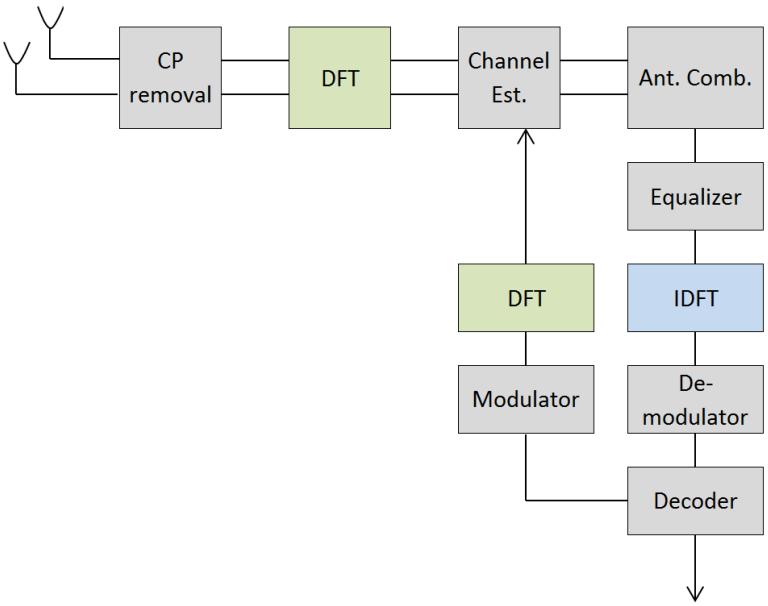
CHALMERS



Iterative Channel Estimation for LTE UL

Master's Thesis in Communication Engineering

ANDERS MARKOFF

Department of Signals and Systems
Chalmers University of Technology
Gothenburg, Sweden 2014

Iterative Channel Estimation for LTE UL

Master's Thesis in Communication Engineering

ANDERS MARKOFF

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Signals and Systems
Chalmers University of Technology
Gothenburg, Sweden 2014
Master's Thesis EX025/2014

Iterative Channel Estimation for LTE UL
Master's Thesis in Communication Engineering
ANDERS MARKOFF

Master's Thesis EX025/2014
Department of Signals and Systems
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 (0)31-772 10 00

Cover:
Illustration of a simple communication receiver model with a feedback loop.

Chalmers University of Technology
Gothenburg, Sweden 2014

Abstract

LTE is a mobile communications standard that is spreading fast across the world. Two of the reasons are the high throughput offered and the adaptability in spectrum allocations. To be able to offer a high throughput the communication channel between the UE and the base station has to be estimated accurately. In order to estimate said channel, two of the fourteen symbols in each subframe are on forehand known reference symbols; if the channel estimate based on the reference symbols is not accurate enough and the decoding of the data bits fail, a retransmission is requested.

This thesis presents a method to reuse *all* the decoded data symbols instead of only the reference symbols in order to open up the opportunity to perform a new iteration of channel estimation if the decoding fails, therefore reduce the number of required retransmissions and increase the throughput. The method is based on a weighted least square estimate combined with a polynomial model approximation. The performance is analysed as well as the implementation aspects for a multicore DSP platform.

Simulations are presented showing a SNR gain for one iteration of 0.5 to 0.6 dB. For two iterations the SNR gain is slightly higher, about 0.15 dB higher than for one iteration. Although the simulations shows a SNR gain, the implementation cost of the proposed algorithm is high. Unfortunately, with today's hardware limitations, it's unlikely that the proposed algorithm can be implemented in a product.

Keywords: Iterative channel estimation, LTE uplink (LTE UL), Weighted least-square channel estimation

Acknowledgements

I would like to take this opportunity to thank Erik Lindén for the cooperation during this project, Alan Said for the many hours of explaning and as sounding board for my idéas, Jinliang Huang, Johan Parin, Fredrik Huss and everyone else at Ericsson who helped me when things didn't always went according to the plan, Alexandre Graell i Amat at Chalmers for accepting to be my examiner despite my very short notice and last but not least all my friends at Chalmers whom has helped me during my entire studies.

Anders Markoff, June 2014

Contents

1

Introduction

1.1 Introduction

In the past 20 years, mobile communication has spread across the world fast. In 1992 the first digital mobile telephone system, GSM, was launched in Europe[1]. Since then the urge for faster connections have been a constant subject for research and development.

The latest standard for mobile broadband is called LTE. The first commersial LTE network was launched by Telia in 2009[2]. Since then the development has continued, and the latest version of LTE is release 13.

In order for the receiver to decode the received data, it has to have knowledge about the channel between the UE and the base station. In order to gain such knowledge, two of the fourteen symbols in each subframe is on forehand known reference symbols. These reference symbols are then used by the receiver to estimate the channel. However, if the channel estimate is not accurate enough the decoding will fail and the data has to be scheduled for retransmission. A retransmission is computationally costly, introduces time delays and requires reallocation of the spectrum. For the end user, it is perceived as a lower throughput.

The scope of this thesis is to investigate if an iterative channel estimation algorithm can be used to improve the throughput of LTE UL. Also investigate the possibility of implementing the algorithm on a hardware multicore DSP platform.

This thesis will only focus on the uplink (UL) part of LTE, that is when data is sent from the user equipment (UE) to the base station. An iterative channel estimation algorithm for LTE downlink (DL) has been analysed in [3]. However, the frame structure and placements of the pilot symbols are different from LTE UL which makes it hard to compare the absolute values between [3] and the proposed algorithm in this thesis.

1.2 Thesis outline

In chapter 2 some background principles of multicarrier transmission is explained, as well as the memory handling of fixed point numbers. The basic structure of the LTE uplink PUSCH channel is also addressed. In chapter 3 the principle of iterative channel estimation is presented together with the proposed algorithm and some implementation aspects. Results from both simulations and a hardware implementation is found in chapter 4, and are discussed in chapter 5.

Abbreviations and acronyms can be found in appendix A.

2

Background

2.1 OFDM

OFDM stand for Orthogonal Frequency Division Multiplexing. Frequency division means that the data is divided into two or more frequency bands in comparison to conventional single carrier transmission where all the data is transmitted on the same carrier frequency. Typically an OFDM system uses a large amount of relatively narrow frequency bands called subcarriers to transmit an OFDM symbol. Multiplexing means that many data streams are combined into one single data stream. In OFDM many modulation symbols are combined together to form one OFDM symbol, see equation 2.1.

The biggest advantage of OFDM compared to conventional single carrier transmission is the capabilities to cope with bad channel conditions including, but not limited to, frequency-selective fading from multipath and intersymbol interference (ISI) without the use of a complex equalizer[4]. The reason for this is that when using OFDM the modulation symbols are spread over many narrow frequency bands. This has many advantages. One is that even if the channel suffers from frequency selective fading over the total bandwidth, the channel over the narrow bandwidth utilised by one of the subcarriers is fairly flat. Another advantage is that N_c parallel subcarriers require a symbol-time N_c times longer than for a single carrier system. This means that the ISI relative the symbol-time is much smaller for the OFDM system. Hence, the use of a guard interval or cyclic prefix doesn't impact the overall system performance as much.

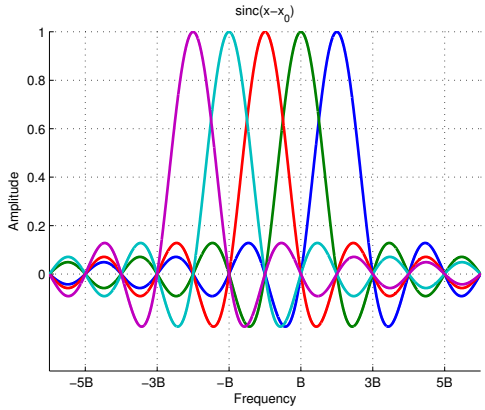


Figure 2.1: Illustration of orthogonal sinc pulses with bandwidth B and relative frequency shift nB

2.1.1 OFDM Modulation

OFDM uses rectangular pulse shape with symbol-time T_u. In frequency domain, this means that each subcarrier is a sinc pulse with bandwidth $B = 1/T_u$. From the nature of the sinc pulse it is orthogonal to all other sinc pulses with the same bandwidth and relative frequency shift nB, where $n \in \mathbb{Z}_{\neq 0}$[5]. This is illustrated in figure 2.1.

By utilizing the previous fact, the OFDM signal uses a subcarrier spacing $\Delta f = B = 1/T_u$. The OFDM signal x(t) can therefore be described by

$$x(t) = \sum_{k=0}^{N_c-1} x_k(t) = \sum_{k=0}^{N_c-1} a_k e^{j2\pi k\Delta f t} \tag{2.1}$$

where a_k is a complex modulation symbol put on the kth subcarrier and N_c is the total number of subcarriers. This means that the OFDM pulse is modulation independent and that the limiting factors for the number of points in the constellation are the channel properties. One should also notice that if the channel is good for subcarrier $k \leq \alpha$ different modulation schemes may be used for subcarriers $k \leq \alpha$ and $k > \alpha$. The modulation schemes used in LTE are QPSK(4QAM), 16QAM and 64QAM, both in uplink (UL) and downlink (DL)[6, Table 5.3.2-1, 6.3.2-1].

The orthogonality of the sinc pulses means that the inner product of the pulses are zero over the symbol-time.

$$\langle f(k_1), f(k_2) \rangle = \int_{mT_u}^{(m+1)T_u} e^{j2\pi k_1 \Delta f t} e^{-j2\pi k_2 \Delta f t} dt = 0 \tag{2.2}$$

Where m notes the mth OFDM symbol and $k_1 \neq k_2$. [7, p.399]

By extending equation 2.2 to the inner product of two modulated OFDM subcarriers, it is clear that any two modulated subcarriers are orthogonal over one symbol-time.

$$\int_{mT_u}^{(m+1)T_u} x_{k_1}(t)x_{k_2}^*(t)dt = \int_{mT_u}^{(m+1)T_u} a_{k_1}a_{k_2}^* e^{j2\pi k_1 \Delta f t} e^{-j2\pi k_2 \Delta f t} dt = 0 \tag{2.3}$$

For $k_1 \neq k_2$, where x^* denotes the complex conjugate of x.

2.1.2 OFDM Demodulation

To demodulate the symbol on one the lth subcarrier, the signal is first multiplied with the complex conjugate of the lth subcarrier and then integrated over the symbol-time. This is mathematically the same thing as equation 2.3.

$$\int_{mT_u}^{(m+1)T_u} \sum_{k=0}^{N_c-1} a_k e^{j2\pi k \Delta f t} e^{-j2\pi l \Delta f t} dt = \begin{cases} a_k, & \text{if } k = l \\ 0, & \text{if } k \neq l \end{cases} \tag{2.4}$$

By executing N_c parallel branches of equation 2.4 with $l = [0, 1...N_c - 1]$ all the complex modulation symbols a_k are extracted.

For a regular frequency division multiplexing system, the orthogonality is achieved from the fact that the symbols are non-interleaving in frequency domain. From equation 2.4 it should be clear that for perfect channel conditions, no intercarrier interference should occur by using OFDM, even though the symbols overlap in frequency domain as seen in figure 2.1. The orthogonality in OFDM rather comes from the choice of subcarrier spacing Δf and it's relation to the symbolrate T_u where $\Delta f = 1/T_u$.

2.1.3 OFDM using DFT

The discrete Fourier transform (DFT) is defined as [8, 13.3]

$$F(k) = \frac{1}{N} \sum_{n=0}^{N-1} f(n) e^{-j2\pi kn/N} \tag{2.5}$$

and the inverse discrete Fourier transform (IDFT) as [8, 13.3]

$$f(n) = \sum_{k=0}^{N-1} F(k) e^{j2\pi kn/N} \tag{2.6}$$

When sampling the OFDM signal in equation 2.1 with a sampling rate f_s that is a multiple N of the subcarrier spacing, $f_s = 1/T_s = N\Delta f$, the resulting sampled OFDM signal x_n is computed

$$x_n = x(nT_s) = \sum_{k=0}^{N_c-1} a_k e^{j2\pi k \Delta f n T_s} = \sum_{k=0}^{N_c-1} a_k e^{j2\pi kn/N} \tag{2.7}$$

If N is chosen such as $N \geq N_c$ and $a_k = 0, \forall k > N_c$ it is possible to create $x(t)$ with an IDFT by defining $F(k)$ as a_k, and as stated above $\Delta f = 1/(NT_s)$. This is illustrated in figure 2.2

$$x_n = \sum_{k=0}^{N} a_k e^{j2\pi kn/N} = \text{IDFT}(a_k) \tag{2.8}$$

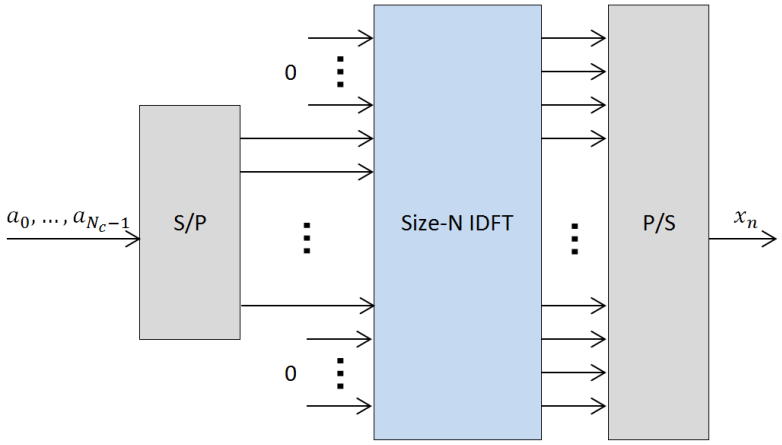


Figure 2.2: Illustration of how the OFDM signal is produced using an IDFT

In figure 2.2 only one UE is transmitting using parts of the entire system bandwidth. Another UE can transmit at the same time using the other subcarriers. In LTE, the scheduler in the base station decides which UE that should transmit when and using which subcarriers.

From the properties of Fourier transform [8, 13.3] the demodulation of an OFDM signal can be made using an DFT.

$$\text{DFT}\{\text{IDFT}\{F(k)\}\} = F(k) \tag{2.9}$$

If N is chosen to be equal to 2^m for any integer m, the fast Fourier transform (FFT) and inverse fast Fourier transform (IFFT) can be used for computational efficiency. The number of operations for an FFT is $\approx N \log_2 N$ as compared to $\approx N^2$ for DFT.

2.1.4 Cyclic prefix

When sending a discrete-time signal $x[n]$ through a linear time-invariant (LTI) channel the output $y[n]$ is the linear convolution between the input $x[n]$ and the channel impulse response $h[n]$. [9, eq:12.15]

$$y[n] = h[n] * x[n] = \sum_k h[k]x[n-k] \tag{2.10}$$

The circular convolution is specified as [9, eq:12.16]

$$y[n] = h[n] \circledast x[n] = \sum_k h[k]x[n-k]_N \tag{2.11}$$

where $x[n-k]_N$ is the periodic extension of $x[n-k]$ with period N. From the DFT definition circular convolution in time is equal to multiplication in frequency. [9, eq:12.17]

$$\text{DFT}\{x[n] \circledast h[n]\} = X[k]H[k], \quad 0 \leq k \leq N-1 \tag{2.12}$$

Adding a cyclic prefix is defined as copying the last μ samples of the input signal $x[n]$ and add them before the actual signal:

$$\tilde{x}[n] = \begin{bmatrix} x[N-\mu] & x[N-\mu+1] & \cdots & x[N-1] & x[0] & x[1] & \cdots & x[N-1] \end{bmatrix} \tag{2.13}$$

for this signal

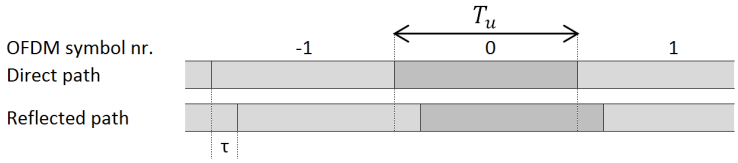
$$\tilde{x}[n] = x[n]_N \quad \text{for } -\mu \leq n \leq N-1 \tag{2.14}$$

For a system with a channel $h[k]$ with impulse response of length $\leq \mu$ and using $\tilde{x}[n]$ as input the output $y[n]$ is equal to the circular convolution of the channel impulse response and $x[n]$. [9, eq:12:18]

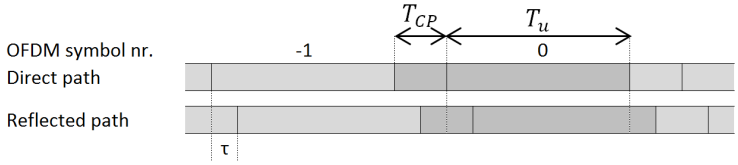
$$y[n] = \tilde{x}[n] * h[n] = \sum_{k=0}^{\mu} h[k]\tilde{x}[n-k] = \sum_{k=0}^{\mu} h[k]x[n-k]_N = x[n] \circledast h[n] \tag{2.15}$$

For a time-dispersive channel, e.g. when the signal has one direct path and one or more reflected paths between the UE and the base station, the reflected signal will be received in the base station with a time shift τ. Since the last part of the signal won't be received, the integration interval over one OFDM symbol will not be equal to a full period. This will introduce a frequency shift in the received signal and the orthogonality between the subcarriers will be lost[4]. By using a cyclic prefix, as long as $\tau \leq T_{CP} = \mu T_s$, the base station will receive one complete period of the signal and the orthogonality will be retained. This is illustrated in figure 2.3. Since equation 2.8 is periodical with period N (property of the IDFT [8]), it doesn't matter if the time shifted signal is sampled during parts of the cyclic prefix instead of at the end. The DFT demodulation will work as long as a full period of the OFDM signal is sampled.

The cyclic prefix also acts as a guard interval between the previous OFDM symbol and the current. In a system where the channel impulse response is longer than the guard interval ISI will be present.



(a) Illustration of one direct path and one time shifted path without the use of cyclic prefix



(b) Illustration of one direct path and one time shifted path using a cyclic prefix of length $T_{CP} \geq \tau$

Figure 2.3: Illustration of the difference using a cyclic prefix in a time dispersive environment

One drawback of using cyclic prefix is the extended length of the signal since the total length is $T_u + T_{CP}$. This results in a loss of capacity. The standard length of the cyclic prefix in LTE is $T_{CP} = 4.69\mu s$ as compared to the symbol time of $T_u = 66.7\mu s$. This results in a capacity loss of about 7% [10]. There is also specified an extended cyclic prefix, for situations where the channel is very time dispersive, of length $T_{CP-e} = 16.7\mu s$ [11]. To decrease the capacity loss the symbol length could be increased. This would however result in a narrower subcarrier spacing and therefore a greater sensitivity to phase shifts.

The reason that cyclic prefix can be used at all, is the long symbol time on OFDM. In W-CDMA the symbol time is 256 times shorter than in LTE and the capacity would drop with a factor of 20 if the same cyclic prefix would be used [10].

2.1.5 Disadvantages of OFDM

OFDM has two big disadvantages compared to single-carrier systems. First, when the number of subcarriers increase, the OFDM time-domain signal starts to look much like Gaussian noise[10]. This is because the amplitude and phase of all subcarriers are independent and has the disadvantage of high peak-to-average power ratio (PAPR). High PAPR requires the power amplifier in the transmitters to be linear over a large span to avoid distortion and saturation. In order to achieve this linearity, either a very expensive amplifier has to be used, or the amplifier has to work with a large backoff from the peak power. This is not very power efficient [12]. For base stations, a lower power efficiency may be tolerated but not for a battery powered user equipment (UE)[4]. Second, the tight spacing between subcarriers makes the system vunerable to frequency errors since a small frequency error will cause the subcarriers to lose orthogonality. As described above in 2.1.4 a long symbol time is desired in order to be able to use a cyclic prefix without a large loss in capacity. But with the relation $\Delta f = 1/T_u$ from section 2.1.1 the longer the symbol time the narrower the subcarrier spacing.

2.2 SC-FDMA

In order to solve the problem with high PAPR in OFDM, but still taking advantage of the benefits of the robustness against multipath propagation, singe carrier frequency division multiple access (SC-FDMA) is used in uplink LTE. SC-FDMA can be described as a regular OFDM system, but with the input data precoded in order to create a signal with lower PAPR. The precoding is done with a DFT of size N_c. For obvious reasons, if $N_c = N$ the DFT and IDFT will cancel each other out and the modulation will not work. However, if $N_c \leq N$ and the other $N - N_c$ inputs to the IDFT is set to zero the output will have the same properties as a single-carrier wideband signal. In figure 2.4 the SC-FDMA modulation system is illustrated. If compared to 2.2 the only difference is the DFT.

It is possible to distribute the DFT modulated symbols in different ways to the input of the IDFT. In figure 2.4 all the symbols are localised to adjacent subcarriers. In [12] Myung, Lim and Goodman shows that even lower PAPR can be achieved by distributing the symbols to equidistant subcarriers across the spectrum allowing symbols from other UEs transmitting at the same time using the subcarriers in between.

By precoding the modulation symbols $a_0, ..., a_{N_c-1}$ through a DFT of size N_c every modulation symbol will be spread across the entire bandwidth. This way the signal amplitude will be more constant in the frequency direction. Figure 2.5 illustrates this. The purpose is to send 8 symbols, using four subcarriers, in the sequence:

$$\Big\{orange \quad green \quad red \quad blue \quad green \quad orange \quad blue \quad red\Big\}$$

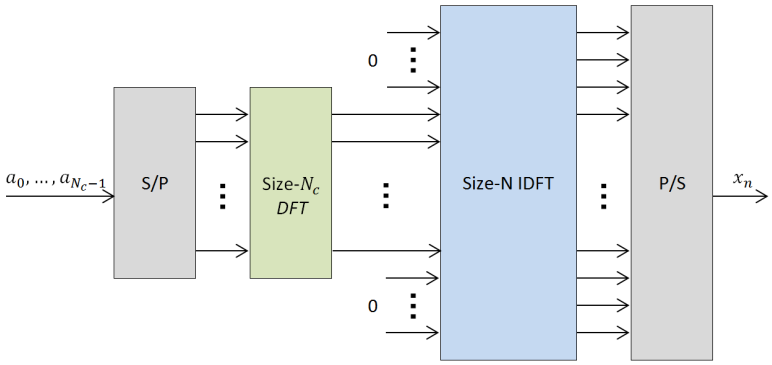
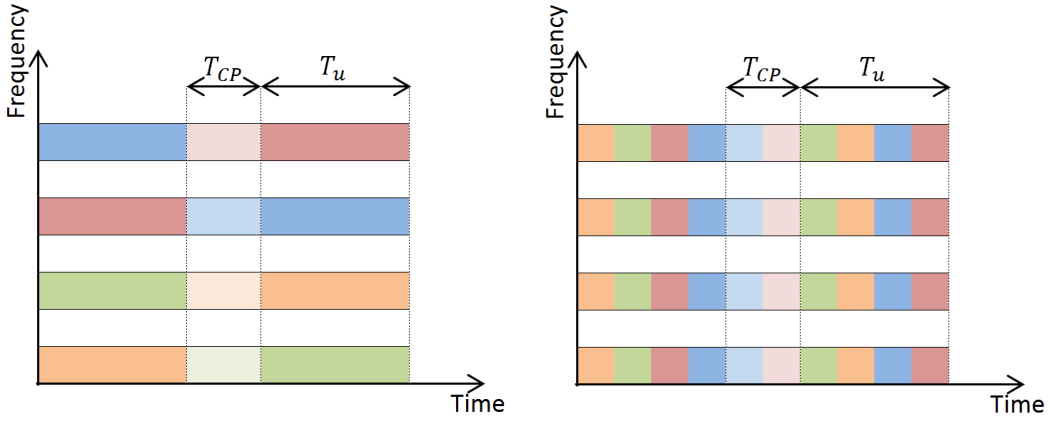


Figure 2.4: Illustration of how the SC-FDMA signal is produced using an DFT to precode the data before regular OFDM modulation. This illustrates the localised SC-FDMA signal where all the signals from one UE is localised to adjacent subcarriers.

The different colors represents different modulation symbols from any modulation alphabet. In figure 2.5a this is illustrated using OFDM. If the amplitude or phase is very different between the symbols, the PAPR will be quite large. In figure 2.5b the symbols are precoded and spread over the bandwidth. However, since the same OFDM modulator is used, the subcarrier is constant over the entire period T_u and the cyclic prefix can be used to cancel the multipath.



(a) Illustration of an OFDM transmission with cyclic prefix

(b) Illustration of the same transmission as in 2.5a but using SC-FDMA

Figure 2.5: Illustration of the difference between OFDM and SC-FDMA in a time-frequency grid

Another name for this operation is DFT-spreading, which quite literally explains that the symbols are spread over the bandwidth using a DFT, as seen in figure 2.5.

The wideband properties of the SC-FDMA is however not only positive. As mentioned before in section 2.1, for a wideband signal the probability of the channel to appear as frequency selective is much larger as apposed to a narrowband signal.

All channels can be described using a set of parameters, one of the most used is the coherence bandwidth B_c. B_c is defined as the frequency where $A_C(\Delta f) \approx 0$ where A_C is the Fourier transform of the channel autocorrelation. Hence, the coherence bandwidth is the frequency separation where the channel fades independently. If the bandwidth B of a signal is much smaller than the coherence bandwidth, $B \ll B_C$, as for the OFDM subcarrier, the channel appears flat. For a wideband signal however, $B \gg B_C$ and the channel appears as frequency selective[9].

In order to handle the frequency selectivity in SC-FDMA, equalization is used in the demodulation.

2.3 Equalization

Equalization is a method to counteract the distortion of the sent signal introduced by the channel.

A simple communication system in baseband can be described by a modulation symbol a_k pulseshaped by a pulse $g(t)$, transmitted over a channel $c(t)$, filtered in the receiver with a matched filter $g_m(t)$ and then sampled. This results in the following equation where $y(t)$ is the signal before sampling and n(t) is noise.

$$y_k(t) = \{a_k * g(t) * c(t) + n(t)\} * g_m(t) \tag{2.16}$$

The matched filter $g_m(t)$ is used to maximize the SNR in $y(t)$. Optimally the matched filter wold be the convolution between the pulseshape and the channel. However, since the channel is unknown, the matched filter is often implemented as the time inverse of the pulseshape conjugate, giving $g_m(t) = g * (-t)$. This is of course not ideal if the channel $c(t) \neq \delta(t)$.

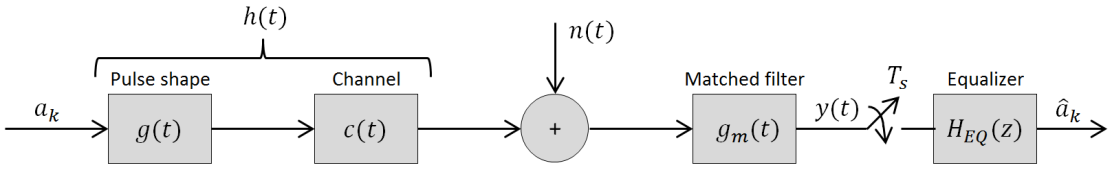


Figure 2.6: Illustration of the simple communication system used to describe the equalization. Se equation 2.16.

By specifying

$$h(t) = g(t) * c(t) \tag{2.17}$$

and

$$f(t) = h(t) * g_m(t) \tag{2.18}$$

equation 2.16 can be rewritten as

$$y(t) = a(t) * f(t) + n(t) * g_m(t) \tag{2.19}$$

and it's z-transform as

$$Y(z) = A(z)F(z) + N_g(z) \tag{2.20}$$

where $N_g(z)$ is the z transform of $n(t) * g_m(t)$ and $a(t) = \sum_k a_k \delta t(t - T_s)$.

2.3.1 Zero-Forcing Equalizer

The simplest form of equalizer is the zero-forcing equalizer. It inverts the channel, and is specified as

$$H_{ZF}(z) = \frac{1}{F(z)} \tag{2.21}$$

This equalizer completely removes any ISI caused by the channel and is in that way ideal. The problem with the zero-forcing equalizer occurs when the channel has deep fades.

For a signal of bandwidth B distorted with white noise with power spectral density $N_0/2$, the noise power of the signal is $N_0 B$. The power spectrum of the noise samples after the equalizer is [9, eq:11.13]

$$N(z) = N_0 |G_m(z)|^2 |H_{ZF}(z)|^2 = \frac{N_0 |G_m(z)|^2}{|H(z)|^2 |G_m(z)|^2} = \frac{N_0}{|H(z)|^2} \tag{2.22}$$

From equation 2.22 it is clear that for notches in the channel where $H(z_0) = 0$, the noise is infinitely large. Even if it's not a notch in the channel but only a fade, the SNR after the equalizer will be very low for those frequencies.

2.3.2 Minimum Mean-square Error Equalizer

The minimum mean-square error equalizer (MMSE) is another equalizer that doesn't remove all ISI as the zero-forcing equalizer, but doesn't have the same noise enhancement for bad channel conditions. The MMSE equalizer minimizes the mean-square error $\mathbb{E}[a_k - \hat{a}_k]$ where $\hat{a}_k$ is the estimated modulation symbol, a linear combination of the filter coefficients w.

$$\hat{a}_k = \sum_{i=-L}^{L} w_i y[k - i] \tag{2.23}$$

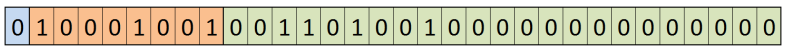


Figure 2.7: Illustration of how the number 1234_{10} is stored in the memory using the IEEE 754 standard binary32[14]. Blue represents sign bit, red represents exponent and green represents the mantissa.

The equalizer then becomes [9, eq:11.26]

$$H_{MMSE}(z) = \frac{1}{F(z) + N_0} \tag{2.24}$$

2.4 Memory handling

For the sake of simplicity, in the following section it is given that the computer uses 32 bits to store the numbers and that all numbers are stored with base 2. A number x of base b is annotated x_b.

2.4.1 Floating point arithmetic

The floating point system divides the number to store in three parts, the sign, mantissa and the exponent. Mathematically it can be described as:

$$(-1)^{sign} \cdot mantissa \cdot base^{exponent} \tag{2.25}$$

For example, the base 10 number 1234_{10} will be stored as $1{,}234 \cdot 10^3$. The name floating point comes from the fact that the exponent ”shifts” the radix point(decimal point for base 10) of the number.

The big advantage of floating point is that the computer can express both small and large values using the same variable. Since the number of significant digits (the size of the mantissa) is the same for all numbers the precision gets worse for large numbers. For example for a mantissa with 4 bits and base 10:

$$\begin{aligned} x_1 &= 1{,}234 \cdot 10^3 \\ x_2 &= 1{,}234 \cdot 10^5 \end{aligned} \tag{2.26}$$

The precision of x_1 is 1, but the precision of x_2 is 100.

How a number is stored in the computer varies from implementation to implementation but in the IEEE 754 standard for a binary32 (commonly known as ”single precision”) the sign 1 bit, the mantissa 23 bits and the exponent 8 bits[13]. This is shown in figure 2.7. This gives a range between approximately 10^{-38} to 10^{38}, base 10.

2.4.2 Fixed point arithmetic

When using fixed point arithmetic all but one (or all if unsigned variables are used) of the bits in the memory are used to store significant bits. This means that the precision is only limited by the size of the memory. The drawback is that the programmer has to remember the exponent rather than letting the computer do it for him or her.

If a programmer wants to store the number $11_2 = 3_{10}$ in the memory he or she stores the sequence $\ldots 0011$ in the memory. If the programmer then wants to store the number $0{,}11_2 = 0{,}75_{10}$ in the memory he or she then would store the sequence $\ldots 0011_2 = 3_{10}$ in the memory, and remember that the radix(binary) point for the second variable has to be moved two steps to the left. In the memory these two variables look the same, but the programmer knows that they represent two different numbers.

The same way of thinking can be applied on very large or small numbers. If the programmer knows that a certain variable will only contain numbers in the interval $[10^{30} \quad 10^{31}]$ he or she can use the full 32 bits for precision as long as he or she remembers that the radix point should be moved back to get the "actual number". This gives higher accuracy than by using floating point arithmetic. Another advantage is that most computers handle fixed point computing much faster than floating point. Especially if the computer doesn't have a special fixed point processing unit.

Q-format

In order to specify how many bits of the memory that should be used for fractional bits and how many that should be used for integers, the Q-format is used. $Qm.n$ means that m bits should be used for integers, and n bits for fractions. It can also be specified as Qn where it means n fractional bits and the rest for integers.

The previous example with the number $0{,}11_2 = 0{,}75_{10}$ is stored in memory as $\ldots 0011_2$ using the format Q2.

Multiplication and division using fixed point arithmetic

Multiplication between two fixed point numbers is a little bit more complicated than multiplication between two floating point numbers if the Q-format should be maintained.

If the number Q2 $x = 011_2 = 0{,}75_{10}$ is multiplied with Q2 $y = 100_2 = 1_{10}$ the result should be Q2 $z = x = 011_2 = 0{,}75_{10}$. Using the bits in the memory the multiplication $z = x \cdot y$ gives:

$$\begin{array}{rrrrrr} & & & 0 & 1 & 1 \\ & & \times & 1 & 0 & 0 \\ \hline & & & 0 & 0 & 0 \\ & & 0 & 0 & 0 & \\ + & 0 & 1 & 1 & & \\ \hline & 0 & 1 & 1 & 0 & 0 \end{array} \tag{2.27}$$

However, Q2 $01100_2 = 3_{10}$. This result is not correct. Q4 $01100_2 = 0{,}75_{10}$ which is correct. If the Q-format of the first factor is to be maintained, the product of the multiplication has to be shifted as many bits right as the number of fractional bits in the second factor.

$$z = (x \cdot y) \gg 2 \tag{2.28}$$

where $\gg$ denotes the bitshift operator gives the correct result.

By using the same example, it can be shown that in order to maintain the Q-format in division, the numerator has to be shifted to the left as many steps as the number of fractional bits in the denominator.

$$z = (x \ll 2)/y \tag{2.29}$$

2.5 LTE-UL

2.5.1 Physical resources

When a UE has data to transmit, the scheduler in the basestation allocates physical resources to that UE in time and frequency where it will listen. The smallest physical resource in LTE is called a resource element. It consists of one subcarrier during one OFDM symbol. Resource elements are then grouped together to form resource blocks consisting of 12 subcarriers and 7 or 6 OFDM symbols, depending on wether normal or extended cyclic prefix is used. See figure 2.8 for an illustration of a resource block with normal cyclic prefix. Either way, the duration of a resource block in time is called a slot and is $T_{slot} = 0.5s$. However, one UE is always allocated at least two subsequent resource blocks in time, one subframe. See figure 2.9.

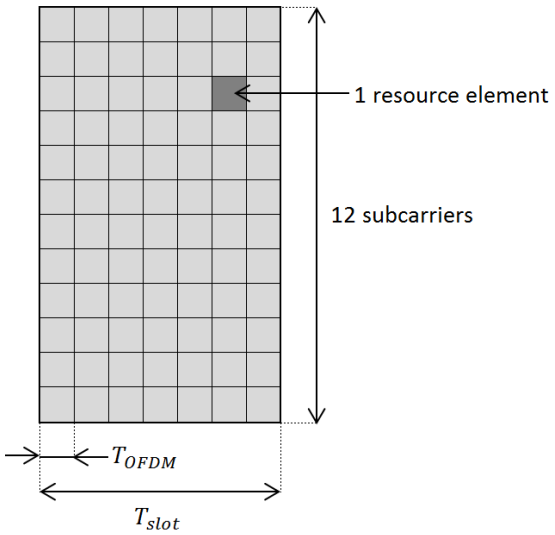


Figure 2.8: Illustration of one resource block using normal cyclic prefix, thus containing 84 resource elements

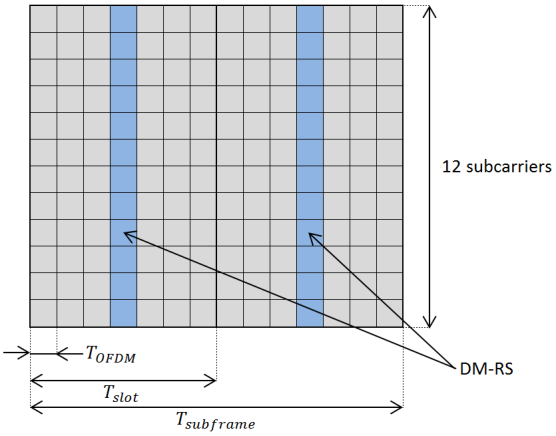


Figure 2.9: Illustration of one subframe using normal cyclic prefix and the two DM-RS

2.5.2 Demodulation Reference Signals

In order to be able to estimate the channel, two of the OFDM symbols in each subframe are not allocated for data symbols, but for demodulation reference signals (DM-RS). When normal cyclic prefix is used the fourth OFDM symbol in the slot is used for DM-RS, see figure 2.9. If extended CP is used, it is the third OFDM symbol that is used.

The symbols used for DM-RS is not regular data, but phase rotated Zadoff-Chu (ZC) sequences[15]. A ZC sequence is a complex data sequence $x[n]$ with constant power in both time and frequency domain.

$$x[n]_{ZC} = e^{-j\pi u \frac{n(n+1)}{N_{ZC}}} \tag{2.30}$$

where N_{ZC} is the length of the ZC sequence and u is the seed. u has to be relative prime[16] N_{ZC}, which limits the number of available ZC sequences. Since a resource block always contain 12 subcarriers, the ZC sequence should be of length $12 \cdot n$ where $n \in \mathbb{N}_{\neq 0}$. In order to be able to use at least 30 different sequences, for $N_{ZC} = 12$ and $N_{ZC} = 24$, special sequences (not from equation 2.30) are specified in [6, 5.5.1.2]. For $N_{ZC} \geq 36$ the reference signal is a cyclic extension of the ZC sequence with length equal to the largest prime number $\leq N_{ZC}$ [4]. Which seed to use depends on the cell identity of the base station.

Two phase rotated Zadoff-Chu sequences from the same seed u are uncorrelated [17]. This fact is used in order to create different demodulation signals from the same seed for the different subcarriers in a resource block.

2.5.3 Matched Filter Channel Estimate

The easiest way to estimate the channel is to use the known DM-RSs and a calculate a matched filter. This is easily done in frequency domain by multiplying the received signal Y at antenna r and subcarrier k with the complex conjugate of the known transmitted DR-RS at subcarrier k. Of course this can be extended for MIMO transmission as well. This is the most basic general example of a channel estimate.

$$\hat{H}_{mf}(k, r) = Y(k, r)X^*(k) \tag{2.31}$$

With this method the channel is estimated for the third OFDM symbol in each slot. By assuming that the channel is roughly time invariant for $T_{slot}/2$ this estimate can be used for all OFDM symbols. This of course requires that the first three OFDM symbols are stored in the receiver memory in order to be decoded once the DM-RS has been decoded. In a realtime environment this can be a bottleneck and has to be accounted for when designing the system.

Another way to estimate the channel is to use the first DM-RS to estimate the channel for all OFDM symbols before the second DM-RS, and use the second DM-RS only for the last three OFDM symbols. This is good for a realtime environment since only the first three symbols in each subframe has to be buffered and all other OFDM symbols can be decoded right away. The drawback of this method is that the channel has to be time invariant for almost the entire subframe time $T_{subframe}$.

2.5.4 DM-RS combinations

If a system isn't time critical and an even better channel estimate is needed, the two DM-RSs can be combined in order to create a common channel estimate for the entire subframe. This can be done in many different ways, and depending on the intended use of the system, this can be beneficial, however often costly. The computational cost will be higher due to the fact that an individual channel estimate is calculated for each OFDM symbol instead for just the DM-RS. The other large cost is the delay caused by the symbols having to be buffered up to 10 T_{OFDM} before decoding can proceed.

2.5.5 System Model

A basic model of a communication system without iterative channel estimation is shown in figure 2.10. However, in a real system many more steps are conducted in between the main tasks shown in the figure.

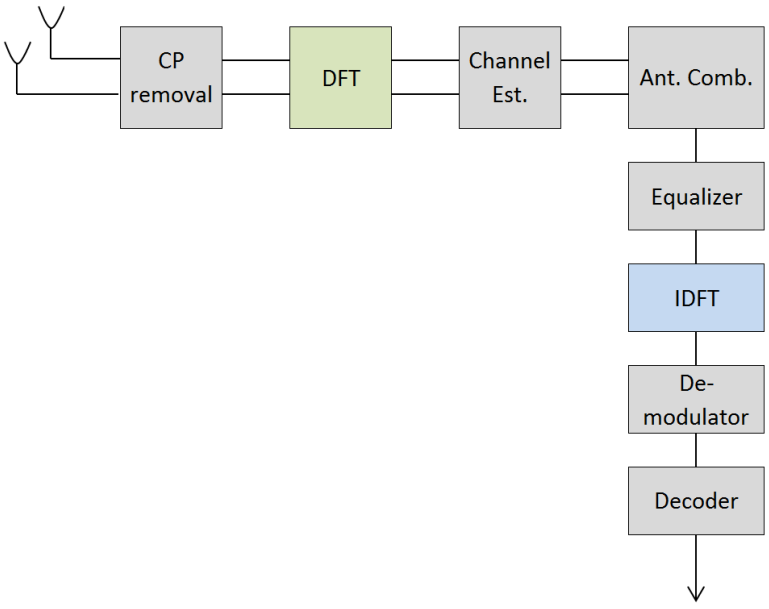


Figure 2.10: Illustration of the basic receiver structure

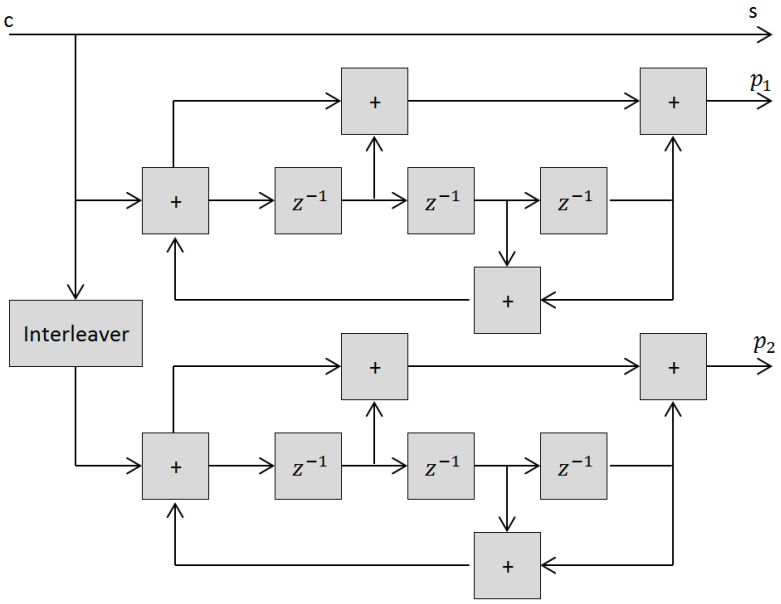


Figure 2.11: Illustration of the systematic rate 1/3 turbo encoder used in LTE

2.5.6 Turbo coding and rate matching

The idea behind coding is to send the data together with parity bits, in the receiver use an algorithm to check if the decoded data is correct, and if possible correct the errors in the data. The more parity bits sent, the bigger is the possibility to be able to correct errors. The drawback is of course that parity bits require spectrum, just as the data bits, reducing the actual amount of data bits that can be sent in one resource block.

The encoder used in LTE UL is a systematic turbo encoder. An illustration of the encoder can be seen in figure 2.11. The encoder is started in the all-zero state and forced to the all-zero state in the end as well. This is however not shown in the illustration. The output s is the same as the input c. The other two outputs, p_1 and p_2, are parity bits constructed by two eight-state convolutional encoders. Since one input bit produces three output bits the coderate of the encoder is 1/3.

A fixed code rate is not a good property for a system if high throughput is requested. If the channel conditions are good, a higher coderate would, for a fixed bitrate, increase the throughput of the system. However, if the coderate is too high, not enough parity bits will be present in the decoder to correct for possible bit errors. An illustration of how the coderate and modulation depend on the channel conditions and the expected throughput can be found in [11], figure 80.

To be able to change the code rate, ratematching is performed after the turbo encoding. Rate matching can simply be described as choosing which of the parity bits to send over the channel, and which to discard. All the data bits are always transmitted, giving the actual transmitted coderate a span of [1/3 1]. To do the rate matching, three interleavers and a circular buffer is used. The exact algorithm is described in [6], section 5.1.4.1.

The decoding of the data works in a similar way as the encoding. A soft demapper transforms the receiver data into soft information, usually bit LLRs. In order for the turbo decoder to work properly, the received data has to be ratedematched. For all positions where the parity bits were discarded in the ratematcher, a bit is inserted with bit probability 1/2. This ensures that the coderate now is 1/3, exactly as the output from the turbo encoder.

In the turbo decoder, two separate decoders perform soft-in soft-out decoding of the ratedematched data. One with the bit LLRs representing c and p_1, the other one with the bit LLRs representing an interleaved version of c and p_2. If both decoders have estimated the same codeword as the most probable, the decoding is assumed done. However, if the decoders have estimated different codewords, they exchange extrinsic information, that is the difference between the a priori and a posteriori information, and perform another decoding iteration.

Worth noticing is that neither the internal interleaver in the turbo encoder and decoder, nor the interleavers in the rate matcher, are the channel interleaver discussed in section 3.1.3 and section 3.2.

3

Channel estimation

3.1 Iterative channel estimation

The idea behind an iterative channel estimation is to feed back the decoded data to the channel estimator and use it as a priori information for a new channel estimation of *the same* channel. A basic illustration of an iterative system can be seen in figure 3.1. As compared to the system in figure 2.10 the iterative system has a feedback loop to the channel estimator.

The decoded data is used in different ways depending on if it is exported to the MAC-layer or if it is used for an iterative channel estimate. The MAC-layer only has interest in a bit stream containing the data bits transmitted over the channel. However, the iterative channel estimator requires the exchange of soft information from the decoder. Usually this soft information is presented as bit LLRs. Since the channel estimator should estimate the channel for all transmitted bits, not only the data bits, the soft information should be containing bit LLRs for both the data bits and the parity bits, as discussed in section 2.5.6. The soft bits are modulated to soft symbols using the same modulation alphabet as the UE used when it transmitted the data, see section 3.1.5.

The soft symbols can be used in two different ways. The intuitive way would be to use the soft symbols as they are. However, since the modulation alphabet is known and fixed, it is impossible that the UE sent a symbol in between two points in the modulation alphabet. Figure 3.2 illustrates the 16-QAM modulation alphabet (rings) and a re-modulated soft symbol (x). Since no symbol is placed at the x, it's impossible for the UE to send symbol x and the modulator could take hard decisions based on the soft values, i.e. move the x to the closest ring. Which method that has better performance will be analysed in section 3.1.5.

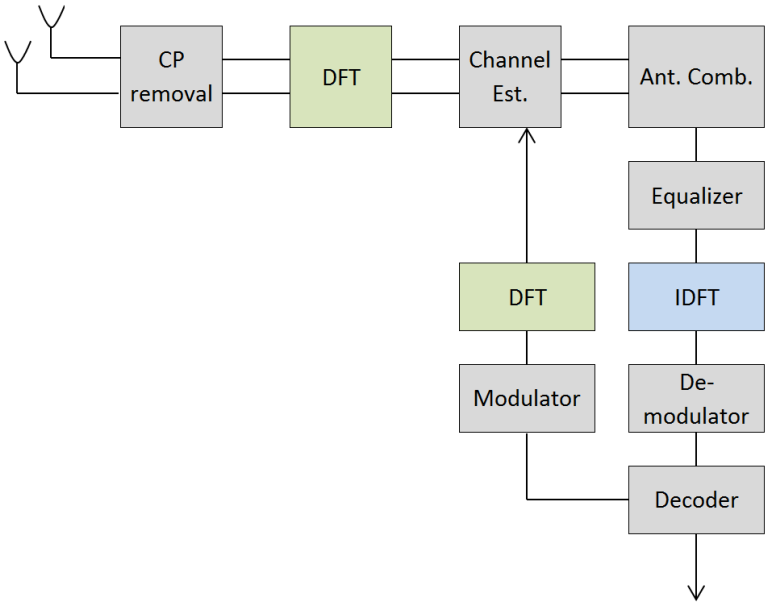


Figure 3.1: Illustration of the basic receiver structure with a feedback loop

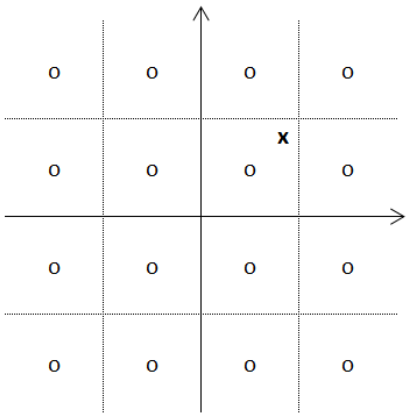


Figure 3.2: 16-QAM constellation (rings) and a soft symbol (x)

3.1.1 The re-estimation algorithm

The algorithm developed for the channel re-estimator is based on a weighted least-square estimate. It is described below as well as in [18] and [19].

For a real system, that is suitable for implementation, the system becomes far more complex than the one showed in figure 3.1. An illustration of the iterative system developed for this thesis is found in figure 3.3. A description of the main parts of the algorithm is described below.

3.1.2 Rate matching and reencoding

As discussed in section 2.5.6 the turbo decoder works at a coderate of 1/3. In order to recreate the signal, the code needs to be reencoded and then ratematched to the same rate as the transmitted signal. In order to simplify this project, it is assumed that this functionality already exists and the regenerated data henceforth is assumed to be reencoded and ratematched.

3.1.3 Burst errors and channel interleaving

Channel interleaving is an important part of the wireless communication system. In order for the reestimation algorithm to perform better than the initial channel estimate, and not only reproduce the errors of the initial estimate, the decoder has to correct for some of the bit errors. As shown in [9] errors in a fading channel tend to occur in bursts. This happends when the channel is in a deep fade. Since most error correction codes are designed to work in a AWGN environment, where the errors are spread equally over the data[9], a burst error may be uncorrectable for the decoder. If the errors are spread over a larger part of the signal they will appear as random, and the codes designed for AWGN error correction may be able correct it. In order to spread the errors over the entire signal, the data is interleaved by a channel interleaver before it is transmitted. In the receiver the data is deinterleaved before decoding. A description of the block interleaver used for channel interleaving in LTE and some hardware constraints can be found in section 3.2. However, this channel interleaver is not the same interleaver as described in the turbo encoder and decoder in section 2.5.6.

Here follows an easy example of burst errors in a data string and the same data string interleaved using a simple 6 column block interleaver. In the original text corrupted with the burst error there is no way of determining the fourth word. In the deinterleaved text however, it is possible to understand the sentence even though the text is corrupted. The 6 column block interleaver is illustrated in figure 3.5.

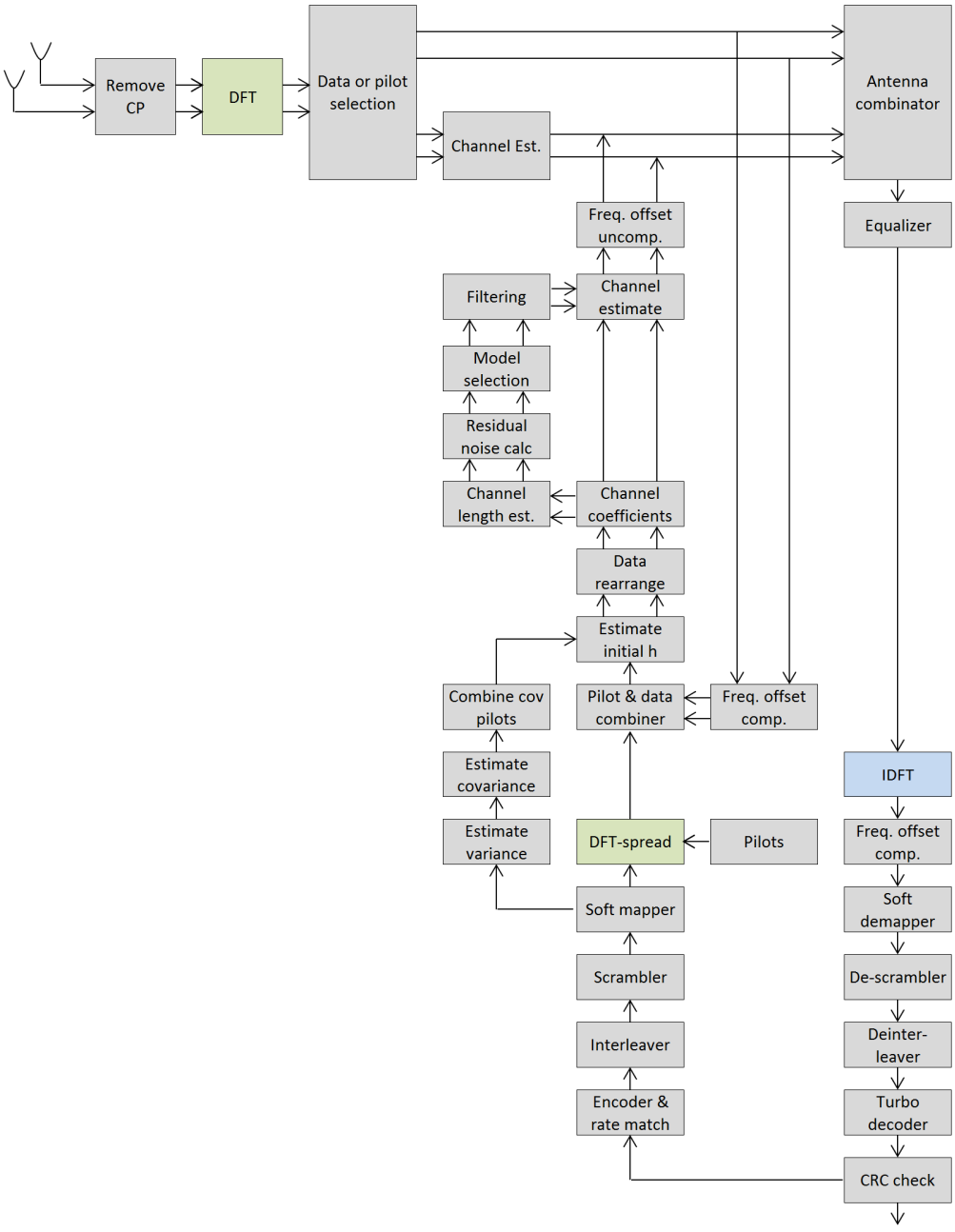


Figure 3.3: An overview of the system developed for this thesis. The block "channel estimator" consists of several sub blocks, but since the exact way of how this is working is an Ericsson company secret it is in this system seen as a black box.

`Test sentence with burst error`	Original text
`Test sentence with error`	Original text with burst error
`Tee en bestwurteirr ntsoschtr`	Interleaved text
`Tee en bestwurteir oschtr`	Interleaved text with burst error
`Test sente ce wi h bur t er or`	Deinterleaved text with burst error

There are two major drawbacks of the block interleaver. One is the memory constraints discussed in section 3.2. The other one is the time constraints. For the interleaver to work according to the LTE specifications, the entire subframe must be exported from the turbo decoder to fill the interleaving buffer. This creates an unwanted delay in a realtime system.

3.1.4 Scrambling

In many parts of the communication system, the data is considered to be independent and identically distributed. This is however not always the case since the data to be sent may be dependent. For example the clock error estimation and the automatic gain control will have problems to lock on the signal if the bits sent is only ones or only zeros. The scrambler prevents this from happening by changing the sign of some of the bits in the input sequence. In the LTE standard, the sequence used for scrambling is specific for each UE and described in [6].

Another advantage of using a scrambler is that the energy in the signal is more evenly spread over the spectrum, reducing the effects of energy peaks in narrow parts of the spectrum. This since the code structure is destroyed. This also makes it harder for another observer to glean the signal and be able to decode it, working as a primitive way of ciphering. Of course, since the scrambling sequences are both fixed and known, there is no guarantee that the signal won't be gleaned and therefore other types of ciphering is used in LTE as well[20].

3.1.5 Soft mapper

The soft mapper converts bit log-likelihood ratios to symbols for any specified symbol alphabet. To calculate the position of the symbol in the complex plane, the probability of a bit being 1 needs to be calculated. Simplified it could be explained as the higher the probability, the closer the symbol should be to the corresponding fixed position in the used symbol alphabet.

Mathematically the regenerated symbol at sample n can be described by

$$\hat{a}_n = E\{a_n \quad | \quad [L_{n,1}, L_{n,2}, \ldots, L_{n,qm}]\} \tag{3.1}$$

where a_n is the transmitted symbol, $L_{n,j}$ represents the bit LLRs for sample n and qm is the number of bits per symbol. For each bit in the symbol the bit LLR $L_{n,j}$ is specified as

$$L_{n,j} = ln\left(\frac{p(c_{n,j}=0)}{1-p(c_{n,j}=0)}\right) \tag{3.2}$$

where $p(c_{n,j}=0)$ is the probability of bit j being 0. Thus the regenerated symbol is expressed as

$$\hat{a}_n = \sum_{i=1}^{2^{qm}} c_i \prod_{j=1}^{qm} L_{n,j} \tag{3.3}$$

where c_i is the bit sequence $[b_1, b_2, \ldots, b_{qm}]^T$ represented by the i:th symbol in the symbol alphabet. [21]

As mentioned earlier in section 2.1.1 the three different modulation alphabets used in LTE is QPSK, 16QAM and 64QAM. For these alphabets, $\hat{a}$ can be analytically specified by linear combinations of the bit probabilities $p_{0,j}$ and $p_{1,j}$. For readability, the symbol index n is dropped.

$$\begin{aligned} p_{0,j} &= \tfrac{exp(L_j)}{1+exp(L_j)} \\ p_{1,j} &= 1 - p_{0,j} \end{aligned} \tag{3.4}$$

$$\begin{aligned}
&\text{QPSK:} && \begin{cases} \hat{a} = \frac{(p_{0,1}-p_{1,1})+i(p_{0,2}-p_{1,2})}{\sqrt{2}} \end{cases} \\
&\text{16QAM:} && \begin{cases} tmp_1 = p_{0,1} - p_{1,1} \quad tmp_2 = p_{0,2} - p_{1,2} \\ tmp_3 = 3 \cdot p_{1,3} + p_{0,3} \quad tmp_4 = 3 \cdot p_{1,4} + p_{0,4} \\ \hat{a} = \frac{(tmp_1 \cdot tmp_3)+i(tmp_2 \cdot tmp_4)}{\sqrt{10}} \end{cases} \\
&\text{64QAM:} && \begin{cases} tmp_1 = p_{0,1} - p_{1,1} \quad tmp_2 = p_{0,2} - p_{1,2} \\ tmp_3 = p_{1,3} - p_{0,3} \quad tmp_4 = p_{1,4} - p_{0,4} \\ tmp_5 = 3 \cdot p_{1,5} + p_{0,5} \quad tmp_6 = 3 \cdot p_{1,6} + p_{0,6} \\ \hat{a} = \frac{(tmp_1 \cdot (tmp_3 \cdot tmp_5 + 4)) + i(tmp_2 \cdot (tmp_4 \cdot tmp_6 + 4))}{\sqrt{42}} \end{cases}
\end{aligned} \tag{3.5}$$

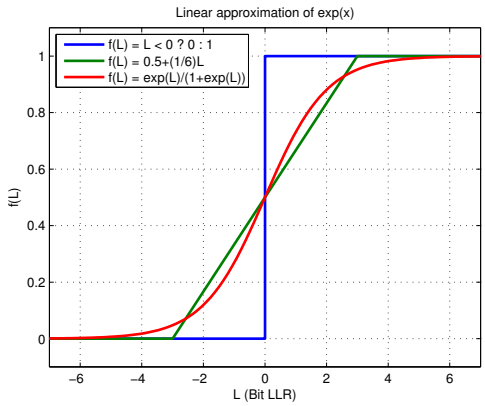


Figure 3.4: Illustration of the different approximations of the exponential function and the correct function.

In order to convert the log-likelihood ratio to the bit probability in equation 3.4 the exponential function is used. This operation is a quite costly operation in the DSP. Especially since it needs to be run once for every reencoded bit. Therefore three different versions of the soft mapper was tested in the LTE simulator. The first one fixing the likelihood to either 0 or 1, the second one using a linear approximation of the exponential function and the third one using the exponential function. The function $exp(L)/(1 + exp(L))$ was studied, and the three versions are illustrated in figure 3.4.

From simulations using the channels epa5, etu70 and etu300[22] it was concluded that for low SNR the soft mapper using hard bits performed a little bit worse than the linear and ideal soft mapper. This is in line with the results in [3]. For higher modulation and/or higher SNR the three mappers performed equally. This, the fact that the difference between the three versions is almost neglectable,the increased computational power required and for the sake of simplicity, motivated the choice of using the hard bits version of the mapper for this initial test setup.

3.1.6 DFT-spread

After the soft symbols has been recreated, they are DFT-spread using a hardware implementation of the FFT as discussed in section 2.2.

$$[\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_N] = FFT\{[\hat{a}_1, \hat{a}_2, \cdots, \hat{a}_N]\} \tag{3.6}$$

3.1.7 Variance and covariance estimation

In the simulations of the iterative channel estimator[19] the variance and covariance of the regenerated data was estimated over all resource blocks allocated to the user according to

$$\sigma_{\hat{x}}^2 = \frac{1}{|\mathbb{D}|N_{SC}} \sum_{\mathbb{D}} \sum_{k=1}^{N_{SC}} |\hat{x}_k|^2 \tag{3.7}$$

$$\sigma_{x\hat{x}} = \Re \left(\frac{\sum_{\mathbb{D}} \sum_{N_{SC}} \sum_{\mathbb{A}} y\hat{h}^* \hat{x}}{\sum_{\mathbb{D}} \sum_{N_{SC}} \sum_{\mathbb{A}} |\hat{h}|^2} \right) \tag{3.8}$$

where $\mathbb{D}$ is all datasymbols in the subframe, N_{SC} is all scheduled subcarriers for the user and $\mathbb{A}$ is all receive antennas. In the hardware however, because of the limited amount of available fast memory it is not feasible to sum over such large data quantities. The simulations showed that a simplification of the estimates could be done by calculating the variance over only the current data symbol and estimate the covariance from the variance

$$\sigma_{\hat{x},m}^2 = \frac{1}{N_{SC}} \sum_{k=1}^{N_{SC}} |\hat{x}_k|^2 \tag{3.9}$$

$$\sigma_{x\hat{x},m} \approx \sqrt{\sigma_{\hat{x},m}^2} \tag{3.10}$$

where m is the symbol index. Equation 3.10 is motivated by Cauchy–Schwarz inequality $|\sigma_{x\hat{x}}| \leq \sigma_x \sigma_{\hat{x}}$ where $\sigma_x = 1$ since the data symbols can be modelled as a zero-mean Gaussian random variable.

3.1.8 Weighted least-square channel estimate

The weighted least square channel estimate is a least square channel estimate where every point in the estimate also has a weight associated with it. The weight is calculated proportionally to the received power in the signal. This to be able to suppress the very high peaks in the least square estimate where otherwise the noise is amplified.

The least square channel estimate is calculated as

$$\hat{h}_{m,k} = \frac{y_{m,k}}{c_m \hat{x}_{m,k}} \tag{3.11}$$

where c is defined

$$c_m = \frac{\sigma_{x\hat{x},m}}{\sigma_{\hat{x},m}^2} \tag{3.12}$$

However, if the simplification in equation 3.10 is used, c_m is simplified to

$$c_m = \frac{1}{\sqrt{\sigma_{\hat{x},m}^2}} \tag{3.13}$$

The weights are calculated as the inverse of the variance of the estimated error[19].

$$w_{m,k} = \frac{c_m \hat{x}_{m,k}^2}{\left(\frac{\hat{h}_{m,k}^2}{\sigma_n^2}\left(1 - \frac{\sigma_{x\hat{x},m}^2}{\sigma_{\hat{x},m}^2}\right) + 1\right)} \tag{3.14}$$

As for c_m, $w_{m,k}$ is simplified if equation 3.10 is used. Since one term in the denominator in equation 3.14 becomes zero:

$$1 - \frac{\sigma_{x\hat{x},m}^2}{\sigma_{\hat{x},m}^2} = 1 - \frac{\sqrt{\sigma_{\hat{x},m}^2}^2}{\sigma_{\hat{x},m}^2} = 0 \tag{3.15}$$

$w_{m,k}$ is in the end calculated only as

$$w_{m,k} = c_m \hat{x}_{m,k}^2 \tag{3.16}$$

This method results in a quite noisy channel estimate, where no distinct trends in the channel estimate can be observed.

3.1.9 Polynomial approximation

In order to be able to model the channel closer to the real channel, the noisy channel estimate is passed to a model fitting process. In the model fitting process, every subcarrier is modelled zeroth or first order polynomial. I.e. as either a constant or as a line. In the simulations a second order polynomial was evaluated as well, this however lead to even larger complexity.

In order for the model fitting to be efficient, any frequency offsets must be compensated for. Frequency shifts occur for different reasons, but two common reasons are difference in speed of the transmitting UE and the basestation and the difference in the oscillators in the UE and the basestation. Therefor the channel estimate is rotated by a factor f_{offset}

$$\hat{h^c}_{m,k} = \hat{h}_{m,k} e^{i2\pi \Delta t m f_{\text{offset}}} \tag{3.17}$$

where Δt here is the time between symbols. The frequency offset is estimated in an earlier stage of the first channel estimation process.

The polynomial approximation is a minimisation problem. For the zeroth order polynomial, the constant is calculated for each subcarrier by

$$\min_{b_{0,k}} \sum_{m=0}^{N_s} \left| \hat{h^c}_{m,k} - b_{0,k} \right|^2 w_{m,k} \tag{3.18}$$

where N_s is the number of symbols in the subframe, i.e. 14 or 12 depending on cyclic prefix.

The first order polynomial is calculated for each subcarrier by minimising

$$\min_{b_{1,k},b_{2,k}} \sum_{m=0}^{N_s} \left| \hat{h^c}_{m,k} - (b_{1,k} + s_m b_{2,k}) \right|^2 w_{m,k} \tag{3.19}$$

where s is a vector of length N_s with uniformly spaced values in the range $[-1\ 1]$. The use of s instead of the symbol number gives the channel coefficients better numerical properties in the DSPs as well as less dependence on the type of cyclic prefix used. The models are explained in more detail in [19].

Both the zeroth and first order polynomial can be solved analytically. The zeroth order polynomial coefficients are given by

$$b_{0,k} = \frac{\sum_{m=0}^{N_s} {h^c}_{m,k} w_{m,k}}{\sum_{m=0}^{N_s} w_{m,k}} \tag{3.20}$$

and the first order coefficients by

$$\begin{bmatrix} b_{1,k} \\ b_{2,k} \end{bmatrix} = \begin{bmatrix} \sum_{m=0}^{N_s} w_{m,k} & \sum_{m=0}^{N_s} s_m w_{m,k} \\ \sum_{m=0}^{N_s} s_m w_{m,k} & \sum_{m=0}^{N_s} s_m^2 w_{m,k} \end{bmatrix}^{-1} \begin{bmatrix} \sum_{m=0}^{N_s} \hat{h}^c_{m,k} w_{m,k} \\ \sum_{m=0}^{N_s} s_m \hat{h}^c_{m,k} w_{m,k} \end{bmatrix} \tag{3.21}$$

where the matrix inversion is calculated as

$$\begin{bmatrix} a & b \\ b & c \end{bmatrix}^{-1} = \frac{1}{ac - b^2} \begin{bmatrix} c & -b \\ -b & a \end{bmatrix} \tag{3.22}$$

3.1.10 Channel length estimation

The channel length is estimated using the same heuristic method as the conventional channel estimator. The method cannot be described in this thesis in detail due to company regulations. However, the result from this method is the optimal channel length in DCT (discrete cosine transform) domain, L, and the residual noise variance, $\sigma^2_{n,LS}$. The noise variance is a measurement of the variance of the signal outside the optimal channel length and can be seen as a simple measurement of the noise affecting the channel estimate.

3.1.11 Model selection

Which model to select, constant or linear polynomial approximation depends on the channel. A higher order model fits a faster varying channel better, but is more affected by the noise. If the channel is slow or very noisy the constant model is a more appropriate. To select a fitting model, a switching variable Θ is calculated as

$$\Theta = \sum_{k=0}^{N_{SC}-1} \frac{|b_{2,k}|^2}{\sigma^2_{n,LS}} \tag{3.23}$$

where $b_{2,k}$ is the slope coefficient for the linear approximation, a measurement on how fast the channel is varying. The model is then chosen as linear if $\Theta >$ Switching value and else as constant. The switching value is determined from simulations and is dependent on the scaling of the input samples. Simulated results for when to use which model can be read in [19] where the second order polynomial is analysed as well.

3.1.12 Filtering of the channel estimate

In order to reduce the noise in the channel estimate further, the channel estimate is filtered over frequencies as well. The polynomial model approximation can be seen as a filtering or weighted average over time. The channel estimate is however still affected by noise in the frequency direction. Therefore the selected channel model is transformed to DCT-domain filtered and transformed back using as IDCT.

$$[d_{x,0}\ d_{x,1} \cdots d_{x,N_{SC}-1}]^T = DCT\left([b_{x,0}\ b_{x,1} \cdots b_{x,N_{SC}-1}]^T\right) \tag{3.24}$$

$$d^{filt}_{x,y} = \begin{cases} d_{x,y}, & y < L \\ 0, & \text{else} \end{cases} \tag{3.25}$$

$$\left[b^{filt}_{x,0}\ b^{filt}_{x,1} \cdots b^{filt}_{x,N_{SC}-1}\right]^T = IDCT\left(\left[d^{filt}_{x,0}\ d^{filt}_{x,1} \cdots d^{filt}_{x,N_{SC}-1}\right]^T\right) \tag{3.26}$$

Of course, for computational efficiency, if the model selection indicates that the constant model is to be used, only the $b_{0,k}$ coefficients are filtered. The same goes for the linear model, then the $b_{1,k}$ and $b_{2,k}$ coefficients are filtered.

3.1.13 Final channel estimate

The final channel estimate is constructed using the filtered polynomial coefficients. For the constant model as

$$\hat{h}_{m,k}^{it,c} = b_{0,k}^{filt} \tag{3.27}$$

and for the linear model as

$$\hat{h}_{m,k}^{it,c} = b_{1,k}^{filt} + s_m b_{2,k}^{filt} \tag{3.28}$$

However, since the channel estimate was frequency compensated in equation 3.17, the final channel estimate has to be uncompensated according to

$$\hat{h}_{m,k}^{it} = \hat{h}_{m,k}^{it,c} e^{-i2\pi\Delta t m f_{\text{offset}}} \tag{3.29}$$

3.2 Implementation aspects

Since the algorithm is supposed to be evaluated for possible use in a real system, and not only as a theoretical model, there are some restrains in memory, complexity and execution speed that needs to be assessed.

The hardware intended for use is a multicore DSP platform. A general design for this type of platform is as following: Each DSP core has access to a small amount of very fast local memory where the program is stored as well as the stack and the heap. Usually there are also one or more larger pools of common memory from where larger chunks of data can be accessed. These common memory pools are however often very slow compared to the local memory. Since these pools cannot be physically close to all DSP cores in a large system, especially the time-to-first-access is large. Because of this, it is desirable to load all parameters and data used in each function to the local memory, run the function in the local memory, and then export the data. This way the data can be written in batches to the slower common memory and thereby spreading the time-to-first-access on a larger amount of data and decrease the total time used.

T	e	s	t	_	s
e	n	t	e	n	c
e	_	w	i	t	h
_	b	u	r	s	t
_	e	r	r	o	r

Figure 3.5: A 6 column block interleaver

3.2.1 Interleaving

In order to reencode the data, it has to be interleaved. Interleaving of the data is in theory a very simple procedure. According to the LTE specifications[23] the interleaver is in its simplest case with no HARQ-ACK- or RI-bits a buffer with N_s = 'Number of symbols per subframe' columns. The data is grouped in groups of 2, 4 or 6 bit LLRs depending on the modulation alphabet used, then written into the buffer row by row. When all bits are in the buffer, the output data is read column by column instead. This is illustrated in figure 3.5 where the example from section 3.1.3 is reused (without burst errors) in a 6 column block interleaver.

`Test sentence with burst error`	Original text
`Tee en bestwurteirr ntsoschtr`	Interleaved text

The size of this buffer would have to be of size

$$N_{\text{buffer}} = N_{PRB} \cdot N_{SC} \cdot N_D \cdot qm \tag{3.30}$$

where N_{PRB} is the number of resource blocks allocated for the user, N_{SC} the number of subcarriers per resource block, N_D the number of data symbols per resource block and qm the number of bits per symbol. In total for a 20MHz cell with 96 PRBs scheduled for one UE:

$$N_{\text{buffer}} = 96 \cdot 12 \cdot 12 \cdot 6 = 82944 \text{ data points} \tag{3.31}$$

When also taking into account that the data has to be read from somewhere before it is put into the buffer, resulting in another factor of two, and the fact that each bit LLR has to be represented by at least a couple of bits, it is easily understood that some kind of memory management is needed to perform the interleaving.

Therefore an algorithm for memory management for the interleaver was developed to fill the local memory with the maximum amount of data, from different positions in the common memory in order to be able to perform a partial interleaving. By then looping through the memory the interleaving result was the same as for the theoretical model.

4

Results

4.1 Simulation results

From the simulations presented in figure 4.1 and figure 4.3, it is shown that the iterative channel estimation algorithm performs better than the conventional pilot aided estimation algorithm. The average SNR gain for one iteration is about 0.5 to 0.6 dB depending on the modulation. For two iterations the SNR gain is slightly higher, about 0.15 dB higher than for one iteration. As a reference the conventional receiver without any iteration loop is used.

For very good channel conditions the SNR gain is lower. This because almost all bits are decoded correctly already in the first iteration, therefor there is not that many, if any, bits that needs to be corrected. In figure 4.1 this is shown for SNR > 14. The same is true for bad channel conditions due to the fact that there are so many errors in the decoded bits that the turbo-decoder cannot correct for them.

Excess tap delay [ns]	Relative power [dB]
0	-1.0
50	-1.0
120	-1.0
200	0.0
230	0.0
500	0.0
1600	-3.0
2300	-5.0
5000	-7.0

Table 4.1: Tap delay and relative power for the ETU channel model. As specified in [22]

Many different scenarios have been simulated in the project. Presented below are the results using the ETU70 channel. ETU stands for extended typical urban model. It is a nine tap channel model with a maximum tap delay of 5000 ns, as shown in table 4.1. The number 70 represents the maximum doppler frequency, 70 Hz[22].

Figure 4.1 shows simulation results for one of many test cases run. The throughput is normalised due to regulations at Ericsson. However, the gain is clearly visible without the use of absolute values. Table 4.2 describes the simulation parameters. The region with the best SNR gain is the region around 9 - 14 dB SNR, and this is the region where the system will work if the scheduler is functioning as intended. From the figure it is also clear that the gain from the first iteration step is much larger than the gain from the second iteration. A plot of the averaged SNR gain is present in figure 4.3 This is described in more detail in [19].

In figure 4.2 no iteration and one iteration for all three modulation formats, QPSK, 16-QAM and 64-QAM, are plotted in the same figure. The SNR gain is harder to distinguish in this figure, but the throughput for the different modulation formats is shown clearly. One interesting part of this figure is the results at around 8 dB SNR, where 64-QAM actually outperforms 16-QAM. This is probably due to the HARQ retransmissions since the wavelike shape of the plot dissapears when HARQ is turned off. In table 4.3 the simulation parameters for figure 4.2 are listed.

Figure 4.3 illustrates the SNR gain for QPSK, 16-QAM and 64-QAM respectivly, using the ETU70 channel for one and two iterations. The gain is an averaged value over the region of normalised throughput between 0.7 and 0.9. As reference the zero iteration SNR is used.

Modulation	16-QAM
SNR range	[-5 20]
Channel	ETU 70
Channel bandwidth	5 MHz
Scheduled RB per user	25
Frames simulated	1 000
HARQ retransmissions	ON (max 4)
OFDM symbols per subframe	12
Code rate	3/4
Uncoded bits	10680
CRC size	24 bits
Number of codeblocks	2
Size of codeblock	16 140 bits
Bits per subframe	14 400
Symbols per subframe	3 600
Antennas (TxR)	1x2 (SIMO)

Table 4.2: Simulation parameters used to simulate the results presented in figure 4.1

4.2 Implementation results

The implementation on the DSP platform is evaluated not by the gain in throughput but rather by the computational cost and timing constraints. In order to have a reference value, all measurements are compared against the average computational cost of the non-iterative decoding of the data. For the signal processing cost (cycles) the scale ranges from very intensive to very low, where very intensive indicates that the process almost completely blocks a DSP core, and very low indicates that the cost is almost neglectable. The memory constraints scale ranges from very high to very low, where very high indicates that a memory handling outside the fast local memory around the DSP core is necessary (see section 3.2) and very low that the memory cost is almost neglectable.

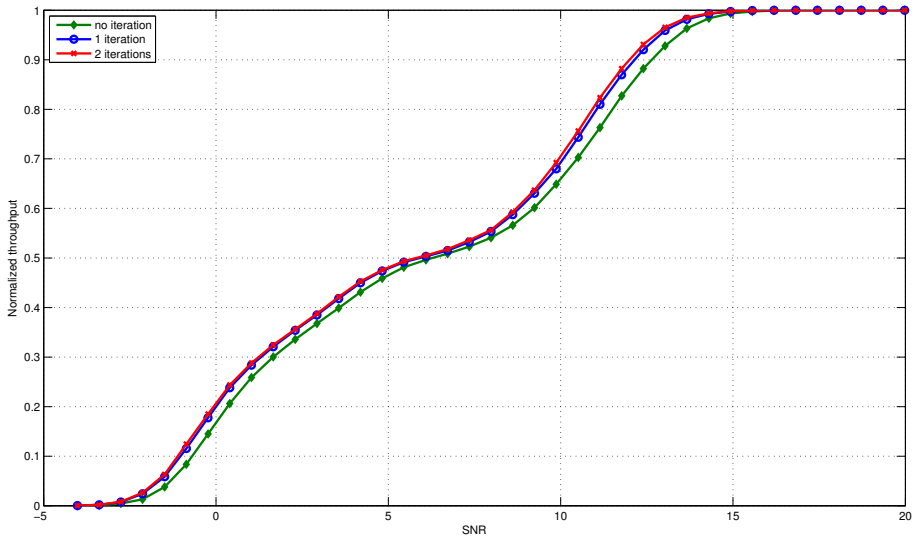


Figure 4.1: Normalised throughput for the PUSCH channel versus SNR. Simulation results with and without an iteration loop. The simulation is based on the channel ETU70[22], using 16-QAM modulation, 25 scheduled resourceblocks per user and 1000 simulated frames, as specified in table 4.2.

In table 4.4 the computational cost of the individual parts of the reestimation process is presented. The presented data is the maximum cost for numerous simulations. i.e. the worst case scenario.

Worth noticing is the fact that code optimisation has not been a priority in this project. Since this is a pilot project and not intended for anything but a pre study, one has to take into consideration that an implementation intended for product deployment can be further optimised in terms of code efficiency and usage of the multicore architecture.

A possible delay is introduced in the receiver chain due to the iterative channel estimation. If the cyclic redundancy check, CRC, is ok no extra iteration is needed, and no delay is added to the receiver chain. However, if the CRC is not ok, then all OFDM-symbols in the frame have to be received and decoded before the channel estimate could be regenerated. This introduces a timing delay of up to $13T_{OFDM}$ plus the actual computational time for the iterative algorithm. The first part similar to the delay discussed in section 2.5.4.

Modulation	QPSK	16-QAM	64-QAM
SNR range	[-12 8]	[-4 20]	[0 26]
Channel	ETU 70	ETU 70	ETU 70
Channel bandwidth	5 MHz	5 MHz	5 MHz
Scheduled RB per user	25	25	25
Frames simulated	1000	1000	1000
HARQ retransmissions	ON (max 4)	ON (max 4)	ON (max 4)
OFDM symbols per subframe	12	12	12
Code rate	1/3	3/4	5/6
Uncoded bits	2216	10680	18 336
CRC size	24 bits	24 bits	24 bits
Number of codeblocks	1	2	3
Size of codeblock	6 732 bits	16 140 bits	18 444 bits
Bits per subframe	7 200	14 400	21 600
Symbols per subframe	3 600	3 600	3 600
Antennas (TxR)	1x2 (SIMO)	1x2 (SIMO)	1x2 (SIMO)

Table 4.3: Simulation parameters used to simulate the results presented in figure 4.2

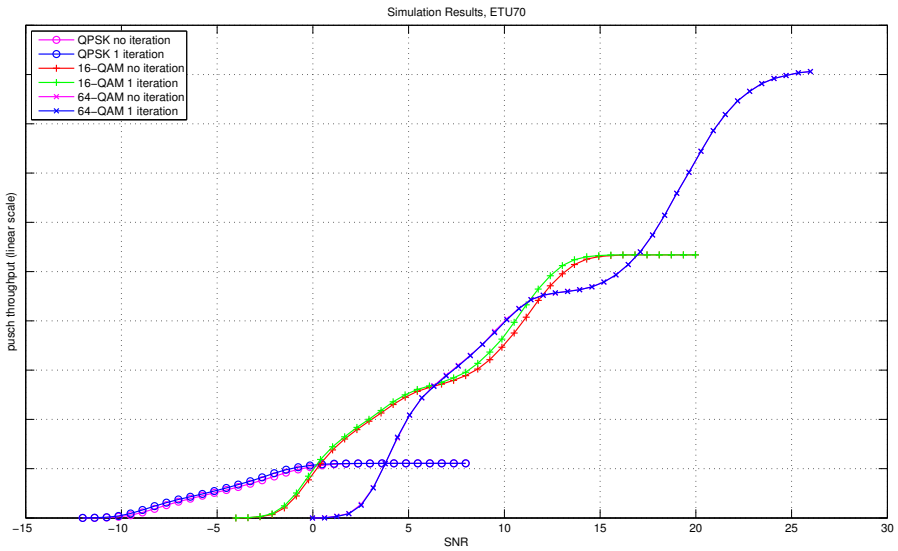


Figure 4.2: Normalised throughput for the PUSCH channel versus SNR. Simulation results with and without an iteration loop for all modulation schemes. The simulation is based on the channel ETU70[22], 25 scheduled resourceblocks per user and 1000 simulated frames, as specified in table 4.3. The scaling on the Y-axis is linear, but the absolute values intentionally removed.

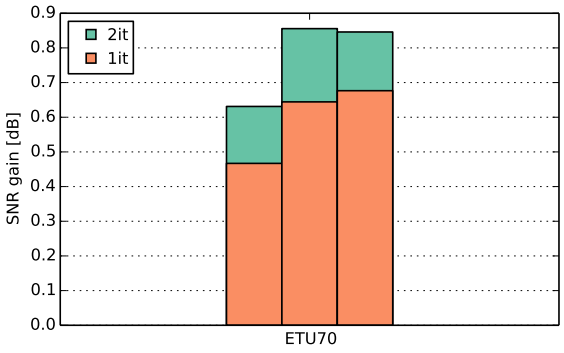


Figure 4.3: Simulation results showing the average gain in SNR for a constant throughput for channel ETU70 and the three different modulation schemes, QPSK, 16-QAM and 64-QAM respectivly.

Function	Cost (cycles)	Cost (memory)	Comments
Interleaver	intensive	very high	1 subframe, 25RB, 16-QAM
Scrambler	low	low	1 subframe, 25RB, 16-QAM
Soft mapper	very intensive	low	1 subframe, 25RB, 16-QAM
Variance estimation	low	moderate	1 subframe, 25RB
Covariance estimation	very low	very low	1 subframe
DFT-spread	low	N/A	1 subframe, 25RB
Initial LS channel estimate	very intensive	moderate	1 subframe, 25RB, 1 antenna
Data rearrangement	intensive	very high	1 subframe, 25RB, 1 antenna
Channel coefficient calculation	very intensive	low	1 subframe, 25RB, 1 antenna
Model selection	low	moderate	1 subframe, 25RB, 1 antenna
Filtering	N/A		
Create channel estimate	N/A		

Table 4.4: Computational costs and memory requirements for individual functions of the channel reestimation algorithm

5

Discussion

The iterative channel estimation process is in theory a very good way of increasing the throughput of the communication system.

The practical implementation of the algorithm showed to be more complex than expected due to the lack of enough memory to process the entire subframe all at once. This gives raise to unavoidable memory handling processes that costs computational time without adding value.

As shown in section 4.2 the most computationally costly part of the reestimation algorithm is the soft mapper. This is partly due to how the data is unpacked from bit LLRs to symbols.

One of the things that could reduce the complexity of the implementation would be if the hardware platform would support floating point arithmetics instead of only fixed point arithmetics. The major benefit would be that the dynamic range would be increased. Another advantage would be the simplified development since mathematical models more easily could be transferred into functions without having to scale everything to fit into the desired dynamic range. A disadvantage of using floating point aritmetics is that processors that support floating point aritmetics generally are slower than fixed point processors, and require more fast local memory to store the same amount of data. The key reasons why the fixed point processors still are so popular might be the fact that fast local memory already is a bottleneck and that floating point processors have higher purchase cost.

The cost for the iterative channel estimation algorithm has shown to be high compared to the conventional algorithm. If it is too high or not, depends on the perspective in which it's analysed. If one is to compare the iterative algorithm side by side with the conventional algorithm it is clear that the algorithm is far too costly to be used in a product. However, if one instead includes the statistics for CRC failure, one can see that only about 10% of the frames are in error after the first channel estimate. This means that in general only every tenth frame need to be sent through the iterative channel estimator. Over time this means that the iterative channel estimation algorithm compared to the conventional algorithm only adds a cost of 10% of the worst case scenario where every frame needs to be fed through the iterative algorithm. This also means that if the scheduler works as intended, and the amount of frames that needs to be fed through the iterative algorithm can be reduced, the implementation of an iterative algorithm may be feasible.

In an even larger perspective, and when speculating about the future, the iterative channel estimate may even be a necessary part of the LTE system. The cost for retransmission of a frame is, compared to the cost of the iterative channel estimation process, very high. When retransmission is needed, the cost is not only placed on the basestation, but the scheduler, UPC and UE are affected as well. Of course the channel has to be estimated for the retransmission as well, so the total computational cost in the basestation will be double the cost for the conventional channel estimation algorithm. Apart from the computational cost, for every retransmission parts of the electromagnetical spectrum have to be reallocated, and the shortage of the electromagnetical spectrum is probably only getting worse in the future.

Given even higher bitrates in the future, which is safe to assume, judging from the past ten years of mobile communication development, the cost for retransmissions will increase. If some of the subframes in error can be corrected for by the use of an iterative channel estimation algorithm, the cost for the iterations might be considered as tolerable. However, today it is still a more theoretical than practically feasable algorithm.

Bibliography

[1] J. Andersson, Digital Transmission Engineering, 2nd Edition, IEEE Press, 2005.

[2] Telia, 4G fyller tre år, Telia–Smartare vardag, Accessed may 29, 2014.
URL http://blogg.telia.se/smartarevardag/2012/12/14/4g-fyller-tre-ar/

[3] F. Kadrija, M. Šimko, M. Rupp, Iterative Channel Estimation in LTE Systems, IEEE WSA 2013.

[4] E. Dahlman, S. Parkvall, J. Sköld, 4G LTE/LTE-Advanced for Mobile Broadband, Academic Press, 2011.

[5] F. Willems, Pulse amplitude modulation, Technische Universiteit Eindhoven, Department of Electrical Engineering, Accessed may 22, 2014.
URL http://www.sps.ele.tue.nl/members/F.M.J.Willems/TEACHING_files/5P350/chapter14.pdf

[6] 3rd Generation Partnership Project, 3GPP TS 36.211 V12.0.0, Tech. Rep. V12.0.0, Technical Specification Group Radio Access Network, E-UTRA, Valbonne, FRANCE (dec 2013).
URL http://www.3gpp.org/DynaReport/36211.htm

[7] U. Madhow, Fundamentals of Digital Communications, Cambridge University Press, New York, 2008.

[8] L. Råde, W. B, Mathematics Handbook, 5th Edition, Studentlitteratur, Lund, 2004.

[9] A. Goldsmith, Wireless Communications, Cambridge University Press, New York, 2005.

[10] M. Rumney, 3GPP LTE: Introducing Singel-Carrier FDMA, Agilent Measurement Journal (4) (2008) 18–27, 5989-7898EN.

[11] C. Johnson, Long Term Evolution IN BULLETS, 2nd Edition, CreateSpace Independent Publishing Platform, 2012.
URL `http://www.lte-bullets.com/`

[12] H. Myung, J. Lim, D. Goodman, Single Carrier FDMA for Uplink Wireless Transmission, IEEE Vehicular Technology Magazine 1 (3) (2006) 30–38.

[13] IEEE, 754-2008 - IEEE Standard for Floating-Point Arithmetic.
URL `http://ieeexplore.ieee.org/servlet/opac?punumber=4610933`

[14] H. Schmidt, IEEE 754 Converter, Accessed feb 24, 2014.
URL `http://www.h-schmidt.net/FloatConverter/`

[15] Y.-H. N. et al., Evolution of Reference Signals for LTE-Advanced Systems, IEEE Communications Magazine 50 (2) (2012) 132–138.

[16] E. Weisstein, Relatively Prime, MathWorld–A Wolfram Web Resource, Accessed mar 04, 2014.
URL `http://mathworld.wolfram.com/RelativelyPrime.html`

[17] M. Gul, S. L., X. M., Robust synchronization for OFDM employing Zadoff-Chu sequence, Annual Conference on Information Sciences and Systems (CISS) 46 (2012) 1–6, Princeton, NJ.

[18] E. Lindén, A. Markoff, J. Huang, A. Said, Iterative Channel Estimation Method for LTE UL, provisional patent US1 62/000295.

[19] E. Lindén, Iterative Channel Estimation in LTE Uplink, KTH, to be published.

[20] M. Solanki, S. Salehi, A. Esmailpour, LTE Security: Encryption Algorithm Enhancements, 2013 ASEE Northeast Section Conference.

[21] L. Fang, Q. Guo, D. Huang, S. Nordholm, A low cost soft mapper for turbo equalization with high order modulation, in: ISOCC 2012 - 2012 International SoC Design Conference.

[22] 3rd Generation Partnership Project, 3GPP TS 36.104 V12.3.0, Tech. Rep. V12.3.0, Technical Specification Group Radio Access Network, E-UTRA, Valbonne, FRANCE (mar 2014).
URL `http://www.3gpp.org/DynaReport/36104.htm`

[23] 3rd Generation Partnership Project, 3GPP TS 36.212 V12.0.0, Tech. Rep. V12.0.0, Technical Specification Group Radio Access Network, E-UTRA, Valbonne, FRANCE (dec 2013).
URL `http://www.3gpp.org/DynaReport/36212.htm`

A

Abbreviations & Acronyms

LTE	long term evolution
UL	uplink
W-CDMA	wideband code division multiple access
SNR	signal to noise ratio
ISI	intersymbol interference
LTI	linear and time invariant
RB	physical resource block
PRB	physical resource block
UE	user equipment
ETU70	extended typical urban model, 70 Hz
CRC	cyclic redundancy check
OFDM	orthogonal frequency division multiplexing
SC-FDMA	single carrier frequency division multiple access
PAPR	peak to average power ratio
UPC	user plane control
DSP	digital signal processor
MAC	media access control
AWGN	additive white gaussian noise

Table A.1: Abbreviations & Acronyms

DFT	discrete fourier transform
IDFT	inverse discrete fourier transform
FFT	fast fourier transform
IFFT	inverse fast fourier transform
DCT	discrete cosine transform
IDCT	inverse discrete cosine transform
LLR	log-likelihood ratio
CP	cyclic prefix
DM-RS	demodulation reference symbol (pilot)
HARQ	hybrid automatic repeat request
ACK	acknowledgement
RI	rank indication
PUSCH	physical uplink shared channel
A_C	fourier transform of channel autocorrelation
a_k	complex modulation symbol on the k:th subcarrier
a_n	transmitted symbol
$\hat{a}_n$	regenerated symbol
B	bandwidth
B_c	coherence bandwidth
$b_{x,k}$	polynomial coefficient
c_i	bit sequence represented by the ith symbol in the alphabet
$d_{x,y}$	polynomial coefficients in DCT domain
D	datasymbols in subframe
Δf	subcarrier spacing
Δt	time between OFDM-symbols
$\delta(t)$	dirac pulse
f_{offset}	frequency offset
$\hat{h}_{m,k}$	initial least square channel estimate
$\hat{h}^c_{m,k}$	frequency offset compensated channel estimate
$\hat{h}^{it}_{m,k}$	channel estimate after iterative algorithm
k	subcarrier index
L	channel length in DCT domain
L_n	bit LLR for sample n

Table A.2: Abbreviations & Acronyms

m	OFDM-symbol index
N	size of DFT/IDFT
N_{buffer}	size of buffer
N_c	number of subcarriers in OFDM-system
N_D	number of data symbols per PRB
N_{PRB}	number of physical resource blocks
N_s	number of symbols in a subframe
N_{SC}	number of subcarriers for current user
N_{ZC}	length of ZC sequence
n	sample index
p	bit probability
qm	bits/symbol
r	antenna index
s	vector of size N_s uniformly spaced $[-1\ 1]$
σ_x^2	variance
$\sigma_{\hat{x},x}$	covariance
$\sigma_{n,LS}^2$	residual noise variance
T_{CP}	cyclic prefix time
T_{CP-e}	extended cyclic prefix time
T_{OFDM}	OFDM-symbol time
T_{slot}	slot time
$T_{subframe}$	subframe time
T_u	symbol time
t	time index
τ	timeshift
Θ	switching parameter
u	seed
w	filter coefficient
$w_{m,k}$	weight assiciated with $\hat{h}_{m,k}$
$\hat{x}$	regenerated data

Table A.3: Abbreviations & Acronyms