FM channels - when this infrastructure exists. Vehicular communication aims to fill this gap, by making it possible to exchange data between vehicles without the need of such an infrastructure. They are therefore seen as a key part of Intelligent Transportation Systems (ITS).

The biggest interest of such systems lies in improving the safety in various situations where the knowledge of the sole close vicinity of the car would not be sufficient. For example, a safe lane change requires a careful assessment of the speed of arriving vehicles, but blind spots can make this difficult. When approaching an intersection with large objects hiding the other road, the risk of collision is increased. In the case of traffic congestions, the last vehicles should use their warning lights to inform incoming drivers; if not, accidents are more likely due to the lack of visual indications. Other scenarios are e.g. emergency vehicle warning, hazardous location notification, wrong-way driving warning, cooperative merging assistance or slow vehicle warning [13].

Another interest for vehicular communication would be to allow in-car entertainment more easily, although 4G networks could also be serious candidates for such uses. Access to the Internet, in-vehicle television or tourist information are some of the possibilities.

These use cases put several constraints on the underlying system. Safety-related scenarios require low latencies and low error rates to be able to efficiently play their role of avoiding accidents. On the other hand, applications such as television need high data rates. Thus, challenges appear on different levels, from a networking point of view, to the physical layer design. The biggest obstacle for the physical layer is to be able to simultaneously fulfill the requirements on data and error rates. This is made especially difficult by the nature of the wireless channel, as will be seen in Section 1.3.

1.2 IEEE 802.11p Specifications

Frequency Band ITS applications are to use a 70MHz spectrum at 5.9GHz, split in seven 10MHz channels. One is meant to be a control channel, while the others will be used for data transmission. It is interesting to note that in order to achieve higher data rates, some of these channels can be merged.

Frame Structure An Orthogonal Frequency Division Multiplexing (OFDM) system has been chosen as the modulation scheme. 64 subcarriers are defined, but only 52 are used for data transmission. The DC subcarrier is set to null to avoid dealing with a DC component at the receiver side. The remaining 11 subcarriers serve as guard bands to prevent frequency leakage. From the 52 subcarriers left, 48 actually transmit the data; the four others send pilot symbols called *comb pilots* to track the phase and frequency for the whole frame duration. A long preamble made of two pilot symbols, called *block pilots*, is also prepended to all subcarriers. A short preamble dedicated to coarse frequency synchronization and timing is sent before it on some subcarriers [3]; this will however not be considered in the simulations, because these symbols cannot be used for channel estimation. The resulting frame structure is summarized in Figure 1.1. While pilot symbols always use Binary Phase-Shift Keying (BPSK) [1], data symbols can be modulated with BPSK, Quadrature Phase-Shift Keying (QPSK), 16 Quadrature Amplitude Modulation (16-QAM) or 64 Quadrature Amplitude Modulation (64-QAM). Before that, the information bits are encoded with a convolutional code of rate 1/2. This rate is adjustable using two different puncturing patterns, which by dropping some of the bits leads to coding rates 1/2, 2/3 and 3/4.

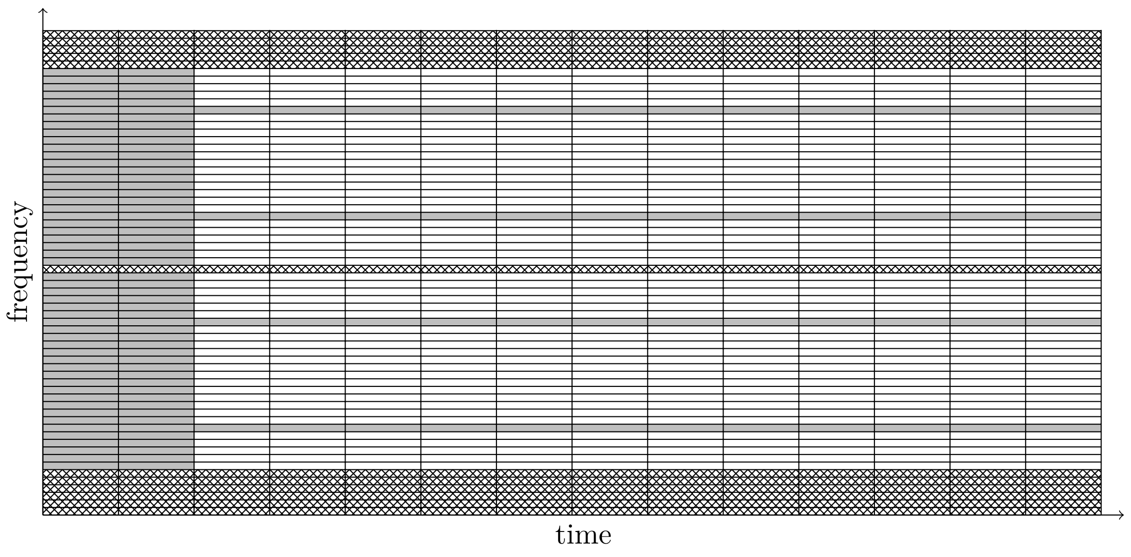


Figure 1.1: Structure of an OFDM frame. Pilots are in gray, data is plain white and null subcarriers are hatched.

Differences with 802.11a Despite being based on 802.11a, 802.11p includes some differences. While systems based on 802.11a communicate over 20MHz bands with a symbol duration of 4 μs, 802.11p uses 10MHz bands with a doubled symbol duration. Using a larger symbol duration makes 802.11p more robust against fading, but divides the achievable transmission rates by a factor of two. The different theoretical rates available are summarized in Table 1.1.

Modulation	Coding rate	Data rate (Mbps)
BPSK	1/2	3
BPSK	3/4	4.5
QPSK	1/2	6
QPSK	3/4	9
16-QAM	1/2	12
16-QAM	3/4	18
64-QAM	2/3	24
64-QAM	3/4	27

Table 1.1: Theoretical rates for 802.11p.

1.3 Current Issues

Since vehicles are by definition mobile, vehicular communications occur in time-varying fast-fading channels. Such channels are said to be *doubly-selective*, because they involve a spread both in time and frequency. It is therefore important to take these characteristics in account when designing a standard.

Thanks to the use of OFDM, the InterSymbol Interference (ISI) can be suppressed. Each subcarrier can then be seen as frequency-flat.

Vehicular communication systems also have to deal with fast variations of the channel. The classical approach for channel estimation consists of using block pilots to estimate the channel realization during the whole frame, for example with the average Least Squares method. While this gives good results for 802.11a, it works poorly for 802.11p [16], especially at high speeds and/or when using long frames. This loss of performance is due to the unsufficient density of pilot symbols. Consequently, more complex systems based on iterative designs and Joint Channel Estimation and Decoding (JCED) have been suggested to compensate this lack of information, such as [9]. This approach gives better results at the expense of much higher execution times.

1.4 Methodology

This thesis aims at improving and extending the work from [9]. In this previous work, a 802.11p transmitter has been implemented and three receiver designs (classical, MMSE and MAP) have been proposed, implemented and assessed with respect to their relative error performance. However, the codes were mostly limited to a single carrier scenario, and complexity was only partly analyzed through execution times.

In this work, the receivers were modified to operate on full frames, and the simplifications for the MAP receiver suggested in [9] have been added and tested. Since we aim at achieving a complexity efficient decoding, the theoretical complexity of the existing algorithms has been evaluated. New MMSE and MAP designs were then proposed along with other initialization methods and their complexities were investigated before implementation. Simulations were carried out using MATLAB.

1.5 Thesis Outline

This report is organized as follows.

In Chapter 2, the OFDM transmitter and the channel model used in the simulations are presented.

In Chapter 3, several receiver designs and the corresponding underlying theories are provided. We introduce and discuss the improvements made when applicable.

In Chapter 4, some elements on algorithm complexity are recalled. We subsequently apply them to the analysis of the receivers detailed in the previous chapter. Especially, analysis of the channel estimation parts can also be found in [10].

In Chapter 5, we assess the performance of the proposed algorithms, with respect to both achieved error rates and execution times [10]. We also compare our findings with previous works.

Finally, in Chapter 6 we draw some conclusions on the results and suggest areas of research for future work.

Chapter 2

Channel and Signal Model

In this chapter, we introduce the models which will subsequently be used in our numerical simulations. The characteristics of the vehicular channels are presented. The building blocks of the 802.11p transmitter are finally described.

2.1 Channel

In this section, the channel model will be discussed. In communication systems, the channel is responsible for various impairments to the original signal which must be accounted for at the receiver side. These impairments have to be reflected in the model by the simulated channel impulse response.

In the specific case of vehicular communications, three phenomena are altering the signal. Firstly, electronics in the device produce noise. Secondly, reflections on various objects lead to frequency selectivity. Finally, relative motion between the transmitter and the receiver make the channel time-varying. Taking in account these characteristics will eventually allow us to derive a model for the OFDM subcarriers.

2.1.1 Additive White Gaussian Noise

Gaussian random variables often arise in communication systems. This can be explained by the central limit theorem, which states that the sum of a sufficiently large number of independent and identically distributed random variables is well approximated as a Gaussian random variable. Here, it will be used to represent the electronic noise in the model. Since the number of electrons involved is high, the assumption of the theorem holds.

The chosen model is an additive white noise, i.e., an additional component with a constant spectral density and a Gaussian amplitude (hence the name of Additive White Gaussian Noise (AWGN)). The received signal can then be written as

$$r(t) = x(t) + z(t), \tag{2.1}$$

where $x(t)$ is the transmitted signal and $z(t)$ the noise component. In the complex case, the real and imaginary parts of the noise are considered independent and Gaussian distributed with variance $\sigma^2/2$. The complex noise consequently follows a circular symmetric complex normal distribution, with zero mean and variance σ^2, denoted by $z \sim \mathcal{CN}(0, \sigma^2)$. The corresponding Probability Density Function (PDF) is

$$p_z(z) = \frac{1}{\pi\sigma^2} \exp\left(-\frac{|z|^2}{\sigma^2}\right).$$

However, we will not deal directly with noise variances. Instead, a common measure is the Signal-to-Noise Ratio (SNR), especially in its normalized form: E_b/N_0, where E_b is the energy per bit and N_0 is the noise spectral density. The relation between the SNR and the noise variance is

$$\sigma^2 = \frac{E_s}{mR \left.\frac{E_b}{N_0}\right|_{lin}}, \tag{2.2}$$

where m is the number of bits per constellation symbol, R is the code rate and E_s is the energy per symbol.

SNRs are usually expressed in decibels (dB). For completeness, we recall here the relation between linear and dB values:

$$\left.\frac{E_b}{N_0}\right|_{dB} = 10\log_{10}\left(\left.\frac{E_b}{N_0}\right|_{lin}\right).$$

2.1.2 Multipath Channel

The channels encountered during vehicular communication will typically present a high number of scatterers: other vehicles, surrounding buildings, road infrastructure... The receiver then observes several copies of the original signal, each one with its own delay, phase and amplitude, as shown in Figure 2.1. These copies will interfere with each other, causing either constructive or destructive interference at the receiver side. This phenomenon accounts for the frequency selectivity of the channel. Such a channel is called time dispersive.

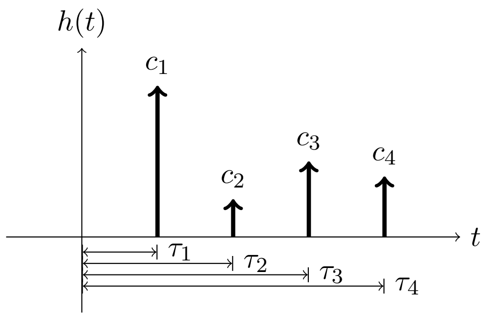


Figure 2.1: Multipath delay profile. Each tap models a different path with its attenuation and delay.

The impulse response of a multipath channel can be represented by a sum of impulses, called a tap delay line:

$$h(t,\tau) = \sum_{i=0}^{L_{ch}-1} c_i(t)\delta(\tau - \tau_i(t)).$$

We can notice that the complex coefficients and the delays of this impulse response will vary with time, due to the motion of the different components of the system. The time for which the channel can be considered constant is the *coherence time*. The number of coefficients used in the impulse response depends on what is seen as a significant level. Hence, an important metric of time dispersive channels is the Delay Spread (DS), which is defined as the maximum delay after which the received signal becomes negligible. A channel with a large delay spread will be modeled using more paths than a channel with a short delay spread.

The transmitted signal will go through a linear time-varying filter, which is

$$r(t) = (h * s)(t) = \sum_{i=0}^{L_{ch}-1} c_i(t) s(t - \tau_i(t)). \tag{2.3}$$

Following the choices of [16] and [9], we will use for the simulations the International Telecommunication Union (ITU) channel model, especially the Vehicular A and B [11]. It is part of a set of empirical channel models specified for three different environments: indoor office, outdoor-to-indoor pedestrian and vehicular. Using the same models will make comparisons of the results much easier. The parameters of the tap delay line for this empirical model are given in Table 2.1.

Tap	Channel A		Channel B	
	Delay (ns)	Average power (dB)	Delay (ns)	Average power (dB)
1	0	0.0	0	-2.5
2	310	-1.0	300	0.0
3	710	-9.0	8900	-12.8
4	1090	-10.0	12900	-10.0
5	1730	-15.0	17100	-25.2
6	2510	-20.0	20000	-16.0

Table 2.1: ITU Vehicular channel models [11].

It clearly appears that both ITU models have a delay spread which is higher than the length of the cyclic prefix (refer to 2.2.4 for more details). ISI can therefore be expected in the simulations.

Another remark is that delays for Channel A do not exactly fit the sampling grid of the simulation, which uses $T_s = 100$ns. Some methods to counter this are suggested in [16]: here, we choose to align the taps to the nearest point on the sampling grid.

This is not the only existing model when dealing with vehicular environments. In 2007, a set of six empirical channel models was proposed in [2], based on measurements collected near Atlanta, USA. In this article, the authors recommend to use the VTV Expressway Oncoming model because it has the highest frame error rate among the six.

Unlike the ITU models, all models from [2] have a delay spread below the length of the cyclic prefix, which according to [3] seems to be a more realistic assumption. However, a drawback of these models is that they do not take in account the non-stationarity of real channels. Consequently, in the remainder of this thesis, we will restrict ourselves to the ITU Vehicular channel model A.

2.1.3 Doppler Shift

When scatterers, the receiver or the transmitter are in motion, the frequency undergoes a shift called the Doppler shift. The maximum Doppler shift is given by

$$f_d = \frac{v}{\lambda} = f_c \cdot \frac{v}{c}, \tag{2.4}$$

where f_c is the carrier frequency, v the relative speed between the nodes, λ the wavelength and c the speed of light.

Considering the order of magnitude of the speeds, the Doppler shifts are relatively small. For example, we can take an extreme situation of two vehicles coming in opposite directions, each one driving at $v = 200$ km/h, trying to communicate with each other [23]. With $f_c = 5.9$ GHz, this leads to $f_d = 2.18$ kHz, which is small compared to the carrier frequency. Consequently, InterCarrier Interference (ICI) is not an issue. However, the Doppler shifting will introduce fading, which is a phenomenon that cannot be neglected. Furthermore, the different paths are likely to be affected by different frequency shifts. These mismatches cannot be totally suppressed by the carrier synchronization system [12].

The time-varying impulse response of the channel $h[n]$ can be modeled by Jakes spectrum, derived from the classical model by Clarke and whose formula for the discrete-time case is

$$h[n] = \sqrt{\frac{2}{L_{ch}}} \sum_{i=1}^{L_{ch}} (\cos\beta_i + j\sin\beta_i)\cos(2\pi f_d T_s n \cos\alpha_i + \theta_i), \tag{2.5}$$

where:

- L_{ch} is the number of paths,
- α_i is the angle of arrival from the scatterers,
- β_i is a random variable introduced to ensure that the I and Q components are uncorrelated and of equal power,
- θ_i is the initial phase of the i-th path,
- T_s is the sampling time,
- f_d is the maximum Doppler shift as in 2.4.

This is the spectrum which applies to all the taps in the ITU Vehicular channel models [11]. It corresponds to a frequency-flat Rayleigh fading process. In such situations, the autocorrelation function of the channel is found to be

$$R(\tau) = J_0(2\pi f_d \tau),$$

where $J_0(\cdot)$ denotes the zero-th order Bessel function of the first kind.

2.1.4 Modeling of the Channel

Discrete-time models will be used in the remainder of this thesis. The discrete-time time invariant channel impulse response, $h[m]$, is such that $h[m] = 0$ if $m < 0$ or $m \geq L_{ch}$. For the time varying case, we obtain

$$h[n, m] = h[m] \cdot h[n]. \tag{2.6}$$

Using Equation (2.6) to lead the discrete-time counterpart of Equation (2.3), we obtain

$$r[n] = h[n,m] * s[n] = \sum_{m=0}^{L_{ch}-1} s[n-m]h[n,m]. \tag{2.7}$$

Adding the noise coefficient forms the following model

$$r[n] = h[n,m] * s[n] + z[n]. \tag{2.8}$$

2.2 Transmitted Signal

In this section, the different components of the transmitter are described. They comply with the standard which was already partly described in Section 1.2. The transmitter is structured as follows.

First, the information bits are randomized by means of a scrambler. The scrambled data then goes through a convolutional encoder of rate 1/2, adaptable through puncturing. The coded bits are then interleaved, and the resulting sequence is subsequently mapped to constellations symbols, namely BPSK, QPSK, 16-QAM and 64-QAM. These symbols are assembled to produce OFDM symbols, and the pilot symbols are added. Finally, an Inverse Fast Fourier Transform (IFFT) is applied to perform pulse shaping. A complete description of an 802.11p-compliant receiver can also be found in [16].

2.2.1 Source

In order to make the transmitted bits appear equally likely, a scrambler is used. However, since this work focuses on the physical layer without any specific requirements on the type of data transmitted, randomly-generated bits are used in the simulations. This makes scrambling unnecessary.

The number of bits N_b to be generated depends on several parameters. Two formulas are possible, depending on whether termination or truncation (to be explained later in this section) are chosen. They are given respectively in Equations (2.9) and (2.10)

$$N_b = RmN_{scd}N_s - (L-1), \tag{2.9}$$

$$N_b = RmN_{scd}N_s, \tag{2.10}$$

where N_s is the number of OFDM symbols, N_{scd} the number of data subcarriers and L the constraint length of the encoder.

The bits can take the values "1" or "0". They are generated by a sequence of N_b values drawn from a uniform distribution on [0,1]: values larger than 0.5 are mapped to "1", while values lower than 0.5 are mapped to "0".

These bits are subsequently passed through a convolutional encoder in order to add some redundancy. This is especially important, because this additional information is not only used to detect and correct errors, but also to refine channel estimates in iterative receiver designs, such as the ones presented in Chapter 3.

A convolutional encoder can be represented with a series of shift registers, each one corresponding to a time delay. At each time index, the encoder takes n data bits as an input and generates k coded bits at the output, leading to a coding rate of n/k. In the standard, $n = 1$ and $k = 2$. Output bits are computed by modulo-2 addition of the values from a specific set of shift registers. The *constraint length* of the code is defined as $L = N_r + 1$, with N_r the number of shift registers. It represents the number of bits taken in account to compute a coded bit. The higher it is, the better the code will perform, at the expense of a higher complexity.

For 802.11p, the generator polynomials are $g_0 = 133_8 = [01011011]_2$ and $g_1 = 171_8 = [01111001]_2$. Their binary representations show which shift registers are connected to the modulo-2 adders: a "1" denotes a connection between the shift register and the modulo-2 adder, while a "0" signifies the absence of connection. They also tell us that this encoder uses $N_r = 6$ shift registers. The resulting diagram is Figure 2.2.

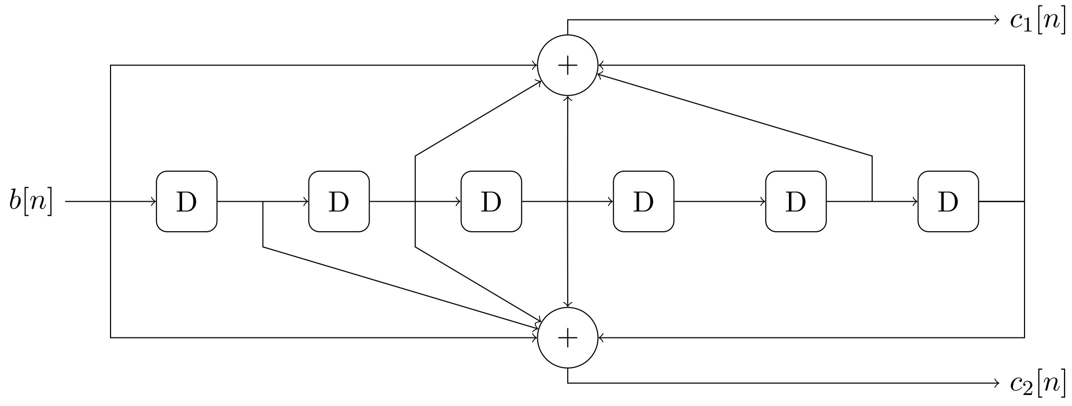


Figure 2.2: Convolutional encoder.

While convolutional encoders typically work with continuous streams of bits, the system defined in the standard uses frames, which are blocks of data. Converting a convolutional code into a block code can be done in two ways: truncation or termination.

Truncation consists of splitting the sequence of data bits into blocks of length N. These blocks are then fed to the receiver, which is reset to the all-zero state between each block. The main drawback of this method is that the last bits are less protected, since they will influence less output bits (or none).

Termination solves this issue by appending a sequence of N_r bits to the block of N data bits, so that the encoder is back to the all-zero state after encoding the sequence. For non-recursive codes such as the one used for 802.11p, N_r zeros will always satisfy this criterion. In this thesis, termination will be used since it leads to a more reliable decoding.

The IEEE 802.11p standard defines a base code of rate 1/2, described above, along with rates 2/3 and 3/4 achieved by puncturing after the encoding. Puncturing consists of transmitting only certain bits according to a pattern. This method relies on the error-correcting ability of the decoder. The puncturing vectors specified in the standard are $p_0 = [1110]$ and $p_1 = [110110]$, where a "0" indicates a deleted bit. With the first pattern, every fourth bit is not transmitted and with the second pattern, every third bit is not sent, eventually leading to code rates 2/3 and 3/4.

2.2.2 Interleaving

In wireless communications, errors typically come in bursts. Unfortunately, long runs of erroneous bits can have disastrous consequences, especially when convolution codes are being employed. Shuffling the coded bits allows to spread errors in a more uniform manner. In a multi-carrier system, interleaving across subcarriers also helps mitigate the effect of strong noise or fading on some of them.

IEEE 802.11p uses a block interleaving scheme. It consists of two permutations: the first one ensures that adjacent coded bits are mapped onto nonadjacent subcarriers, while the second ensures that adjacent coded bits are mapped alternately onto less and more significant bits of the constellation symbols [1].

The block size is the number of coded bits per OFDM symbol, N_{cbps}, which is computed as follows:

$$N_{cbps} = mN_{scd}, \tag{2.11}$$

with N_{scd} the number of data subcarriers in the frame.

In the following formulas, k denotes the index of the bit before any permutation, i its index after the first and j its index after the second permutation, when it is ready for mapping. The first permutation is defined as

$$i = \frac{N_{cbps}}{16}(k \bmod 16) + \left\lfloor \frac{k}{16} \right\rfloor, \quad k = 0, 1, \ldots, N_{cbps} - 1, \tag{2.12}$$

where $\lfloor \cdot \rfloor$ represents the largest integer not exceeding the parameter. It is equivalent to defining a matrix with 16 columns and a number of rows depending on the modulation scheme used; the matrix is filled row by row by the input bits and read column by column. The second permutation is

$$j = s \cdot \left\lfloor \frac{i}{s} \right\rfloor + \left(i + N_{cbps} - \left\lfloor \frac{16 \cdot i}{N_{cbps}} \right\rfloor \right) \bmod s, \quad i = 0, 1, \ldots, N_{cbps} - 1. \tag{2.13}$$

The value of s depends on the number of coded bits per subcarrier, N_{bpsc}, according to

$$s = \max \left(\frac{N_{bpsc}}{2}, 1 \right). \tag{2.14}$$

At this point, it can be noted that the second permutation will leave the bits unchanged when using BPSK or QPSK.

2.2.3 Mapping

In order to achieve a high bandwidth efficiency and high transmission rates, the coded bits are mapped to symbols which are usually non-binary. These symbols are chosen from an M-element symbol alphabet, denoted by $\mathcal{X}$. Each symbol can carry m bits, with $m = \log_2(M)$. It is noteworthy that the symbol rate R_s is limited by the available channel bandwidth, while the bit rate $R_b = mR_s$ can be arbitrarily increased by adjusting m. In practice, noise and available power at the transmitter tend to restrict the possible values for m.

The complex symbols correspond to the complex baseband signal. The set of points representing them in the complex plane is called the signal constellation. The design of the constellation aims to keep a low mean signal power, while maintaining a sufficient distance between symbols to minimize the error rate. These error rate concerns motivate the choice of Gray mapping. In this case, neighboring symbols differ by exactly one bit. Since choosing a neighboring symbol instead of the proper one is the most probable error, we ensure that, in most cases, a symbol error will lead to a single bit error.

The IEEE 802.11p standard defines four possible constellations: BPSK, QPSK, 16-QAM and 64-QAM, shown in Figure 2.3. In order to achieve equal average symbol power for all these constellations, a normalization factor K is applied. It is defined as

$$\frac{1}{K} = \frac{1}{M} \sum_{x \in \mathcal{X}} P_x,$$

with P_x the power of the symbol x. The resulting normalization factors are provided in Table 2.2.

Modulation	**K**
BPSK	1
QPSK	$\frac{1}{\sqrt{2}}$
16-QAM	$\frac{1}{\sqrt{10}}$
64-QAM	$\frac{1}{\sqrt{42}}$

Table 2.2: Normalization factors for constellations from [1].

2.2.4 The OFDM frame

IEEE 802.11p uses Orthogonal Frequency-Division Multiplexing, a multicarrier transmission technique which suppresses ISI over frequency-selective channels.

Principle The total bandwidth is divided into $N_{sc} = 64$ sub-bands, called subcarriers. Their bandwidth is chosen to be narrow enough so that the channel can be considered frequency-flat on each. This concept of bandwidth division is also used in Frequency Division Multiple Access (FDMA). However, while FDMA requires guard frequency intervals between the sub-bands to guarantee carrier independence, OFDM uses orthogonal and overlapping carriers, making it much more bandwidth efficient.

OFDM subcarriers are chosen among the eigenvectors of the channel, which ensures that they will remain orthogonal during the transfer through the channel. The complex exponential waveforms $\mathrm{e}^{j2\pi ft}$ always match this criterion [12].

There is an infinite number of such waveforms, defined over an infinite time horizon. For practical reasons, we will only consider a finite number of time-limited complex exponentials. The frequency and time spacing must be carefully chosen to maintain the eigenvector property and the orthogonality between the waveforms. These requirements will be only approximately met, due to the limits set on frequency and time. The first requirement will be fulfilled if the symbol period T is large compared to the channel DS. The second one will be met if the N_{sc} subcarrier frequencies are spaced apart by an integer multiple of $1/T$, that is $(f_n - f_m)T = k$, $\forall (n, m)$ with k a non-zero integer.

The transmitted complex baseband signal is then given by

$$u(t) = \frac{1}{\sqrt{N_{sc}}} \sum_{n=0}^{N_{sc}-1} S[n] \mathrm{e}^{j2\pi nt/T} \mathbb{I}_{[0,T]}(t), \tag{2.15}$$

where $S[n]$ is the n-th transmitted modulated constellation symbol.

Discrete-time Model Since most implementations (and computer simulations) are based on discrete-time baseband processing, we have to obtain a description of the considered signal in discrete time. The bandwidth of the transmitted OFDM signal $u(t)$ is approximately N_{sc}/T. In baseband representation, it is in fact spread between $N_{sc}/2T$ and $N_{sc}/2T$. According to the Nyquist-Shannon theorem, which states that a signal can be perfectly reconstructed when it is sampled at a frequency greater than twice its maximum frequency, we can sample $u(t)$ at a frequency $F_s = N_{sc}/T$ without losing information. $T_s = 1/F_s$ consequently denotes the sampling interval. The sampled version of the transmitted signal is therefore given by

$$u(kT_s) = \frac{1}{\sqrt{N_{sc}}} \sum_{n=0}^{N_{sc}-1} S[n] \mathrm{e}^{j2\pi nk/N_{sc}}. \tag{2.16}$$

This can be recognized as the inverse Discrete Fourier Transform (DFT) of the symbol sequence $S[n]$. We can write this

$$s[k] = u(kT_s) = \frac{1}{\sqrt{N_{sc}}} \sum_{n=0}^{N_{sc}-1} S[n] \mathrm{e}^{j2\pi nk/N_{sc}}. \tag{2.17}$$

If N_{sc} is a power of 2, this operation can be efficiently performed with an IFFT. The receiver can then do the inverse operation (namely, a Fast Fourier Transform (FFT)), which is written as follows:

$$R[n] = \frac{1}{\sqrt{N_{sc}}} \sum_{k=0}^{N_{sc}-1} r[k] \mathrm{e}^{-j2\pi nk/N_{sc}}. \tag{2.18}$$

The mapped symbols $S[n]$ are considered to be in the frequency domain and the transmitted samples $s[k]$ to be in the time domain. $S[n]$ is broken into N_{sc} parallel low rate streams before generating $s[k]$. Similarly, at the receiver, $r[n]$ is broken into N_{sc} parallel low rate streams.

OFDM and Multipath Channels The signal encounters multipath channels. The channel can be modeled using a Finite Impulse Response (FIR) filter of length L_{ch} (see also Equation (2.7)). After filtering, the sampled noiseless received signal is given by

$$r[m] = (h * b)[m] = \sum_{l=0}^{L_{ch}-1} h[l] s[m-l]. \tag{2.19}$$

The N_{sc}-point DFT of the channel impulse response, H, is

$$H[n] = \frac{1}{\sqrt{N_{sc}}} \sum_{l=0}^{L_{ch}-1} h[l] \mathrm{e}^{-j2\pi nl/N_{sc}}. \tag{2.20}$$

If we replaced the linear convolution in Equation (2.19) by a circular one, defined as

$$\tilde{r}[m] = (h \odot s)[m] = \sum_{l=0}^{N_{sc}-1} h[l \bmod N_{sc}] s[(m-l) \bmod N_{sc}], \tag{2.21}$$

we would obtain

$$\tilde{R}[n] = H[n] S[n], \tag{2.22}$$

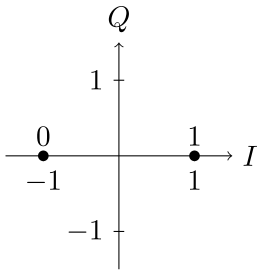
where it is clear that each subcarrier would be affected by a single channel coefficient. However, a linear convolution is equivalent to a circular convolution only if the signals have an infinite time horizon or if one of the signals is periodic; neither condition is verified here. We can nevertheless emulate a periodic signal (and satisfy the second condition) by appending a Cyclic Prefix (CP). It consists of sending the samples

$$s[k] = s[N_{sc} + k],\ k = -(L_{cp} - 1), \ldots, -1,$$

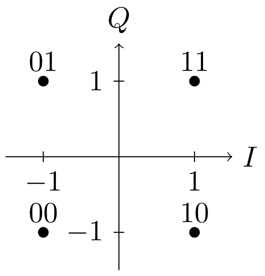
before $s[0], \ldots, s[N_{sc} - 1]$, or, in other words, appending a copy of the last $L_{cp} - 1$ samples of the OFDM symbol to its beginning. The receiver will discard the CP. In order to be able to generate the CP and efficiently avoid ISI, it clearly appears that L_{cp} must be smaller than N_{sc}; this influences the choice of the number of subcarriers. Moreover, adding a CP introduces an overhead of $(L_{cp} - 1)/N_{sc}$, which is reduced by choosing a high value for N_{sc}. More detailed descriptions of OFDM systems can be found in [12].

Advantages and Drawbacks OFDM has a high spectral efficiency. The narrow subcarrier frequency bands also make it especially relevant for dispersive channels, and the CP make it able to suppress ISI. Moreover, efficient solutions based on FFT and IFFT allow to implement it when using time-discrete processing, solutions which also have the benefit of distributing the complexity equally between the transmitter and the receiver. For classical receivers, complexity gains are even higher, because the complex equalizers commonly used to mitigate ISI can now be replaced by simple one-tap equalizers.

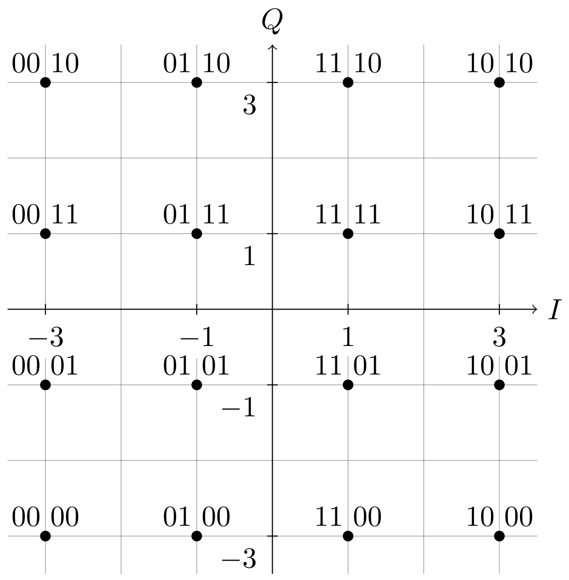
OFDM has also a few disadvantages: a high Peak-to-Average Power Ratio (PAPR) and possible ICI. ICI can be caused by improper carrier synchronization. To alleviate this issue, a certain number of carriers are dedicated to carrier synchronization. ICI may also be caused by time-varying channels, but this is unlikely to happen for vehicular communications because relative speeds would need to be higher than 1440 km/h [21].



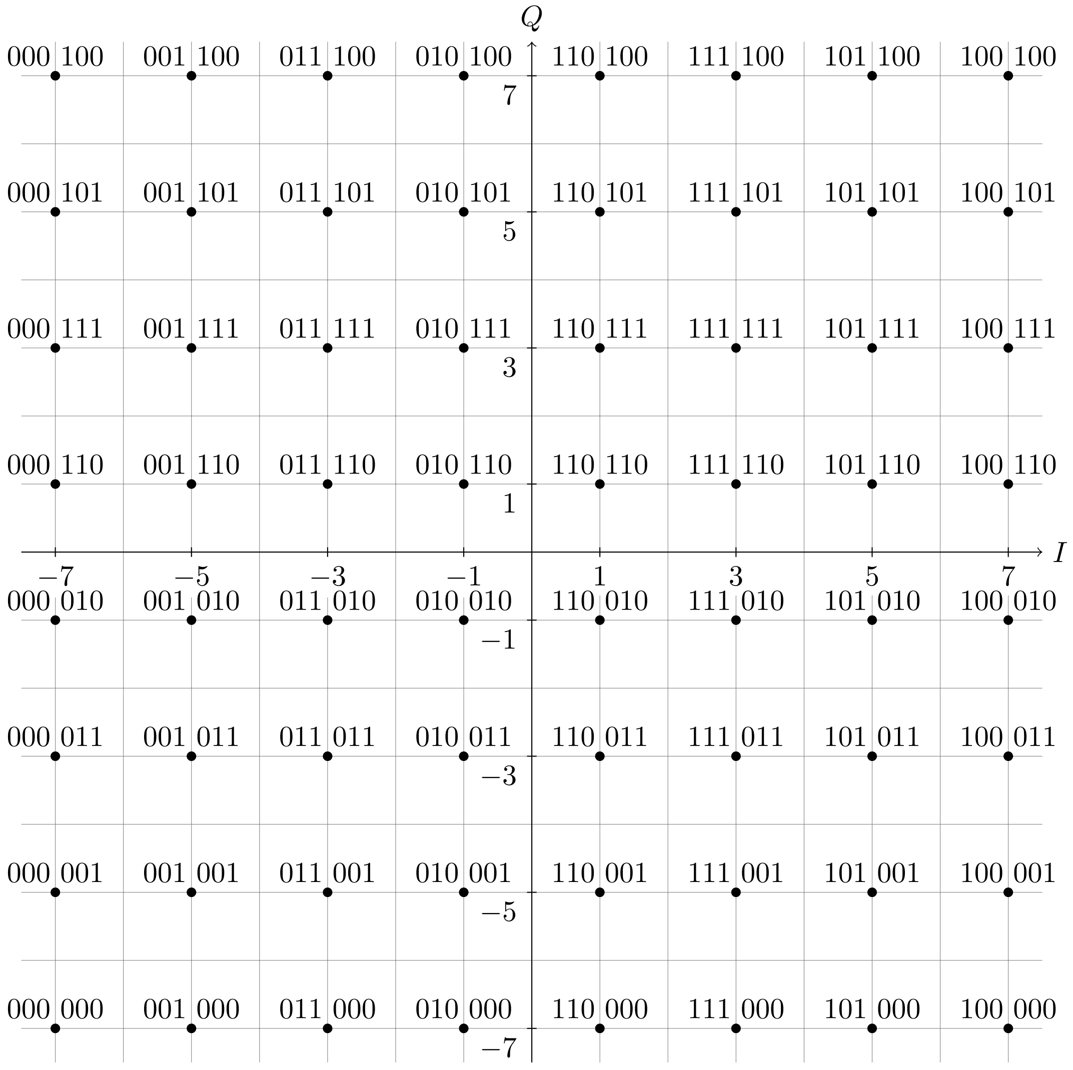
(a) BPSK



(b) QPSK



(c) 16-QAM



(d) 64-QAM

Figure 2.3: Non-normalized constellations for 802.11p from [1].

Chapter 3

Receiver Design

In this chapter, we turn to the design of the receivers proposed in this work. In Section 3.1, both classical and iterative receivers are presented. Two initialization methods are provided. In Section 3.2, the MMSE designs from [9] and [23] are recalled, and our modifications are explained. Finally, Section 3.3 introduces the MAP designs studied. Especially, we justify the modifications that were made to the receiver from [9].

3.1 Receiver Types

In this section, we present the two categories of receivers which are used in this work: classical receivers and iterative receivers. The initialization methods we propose are also provided.

3.1.1 Classical Receivers and Initializations

Classical receivers use a non-iterative algorithm. First, the channel is estimated using pilots. The resulting channel coefficients are then used by the equalizer, and the output is fed to a decoder after being demapped and deinterleaved. This design is summarized in Figure 3.1. Due to its poor performance with 802.11p [16], this receiver is mainly implemented for comparison purposes. Two initialization methods, using different parts of the frame pilots, are used: block LS (implemented in [9]) and comb LS with linear interpolation, as mentioned in [7].

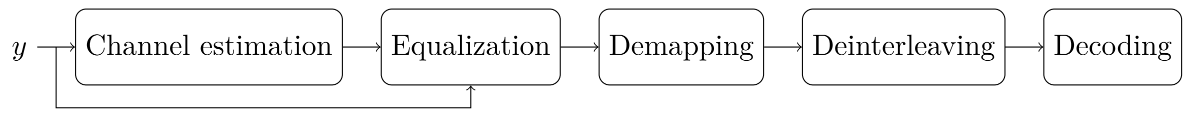


Figure 3.1: Structure of a classical receiver.

Block Pilots

The chosen frame structure adds a preamble made of two pilot symbols at the beginning of each subcarrier. In classical receivers, such as the ones used for 802.11a, these symbols are used to obtain the channel estimation for the subcarrier for the whole frame.

If we consider the subcarriers to be orthogonal and neglect any intersymbol interference, each subcarrier can be seen as a Gaussian channel. This leads to the following model for the n-th symbol on the k-th subcarrier [16]

$$y_k[n] = H_k[n]x_k[n] + z_k[n]. \tag{3.1}$$

The Least Squares (LS) estimate of the channel realization is then given by

$$\hat{H}_k[n] = \frac{y_k[n]}{x_k[n]} = H_k[n] + \frac{z_k[n]}{x_k[n]}. \tag{3.2}$$

It may then be advantageous to use both pilots symbols: this can be done with, for example, the average least squares method [16]. In this method, we first independently compute estimates for the two pilots according to Equation (3.2). To obtain the final channel estimate, we average the two estimates, leading to

$$\hat{H}_k = \frac{\hat{H}_k[1] + \hat{H}_k[2]}{2}. \tag{3.3}$$

This channel coefficient will subsequently be used to equalize the whole subcarrier.

Comb Pilots

In 802.11, the subcarriers entirely dedicated to pilots, referred to as comb pilots, are initially provided to help estimating the phase throughout the frame. The use of these pilots for channel estimation with 802.11p is evaluated in [7], where several initialization schemes using block and/or comb pilots are compared. According to the authors, comb pilots give good results at high SNR for moderate complexity, although their frequency spacing violates the sampling theorem.

We chose in our work the scheme called CLS-linear in [7], as it had a low complexity and was shown to outperform all other schemes for error rates at high SNR. In this scheme, least square estimates of the channel coefficients are first computed for all comb pilot symbols according to Equation (3.2). The channel coefficients for the other subcarriers are then obtained by linear interpolation on each OFDM symbol. In the remainder of this thesis, this scheme will be referred to as Comb Least Squares (CLS).

3.1.2 Iterative Receivers

Iterative receivers have been suggested as a way to refine channel estimations, and therefore improve the error rates. The main concept behind them is to exchange information between the decoder and the channel estimator. More precisely, it allows channel estimation to benefit from the error-correcting capacities of the convolutional code, by making it use the decoded data as soft pilots after the first iteration. In the case of vehicular communications, this ability becomes especially interesting to tackle the lack of information in the channel estimator, due to the unsufficient number of pilot symbols.

This receiver design is sometimes referred to as JCED, Joint Channel Estimation and Decoding. In the following sections, three methods are investigated and developed. Two use a MMSE estimation of the channel, while the third performs MAP estimation. In all cases (including classical receivers), channel estimation is done subcarrier by subcarrier, to take advantage of the OFDM ability to suppress ISI. The complete system will however operate on whole frames.

3.2 Iterative Receivers and MMSE Estimation

A MMSE design based on Wiener filtering was already implemented in [9] for a single subcarrier, based on the work in [20]. It used block pilots for its first iteration. In this work, this design was extended to a whole frame, and it was made compatible with both initialization methods detailed in Section 3.1.1. Furthermore, another receiver detailed in [23] was developed.

3.2.1 Wiener Filtering

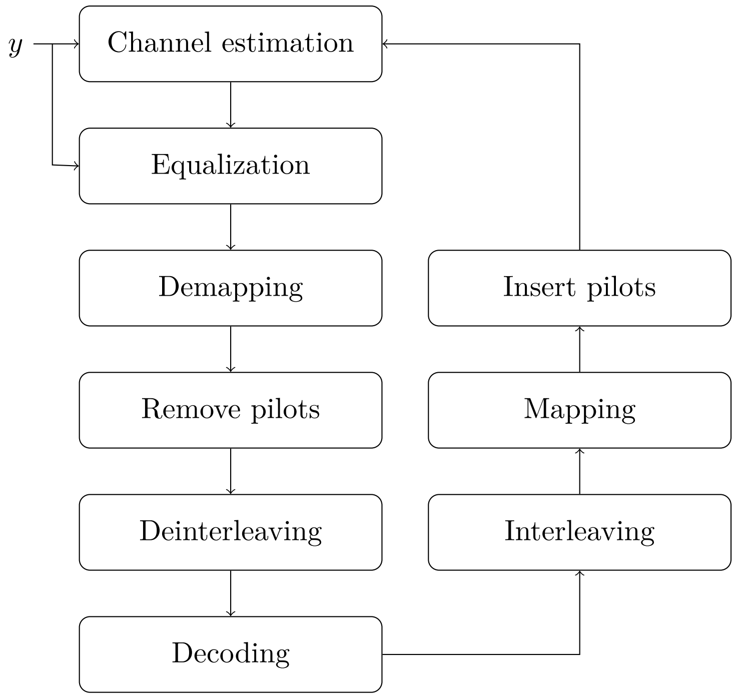


Figure 3.2: Structure of the MMSE receiver from [9].

Receiver Structure

For the time index n on the k-th subcarrier, we use the received signal model from Equation (3.1). The complete structure of the receiver is depicted in Figure 3.2. The channel estimator provides the estimated channel coefficients $\hat{H}_k[n]$. Each received sample $y_k[n]$ is processed by the demapper, which will output for the m coded bits of the transmitted symbol a Log-Likelihood Ratio (LLR) reflecting soft information on these bits and defined as

$$L_{k,n}^{(i)}(y) = \log\left(\frac{P(c_i=0|y_{k,n})}{P(c_i=1|y_{k,n})}\right) = \log\left(\frac{P(y_{k,n}|c_i=0)}{P(y_{k,n}|c_i=1)}\right), \tag{3.4}$$

for $i = 1, \ldots, m$. For example, using BPSK modulation, $m = 1$ and

$$L_{k,n}(y) = -\frac{2\sqrt{E_s}\mathrm{Re}(y_k[n]H_k^*[n])}{\sigma^2},$$

where * denotes complex conjugation, σ^2 is the noise variance and E_s the energy per symbol.

The BCJR decoder produces soft information on information bits b to make the decisions $\hat{b}$, and on coded bits c, which is used for the next iteration of the decoding algorithm. It is defined as a LLR feedback given by [9]

$$L_{k,n}^{(i)}(c) = \log\left(\frac{P(c_i = 0)}{P(c_i = 1)}\right). \tag{3.5}$$

From this feedback, the probabilities for the coded bit c_i are written as

$$\begin{cases} P(c_i = 0) = \dfrac{1}{1 + \mathrm{e}^{-L_{k,n}^{(i)}(c)}}, \\ P(c_i = 1) = \dfrac{\mathrm{e}^{-L_{k,n}^{(i)}(c)}}{1 + \mathrm{e}^{-L_{k,n}^{(i)}(c)}}. \end{cases} \tag{3.6}$$

Based on these probabilities, the mapper produces soft symbol estimates defined as $\hat{x}_{k,n} = \mathbb{E}\{X_{k,n}\}$. In the BPSK case, they are defined as [9]

$$\begin{aligned} \hat{x}_{k,n} &= \mathbb{E}\{X_{k,n}\} \\ &= \sum_{x \in \{-1,1\}} x \cdot P(X_{k,n} = x) \\ &= (-1) \cdot P(c_{k,n} = 0) + (1) \cdot P(c_{k,n} = 1) \\ &= (-1)\frac{1}{1 + \mathrm{e}^{-L_{k,n}^{(i)}(c)}} + (1)\frac{\mathrm{e}^{-L_{k,n}^{(i)}(c)}}{1 + \mathrm{e}^{-L_{k,n}(c)}} \\ &= -\tanh\left(\frac{L_{k,n}^{(i)}(c)}{2}\right). \end{aligned} \tag{3.7}$$

For the other modulation schemes, such expressions are much more complex. First, we provide an equation for coded bits similar to Equation (3.7) which can subsequently be used for all constellations, as follows.

$$\begin{aligned} \mathbb{E}\left\{c_{k,n}^{(i)}\right\} &= (0) \cdot P\left(c_{k,n}^{(i)} = 0\right) + (1) \cdot P\left(c_{k,n}^{(i)} = 1\right) \\ &= (1)\frac{\mathrm{e}^{-L_{k,n}^{(i)}(c)}}{1 + \mathrm{e}^{-L_{k,n}^{(i)}(c)}} \\ &= \frac{\mathrm{e}^{-2\frac{L_{k,n}^{(i)}(c)}{2}}}{1 + \mathrm{e}^{-2\frac{L_{k,n}^{(i)}(c)}{2}}} \\ &= \frac{1}{2}\left(1 - \frac{1 - \mathrm{e}^{-2\frac{L_{k,n}^{(i)}(c)}{2}}}{1 + \mathrm{e}^{-2\frac{L_{k,n}^{(i)}(c)}{2}}}\right) \\ &= \frac{1}{2}\left(1 - \tanh\left(\frac{L_{k,n}^{(i)}(c)}{2}\right)\right). \end{aligned} \tag{3.8}$$

Now that we have the a posteriori expectation on the coded bits, it becomes possible to express soft symbols for all constellations as functions of these expectations. Considering the mapping given in Section 2.2.3, we derive for 16-QAM and 64-QAM the following equations. The equation for QPSK [21] is also provided. For clarity, we drop the indices k, n and shorten $\mathbb{E}\left\{c_{k,n}^{(i)}\right\}$ as $\hat{c}^{(i)}$.

$$\hat{x}^{QPSK} = \frac{1}{\sqrt{2}}\left(2\hat{c}^{(1)} - 1 + j\left(2\hat{c}^{(2)} - 1\right)\right) \tag{3.9}$$

$$\hat{x}^{16QAM} = \frac{1}{\sqrt{10}}\left(6\hat{c}^{(1)} + 2\hat{c}^{(2)} - 4\hat{c}^{(1)}\hat{c}^{(2)} - 3 + j\left(6\hat{c}^{(3)} + 2\hat{c}^{(4)} - 4\hat{c}^{(3)}\hat{c}^{(4)} - 3\right)\right) \tag{3.10}$$

$$\begin{aligned}\hat{x}^{64QAM} = \frac{1}{\sqrt{42}}\Big(&8\hat{c}^{(1)}\hat{c}^{(2)}\hat{c}^{(3)} - 12\hat{c}^{(1)}\hat{c}^{(2)} - 4\hat{c}^{(1)}\hat{c}^{(3)} - 4\hat{c}^{(2)}\hat{c}^{(3)} + 14\hat{c}^{(1)} + 6\hat{c}^{(2)} + 2\hat{c}^{(3)} - 7\\ &+j\left(8\hat{c}^{(4)}\hat{c}^{(5)}\hat{c}^{(6)} - 12\hat{c}^{(4)}\hat{c}^{(5)} - 4\hat{c}^{(4)}\hat{c}^{(6)} - 4\hat{c}^{(5)}\hat{c}^{(6)} + 14\hat{c}^{(4)} + 6\hat{c}^{(5)} + 2\hat{c}^{(6)} - 7\right)\Big).\end{aligned} \tag{3.11}$$

Note that the formulas above produce symbols normalized by the K factors from Table 2.2.

Channel Estimation

In [9], a derivation of a channel estimator was made in the case of BPSK modulation. The signal model was modified, leading to, for a given sample [20]

$$y' = y \cdot x = H \cdot x^2 + z \cdot x = H + z', \tag{3.12}$$

exploiting the fact that $x^2 = 1$ with BPSK. However, this is not the case with other modulations; consequently, another model is proposed in this work, defined as

$$y'' = \frac{y}{x} = H + \frac{z}{x} = H + z'', \tag{3.13}$$

which holds for all constellation choices. Before we proceed with the actual channel estimation, we need to check the statistical properties of the noise terms z' and z'' of the new models. In both cases, the mean value of z (namely, 0) is left unchanged. However, the variances are modified: we have $\mathbb{E}\{z'^2\} = |x|^2\sigma^2$ and $\mathbb{E}\{z''^2\} = \sigma^2/(|x|^2)$. It becomes clear that with QAM constellations, the noise variance will be different for each sample, since the symbols may have different amplitudes. If the noise process has a time-varying variance, it can no longer be considered as Wide-Sense Stationary. Since the theory of the Wiener filter assumes stationary processes, this means we cannot use QAM modulations with this system. Consequently, the only constellations left for use with this algorithm are BPSK and QPSK. Since the first model only holds with BPSK, we will choose for this work the second model and use only PSK modulations.

The goal is to compute $\hat{H}_{k,n}$, the MMSE estimate of $H_{k,n}$. With the chosen model, the problem reduces to the estimation of a signal (the channel coefficients) in noise. It is possible to compute $\hat{H}_{k,n}$ as the Wiener filter estimate of $H_{k,n}$. First, we define some notation. The whole frame (including null and guard subcarriers) will be vectorized as $\mathbf{y}''$, where $y''_{k+N_{sc}(n-1)}$ corresponds to the sample on the k-th subcarrier of the n-th OFDM symbol. This notation will be applied similarly to the transmitted symbols x. We have

$$\begin{aligned}\hat{H}_{k,n} &= \mathbf{w}^T \mathbf{y}''_{k,n} \\ &= \sum_{l=-L}^{L} w_l y''_{k+N_{sc}(n-1)-l} \\ &= \sum_{l=-L}^{L} w_l \frac{y_{k+N_{sc}(n-1)-l}}{x_{k+N_{sc}(n-1)-l}}\end{aligned} \tag{3.14}$$

where $\mathbf{w} = [w_{-L}, w_{-L+1}, \ldots, w_L]^T$ and $\mathbf{y}''_{k,n} = [y''_{k+N_{sc}(n-1)+L}, y''_{k+N_{sc}(n-1)+L-1}, \ldots, y''_{k+N_{sc}(n-1)-L}]^T$. For samples corresponding to null subcarriers, the division is not feasible; hence, we set the corresponding coefficients in $\mathbf{y}''$ to zero. The tap vector $\mathbf{w}$ should minimize the quadratic error, which reads [9]

$$\begin{aligned}\frac{\partial}{\partial \mathbf{w}} \mathbb{E}\{|H_{k,n} - \hat{H}_{k,n}|^2\} &= 0 \\ \frac{\partial}{\partial \mathbf{w}} \mathbb{E}\{(H_{k,n} - \hat{H}_{k,n})^H (H_{k,n} - \hat{H}_{k,n})\} &= 0 \\ \mathbb{E}\{-2(H_{k,n} - \mathbf{w}^H \mathbf{y}''_{k,n}) \mathbf{y}''^H_{k,n}\} &= 0 \\ \mathbb{E}\{\mathbf{y}''_{k,n} \mathbf{y}''^H_{k,n}\} \mathbf{w} &= \mathbb{E}\{H_{k,n} \mathbf{y}''^H_{k,n}\}.\end{aligned} \tag{3.15}$$

The left part of the equation can be written [20]

$$\mathbb{E}\{\mathbf{y}''_{k,n} \mathbf{y}''^H_{k,n}\} = \mathbf{R} + \sigma^2 \mathbf{I}, \tag{3.16}$$

where $\mathbf{R}$ denotes the $(2K+1) \times (2K+1)$ autocorrelation matrix such that $\mathbf{R}_{ij} = [r(|i-j|)]_{-L \leq i,j \leq L}$ with $r(k) = J_0(2\pi f_d k T_s)$. r is essentially the discrete-time version of the autocorrelation function provided for the frequency-flat Rayleigh fading channel in Section 2.1.3. $\mathbf{I}$ denotes the $(2K+1) \times (2K+1)$ identity matrix. The right side of the last equation in 3.15 can be expressed as

$$\mathbb{E}\{H_{k,n} \mathbf{y}''^H_{k,n}\} = \mathbf{P}, \tag{3.17}$$

with $\mathbf{P} = [r(L), \ldots, r(1), r(0), r(1), \ldots, r(L)]^T$, the cross correlation vector [20]. $\mathbf{w}$ is the solution of the Wiener-Hopf equations [20]

$$\mathbf{w} = (\mathbf{R} + \sigma^2 \mathbf{I})^{-1} \mathbf{P}. \tag{3.18}$$

For initialization, the two methods detailed in Section 3.1.1 can be used on a subcarrier basis. More precisely, the block pilot scheme employs the "closest" estimated channel coefficient for each symbol on a given subcarrier [20], which is the one from the first pilot symbol for this first symbol and the one from the second block pilot for the rest of the subcarrier [9]. With the comb pilot scheme, linearly interpolated coefficients based on the ones computed for the pilot subcarriers are used for all symbols in each subcarrier - including the block pilots. In both cases, the estimated coefficients in the subsequent iterations (say, the i-th) will be computed as

$$\hat{H}^{(i)}_{k,n} = \sum_{l=-L}^{L} w_l \frac{y_{k+N_{sc}(n-1)-l}}{\hat{x}^{(i-1)}_{k+N_{sc}(n-1)-l}}. \tag{3.19}$$

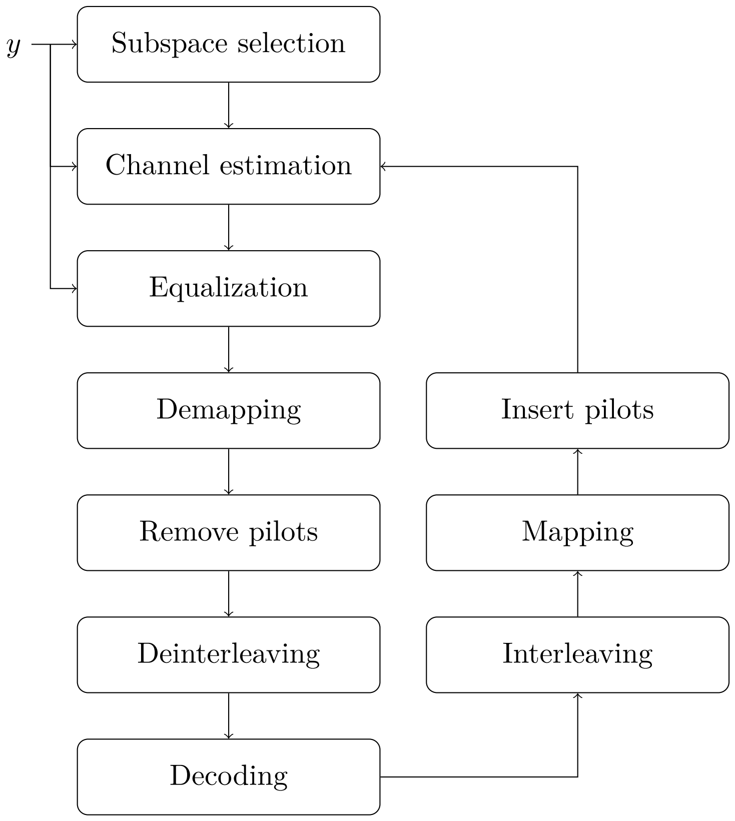


Figure 3.3: Structure of the MMSE receiver from [23].

3.2.2 Adaptive Reduced Rank Filter

An iterative MMSE receiver design using the Discrete Prolate Spheroidal Sequences (DPSS) [17] was proposed for 802.11p in [21]. A refinement, using a subspace adaptation in order to reduce complexity and accelerate the convergence, was first proposed in [22] and applied to vehicular communications in [23]. In this work, we implement this last algorithm, motivated by the interesting complexity and the low error rates achieved. Its structure is given in Figure 3.3.

Like the other MMSE design previously presented, [23] relies on a Wiener filter for channel estimation.

Notations

Before introducing the key parts of the algorithm, we need to define some notation. The Kronecker product will be denoted by $\otimes$, while the Tracy-Singh product [18] of column-wise partitioned matrices will use the $\diamond$ symbol. A coefficient c related to the n-th OFDM symbol on the k-th subcarrier will be written $c[n,k]$, and the corresponding vector $\mathbf{c}$ is defined as

$$\mathbf{c} = [c[1,1], \dots, c[1,N_{sc}], \dots, c[N_s+N_p,1], \dots, c[N_s+N_p,N_{sc}]]^T.$$

We apply this definition to the symbol $\mathbf{d}$, the channel coefficients $\mathbf{h}$, received data $\mathbf{y}$ and noise $\mathbf{z}$. Soft estimates of a coefficient c will be denoted $\tilde{c}$; in particular, the symbol vector $\tilde{\mathbf{d}}$ contains pilot symbols and soft data symbols.

The signal model is defined as

$$\mathbf{y} = \mathbf{D}\mathbf{h} + \mathbf{z}, \tag{3.20}$$

where $\mathbf{D} = \mathrm{diag}(\mathbf{d})$.

Wiener Filter Definition

A Wiener filter is used for channel estimation. The starting point for the developments made in [23] is

$$\hat{\mathbf{h}} = \mathbf{R}_h \tilde{\mathbf{D}}^{\mathrm{H}} \left(\tilde{\mathbf{D}} \mathbf{R}_h \tilde{\mathbf{D}}^{\mathrm{H}} + \mathbf{\Lambda} + \sigma^2 \mathbf{I}_{N_s N_{sc}} \right)^{-1} \mathbf{y}, \tag{3.21}$$

where $\mathbf{R}_h = \mathbb{E}\{\mathbf{h}\mathbf{h}^H\}$ is the covariance matrix of $\mathbf{h}$, and the diagonal matrix $\mathbf{\Lambda}$ is such that [23]

$$[\mathbf{\Lambda}]_{n+N_s k, n+N_s k} = [\mathbf{R}_h]_{n+N_s k, n+N_s k} \left(1 - \left| \tilde{d}[n,k] \right|^2 \right). \tag{3.22}$$

The matrix which needs to be inverted has a size of $N_s N_{sc}$, which would make its inversion particularly costly. This motivates the authors to find the dominant subspace of the covariance matrix and develop a reduced-rank implementation to decrease the complexity.

However, this matrix is not known at the receiver side. $\mathbf{R}_h$ will therefore be approximated by $\tilde{\mathbf{R}}_h$, which assumes a delay-Doppler scattering function prototype with flat spectrum in a region defined as [23]

$$\mathcal{W} = W_t \times W_f = [-\nu_D, \nu_D] \times [0, \theta_P], \tag{3.23}$$

where W_t defines the support region of the Doppler power Spectral Density (DSD) with $0 \leq \nu_D \leq \nu_{Dmax}$, and W_f the support region of the Power Delay Profile (PDP) with $0 \leq \theta_P \leq \theta_{Pmax}$. In these inequalities, ν_{Dmax} is the one-sided maximum normalized Doppler bandwidth, that is $\nu_{Dmax} = f_{Dmax} T$, and θ_{Pmax} is the maximum normalized path delay such that $\theta_{Pmax} = \tau_{max}/(T_s N_{sc})$.

The process is observed on a finite index set

$$\mathcal{I} = I_t \times I_f = [0, \ldots, N_s - 1] \times [-N_{scu}/2, N_{scu}/2 - 1]. \tag{3.24}$$

I_t is the observation interval in the time domain, and I_f its counterpart in the frequency domain.

Factorization

The covariance matrix $\tilde{\mathbf{R}}_h$ can now be factorized as [23]

$$\tilde{\mathbf{R}}_h = \mathbf{R}(W_t, I_t) \otimes \mathbf{R}(W_f, I_f), \tag{3.25}$$

where $\mathbf{R}(W, I)$ has elements

$$\mathbf{R}(W, I)_{k,l} = \frac{1}{|W|} C[k - l, W] \tag{3.26}$$

at its k-th line and l-th column, with $k, l \in I$ and $W = [\nu_1, \nu_2]$ with

$$
\begin{aligned}
C[k,W] = \int_W \mathrm{e}^{j2\pi k\nu}\,\mathrm{d}\nu &= \frac{1}{j2\pi k}(\mathrm{e}^{j2\pi k\nu_2} - \mathrm{e}^{j2\pi k\nu_1}) \\
&= \frac{1}{j2\pi k}\,\mathrm{e}^{j2\pi k\nu_1}(\mathrm{e}^{j2\pi k|W|} - 1) \\
&= \frac{1}{j2\pi k}\,\mathrm{e}^{j2\pi k\nu_1}\,\mathrm{e}^{j2\pi k|W|/2}(\mathrm{e}^{j2\pi k|W|/2} - \mathrm{e}^{-j2\pi k|W|/2}) \\
&= \frac{\sin\left(2\pi k\frac{|W|}{2}\right)}{\pi k}\,\mathrm{e}^{j2\pi k\frac{\nu_1+\nu_2}{2}}.
\end{aligned}
\tag{3.27}
$$

The eigenvector decomposition of $\tilde{\mathbf{R}}_h$ can now be performed. Since this matrix has a high condition number, specific algorithms will be used, as follows. They take advantage of the factorization given in Equation (3.25). The eigenvectors of $\mathbf{R}(W,I)$ are the generalized DPSS $u_i[m,W,I]$, $i \in I$ for the band limit W and time-limited to the observation interval $m \in I$ [17]. They verify the identity

$$\mathbf{R}(W,I)\mathbf{u}_i(W,I) = \lambda_i(W,I)\mathbf{u}_i(W,I), \tag{3.28}$$

where $\mathbf{u}_i(W,I) = [u_i(0,W,I),\ldots,u_i(|I|-1,W,I)]^T$. If we denote by $\mathbf{U}(W,I)$ the matrix whose columns are given by the generalized DPSS $\mathbf{u}_i(W,I)$, and by $\boldsymbol{\sigma}(W,I)$ the eigenvalue matrix with diagonal elements $\lambda_i(W,I)$, we can write

$$\mathbf{R}(W,I) = \mathbf{U}(W,I)\boldsymbol{\sigma}(W,I)\mathbf{U}(W,I)^H. \tag{3.29}$$

In order to solve this eigenvalue problem, [23] uses a result given in [17]. It states that the $N \times N$ tridiagonal matrix $\mathbf{S}$, defined as

$$
\mathbf{S}(N,W)_{i,j} = \begin{cases}
\frac{1}{2}i(N-i), & j = i-1 \\
\left(\frac{N-1}{2} - i\right)^2 \cos 2\pi W, & j = i \\
\frac{1}{2}(i+1)(N-1-i), & j = i+1 \\
0, & |j-i| > 1,
\end{cases}
\tag{3.30}
$$

with $i,j = 0,1,\ldots,N-1$, has the same eigenvectors (namely, the DPSS) as the $N \times N$ matrix $\boldsymbol{\rho}$ with elements

$$\boldsymbol{\rho}(N,W)_{k,l} = \frac{\sin 2\pi W(k-l)}{\pi(k-l)},\quad k,l = 0,1,\ldots,N-1. \tag{3.31}$$

According to Equations (3.26) and (3.27), the matrices $\mathbf{R}(W,I)$ have coefficients

$$\mathbf{R}(W,I)_{k,l} = \frac{1}{|W|}\,\frac{\sin\left(2\pi(k-l)\frac{|W|}{2}\right)}{\pi(k-l)}\,\mathrm{e}^{j2\pi(k-l)\frac{\nu_1+\nu_2}{2}}. \tag{3.32}$$

The DSD support region, W_t, is symmetric, which means $\nu_1 + \nu_2 = 0$. The exponential then equals 1, and it becomes possible to use [17] if taking care of the multiplicative constant $1/|W|$. On the contrary, the PDP support is asymmetric, and the exponential term is different for each matrix coefficient (due to the presence of the $(k-l)$ term). For this reason, we believe that in this case the result in [17] cannot be applied. Consequently, our implementations will call MATLAB's eigensolver for these matrices.

The aim is to obtain an eigenvector decomposition of $\tilde{\mathbf{R}}_h$, defined as [23]

$$\tilde{\mathbf{R}}_h = \mathbf{U}\boldsymbol{\Sigma}\mathbf{U}^H. \tag{3.33}$$

Based on Equation (3.25), the eigenvector matrix $\mathbf{U}$ is factorized as [23]

$$\mathbf{U} = \mathbf{U}(\mathcal{W},\mathcal{I}) = \Pi(\mathbf{U}(W_t, I_t) \diamond \mathbf{U}(W_f, I_f)) \tag{3.34}$$

and the diagonal eigenvalue matrix $\boldsymbol{\Sigma}$ can be written [23]

$$\boldsymbol{\Sigma} = \boldsymbol{\Sigma}(\mathcal{W},\mathcal{I}) = \Pi(\mathrm{diag}(\boldsymbol{\sigma}(W_t, I_t) \otimes \boldsymbol{\sigma}(W_f, I_f))). \tag{3.35}$$

The permutation operator $\Pi(\cdot)$ is chosen such that the eigenvalues in $\boldsymbol{\Sigma}$ - and the corresponding columns in $\mathbf{U}$ - are sorted according to $\lambda_0(\mathcal{W},\mathcal{I}) \geq \lambda_1(\mathcal{W},\mathcal{I}) \geq \cdots \geq \lambda_{(|I|-1)}(\mathcal{W},\mathcal{I})$.

Low-Complexity Reduced Rank Filter

To reduce the rank, we will only consider the D dominant eigenvectors to approximate the covariance matrix $\tilde{\mathbf{R}}_h$ as $\breve{\mathbf{R}}_h$, which reads [23]

$$\tilde{\mathbf{R}}_h \approx \breve{\mathbf{R}}_h = \mathbf{U}_D\boldsymbol{\Sigma}_D\mathbf{U}_D^H, \tag{3.36}$$

where $\mathbf{U}_D$ and $\boldsymbol{\Sigma}_D$ contain the first D columns of $\mathbf{U}$ and $\boldsymbol{\Sigma}$ respectively. The dimension $D = D(\mathcal{W},\mathcal{I})$ minimizing the Mean Square Error for a given noise variance σ^2 is, according to [22] and [15],

$$D = \underset{\mathcal{D}\in\{1,\ldots,|\mathcal{I}|\}}{\arg\min} \left(\frac{1}{|\mathcal{I}|} \sum_{i=\mathcal{D}}^{|\mathcal{I}|-1} \lambda_i(\mathcal{W},\mathcal{I}) + \frac{\mathcal{D}}{|\mathcal{I}|}\sigma^2 \right), \tag{3.37}$$

where the two summands represent the square bias and the variance term. Compared to Equation (30) in [23], a factor $1/|\mathcal{W}|$ is missing in front of the first summand. Considering the original formula in [15] and the subsequent derivations made in [22], this factor is in fact not needed if we consider, as it is the case here, that the $\lambda_i(\mathcal{W},\mathcal{I})$ are the eigenvalues of the $\mathbf{R}(W, I)$ matrices and not of the C matrix whose coefficients are given in Equation (3.27).

Inserting (3.36) into (3.21), and applying the matrix inversion lemma, the expression for the reduced-rank Wiener filter is found to be [23]

$$\hat{\mathbf{h}} = \mathbf{U}_\mathrm{D} \left(\mathbf{U}_\mathrm{D}^\mathrm{H}\tilde{\mathbf{D}}^\mathrm{H}\boldsymbol{\Psi}^{-1}\tilde{\mathbf{D}}\mathbf{U}_\mathrm{D} + \boldsymbol{\Sigma}_\mathrm{D}^{-1} \right)^{-1} \mathbf{U}_\mathrm{D}^\mathrm{H}\tilde{\mathbf{D}}^\mathrm{H}\boldsymbol{\Psi}^{-1}\mathbf{y}, \tag{3.38}$$

where $\boldsymbol{\Psi} = \boldsymbol{\Lambda} + \sigma^2\mathbf{I}_{N_sN_{sc}}$.

Subspace Selection

First, finite sets of hypotheses are defined for the DSD support $\{W_t(1), \ldots, W_t(A)\}$ and for the PDP support $\{W_f(1), \ldots, W_f(A')\}$. These different hypotheses will be tested for each received frame.

They are respectively defined as [23]

$$W_t(a) = \left(-\frac{a}{A}\nu_{Dmax}, \frac{a}{A}\nu_{Dmax} \right) \tag{3.39}$$

and

$$W_f(a') = \left(0, \frac{a'}{A'}\theta_{Pmax}\right), \tag{3.40}$$

with $a \in \{1, \ldots, A\}$ and $a' \in \{1, \ldots, A'\}$.

Each hypothesis is represented by a subspace spanned by the columns of $\mathbf{U}(W_t(a), I_t)$ in the first case or $\mathbf{U}(W_f(a'), I_f)$ in the second case. These matrices are precomputed and stored.

Once the hypotheses are set up, they need to be tested to choose the correct subspace. This test will be made on a subset of the pilot symbols of the frame. For this purpose, a signal model for channel estimation only at the pilot positions is defined as follows

$$\mathbf{y}^{(\mathcal{P})} = \mathbf{D}^{(\mathcal{P})}\mathbf{h}^{(\mathcal{P})} + \mathbf{z}^{(\mathcal{P})}, \tag{3.41}$$

where the different elements are defined as in Equation (3.20), except that they only contain the elements at pilot positions *in the same order as* in $\mathbf{y}$. The channel observations at pilot positions are therefore given by [23]

$$\mathbf{w}^{(\mathcal{P})} = \mathbf{D}^{(\mathcal{P})^H}\mathbf{y}^{(\mathcal{P})} = \mathbf{h}^{(\mathcal{P})} + \mathbf{D}^{(\mathcal{P})^H}\mathbf{z}^{(\mathcal{P})} = \mathbf{h}^{(\mathcal{P})} + \mathbf{z}'^{(\mathcal{P})}, \tag{3.42}$$

where $\mathbf{z}'^{(\mathcal{P})} \sim \mathcal{CN}(0, \sigma^2\mathbf{I}_{|\mathcal{P}|})$ has the same statistics as $\mathbf{z}^{(\mathcal{P})}$ (recall that pilot symbols are BPSK modulated, so their amplitude always equals one, which leaves the noise variance unchanged).

In [23], the authors state that the subspace spanned by the observations in the vector $\mathbf{h}^{(\mathcal{P})}$ is also spanned by the eigenvectors of the covariance matrix

$$\tilde{\mathbf{R}}_{\mathbf{h}^{(\mathcal{P})}} = \mathbb{E}\left\{\mathbf{h}^{(\mathcal{P})}\mathbf{h}^{(\mathcal{P})^H}\right\}. \tag{3.43}$$

Due to the pilot pattern, a time-frequency factorization such as the one which was discussed for the whole covariance matrix is not possible. Instead, the pilots are split into two parts which enable a factorization: comb pilots and block pilots.

More precisely, the comb pilots index set is defined as $\mathcal{T} = I_t \times I_{\mathcal{P}_f}$, where $I_{\mathcal{P}_f} = \{k_1, k_2, k_3, k_4\}$ is the set of comb pilots subcarrier frequencies. The corresponding signal model is given by Equations (3.42) and (3.43) by replacing $(\mathcal{P})$ by $(\mathcal{T})$. The subspace spanned by $\mathbf{h}^{(\mathcal{T})}$ is also spanned by the columns of the matrix $\mathbf{U}(\mathcal{W}, \mathcal{T})$.

The block pilots index set is defined similarly as $\mathcal{F} = I_{\mathcal{P}_t} \times I_f$, where $I_{\mathcal{P}_t} = \{n_1, n_2\}$. The subspace spanned by $\mathbf{h}^{(\mathcal{F})}$ will also be spanned by the columns of the matrix $\mathbf{U}(\mathcal{W}, \mathcal{F})$. It should be noted at this point that in [23], the time indexes considered are not the two block pilots specified by the standard at the beginning of the frame, but the first of these two and an additional block at the end of the frame. This modification, commonly referred to as a "postamble", is not part of the IEEE 802.11p standard.

We now want to find the best hypothesis for the DSD support from $\{W_t(a), a \in \{1, \ldots, A\}\}$ and for the PDP support from $\{W_f(a'), a' \in \{1, \ldots, A'\}\}$. At first glance, this would require the test of AA' hypotheses. Consequently, the authors propose in [23] a reduced complexity algorithm which needs to test only $A + A'$ different hypotheses, as follows.

First, the DSD support is tested on the subcarrier index set $\mathcal{T}$. The PDP is assumed to have maximum support $W_f(A')$. As previously mentioned, the subspace spanned by the values of $\mathbf{h}^{(\mathcal{T})}$ can be approximated by the columns of the matrix $\mathbf{U}(W_t(a) \times W_f(A'), I_t \times I_{\mathcal{P}_f})$. Once generated from the precomputed $\mathbf{U}$ hypothesis matrices, its columns are sorted by means of the permutation Π used in Equation (3.34). The first

$D_a = D(W_t(a) \times W_f(A'), I_t \times I_{\mathcal{P}_f})$ columns are then collected in matrix $\mathbf{U}_a$, while the remaining columns make the $\mathbf{V}_a$ matrix. With $\mathbf{U}_a$, $a \in \{1, \ldots, A\}$, we can obtain A reduced-rank maximum likelihood channel estimates

$$\hat{\mathbf{h}}_a = \mathbf{U}_a \mathbf{U}_a^H \mathbf{w}^{(\mathcal{T})} \tag{3.44}$$

testing all A hypotheses. The data error is expressed as [23]

$$x_a = \frac{1}{|\mathcal{T}|} \|\mathbf{w}^{(\mathcal{T})} - \hat{\mathbf{h}}_a\|^2. \tag{3.45}$$

The metric to be minimized is instead the reconstruction error

$$z_a = \frac{1}{|\mathcal{T}|} \|\mathbf{h}^{(\mathcal{T})} - \hat{\mathbf{h}}_a\|^2 = \frac{1}{|\mathcal{T}|} \left(\left\| \mathbf{U}_a^H \mathbf{z}^{(\mathcal{T})} \right\|^2 + \left\| \mathbf{V}_a^H \mathbf{h}^{(\mathcal{T})} \right\|^2 \right), \tag{3.46}$$

which is unknown at the receiver side since we cannot directly observe $\mathbf{z}^{(\mathcal{T})}$. Knowing x_a, we are interested in obtaining a probabilistic upper bound on z_a, such that $z_a < \overline{z_a}(x_a, p_1, p_2)$, which only depends on x_a and two constant probabilities, p_1 and p_2. This bound allows to select the best hypothesis $W_t(\hat{a})$, defined as

$$\hat{a} = \arg\min_{a} \overline{z_a}(x_a, p_1, p_2), \tag{3.47}$$

which minimizes the reconstruction error. The detailed computation algorithm is presented in the next subsection.

Once $\hat{a}$ has been obtained, the PDP support is tested using the block pilot index set $\mathcal{F}$. Again, the subspace spanned by the values of $\mathbf{h}^{(\mathcal{F})}$ can be approximated by the columns of the matrix $\mathbf{U}(W_t(\hat{a}) \times W_f(a'), I_{\mathcal{P}_t} \times I_f)$. The first $D_{a'} = D(W_t(\hat{a}) \times W_f(a'), I_{\mathcal{P}_t} \times I_f)$ columns are then collected in matrix $\mathbf{U}_{a'}$, and the remaining ones in matrix $\mathbf{V}_{a'}$. The algorithms proceed as previously explained, replacing $\mathcal{T}$ by $\mathcal{F}$ and a with a' to finally lead

$$\hat{a}' = \arg\min_{a'} \overline{z_{a'}}(x_{a'}, p_1, p_2). \tag{3.48}$$

As previously pointed out, the PDP support test relies on a non-standard pilot pattern. Since we aim at designing standard-compliant receivers, this test will be skipped in our simulations, and the maximum PDP will always be assumed.

Hypothesis Testing

In this subsection, we introduce the algorithm used in [23] to compute the upper bounds presented above, when running it with comb pilots. A more complete explanation of the underlying theories and hypotheses can also be found in [6].

The reconstruction error z_a is near its mean with probability p_1, which reads

$$P(|z_a - \mathbb{E}\{z_a\}| < G_a) = p_1. \tag{3.49}$$

It is afterwards assumed that the term $(1/|\mathcal{T}|)\|\mathbf{V}_a^H \mathbf{h}^{(\mathcal{T})}\|^2$ is known. From Equation (3.49), z_a is bounded with probability p_1 according to $\underline{z_a'}(p_1) \leq z_a \leq \overline{z_a'}(p_1)$ where

$$\underline{z'_a}(p_1) = \mathbb{E}\{z_a\} - G_a(p_1, \sigma, 2D_a) = \frac{D_a}{|\mathcal{T}|}\sigma^2 + \frac{1}{|\mathcal{T}|}\left\|\mathbf{V}_a^H \mathbf{h}^{(\mathcal{T})}\right\|^2 - G_a(p_1, \sigma, 2D_a) \quad (3.50)$$

$$\overline{z'_a}(p_1) = \mathbb{E}\{z_a\} + G_a(p_1, \sigma, 2D_a) = \frac{D_a}{|\mathcal{T}|}\sigma^2 + \frac{1}{|\mathcal{T}|}\left\|\mathbf{V}_a^H \mathbf{h}^{(\mathcal{T})}\right\|^2 + G_a(p_1, \sigma, 2D_a). \quad (3.51)$$

The term G_a is calculated by solving numerically [23]

$$p_1 = F\left(2D_a + 2G_a\frac{|\mathcal{T}|}{\sigma^2}, 2D_a\right) - F\left(2D_a - 2G_a\frac{|\mathcal{T}|}{\sigma^2}, 2D_a\right), \quad (3.52)$$

where $F(x, i)$ denotes the Chi-square cumulative distribution function with i degrees of freedom, and D_a is the subspace dimension introduced in the last subsection. It appears that G_a is in fact independent of the channel realization. For real implementations, it should therefore be precomputed for each hypothesis matrix (due to the dependence on D_a) using an appropriate quantization of σ^2.

Now, x_a will be used to obtain a probabilistic bound on $(1/|\mathcal{T}|)\|\mathbf{V}_a^H\mathbf{h}^{(\mathcal{T})}\|^2$, the square bias term, which was assumed to be known for the first step. If $2(|\mathcal{T}| - D_a)$ is sufficiently large, the central limit theorem lets us approximate x_a as a Gaussian random variable. The square bias term is bounded with probability p_2, which can be written as

$$\underline{B_a}(x_a, p_2) \leq \frac{1}{|\mathcal{T}|}\|\mathbf{V}_a^H\mathbf{h}^{(\mathcal{T})}\|^2 \leq \overline{B_a}(x_a, p_2). \quad (3.53)$$

Let us now define $m_a = (1 - D_a/|\mathcal{T}|)\sigma^2$. We are interested in the upper bound for the square bias term, $\overline{B_a}(x_a, p_2)$, which is [23]

$$\overline{B_a}(x_a, p_2) = x_a - m_a + \frac{\alpha^2\sigma^2}{|\mathcal{T}|} + 2\alpha\frac{\sigma}{\sqrt{2|\mathcal{T}|}}\sqrt{x_a - \frac{m_a}{2} + \frac{\alpha^2\sigma^2}{2|\mathcal{T}|}}, \quad (3.54)$$

where α is defined below. Finally, by inserting Equation (3.54) into (3.51), we obtain an expression for the upper bound $\overline{z_a}(x_a, p_1, p_2) \geq \overline{z'_a}(p_1)$:

$$\overline{z_a}(x_a, p_1, p_2) = \frac{D_a}{|\mathcal{T}|}\sigma^2 + \overline{B_a}(x_a, p_2) + G_a(p_1, \sigma, 2D_a). \quad (3.55)$$

So far, we have not defined p_1, p_2 or α. The probabilities p_1 and p_2 mentioned above are defined such that $p_1 = Q(\beta)$ and $p_2 = Q(\alpha)$, with

$$Q(\alpha) = \int_{-\alpha}^{\alpha} \frac{1}{\sqrt{2\pi}}\,\mathrm{e}^{-\frac{x^2}{2}}\,\mathrm{d}x.$$

In [6], the authors recommend to choose p_1 and p_2 as close to one as possible, but advise to keep $\alpha/\sqrt{P}$ small and β/P small as well, with P the number of pilot symbols considered in the test. Our first choice was then to take $p_1 = 0.99$ (which gives $\beta = 2.6$) and $\alpha = 1.6$ (leading to $p_2 = 0.89$). However, better results were noticed when following the choices made in [22], namely $p_1 = 0.68$ and $\alpha = 8$, although such choices clearly go against the previously mentioned recommendations.

Once both hypothesis tests have been done, $\hat{a}$ and $\hat{a}'$ are known. The corresponding precalculated matrices $\mathbf{U}(W_t(\hat{a}), I_t)$ and $\mathbf{U}(W_f(\hat{a}'), I_f)$ can then be selected, along with their corresponding $\boldsymbol{\sigma}$ matrices. We can subsequently obtain $\mathbf{U}$ with Equation (3.34) and $\boldsymbol{\sigma}$. Once the dimension D has been computed with Equation (3.37), we are now able to obtain the reduced-rank channel estimates using Equation (3.38).

3.3 Iterative Receivers and MAP Estimation

In the previous section, we dealt with receivers whose criterion was the minimization of the mean square error. In this section, we present algorithms which aim to maximize a posteriori probabilities, which represent the probability of having sent a certain bit given the received signal.

3.3.1 Factor Graphs

A receiver design using the framework of factor graphs is depicted in detail in [19]. Factor graphs are a way to express in a more graphical manner inference problems, and can thus be applied to situations which are not limited to signal processing. The theoretical bases behind them are Bayes' theorem and the law of total probabilities. Once the problem has been turned into a graph, marginal probabilities are computed using the sum-product algorithm on each node. PDFs of random variables are exchanged.

Although it served as an inspiration for the MAP receiver developed in Section 3.3.2, it was not used directly in [9]. Consequently, we tried in this work to investigate whether using the full factor graph formalism could lead to complexity improvements. Two designs were considered: the generic one from [19] and a design applied to OFDM using the 3GPP standard as presented in [4].

None of them were chosen for implementation in this work. It appeared that the design in [19] had a complexity which was too close to the one of the existing MAP design to expect any improvements. Regarding [4], channel estimation would have required at each iteration in the graph to invert matrices whose size was the number of constellation symbols in the considered frame. Although the authors claim that this inversion could in fact be avoided, such a costly operation is very unlikely to be efficiently performed in a hardware implementation.

3.3.2 MAP

A MAP-based iterative receiver was the main topic developed in [9]. The different parts and interactions between them are summarized in Figure 3.4. Since it is not the key point of the present work, we will only mention the relevant parts needed to introduce the simplifications or the justifications of complexity.

The receiver works as follows. First, for each received frame, likelihoods $p(y|q,x)$ are computed for all the symbols, all possible channel coefficients q and all possible constellations symbols in the modulation scheme chosen for the frame. This step is done only once, and its results are subsequently used by the mapper and demapper. Then the iterative part starts by the mapper, which outputs $p(y|q)$. The channel estimator operates separately on each subcarrier, producing extrinsic data on channel coefficients $p_{EXT}(q)$. The demapper follows, which gives conditional probabilities on the coded bits $p(y|c)$. Finally, the BCJR decoder lets us obtain a posteriori probabilities on the coded bits $p(c|y)$, the input bits $p(b|y)$ along with extrinsic information on the coded bits $p_{EXT}(c)$ to be used for the next iteration.

One of the characteristics of this MAP design is that it uses a Markov chain to model the channel. Channel coefficients are distributed as zero-mean complex Gaussian variables. When written in exponential form, $ae^{i\theta}$, this results in a uniform distribution on $[0, 2\pi]$ for the phase θ and a Rayleigh distribution for the amplitude a. Clearly, there exists an infinity of possible channel coefficients with such a model, which makes it problematic for an implementation as a Markov chain.

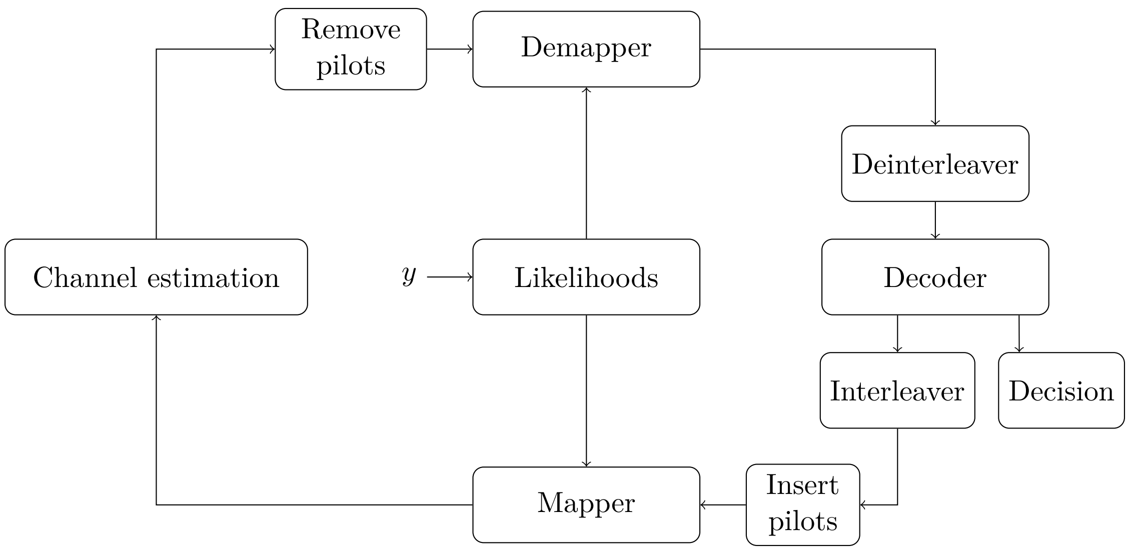


Figure 3.4: Structure of the multicarrier MAP receiver from [9].

Consequently, both amplitude and phase were quantized respectively in N_a and N_{ph} values, chosen as $N_a = 4$ and $N_{ph} = 8$ for the simulations in [9]. The transition matrix was furthermore built to only allow jumps to neighbouring amplitudes, but the phase changes were not constrained. Following a suggestion in [9], whose correctness was shown by numerical simulations, the model was modified to limit jumps to neighbouring phases and amplitudes. Subsequent trials showed no noticeable influence on error rates or on algorithmic complexity. However, the algorithm total running time was divided by a factor of three, mainly due to the reduced number of possible transitions in the BCJR channel estimators. Unless otherwise mentioned, this "simplified" version is the one which will be used in the remainder of this work.

Chapter 4

Theoretical Complexity

Now that the different receiver designs have been presented in Chapter 3, we can now proceed to the comparison of their algorithmic complexity. After a quick reminder on the underlying theory, we present the results of this theoretical assessment for the various building parts of each family of designs.

4.1 Complexity Theory

We will be interested here in comparing different designs both in term of error performance and complexity. Complexity analysis aims at evaluating the number of operations performed by an algorithm as a function of the size of its inputs. It is also possible to consider the memory usage, but this will not be done in this thesis. However, a precise evaluation of the total number of operations is often either not possible or difficult. Instead, a common practice is to give an approximation of this number using Landau's $\mathcal{O}$ notation, whose definition - restricted to our algorithmic case - is given below [8].

Landau's $\mathcal{O}$ Notation For two functions f and g of an integer value n,

$$f(n) = \mathcal{O}(g(n)) \iff \exists N \in \mathbb{N}, C \in \mathbb{N}, \forall\ n > N, 0 \leq f(n) \leq Cg(n). \tag{4.1}$$

Unlike most of the usual algorithms studied in complexity courses (such as sorting algorithms), the different parts of the receivers depend on multiple factors. For this reason, approximations given in the following subsections will try to reflect all the dependencies of each function, instead of keeping only the one with the biggest value or the biggest exponent.

At this point, we can introduce some notation which will be used extensively in this chapter. Let us denote by N_{csif} the number of *constellation* symbols (PSK or QAM) in the frame (not to be confused with N_s, the number of OFDM symbols in the frame). We also define N_{scu}, the number of subcarriers used for data or pilots, N_{scd} the number of data subcarriers and $N_{scp} = 4$, the number of pilot subcarriers, with $N_{scu} = N_{scd} + N_{scp}$. Hence, we can write N_{csif} as $N_s \cdot N_{scu}$.

4.2 Elements Common to All Designs

In this section, we present the receiver subroutines which are shared by many of the algorithms presented so far in this work. The number of calls to these functions varies slightly among the different designs; this will be investigated in Section 4.5.

Fast Fourier Transform

Recall from Section 2.2.4 that the receiver needs to perform a FFT on the time-domain samples before we can proceed with the decoding. This has to be done once per frame, each OFDM symbol at a time. The complexity of the FFT for an input of length n is known to be $\mathcal{O}(n \log_2 n)$ [8]. In this case, we get for the whole frame

$$\boxed{\mathcal{O}\left(N_s \cdot N_{sc} \log_2(N_{sc})\right)}. \tag{4.2}$$

Transitions

The BCJR decoder requires a list of all possible transitions in the convolutional code. Since the convolutional code is a part of the standard and is very unlikely to change, transitions are computed once at the beginning of each simulation. In a real implementation, this would simply be stored in the receiver. Consequently, the complexity of this part of the algorithms is not relevant and will not be computed.

Interleaver and Deinterleaver

Interleaving and deinterleaving bits only consists of swapping bits. Such operations are not considered in complexity evaluation. However, generating the required indices is done using Equations (2.12)-(2.14) for the interleaver, which require some operations. More precisely, the two consecutive permutations will assign a new index to each bit within the interleaved block, which according to Equation (2.11) has a size of N_{cbps} bits. The pattern is the same for all the OFDM symbols in a frame, so it needs to be computed only once. This will therefore have a cost of

$$\boxed{\mathcal{O}(mN_{scd})}. \tag{4.3}$$

where we replaced N_{cbps} by its expression. The value of N_{scd} is indeed specified by the standard, as well as the four modulation schemes available (see Section 2.2.3) which give four possible values for m. Real implementations are likely to have the corresponding interleaving patterns already saved in memory.

The deinterleaver works similarly, and the corresponding equations can be found in [1]. Since the block size is the same for the interleaver and the deinterleaver, the same complexity applies.

Least-Square Initializations

For the *block pilots* scheme, estimates for the two pilots are being made, then averaged and used for the rest of the subcarrier according to Equations (3.2) and (3.3). This will be repeated for each data subcarrier. Hence, the operations are made on a subcarrier basis, which leads for a full frame to a complexity of

$$\boxed{\mathcal{O}(N_{scd})}. \tag{4.4}$$

For the *comb pilots* scheme, we estimate channel coefficients on each constellation symbol on the four comb pilot subcarriers, then perform a linear interpolation within each OFDM symbol to obtain channel estimates for the remaining modulated symbols on the data subcarriers. Hence, one computation is needed for each constellation symbol in the frame. This is made with Equation (3.2) for the comb pilot symbols, or with a linear interpolation of the channel coefficients of the two closest comb pilots for data symbols. This can be written as

$$\boxed{\mathcal{O}(N_{csif})}. \tag{4.5}$$

Mapper and Demapper

The equations used by the mapping routines have been presented in Section 3.2 as Equations (3.7) and (3.9)-(3.11). If at first glance it may be tempting to say that the mapper has a linear complexity with respect to the number of constellation symbols it has to generate, a closer study of the required number of operations for each symbol to map (additions and multiplications) shows that their number also follows linearly the number of possible symbols in the chosen constellation. The associated complexity is therefore

$$\boxed{\mathcal{O}(MN_{csif})}. \tag{4.6}$$

Similarly, the demapper complexity depends linearly on the number of symbols to demap. Again, looking closely to the exact number of operations required gives a linear dependency on the number of possible symbols. Hence, the complexity of the demapper is the same as the mapper complexity.

The demapper is required at each iteration. The mapper is only needed by iterative algorithms so that the feedback from the decoder can be used by the channel estimation part; hence, it is called at each iteration except the last one for which no feedback has to be produced.

BCJR Decoder

In Chapter 2, we introduced the convolutional encoder specified by the standard. It defines a rate 1/2 code with a constraint length $L = 7$. Iterative receivers require that information from the decoder can be subsequently fed to the channel estimator. Consequently, we need a Soft-Input Soft-Output (SISO) decoder, such as the BCJR. In order to justify our complexity analysis, we briefly describe it here. More detailed explanations can be found in [12] or in the original article [5].

First, we consider a sequence of N_b input bits sent. Let us denote by, for $n = 1, 2, \ldots, N_b$,

- b_n, the n-th input bit;
- $c_n^{(1)}$ and $c_n^{(2)}$, the corresponding output bits;
- $y_n^{(1)}$ and $y_n^{(2)}$, the received samples and
- S_n the current state of the encoder. With $N_r = 6$, there are $2^{N_r} = 64$ possible states $\{s_i\}_{0 \leq i \leq 2^{N_r}-1}$.

In this subsection, we also note $\mathbf{y}_n$ the observation vector $[y_n^{(1)}, y_n^{(2)}]^T$ for one input bit and $\mathbf{y}$ the vector of all observations of size $2N_b$. The transition between a state s_r and a state s_s, caused by an input bit $b_{r,s}$ and producing coded bits $c_{r,s}^{(1)}$ and $c_{r,s}^{(2)}$, will be written (s_r, s_s). With these notations, we can now define the following sets

- $\mathcal{S}_+$ is the transition set for which the encoder input bit is 1;
- $\mathcal{S}_-$ is the transition set for which the encoder input bit is 0;

- ${\mathcal{S}'}^{(i)}_{+}$ is the transition set for which the i^{th} coded bit is 1 (with $i = 1, 2$);
- ${\mathcal{S}'}^{(i)}_{-}$ is the transition set for which the i^{th} coded bit is 0.

We want to compute the a posteriori probabilities on bits, which are, for $i = 1, 2$,

$$\begin{aligned}
p(b_n = 1|\mathbf{y}) &= \sum_{(s_r,s_s)\in\mathcal{S}_+} p(S_{n-1} = s_r, S_n = s_s|\mathbf{y}) \\
p(b_n = 0|\mathbf{y}) &= \sum_{(s_r,s_s)\in\mathcal{S}_-} p(S_{n-1} = s_r, S_n = s_s|\mathbf{y}) \\
p(c_n^{(i)} = 1|\mathbf{y}) &= \sum_{(s_r,s_s)\in{\mathcal{S}'}^{(i)}_{+}} p(S_{n-1} = s_r, S_n = s_s|\mathbf{y}) \\
p(c_n^{(i)} = 0|\mathbf{y}) &= \sum_{(s_r,s_s)\in{\mathcal{S}'}^{(i)}_{-}} p(S_{n-1} = s_r, S_n = s_s|\mathbf{y}).
\end{aligned}$$

For the remainder of this subsection, we will write $p(S_n = s_r)$ as $p(s_r)$ for conciseness. The previous equations all require the state transition probabilities given the observation, $p(s_r, s_s|\mathbf{y})$. Using Bayes' rule, we have

$$p(s_r, s_s|\mathbf{y}) = \frac{p(s_r, s_s, \mathbf{y})}{p(\mathbf{y})}. \tag{4.7}$$

$p(\mathbf{y})$ does not depend on the states, and can be seen as a scaling factor. Now, we have to split the observations in three parts: $\mathbf{y} = [\mathbf{y}_n^-, \mathbf{y}_n, \mathbf{y}_n^+]^T$, which respectively represent the samples received before, at, and after time index n. With such notation, $p(s_r, s_s, \mathbf{y})$ can also be split, leading to [5]

$$p(s_r, s_s, \mathbf{y}) = p(s_r, \mathbf{y}_n^-)p(s_s, \mathbf{y}_n|s_r)p(\mathbf{y}_n^+|s_s). \tag{4.8}$$

In [5], the factors of the right part are denoted as

$$\alpha_{n-1}(s_r) = p(s_r, \mathbf{y}_n^-), \tag{4.9}$$
$$\beta_n(s_s) = p(\mathbf{y}_n^+|s_s), \tag{4.10}$$
$$\gamma_n(s_r, s_s) = p(s_s, \mathbf{y}_n|s_r). \tag{4.11}$$

This lets us rewrite Equation (4.8) as

$$p(s_r, s_s, \mathbf{y}) = \alpha_{n-1}(s_r)\gamma_n(s_r, s_s)\beta_n(s_s). \tag{4.12}$$

$\alpha_{n-1}(s_r)$ represents the probability of being in state s_r at time index $n-1$, knowing the previous observations $\mathbf{y}_n^-$. It can be computed recursively [5]:

$$\alpha_{n-1}(s_r) = \sum_{i=0}^{2^{N_r}-1} \alpha_{n-2}(s_i)\gamma_{n-1}(s_i, s_s). \tag{4.13}$$

With the code used, there are only two transitions available from each state (since the input bit can only take two values, 0 or 1). This means that some transitions are not possible, so some terms in the previous sum equal zero. This computation is performed during the forward phase. For initialization, we know that the encoder always starts in state s_0, which reads [5]

$$\alpha_0(s_i) = \begin{cases} 0, & i \neq 0, \\ 1, & i = 0. \end{cases} \tag{4.14}$$

$\beta_n(s_s)$ is the probability of being in state s_s at time index n, based on the future observations $\mathbf{y}_n^+$. Like α, β can be computed recursively during the backward phase, according to [5]

$$\beta_n(s_s) = \sum_{i=0}^{2^{N_r}-1} \beta_{n+1}(s_i)\gamma_{n+1}(s_s, s_i). \tag{4.15}$$

If termination is used, the last state is s_0 and the recursion can be started with [5]

$$\beta_{N_b}(s_i) = \begin{cases} 0, & i \neq 0, \\ 1, & i = 0. \end{cases} \tag{4.16}$$

However, when truncation is used, the final state of the encoder is unknown, so we need to initialize $\beta_{N_b}(s_i)$ as $1/2^{N_r}$ for all values of i.

In the previous equations, a factor γ appeared. $\gamma_n(s_r, s_s)$ is the probability to observe the transition (s_r, s_s) based on the current time observation $\mathbf{y}_n$. It is computed as follows.

$$\gamma_n(s_r, s_s) = p(b_n = b_{r,s})p(y_n^{(1)}|c_{r,s}^{(1)})p(y_n^{(2)}|c_{r,s}^{(2)}). \tag{4.17}$$

Once the terms $p(s_r, s_s, \mathbf{y})$ have been obtained, $p(s_r, s_s|\mathbf{y})$ can be inferred from Equation (4.7) if we define $p(\mathbf{y}) = \sum_{s,r} p(s_r, s_s, \mathbf{y})$. This in turn allows the computation of the a posteriori probabilities on bits.

To avoid the numerical instability created by multiplications of small numbers, the logarithmic version of the BCJR algorithm is used. This also reduces the computational costs of multiplications by replacing them by additions.

The logarithmic versions of the variables α, β and γ are [12]

$$\begin{aligned} a_{n-1}(s_r) &= \log \alpha_{n-1}(s_r), \\ b_n(s_s) &= \log \beta_n(s_s), \\ g_n(s_r, s_s) &= \log \gamma_n(s_r, s_s). \end{aligned}$$

From (4.13), (4.15) and (4.17), we have

$$\begin{aligned} a_{n-1}(s_r) &= \log \sum_{i=0}^{2^{N_r}-1} \exp(a_{n-2}(s_i) + g_{n-1}(s_i, s_s)), \\ b_n(s_s) &= \log \sum_{i=0}^{2^{N_r}-1} \exp(b_{n+1}(s_i) + g_{n+1}(s_s, s_i)), \\ g_n(s_r, s_s) &= \log p(b_n = b_{r,s}) + \log p(y_n^{(1)}|c_{r,s}^{(1)}) + \log p(y_n^{(2)}|c_{r,s}^{(2)}). \end{aligned}$$

At this point, we introduce a new function, known as the Jacobian logarithm, to compute sums of exponentials. It is defined as

$$\overset{*}{\max}(a_1, a_2, \ldots, a_N) = \log(\mathrm{e}^{a_1} + \ldots + \mathrm{e}^{a_1}), \tag{4.18}$$

which for $N = 2$ can also be written as

$$\overset{*}{\max}(a,b) = \max(a,b) + \log(1 + \mathrm{e}^{-|a-b|}). \tag{4.19}$$

This formula can be extended to any number of terms. For example, with $N = 3$:

$$\overset{*}{\max}(a,b,c) = \overset{*}{\max}(\overset{*}{\max}(a,b),c). \tag{4.20}$$

The Jacobian logarithm will be used for computations of the forward and backward recursions. In the case of the log BCJR algorithm, dealing with LLRs instead of probabilities is more convenient, especially in the case of binary variables because one value is stored instead of two.

Now that we have presented the algorithm, we can turn to its complexity analysis. Each frame carries mN_{csif} bits. However, these bits are the result of an encoding by the convolutional encoder of rate 1/2, sometimes followed by puncturing (see Section 2.2.1). For the bits which are input to the BCJR decoder, puncturing is however not relevant since the removed bits would have been replaced by zeros at this stage. Consequently, the frame payload always represents for the decoder a block of $mN_{csif}R$ input bits subsequently encoded with a code rate 1/2.

For each of the input bits, we need to compute the 128 γ metrics corresponding to the available transitions (64 encoder states, each having two transitions depending on the value of the input bit). Then, the forward and the backward recursion are performed. Finally, we obtain the LLRs for the input bit and the two coded bits by taking Jacobian logarithms based on the logarithmic version of $p(s_r, s_s, \mathbf{y})$.

Since the number of transitions (128) is a constant of the code, it will not appear in the final complexity which becomes

$$\boxed{\mathcal{O}(mN_{csif}R)}. \tag{4.21}$$

The decoder is required at each iteration of the presented receivers, or once for the non-iterative design.

Decision

Recall from the previous subsection that the BCJR algorithm outputs soft decisions. However, the upper network layers need bits, which require hard decisions. Consequently, we use a threshold to take hard decisions based on the LLRs. Considering the definition given in Equation (3.5), values above 0 will denote a "0", while negative values denote a "1". A decision has to be taken for all the $mN_{csif}R$ input bits of the frame. The cost of this comparison operation is

$$\boxed{\mathcal{O}(mN_{csif}R)}. \tag{4.22}$$

Decisions are taken once all the other subroutines, iterated if applicable, are finished. This step consequently occurs once per frame for all the receivers presented.

4.3 Elements Specific to MMSE Designs

In this section, we turn to algorithms used in the MMSE receivers. Especially, the building blocks presented in [23] will be considered.

Wiener filter

This part is specific to the MMSE receiver developed in [9]. First, we need to compute $\mathbf{R}$, the $(2K+1) \times (2K+1)$ autocorrelation matrix with $K = 10$ [20] presented in Equation (3.16). According to the definitions given in Section 3.2.1, it is a symmetric Toeplitz matrix; hence, we only need to compute $2K+1$ coefficients, which are values of the zeroth order Bessel function of the first kind. We also need the cross-correlation vector $\mathbf{P}$, but since its coefficients are the same as the values previously computed for the matrix (although in a different order), no additional operations are required. Solving the matrix equation (3.18) for the Wiener tap vector $\mathbf{w}$ has a complexity of

$$\boxed{\mathcal{O}\left((2K+1)^3\right)}. \tag{4.23}$$

Since K has a fixed value, the equation above only contains constants. Hence, complexity theory tells us that it has to be reduced to $\mathcal{O}(1)$.

The tap vector will then be used to perform filtering of the sample sequence and obtain channel estimates, as in Equation (3.19). This filtering operation will be done at each iteration of the algorithm. The complexity of such a computation is

$$\boxed{\mathcal{O}(N_{csif})}. \tag{4.24}$$

At this point, we need to mention that the correlation coefficients depend on the relative speed between the transmitter and the receiver through the Doppler shift. This value is fixed for each frame, so the tap vector of the filter does not have to be computed iteratively. However, we need to choose between computing the filter once for each frame (as was done in our simulations), or using a quantization of the speed to rely on precomputed values stored in the memory of the receiver.

Subspace Hypothesis Matrices

The subspace selection algorithm in [23] chooses the best fit among a number of hypotheses A on the DSD and A' on the PDP. These numbers are fixed at the beginning of the computations and would also be constants in a real implementation.

Each hypothesis is represented by a matrix of eigenvectors and eigenvalues, which are computed either using the method in [17] for the DSD hypothesis or MATLAB eigensolver for the PDP. Such computations are costly, but they need to be performed only once. This allows the subspace hypothesis matrices to be generated on a computer and then stored, for instance in the memory of a receiver or for further use by subsequent numerical simulations.

Subspace Selector

In [23], the complexity of the subspace selection algorithm for a single frame is given as

$$C_{ss} = \sum_{a=1}^{A} |\mathcal{T}| D_a^2 + \sum_{a'=1}^{A'} |\mathcal{F}| D_{a'}^2,$$

where the first term accounts for the DSD and the second for the PDP. D_a represents the dimension of the dominant subspace for the a-th hypothesis matrix, computed with Equation (3.37).

Since our frames do not have any postamble, the PDP support test could only be run on the two block pilots at the beginning of the frame. Simulations showed that this approach underestimates the dimension of the subspace, leading to a mismatched filter and very low performance. Consequently, we do not perform the test for the PDP and always take the maximum support.

Following [23], we take $A = 10$. From the definition of $\mathcal{T}$, $|\mathcal{T}|$ is found to be $N_{scp} \cdot (N_s + 2)$. The expression for the complexity of the subspace selection algorithm now becomes

$$C_{ss} = \sum_{a=1}^{A} |\mathcal{T}| D_a^2 = N_{scp}(N_s + 2) \sum_{a=1}^{10} D_a^2 \leq N_{scp} \cdot 10 \cdot (N_s + 2) \cdot D_{10}^2$$

The last inequality comes from the fact that the submatrix dimension will increase with an increasing DSD support. We found the values of D_{10} to be 10, 29 and 35 respectively for frame lengths of 36, 72 and 136 symbols. This leads to the final expression

$$\boxed{\mathcal{O}(N_{scp} \cdot A \cdot {N_s}^3)}. \tag{4.25}$$

Here, one may wonder why the computations of the $\mathbf{U}$ matrices and their associated dimensions are not taken in account in the previous analysis. Recall from the previous subsection that the submatrices associated with each DSD or PDP hypothesis are generated only once and stored for future use. This allows precomputation of the $\mathbf{U}$ matrices (which directly depend on the hypothesis submatrices), and in turn sorting of their eigenvalues and computation of their D_a. Hence, these costly computations do not need to be performed at runtime.

As far as the term G_a is concerned, it can also be precomputed by using an appropriate quantization of the noise variance σ^2 [23] and solving the equations for the different values of D_a, since they also appear in the computation.

Following [23], the subspace selection is only performed once, as the authors claim that making this selection at each iteration of the receiver does not improve the performances.

Channel Estimation

An assessment of the number of operations required for producing the channel estimates is provided in [23]. We reproduce it here, adapted to the notations of the thesis.

$$C_{ch.est.} \approx 8 N_{sc} N_s D^2 + \frac{8}{3} D^3$$

At this point, the complexity of this algorithm may look linear with respect to the number of OFDM symbols. However, [21] and [23] give example values for D: 36 for $N_s = 38$ and 75 in the worst case for $N_s = 73$, which suggest a linear dependency between D and N_s. We also performed simulations for different frame lengths, whose results confirmed this linear dependency. The final expression for the complexity of the channel estimator becomes

$$\boxed{\mathcal{O}(N_{sc} \cdot {N_s}^3)}. \tag{4.26}$$

As in the case of the non-adaptive MMSE receiver, channel estimation has to be performed for each iteration of the algorithm [23].

LMMSE Detector

The LMMSE detector performs equalization on a symbol basis, according to [21]

$$\hat{x} = \frac{y\hat{H}^*}{\sigma^2 + |\hat{H}|^2}. \tag{4.27}$$

For a complete frame, this formula has to be repeated for its N_{csif} constellation symbols, which leads to a complexity of

$$\boxed{\mathcal{O}(N_{csif})}. \tag{4.28}$$

4.4 Elements Specific to the MAP Design

In this section, we discuss the parts specific to the MAP design proposed in [9]. We will only introduce here the main lines of the algorithms and the elements needed to justify our complexity estimations. A more rigorous approach of the theory can be found in Chapter 3 of [9].

Mapper and Demapper

Unlike the ones in Section 4.2 which operate on complex symbols, the mapper and demapper used in the MAP design work with conditional probabilities, as follows.

The mapping routine operates with probabilities on coded bits $P(c = 1)$ provided by the interleaver. It also takes as input $P(y|q, x)$, which are the probabilities of observing a received signal y given a channel coefficient q and a symbol sent x, provided by the channel likelihoods function. It outputs $P(y|q)$, which is a matrix of size $N_a \cdot N_{ph} \cdot N_{csif}$. Computation of each of these coefficients requires $\mathcal{O}(m \cdot M)$ operations, since we need to compute probabilities for the m coded bits forming the symbols. These probabilities are based on the values the coded bits can take for the M symbols in the constellation and on the input provided by the interleaver. In the end, the complexity is

$$\boxed{\mathcal{O}(N_{csif} \cdot N_a \cdot N_{ph} \cdot M \cdot m)}. \tag{4.29}$$

The demapper outputs LLRs on the $m \cdot N_{csif}$ coded bits, based on $P(y|h, x)$ and $P(h)$. According to Bayes' rule, summing the products of $P(y|h, x)$ and $P(h)$ for the $N_a \cdot N_{ph}$ possible values of h gives $P(y|x)$. Obtaining $P(y|c)$ for a given value of c (0 or 1) is made by summing the $P(y|x)$ of the corresponding symbol x on all its possible $M/2$ values (fixed by the modulation) for which the bit c will have the required value 0 or 1. Such summations will be repeated for all coded bits. This leads to a complexity of

$$\boxed{\mathcal{O}(N_{csif} \cdot N_a \cdot N_{ph} \cdot M \cdot m)}. \tag{4.30}$$

Note that as in the non-probabilistic case, both the mapper and the demapper have the same complexity. These two subroutines are called at each iteration. If for the demapper this is similar to the other receivers, the difference for the mapper comes from the fact that the MAP receiver starts each iteration by a call to the mapper (see the description in Section 3.3.2), while the other iterative receivers only call the mapper when subsequent iterations are needed. Consequently, the MAP receiver will require the mapper for the last iteration, unlike the MMSE designs.

BCJR Channel Estimation

The algorithm is essentially the same as the decoder version, which was already presented. The only difference is the type of data computed: while the decoder deals with bits and states of the convolutional encoder, the channel estimator deals with channel estimates, quantized coefficients and non uniform transition probabilities. However, the number of states and available transitions are not fixed by the standard, and need to be taken in account in the analysis.

Recall from Section 3.3.2 that there are $N_a N_{ph}$ different quantized channel coefficients, representing the different states available in the Markov model. Thanks to the limitations on the possible jumps, it is only possible to reach from a given state states with neighbouring phases and/or amplitudes. It is also possible to keep the same phase and/or amplitude. This gives a choice of 3 different amplitudes (or 2 if we already are at the lowest or highest amplitude available), and 3 phases, which gives 9 (or 6) possible transitions from a given state. Consequently, there are $\mathcal{O}(N_a N_{ph})$ possible transitions in total. This subroutine operates separately on each subcarrier; for a N_s-symbol long subcarrier, $\mathcal{O}(N_s N_a N_{ph})$ are therefore required. Since the receiver deals with frames, the process has to be repeated for each subcarrier. The complexity of the channel estimation for a complete frame is

$$\boxed{\mathcal{O}(N_{csif} \cdot (N_a \cdot N_{ph}))}. \tag{4.31}$$

Channel estimation is required at each iteration.

Channel Transitions

Just like we had to specify the available transitions for the convolutional encoder states, we need to define the possibilities of jumps between different channel quantized coefficients. We already saw in the description made in the previous chapter that the number of quantized amplitudes N_a and of quantized phases N_{ph} were fixed at the beginning. Furthermore, transitions are only allowed between coefficients of neighbouring amplitudes and/or phases. Consequently, it becomes possible to compute these transitions only once, and store them for future use.

Transition Matrix

The Markov chain model requires a transition matrix to store the probabilities for each jump. Since the amplitude and the phase are independent, we can compute the corresponding transition matrices separately and obtain the complete matrix by multiplying. Taking in account the fact that jumps are only possible between adjacent amplitudes and adjacent phases (due to the simplifications introduced in Section 3.3.2), $\mathcal{O}(N_a)$ computations are needed for amplitudes and $\mathcal{O}(N_{ph})$ for phases. The complete transition matrix stores the probabilities for each of the theoretically possible jumps between the $N_a N_{ph}$ states of the Markov chain, which are here channel coefficients. This makes a total of $(N_a N_{ph})^2$ probabilities in the matrix, although most of them are zeros due to the previously introduced limitations. $\mathcal{O}\left((N_a N_{ph})^2\right)$ products are required to generate the final matrix. The final complexity is

$$\boxed{\mathcal{O}\left((N_a \cdot N_{ph})^2\right)}. \tag{4.32}$$

Note that this algorithm takes in account the Doppler shift; hence, it would have to be either called for each frame, or run once for different quantized values of the speed and stored for subsequent use. Our simulations choose the first approach. The transition matrix has to be produced (or fetched from memory) once per frame.

Channel Likelihoods

Recall from Section 3.3.2 that the first block of the MAP estimator is dedicated to the computation of the likelihood of each of the $N_a N_{ph}$ possible quantized channel coefficients, and this has to be done for the N_{csif} constellation symbols in the frame. However, since the constellation symbols which have been sent are still unknown to the receiver when this algorithm is called, it needs to perform these likelihood computations for each of the M possible symbols which could have been sent.

This leads to a complexity of

$$\boxed{\mathcal{O}(N_{csif} \cdot (N_a \cdot N_{ph}) \cdot M)}. \tag{4.33}$$

As already mentioned in the description of the receiver in Section 3.3.2, this subroutine is called once per frame before starting the iterations.

4.5 Summary

Now that the complexity of each part of the design has been assessed, it would be interesting to see how many times each building block is called during the decoding of a single frame. In this section, iterative algorithms will perform N_{it} iterations on each frame. For the classical and MMSE algorithms, we do not make separate tables for the two initialization methods.

We start with the non-iterative case, described in Table 4.1.

Algorithm	**Complexity in $\mathcal{O}$**	**Number of calls**
FFT	$(N_s N_{sc}) \log_2(N_{sc})$	1
Channel estimation with block pilots *or* with comb pilots	 N_{scd} N_{csif}	1
Demapping and equalization[1]	$M N_{csif}$	1
Decoding	$m N_{csif} R$	1
Decision	$m N_{csif} R$	1

Table 4.1: Complexity breakdown for the classical receiver.

It can be seen from the data in Table 4.1 that the overall complexity of the algorithm is linear with respect to the frame length.

We can now turn to iterative designs. Complexities associated with the MMSE receiver in [9] are presented in Table 4.2.

Again, linear dependencies on both frame lengths and number of iterations appear. The MMSE design in [23] is presented in Table 4.3.

Here, the two algorithms specific to this MMSE design (namely, the subspace selection and the channel estimator) prove to have a cubic dependency on the number of OFDM symbols, which is very poor [10]. Although using a reduced dimension D for the Wiener filter offers significant computational gains compared to the full rank filter, this

[1]Both are made simultaneously in our implementations.

Algorithm	**Complexity in $\mathcal{O}$**	**Number of calls**
FFT	$(N_s N_{sc}) \log_2(N_{sc})$	1
Initial channel estimation with block pilots *or* with comb pilots	 N_{scd} N_{csif}	1
Wiener tap vector computation	1	1
Wiener filtering	N_{csif}	N_{it}
Demapping and equalization	MN_{csif}	N_{it}
Decoding	$mN_{csif}R$	N_{it}
Mapping	MN_{csif}	$N_{it} - 1$
Decision	$mN_{csif}R$	1

Table 4.2: Complexity breakdown for the MMSE receiver [9].

Algorithm	**Complexity in $\mathcal{O}$**	**Number of calls**
FFT	$(N_s N_{sc}) \log_2(N_{sc})$	1
Subspace selection	$N_{scp} A {N_s}^3$	1
Channel estimation	$N_{sc} {N_s}^3$	N_{it}
Equalization	N_{csif}	N_{it}
Demapping	MN_{csif}	N_{it}
Decoding (max-log BCJR)	$mN_{csif}R$	N_{it}
Mapping	MN_{csif}	$N_{it} - 1$
Decision	$mN_{csif}R$	1

Table 4.3: Complexity breakdown for the adaptive MMSE receiver [23].

algorithm will have a consequent weight in the overall complexity of this design.

Finally, we can turn to the MAP design, whose complexity was already partly assessed in [9]. Our results are presented in Table 4.4.

Algorithm	**Complexity in $\mathcal{O}$**	**Number of calls**
FFT	$(N_s N_{sc}) \log_2(N_{sc})$	1
Transition matrix	$(N_a N_{ph})^2$	1
Channel likelihoods	$N_{csif} N_a N_{ph} M$	1
Mapping	$N_{csif} N_a N_{ph} M m$	N_{it}
Channel estimation	$N_{csif}(N_a N_{ph})$	N_{it}
Demapping and equalization	$N_{csif} N_a N_{ph} M m$	N_{it}
Decoding	$mN_{csif}R$	N_{it}
Decision	$mN_{csif}R$	1

Table 4.4: Complexity breakdown for the MAP receiver.

At this point, it is interesting to compare the findings expressed in Table 4.4 to the data provided in [9]. First, an expression similar to Equation (4.1) in [9] can be derived from our results. Then, there is a linear dependency on the frame length, as shown in Figures 4.15 and 4.16 of [9]. The quadratic dependence on the number of quantized levels $N_a N_{ph}$ (referred to as N_q in [9]) no longer appears in the channel estimation part due to the simplifications made. However, for the non simplified algorithm used in [9], a quadratic dependence on the number of phases N_{ph} would have been found.

However, while the complexity assessment in this work indicates an influence of the size of the constellation M as $Mm = M\log_2(M)$, previous results based on execution times only showed a linear relation [9]. Although the repartition of the total execution time among subroutines will be treated in the next chapter, we believe it to be the explanation of this difference. Indeed, the subroutines showing the quasilinear complexity have very low computation times, while the channel likelihoods exhibiting the linear complexity in M accounts for a large part (30 to 50%) of the total running time.

Finally, if we compare the complexity of subroutines serving the same purpose between the MAP and the MMSE design, MAP algorithms show an increase by a factor of $N_a N_{ph}$, which can account for the gap in execution times showed in Figure 4.15 in [9].

Chapter 5

Performance Analysis

In this chapter, the data from the simulations is presented. Section 5.1 gives the achieved error rates for each algorithm, and draws some comparisons. In Section 5.2, we provide the execution times which validate and extend the estimates in Section 4. Section 5.3 tries to sort the receivers proposed so far by taking simultaneously in account the error rates and the complexity. Finally, Section 5.4 addresses some of the concerns which may arise when implementing the receivers on testbeds.

5.1 BER and FER Performance

In this section, we introduce the results of the implemented algorithms in terms of error rates. The first rate is the Bit Error Rate (BER). Usually expressed as a function of the SNR in the form E_b/N_0, it represents an interesting measure from the theoretical point of view. However, the systems considered in this work are frame-based. For this reason, we will also consider the Frame Error Rate (FER), where a frame is said to be in error if at least one of its bits is wrongly decoded. It can be noticed here that the FER will not only depend on the overall BER, but also on the spreading of erroneous bits. Consider for example a batch of 100 frames, where 100 bits in total are in error. It is straightforward that although the BER will not change, the FER will experience large variations if one bit is wrong in each frame or if the 100 wrong bits are within a single frame.

In the remainder of this section, we use the same SNR range as in [9], namely from -4 to 28 dB. Similarly, BPSK modulation is used, and we consider a relative speed of $v = 100$ km/h. We simulated frame lengths of 36, 72 and 136 OFDM symbols (excluding pilots) as in [9] and previously [21], corresponding to lengths of 100, 200 and 400 bytes with BPSK. Unless otherwise mentioned, simulations run over 500 frames for each SNR value.

5.1.1 Classical Receivers

First, we present in Figure 5.1 and 5.2 the results for the classical receiver, with the two initialization methods mentioned in Chapter 3.

It can be seen that with the chosen channel model, comb pilots offer improved performance at high SNRs and allow to go beyond the error floor occuring with block-based receivers at 10 dB. Interestingly, the ability to have pilots for the whole length of the frame make the performance of this design almost independent of the frame length. On the contrary, the block-based systems is affected by a decreased performance when

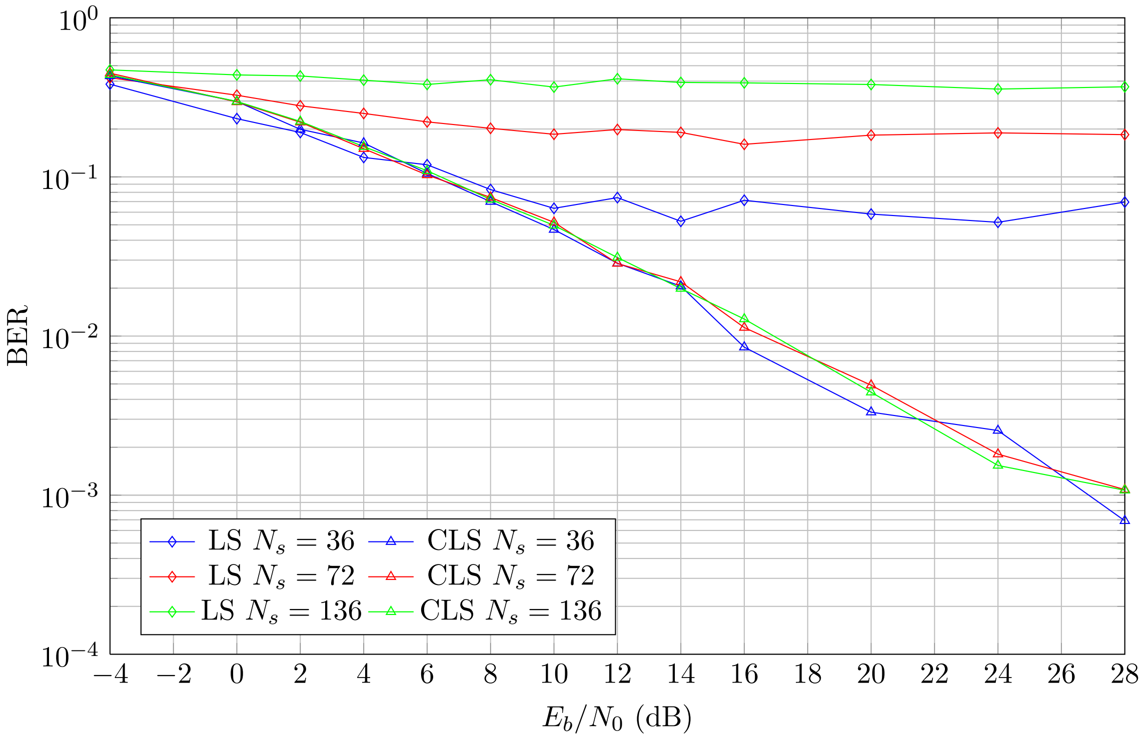


Figure 5.1: Comparison of the BER performance of the classical receiver for LS and CLS initializations and different frame lengths.

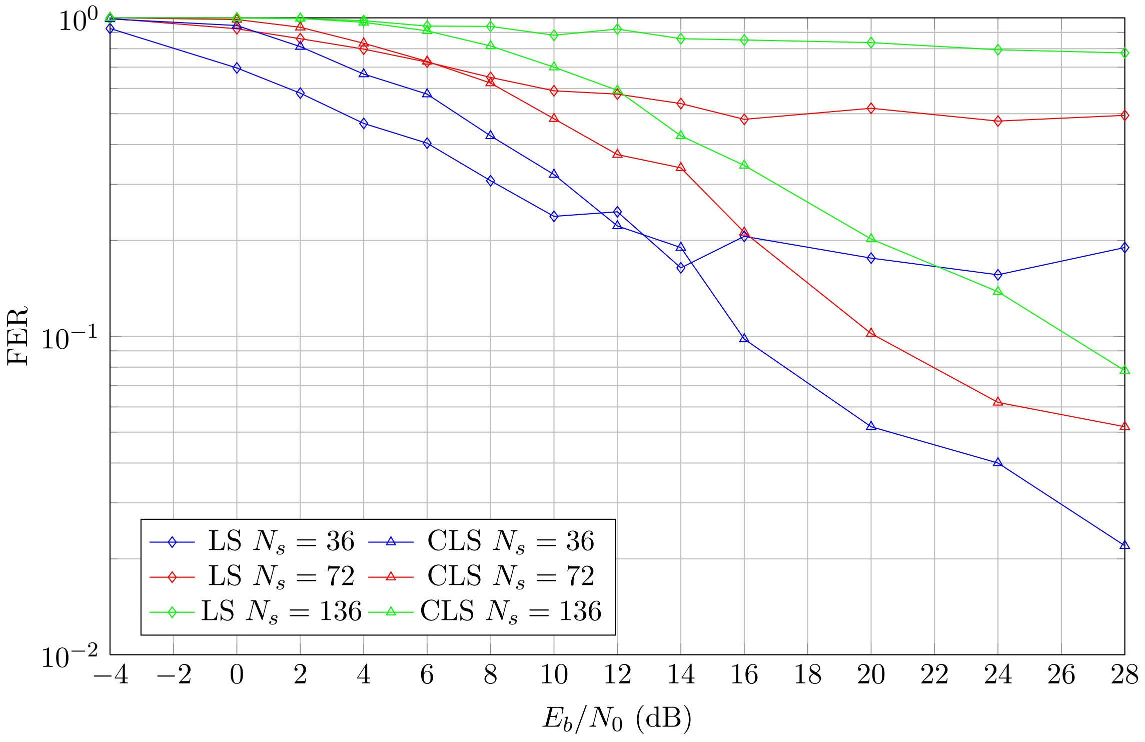


Figure 5.2: Comparison of the FER performance of the classical receiver for LS and CLS initializations and different frame lengths.

the frame length is increased: the BER is almost doubled each time the frame length doubles, reaching a floor at 40% for the longest 136-symbol frames.

When dealing with the FER, the comb-based receiver still performs better than its block-based counterpart. However, performance degrades with longer frames. The FER threshold of 10^{-1} mentioned in [2] and [23] would never be reached by the block-based receivers, while it would require a minimum SNR of 16, 20 and 26 dB for respective lengths of 36, 72 and 136 symbols with the comb-based designs.

5.1.2 MMSE Receivers

We now turn to the MMSE designs. Again, the two initialization methods are used for the design from [9]. The results are given in Figure 5.3 and 5.4.

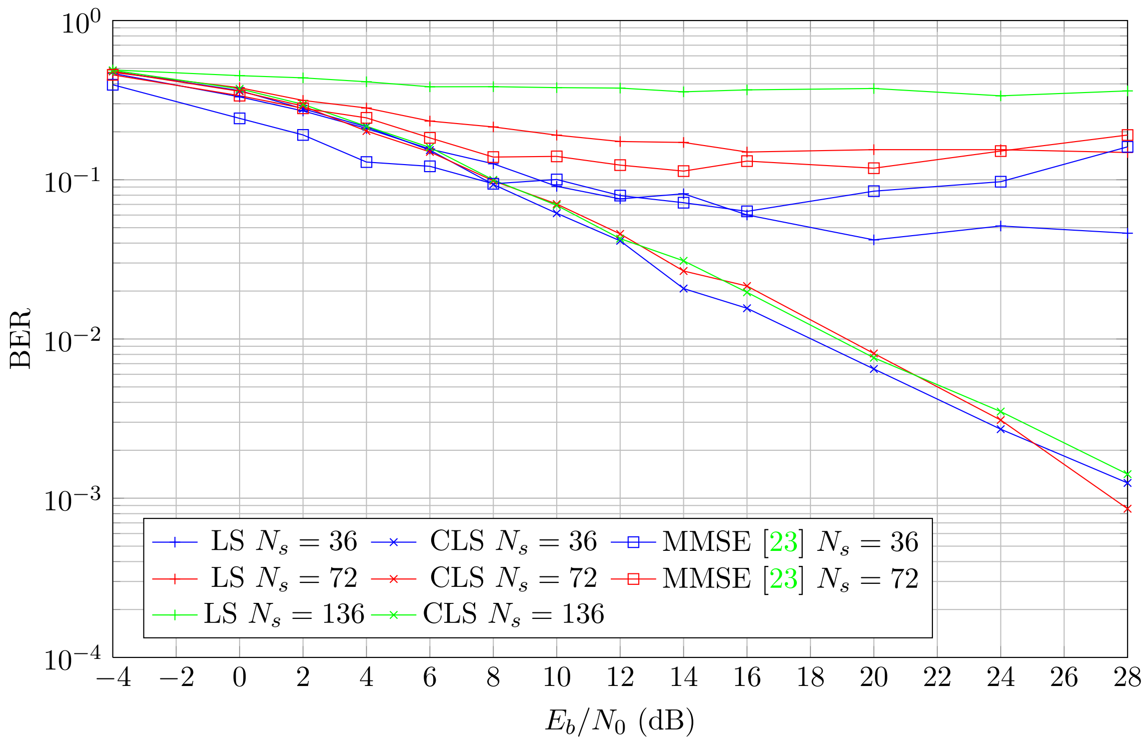


Figure 5.3: Comparison of the BER performance of the MMSE receivers for LS and CLS initializations and different frame lengths.

From the simulations, we see that the difference of behaviour between comb-based and block-based initializations observed for the classical receiver also applies here: comb pilots make the BER performance almost independent of the frame length, and overcome an error floor. FER curves are also very similar. Especially, the FER threshold of 10^{-1} is never reached by the block-based receivers, while it would require a minimum SNR of 17, 22 and 27 dB for respective lengths of 36, 72 and 136 symbols. This is 1 dB more than for classical receivers. Compared to the results in [9] where the MMSE design performed better than the classical receiver, this finding was not expected.

One may notice that the results provided here for the MMSE design in [23] do not exactly match the curves in [23]. Several factors can account for these differences. First, the channel model used here has a longer DS: 2.5 μs against 1.6, which creates ISI. Then, Zemen et al. consider a frame which has a higher number of pilots due to its postamble, which is not implemented in our simulation because it is not part of the

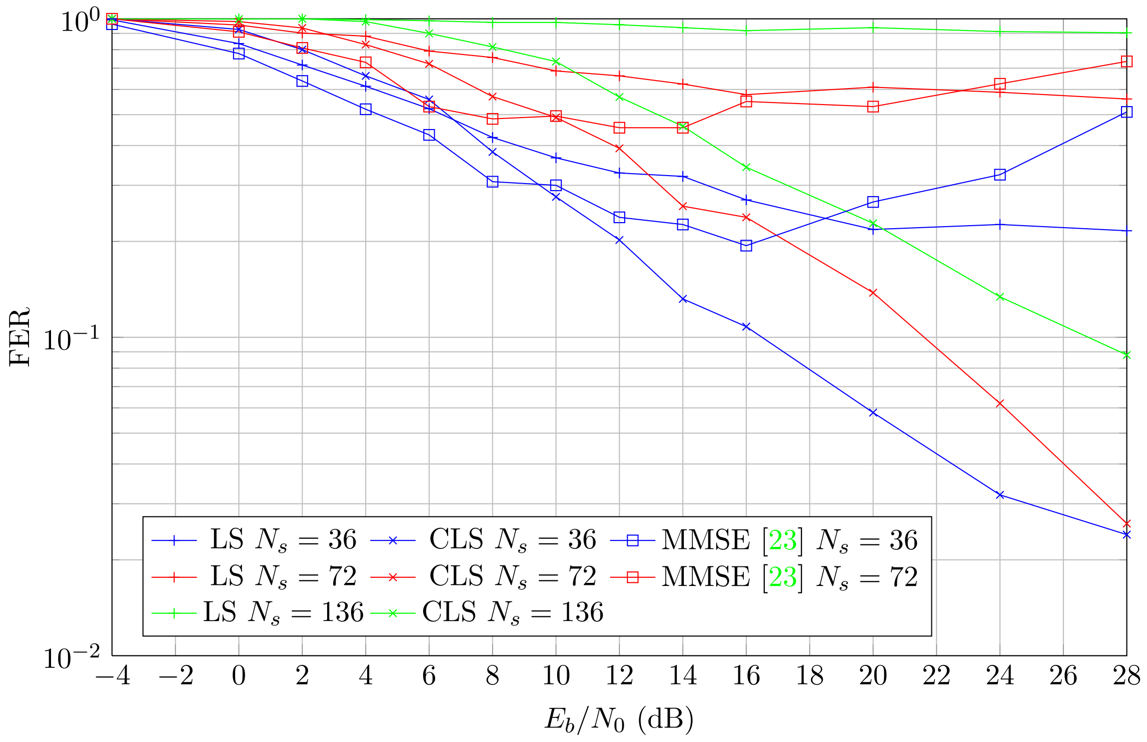


Figure 5.4: Comparison of the FER performance of the MMSE receivers for LS and CLS initializations and different frame lengths.

standard. These extra pilots were however shown, for example in [21], to dramatically improve the performance of the receivers. Such improvements were also noted in [9] with the MAP design for the subcarrier case. Finally, the formula for the computation of noise variance is different.

It can also be seen that the algorithm from Zemen et al. [23] suffers from a performance loss at high SNRs (above 16 dB). This may be caused by incorrectly demapped symbols which are nevertheless considered correct by the algorithm. The fact that no error rate curve in the corresponding literature is plotted for SNR values above 14 dB tends to suggest that this particularity was also known to the authors.

5.1.3 Comparison

The complete results of our simulations, summarizing the two previous subsections and adding the MAP design are presented in Figure 5.5 and 5.6. For each SNR value, 500 frames were simulated, except for [23] and MAP algorithms with a frame length of 72 symbols for which only 200 frames were simulated. This reduction was motivated by time considerations (see next section for more details on the execution times). Due to the high number of curves, the legend is split between the BER and FER plots. Furthermore, the curves representing the MMSE algorithm with comb initializations were not added for clarity, as they match very closely the corresponding curves obtained with the classical receiver.

In [9], the MAP design was shown to outperform the other designs in terms of error rates for the subcarrier case. From the results given here, we can see that this design is much less adapted to the multicarrier case, having only a small advantage for SNR values up to 10 dB for BER values before being outperformed by the MMSE design using comb

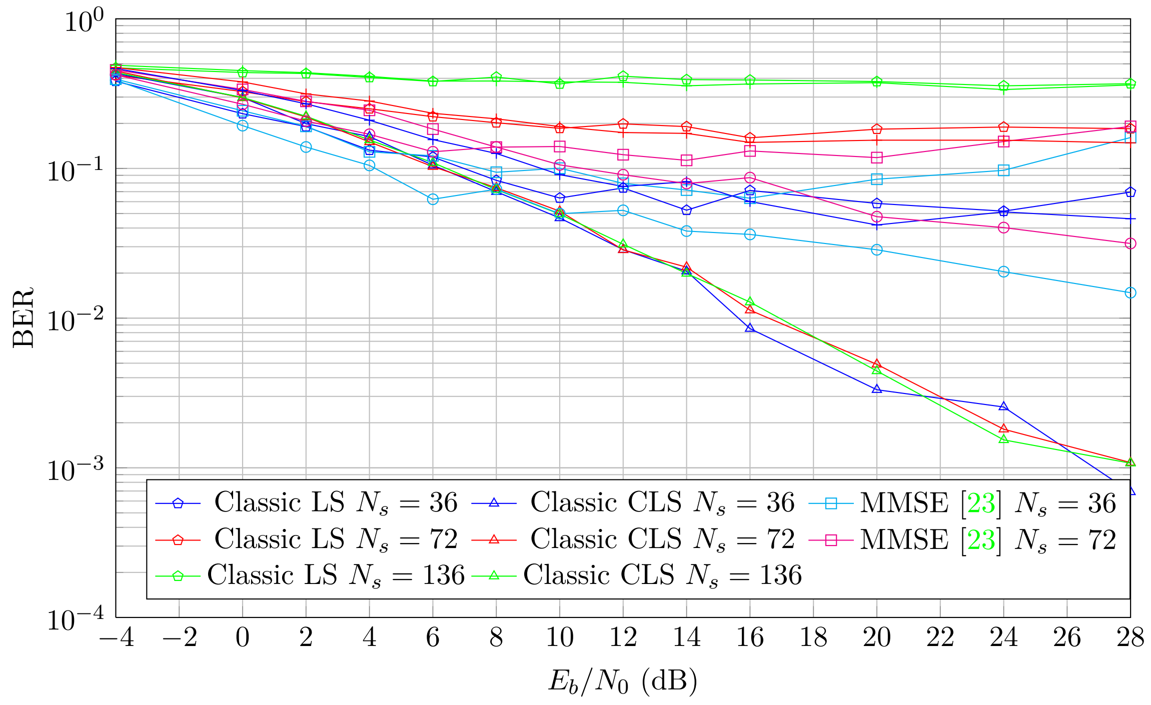


Figure 5.5: Comparison of the BER performance of all the studied receivers for three different frame lengths.

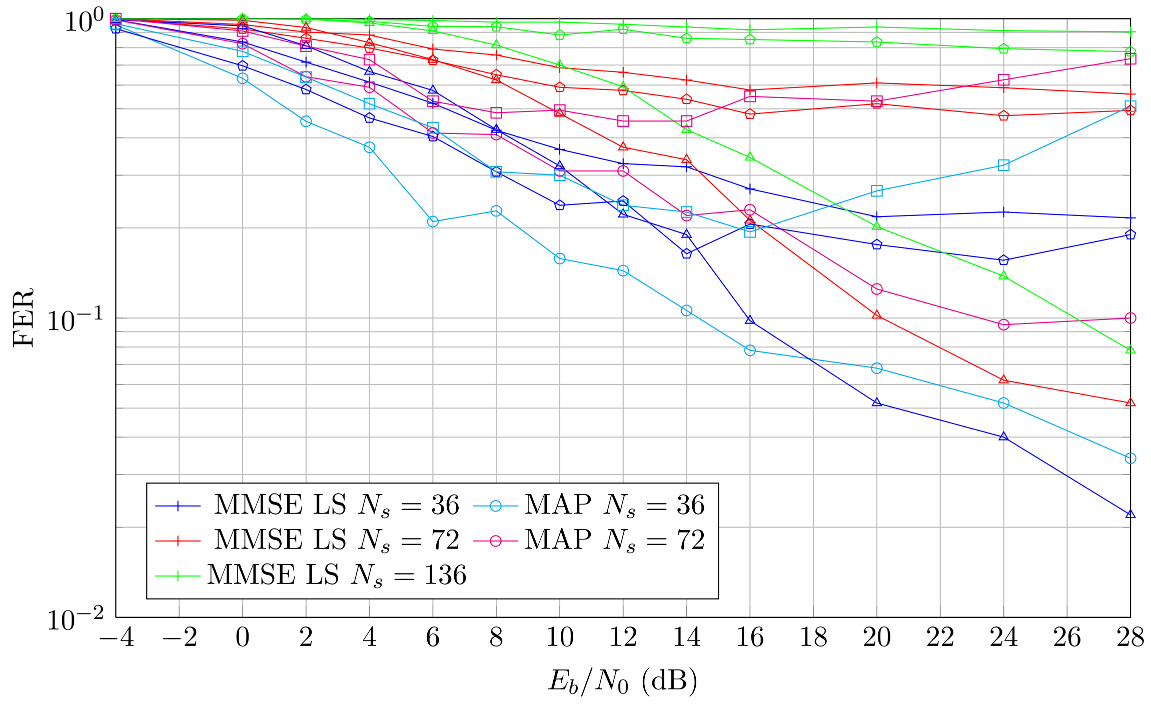


Figure 5.6: Comparison of the FER performance of all the studied receivers for three different frame lengths.

pilots. This difference of performance is likely to come from the inner design of the MAP algorithm, which performs channel estimation separately on each subcarrier without taking in account the whole frame. However, for the FER, the advantage lasts up to 18 dB; afterwards, MAP and MMSE designs with comb pilots offer similar performances. This would tend to indicate that errors are more "concentrated" with the MAP receiver than with its MMSE counterpart.

For algorithms run with either block or comb pilots used for initialization, we see that block-based estimators perform slightly better at low SNR values, while comb pilots provide the best error rates for high SNRs. This was already noticed in [7]. Another advantage of comb pilots is that they allow to reach lower error rates at high SNRs, where the same algorithms would experience an error floor when using block pilots.

5.2 Complexity and Execution Times

Theoretical complexity assessments have been presented in Chapter 4. If they are a valuable tool to evaluate the total number of operations as well as the dependencies on different parameters, they do not necessarily reflect the real performance of the algorithms. Hence, we give in this section another measure of complexity through execution times. It should be noted that the execution times provided in this thesis have been obtained through MATLAB simulations, and are dependent on several factors such as the load of the computer used.

Table 5.1 present the execution times per frame for the proposed algorithms of this work. The evaluations were performed at the same SNR value using BPSK modulation and with similar computer loads. They were averaged over several frames, and did not make use of MATLAB's profiler. This tool indeed gives detailed timings, but at the cost of a large overhead which does not reflect the execution times achieved during normal operation. Iterative algorithms made $N_{it} = 5$ iterations. This value was indeed shown in [9] to be sufficient for the algorithms to converge when dealing with short frames, and this observation was subsequently confirmed for all receivers by our simulations. We also chose to apply this value to long frames so that we could focus on the influence of the other parameters.

Algorithm	N_s	**Time (s)**
Classic	36	2
	72	4
	136	8
MMSE	36	11
	72	20
	136	39
MMSE [23]	36	25
	72	171
	136	1104
MAP	36	60
	72	134
	136	249

Table 5.1: Execution time per frame for the proposed algorithms

These execution times confirm the dependency on the frame lengths, which was found analytically in Chapter 4 (and previously in [9]) to be linear for all algorithms, except [23] for which it is cubic [10]. For the classical and MMSE design, we did not separate LS and comb LS (CLS) initializations, as they proved to lead to similar execution times.

5.2.1 Repartition of the Execution Times

For further optimizations, it is interesting to know how the total execution time is split between the different parts. Hence, we present in this section the breakdown for all the algorithms implemented, operating on full frames. We use BPSK modulation and $N_s = 36$ symbols. Iterative algorithms perform $N_{it} = 5$ iterations.

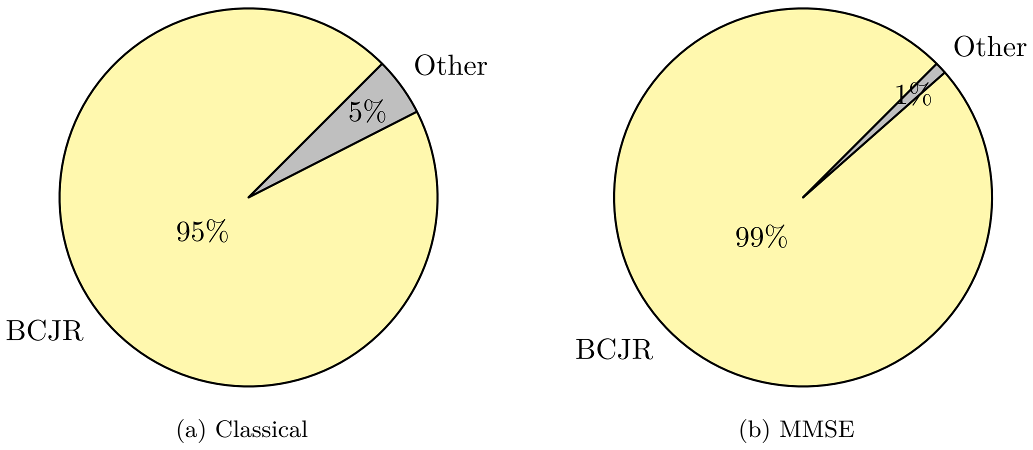


Figure 5.7: Repartition of the execution time for the classical and MMSE receivers.

It is clear that for these two algorithms, the BCJR decoder represents by far the largest part of the execution time. Hence, being able to optimize this part would lead to considerable computational savings. One of the most popular optimizations for the BCJR is discussed in Section 5.4.

We now turn to the receiver design of [23].

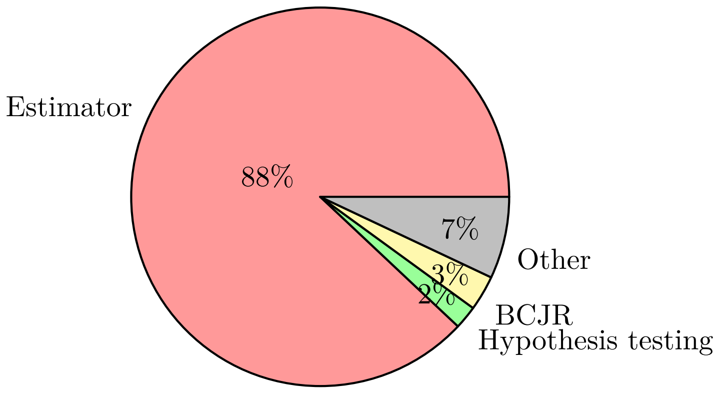
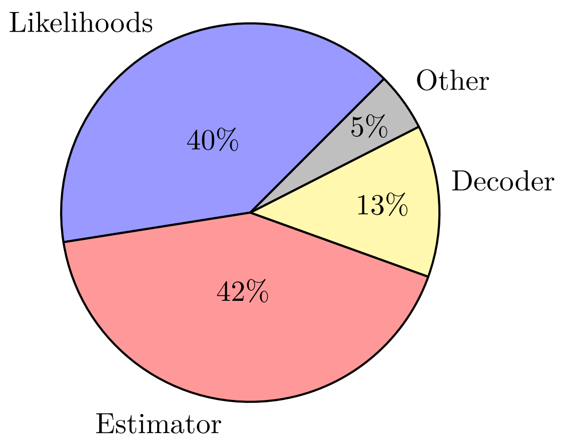


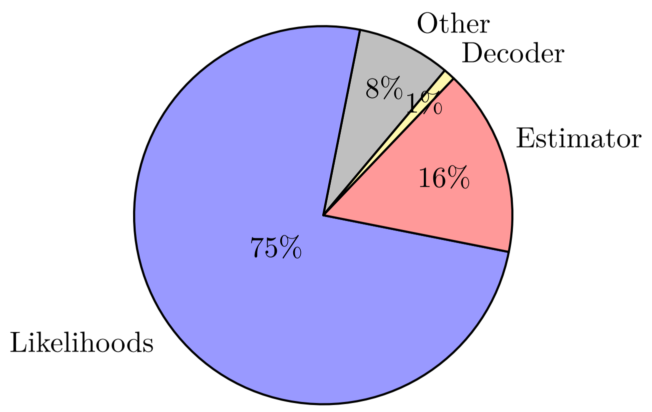
Figure 5.8: Repartition of the execution time for the MMSE receiver [23].

Here, the total time is dominated by the channel estimation process. This can explain the fact that the total execution time reflects the cubic dependency on the frame length of the estimator. Unlike the two previous receivers, the BCJR only plays a minor role, which is not only due to the estimator but also to the optimization mentioned earlier, which is implemented in this decoder.

To conclude this section, we provide the MAP design performance, both with and without the BCJR optimization. Recall that the MAP receiver used in this work features some simplifications which were not implemented in [9], and that the BCJR algorithm contributes both to channel estimation and to decoding. Like in [9], the channel coefficients can take up to 32 different quantized values.



(a) max* BCJR



(b) max-log BCJR

Figure 5.9: Repartition of the execution time for the MAP decoder.

From Figure 5.9, we can see that the likelihood computation, although occuring only once per frame, accounts for a large part of the execution time, as mentioned in Section 4.5. This is especially true when using the max-log version of the BCJR, which decreases the time spent by the estimator and decoder. It should be noted that the total time for the max BCJR case is halved compared to the max* BCJR case.

In [9], a similar diagram was provided in Figure 4.19 for a subcarrier basis, for the non-simplified design. The percentages were found to be 8% for the likelihood computation, 77% for the channel estimator and 15% for the decoder. If the percentage representing the decoder is very similar to our results, we were not able to reproduce a situation in which the channel estimator would be the dominant algorithm.

5.3 Overall Performance

The last two sections presented separately the results of the algorithms in terms of error rates and complexity. In this section, we try to give an insight on the overall performance of our receivers by simultaneously taking in account both parameters.

We proceed as follows. First, we use for all algorithms the same frame length (36 OFDM symbols), relative speed ($v = 100$ km/h), modulation (BPSK) and channel model

(ITU Vehicular A). We also choose a SNR value, namely $E_b/N_0 = 20$dB. This choice is explained by the fact that the receivers show high performance differences for this value. In Figure 5.10, we present for each algorithm the BER achieved for this SNR value, and the associated execution time.

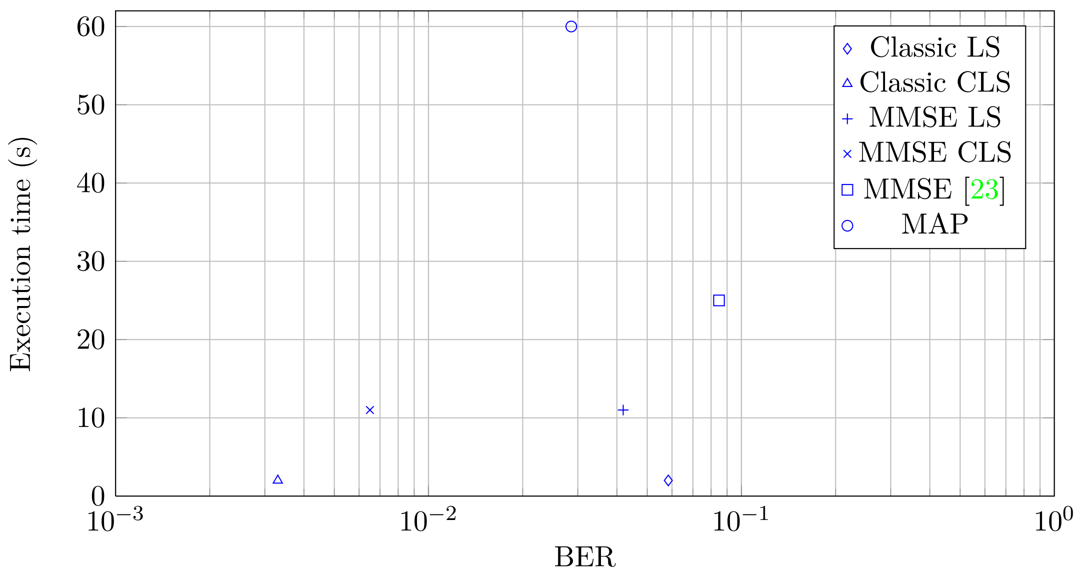


Figure 5.10: BER/complexity tradeoff for the implemented receivers.

On Figure 5.10, an ideal design (low execution time and low bit error rates) would be in the bottom left-hand corner. However, in real situations it is often not possible to have both advantages simultaneously, and tradeoffs are required. We believe that this diagram is a good tool if one needs to make such decisions. However, it is only a snapshot of a given error rate for a given situation (frame length, speed, modulation, SNR value), so the parameters have to be carefully chosen.

With the channel model used, we see that comb initializations give the best performance, both for the MMSE and classical receivers. The MAP receiver features a very high execution time but for a limited performance.

5.4 A Note Regarding Hardware Implementation

The final goal is to have the receiver designs implemented on testbeds so they can be evaluated in real use conditions. Consequently, in this section we provide some elements which would have to be specifically taken in account when dealing with such systems. Following the guidelines provided in [14], we focus on the required numerical precision and the number of operations. Memory footprints will also be discussed.

5.4.1 The Max-Log Approximation

It can be seen from Section 5.2.1 that for the classical and MMSE designs, the most time-consuming part is the BCJR decoding. More precisely, the Jacobian logarithm operation introduced in Section 4.2 is intensively called by this algorithm and accounts for approximately two thirds of its running time. It would therefore be interesting to improve the computation of this operation.

Recall from Equation (4.19) that the Jacobian logarithm formula consists of the sum of two parts: a max operation and a logarithm with an exponential in its argument. One possibility is to quantize the values taken by the logarithmic part and use a table look-up to pick the closest match. This introduces errors, but since the values of the logarithmic part lie between 0 and $\log 2$, these errors are limited. This observation leads to a second approximation, known as the max-log approximation. In this second case, the Jacobian logarithm equals to a max operation. By dropping the log and exp computations, it makes the algorithms much faster thanks to the reduced number of operations, although the theoretical overall complexity of the BCJR decoder remains unchanged.

We were interested in the influence of this second approximation on the error rates. For this purpose, we took two "extreme" cases: the classical design, for which a BCJR is only used for decoding, and the iterative MAP design which uses BCJR both for decoding and channel estimation. The data was gathered with the single subcarrier version of the MAP and the frame version of the classical design, using BPSK modulation and with frames of 36 OFDM symbols. 100 frames were simulated for each SNR value, which can account for the lack of smoothness of the presented curves.

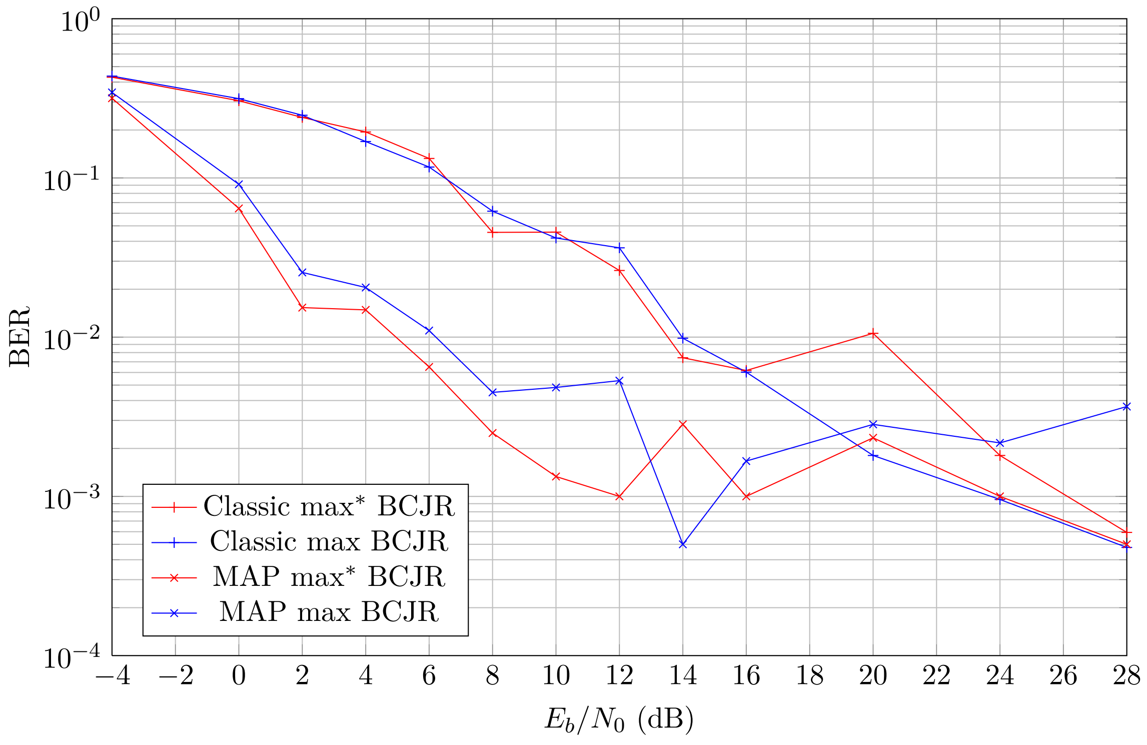


Figure 5.11: Influence of the max-log approximation on the classical and MAP receivers on the BER.

From Figure 5.11, we see that in the case of the classical receiver, the performance is very similar for both cases, while for the MAP receiver, the error rate can sometimes be doubled. Such a simplification would therefore not be desirable in all cases; instead, it clearly requires a trade-off between execution time and accuracy. As an example, we run the classical receiver using MATLAB's profiler. We obtain a running time for the conventional BCJR of 3.57 s when the approximation gives a running time of 140 ms. This suggests that the max-log approximation would be especially interesting to reduce computation times of non iterative receivers. It should be noted here that following the

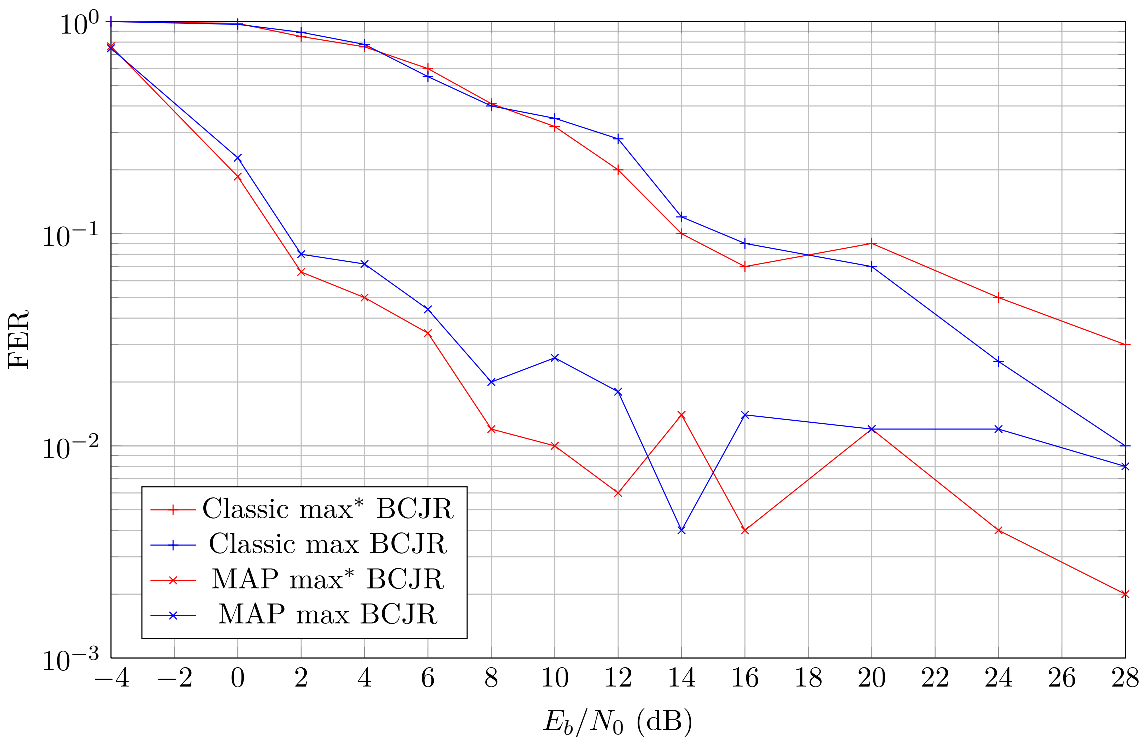


Figure 5.12: Influence of the max-log approximation on the classical and MAP receivers on the FER.

description in [23], the design of Zemen et al. was only simulated with the max-log version of the BCJR.

5.4.2 Comparison of the Algorithms

The MAP receiver requires for initialization channel likelihoods and the channel transition probability matrix. These two parts perform several function integrations, which account for most of the running time of the likelihood computations. Therefore, the loss of precision due to fixed-point systems should be carefully considered. Furthermore, a lot of data needs to be stored at runtime, such as $MN_aN_{ph}N_{csif}$ likelihoods and $N_aN_{ph}N_{csif}$ probabilities from the mapper. The BCJR channel estimation will also have a high memory footprint due to the quadratic dependency on the number of possible channel coefficients.

The algorithm in [23] uses a specific matrix factorization because of the poor conditioning of the matrix. In one of the simulations, the eigenvalues had an order of magnitude ranging from 10^2 to 10^{-30}. Consequently, trying to set a precision limit to accomodate the fixed-point architecture of testbeds [14] would prove to be difficult. The other problem of this algorithm is that, in order to prevent excessively costly computations, several values have to be stored: $A+A'$ hypothesis submatrices split in eigenvectors and eigenvalues, $A+A'$ submatrices used solely for the hypothesis test process, and values of G_a for quantized noise variances for each possible submatrix for the test. The size of the matrices involved in Equation (3.38) can not be neglected either: typically, a short 36-symbol frame requires a 2432×2432 matrix of complex numbers. Finally, this receiver requires matrix inversions, which are especially costly both in storage and number of operations. All these factors make this algorithm particularly unsuitable for an embedded system.

Based on the results found in this chapter, we would recommend for implementation the classical receiver with comb linear interpolation, because it is fast, efficient and works for all constellations in the standard. However, these results would have to be verified with other scenarios, involving different channel models and/or implementation for use in real conditions.

Chapter 6

Conclusions

Wireless vehicular communication is a key technology to enable ITS, which will allow for more safety on the roads and better traffic management. Standards are currently in place, and the corresponding frequency bands are allocated. Unfortunately, the channels usually encountered with vehicular communication pose challenges both regarding the scale of the network and the high mobility of the nodes. Several research projects, also involving car equipment manufacturers, are ongoing in order to deal with these issues. Especially, the time-varying and dispersive nature of the channel impedes link robustness. In this thesis, we dealt with receivers able to reach an acceptable performance over such channels.

To ensure interoperability, we restricted ourselves to fully 802.11p compliant frames. Following a previous work, we used a standard-compliant transmitter, and a channel model which took into account the noise and selectivities of real channels. For systems operating in such poor conditions, a crucial point is the channel estimation step. However, the standard only specifies two pilot symbols at the beginning of each subcarrier, and four pilot subcarriers whose frequency spacing is too large to allow for proper channel estimation. Our first step was to make the existing algorithms (classical, MMSE and MAP) operate on complete frames. The MMSE receiver uses a Wiener filter, and takes in account the autocorrelation of the channel. The MAP receiver relies on a Markov modeling of the channel. These two designs are iterative, which means they refine their channel estimates by taking in account the extra information provided by the decoder. Motivated by the promising results previously obtained, we also implemented another MMSE algorithm from the literature for comparison purposes. It used a subspace selection to adapt itself as precisely as possible to the exact delay and Doppler spread induced by the channel, instead of always supposing the maximum values. Finally, we gave the classical and MMSE receivers the ability to initialize themselves either with block pilots or with comb subcarriers.

We were interested both in the error rates and complexities of the implemented designs. The complexity was measured by approximating the theoretically required number of operations, and through execution times in order to give a more practical overview. It turned out that the MAP decoder, which outperformed all the others on the subcarrier case, is still the best in terms of FER, but at the cost of a very high complexity. The adaptive receiver design in [23], despite promising results in the literature, was shown to achieve much worse error rates without the extra "postamble" at the end of the frame. Furthermore, its computational costs were found to be unacceptably high [10], especially when dealing with long frames. For the classical and MMSE designs, comb pilots improved the performance, especially at high SNRs, by suppressing the error floor encountered while using block pilots. MMSE had a small advantage; unfortunately, this design only works with PSK modulations. It appears that a design matching all constellations with an acceptable performance and complexity is not available yet. One of the biggest difficulties was to compare designs with the existing literature, since there is no common channel model between all the articles.

Future Work As far as entertainment is concerned, recent news from car manufacturers currently show a preference for 4G networks[1]. For safety-related uses, it would be of much interest to slightly modify the protocol in order to add a postamble, as it can lead to high performance improvements with only minor changes to existing algorithms. Using the pilots of the next frame could have been an interesting workaround, but the delay between frames for security functions is too high for this scheme to work effectively [23]. Another suggestion would be to use diversity through a MIMO system. In this work, estimation of the noise variance from measurements has not been taken in account, although it would be required in real receivers. A possibility would then be to implement several decoding algorithms, each one with its optimal SNR domain, and call at runtime the most efficient one given the estimated SNR. Finally, hardware implementations clearly appear as compulsory for performance evaluation, given the high number of different channel models and parameters used in the literature. They would not only provide an insight on performance in real conditions, but also weigh in favour of (or against) some of the channel models currently in use for vehicular scenarios. In the long run, this would ideally reduce the number of models chosen for numerical simulations, hence allowing easier comparison of the results.

[1]For example, General Motors announced in February 2013 that their new 2014 models would have an embedded 4G connection for such purposes, at least in the US. `http://www.usatoday.com/story/money/cars/2013/02/25/gm-general-motors-4g-att-onstar/1939583/`

Bibliography

[1] IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pages 1–1076, 2007.

[2] G. Acosta-Marum and M.-A. Ingram. Six time- and frequency- selective empirical channel models for vehicular wireless LANs. *Vehicular Technology Magazine, IEEE*, 2(4):4–11, December 2007.

[3] P. Alexander, D. Haley, and A. Grant. Cooperative intelligent transport systems: 5.9-GHz field trials. *Proceedings of the IEEE*, 99(7):1213–1235, July 2011.

[4] M.-A. Badiu, G.E. Kirkelund, C.N. Manchon, E. Riegler, and B.H. Fleury. Message-passing algorithms for channel estimation and decoding using approximate inference. In *Information Theory Proceedings (ISIT), IEEE International Symposium on*, Cambridge, MA, July 2012.

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

[6] S. Beheshti and M.A. Dahleh. A new information-theoretic approach to signal denoising and best basis selection. *Signal Processing, IEEE Transactions on*, 53(10):3613–3624, October 2005.

[7] L. Bernadó, N. Czink, T. Zemen, and P. Belanovic. Physical layer simulation results for IEEE 802.11p using vehicular non-stationary channel model. In *Communications Workshops (ICC), IEEE International Conference on*, Cape Town, South Africa, May 2010.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms.* MIT Press, 2001.

[9] O. Goubet. Receiver design for vehicular communications. Master's thesis, Royal Institute of Technology, 2013.

[10] O. Goubet, G. Baudic, F. Gabry, and T. J. Oechtering. Low complexity scalable iterative receiver algorithm for IEEE 802.11p. *Vehicular Technology, IEEE Transactions on*, 2013. (to be submitted).

[11] R. Jain. Channel models, a tutorial, February 2007.

[12] U. Madhow. *Fundamentals of Digital Communication.* Cambridge University Press, 2008.

[13] C.F. Mecklenbräuker, A.F. Molisch, J. Karedal, F. Tufvesson, A. Paier, L. Bernadó, T. Zemen, O. Klemp, and N. Czink. Vehicular channel characterization and its implications for wireless system design and performance. *Proceedings of the IEEE*, 99(7):1189–1212, July 2011.

[14] P. Paschalidis. Complexity guidelines for implementation of signal processing algorithms on FPGA-board. (private communication).

[15] L.L. Scharf and D.W. Tufts. Rank reduction for modeling stationary signals. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 35(3):350–355, March 1987.

[16] V. Shivaldova. Implementation of IEEE 802.11p physical layer model in SIMULINK. Master's thesis, Vienna University of Technology, 2010.

[17] D. Slepian. Prolate spheroidal wave functions, Fourier analysis, and uncertainty-V: The discrete case. *Bell System Technical Journal*, 57(5):1371–1430, May/June 1978.

[18] D. S. Tracy and R. P. Singh. A new matrix product and its applications in partitioned matrix differentiation. *Statistica Neerlandica*, 26(4):143–157, December 1972.

[19] H. Wymeersch. *Iterative Receiver Design.* Cambridge University Press, 2007.

[20] Y. Yapici and A.O. Yilmaz. Joint channel estimation and decoding with low-complexity iterative structures in time-varying fading channels. In *Personal, Indoor and Mobile Radio Communications, IEEE 20th International Symposium on*, Tokyo, Japan, September 2009.

[21] T. Zemen, L. Bernadó, N. Czink, and A.F. Molisch. Iterative time-variant channel estimation for 802.11p using generalized discrete prolate spheroidal sequences. *Vehicular Technology, IEEE Transactions on*, 61(3):1222–1233, March 2012.

[22] T. Zemen, C.F. Mecklenbräuker, F. Kaltenberger, and B.H. Fleury. Minimum-energy band-limited predictor with dynamic subspace selection for time-variant flat-fading channels. *Signal Processing, IEEE Transactions on*, 55(9):4534–4548, September 2007.

[23] T. Zemen and A.F. Molisch. Adaptive reduced-rank estimation of nonstationary time-variant channels using subspace selection. *Vehicular Technology, IEEE Transactions on*, 61(9):4042–4056, November 2012.