[ELEC-H401] - Project Modulation and Coding

Simulating DVB-S2 satellite communication

Rogier De Nys, Warre De Winne

31/05/2023

Contents

1 Introduction

This report discusses the simulation of a DVB-S2 satellite communication channel. The first part discusses symbol mapping, noise on the channel and the effect of that noise on the performance of the channel. The second part talks about low-density parity check code and its performance on the noisy channel. In the third part the influence of CFO, carrier phase error, and sample time shift on the channel is discussed. Finally, time and frequency synchronisation is implemented and its performance is discussed.

2 Part 1: Optimal communication over the ideal channel

First, a short overview of the steps taken in this part is given. Then the used filter and the noise addition is discussed. Finally the performance of the channel is evaluated by constructing the bit-error-rate curve.

2.1 Overview

First the bits that need to be transmitted, are converted into symbols by the use of an already given Matlab function. The constellations used are BPSK, QPSK, 16QAM and 64QAM. Then the symbols are filtered using a halfroot Nyquist filter (see section 2.2). This reduces the bandwidth the transmission and also maximizes the signal-to-noise ratio if a matched filter is used at the receiver.

During transmission, noise is added. For this channel, additive white Guassian noise is used (see section 2.3).

Finally, when the symbols are received, they go through the matched halfroot Nyquist filter at the receiver and are turned back into bits.

2.2 Nyquist Filter

The halfroot Nyquist filter aims to reduce inter-symbol interference and to maximize signal-to-noise ratio by using two filters that are matched. One filter is placed at the receiver and the other one is placed at the transmitter. The Nyquist filter is constructed using following equations:

$$H(f) = \begin{cases} T & |f| \leq \frac{1-\beta}{2T} \\ \frac{T}{2}(1 + cos(\frac{\pi T}{\beta}(|f| - \frac{1-\beta}{2T}))) & \frac{1-\beta}{2T} \leq |f| \leq \frac{1+\beta}{2T} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Where β is the roll-off factor, T the symbol period and f the frequency. Using $\beta = 0.3$ and $T = 5 \times 10^{-7}$ a filter shown in figure 1 is obtained. This filter limits the communication bandwidth as it acts as a lowpass filter. Note that this filter is used in the base-band equivalent channel.

To obtain the halfroot Nyquist filter, one has to take the root of the Nyquist filter in the frequency domain. Note that the filter has to be normalised in the time domain.

The obtained impulse response is given in figure 2. For the standard Nyquist filter the impulse response is zero for multiples of the symbol duration, here 5×10^{-7} seconds, except for zero where the amplitude is one. This shows that the use of this filter cancels inter-symbol-interference.

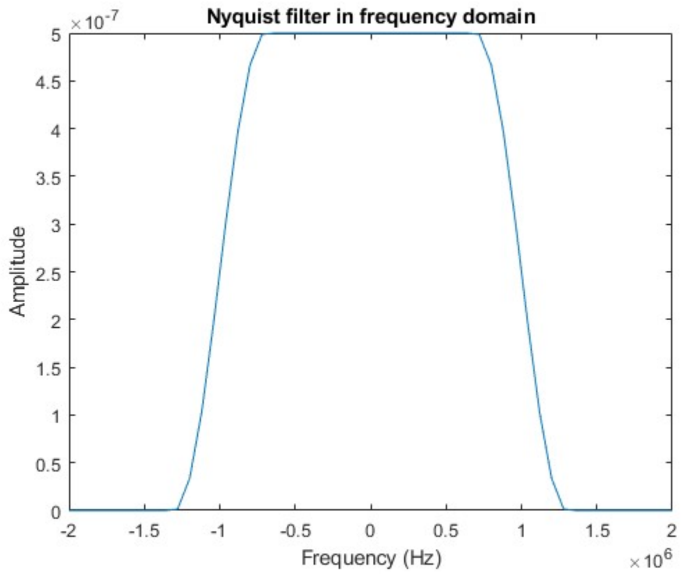


Figure 1: Nyquist filter in frequency domain

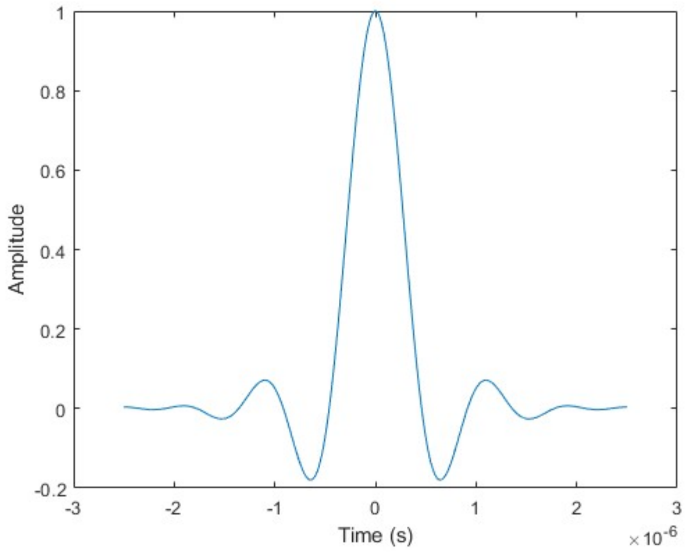


Figure 2: Impulse response of Nyquist filter

2.3 Noise Addition

In order to assess the bit-error-rate, one has to obtain the energy per bit E_b, such the wanted noise spectral density N_0 can be determined. To obtain E_b, the signal energy E_s should be determined using following formula:

$$E_s = \sum_{n=0}^{N} |s(n)|^2 \tag{2}$$

Then one finds for E_b:

$$E_b = \frac{E_s T_{sample}}{2N_b} \tag{3}$$

With T_{sample} the sampling rate, N_b number of bits in the signal, and where the factor 2 occurs due to the fact that the power of a bandpass signal is half of the power of its baseband equivalent signal. Using E_b one can obtain N_0 from the wanted $\frac{E_b}{N_0}$ ratio. Finally, the actual noise power P_n is determined using following formula:

$$P_n = \sqrt{N_0 f_{sample}} \tag{4}$$

With f_{sample} the sampling frequency and where the root is due to the fact that the baseband equivalent signal is used.

One obtains the noisy signal by adding the noise to the original signal. Note that the noise is generated independently for the real and imaginary part of the signal since they are uncorrelated and use a Gaussian distribution.

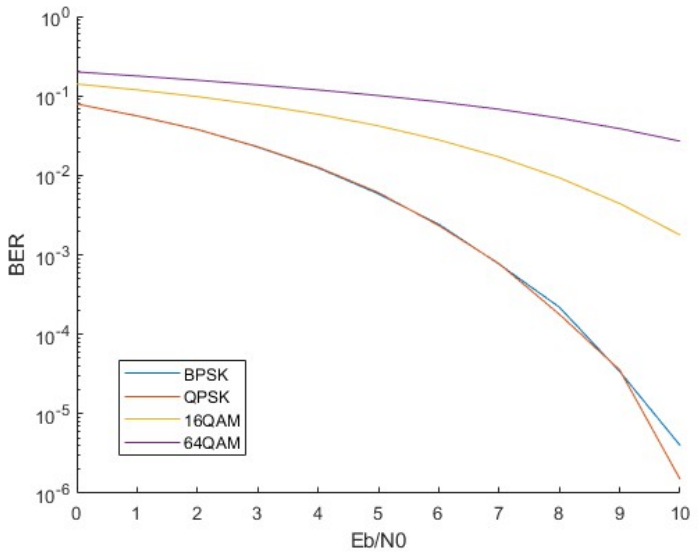


Figure 3: Bit Error Rate of BPSK, QPSK, 16QAM and 64QAM

2.4 Bit error rate

The Bit Error Rate curves for the different modulations are shown in figure 3.

These curves match the expected ones seen in the theory of this course. The higher the constellation size, the lower the Euclidean distance between two symbols and the higher the likelihood a symbol is wrongly interpreted when disturbed by white noise. This can be seen by the fact that 16QAM and 64QAM have a higher Bit Error Rate for the same E_b/N_0 compared to BPSK and QPSK. BPSK and QPSK overlap since QPSK is the combination of two orthogonal BPSK modulations. Also, as expected, the BER goes down for higher E_b/N_0 ratios.

2.5 Questions

- It is proposed to use the baseband equivalent model of the AWGN channel. Would it be possible to work with a bandpass implementation of the system?

  One could simulate the channel with the bandpass model. The carrier frequency is not important for the simulation since it is not simulated in real time. However the baseband model allows one to work with a lower sampling frequency and since the AWGN has also a baseband representation that can be added to the baseband signal, one can completely skip the transformation to a bandpass signal. And thus simplifying the simulation.

- How do you choose the sample rate in Matlab?

  The sampling rate should be at least twice the symbol rate. This sampling rate is chosen to avoid aliasing after filtering with the halfroot Nyquist filter. One can see from equation (1) that the frequency where the signal becomes zero is bigger then $\frac{1+\beta}{2T}$. For this filter, the frequency is equal to 2.6 MHz. So in theory one could upsample with a sampling rate of 1.3, but since this is not possible, the closest integer is chosen, and this is two.

- How do you make sure you simulate the desired Eb/N0 ratio?

  See Section 2.3.

- How do you choose the number of transmitted data packets and their length?

  In order to get an accurate bit error rate, one should send as many data packets as possible. Ideally this would be an infinite amount of packets. But since this is not possible one can use a rule of thumb that states that in order to observe an accurate bit error rate of 10^{-n}, one should send 10^{n+2} bits of data. This corresponds to having around 100 bit errors. Finally, the packet length should be equal to an integer amount of bits needed to encode a symbol. For example, this is equal to $4n$ if one uses 16QAM modulation.

- Determine the supported (uncoded) bit rate as a function of the physical bandwidth.

  The bit rate R is equal to $R = \log_2 M/T$ with M the amount of symbols used by a modulator and T the symbol rate. For this channel, the symbol rate is equal to 5×10^{-7} seconds. The bit rate for each modulation is given below:

  - BPSK (2 symbols):
$$R_{BPSK} = \frac{\log_2 2}{5 \times 10^{-7}} = 2\,\text{Mb/s} \tag{5}$$
  - QPSK (4 symbols):
$$R_{QPSK} = \frac{\log_2 4}{5 \times 10^{-7}} = 4\,\text{Mb/s} \tag{6}$$
  - 16QAM (16 symbols):
$$R_{16QAM} = \frac{\log_2 16}{5 \times 10^{-7}} = 8\,\text{Mb/s} \tag{7}$$
  - 64QAM (64 symbols):
$$R_{64QAM} = \frac{\log_2 64}{5 \times 10^{-7}} = 12\,\text{Mb/s} \tag{8}$$

- Explain the trade-off communication capacity/reliability achieved by varying the constellation size.

  Since the same bandwidth and transmitted power is used for all constellation sizes, one can conlcude that for a smaller constellation size, the Euclidian distance between the symbols is larger. So the reliability would increase (lower BER) at the cost of capacity (lower bits/s). The opposite can be said when the constellation size is increased.

- Why do we choose the halfroot Nyquist filter to shape the complex symbols?

  The halfroot Nyquist filter minimizes inter-symbol interference and maximizes the signal-to-noise ratio if two matched filters are used, where one is placed at the receiver and the other one at the transmitter. It also reduces the bandwidth of the signal. The unfiltered signal consists of pulses, these take up a very large bandwidth. In order to reduce this bandwidth, the filter is used.

- How do we implement the optimal demodulator?

  The optimal demodulator maximises the signal-to-noise ratio. This is achieved by using matched filters at the transmitter and receiver. In this case the halfroot Nyquist filter is used. The filter at the receiver and sender have the same frequency spectrum properties.

- How do we implement the optimal decoder?

  The optimal decoder can be implemented using two optimization criteria: Maximum a posteriori and Maximum likelihood.

  Maximum a posteriori maximizes the probability of a correct decision by using following formula:

$$\tilde{\underline{s}}_m^{MAP} = \max_{\underline{s}_m} p(\underline{s}_m|\underline{r}) \tag{9}$$

  Where $\underline{s}_m$ is the original transmitted signal and $\underline{r}$ the received signal. The decoder maps the received signal to one of the possible transmitted signals using the above formula. Using Bayes' rule one can rewrite equation (9) as:

$$\tilde{\underline{s}}_m^{MAP} = \max_{\underline{s}_m} \frac{p(\underline{r}|\underline{s}_m)p(\underline{s}_m)}{p(\underline{r})} \tag{10}$$

  the denominator can be neglected since it is independent of the transmitted signal. So one gets for maximum a posteriori:

$$\tilde{\underline{s}}_m^{MAP} = \max_{\underline{s}_m} p(\underline{r}|\underline{s}_m)p(\underline{s}_m) \tag{11}$$

  Note that knowledge about $p(\underline{s}_m)$ and $p(\underline{r}|\underline{s}_m)$ is required.

  Maximum likelihood uses following formula to map the received signal to a possible transmitted signal:

$$\tilde{\underline{s}}_m^{MAP} = \max_{\underline{s}_m} p(\underline{r}|\underline{s}_m) \tag{12}$$

  This criterion aims to finding the signal $\underline{s}_m$ that is closest in Euclidian distance to the received signal $\underline{r}$ and can be rewritten as:

$$\tilde{\underline{s}}_m^{MAP} = \min_{\underline{s}_m} \sum_{k=1}^{K} (r_k - s_{mk})^2 \tag{13}$$

  Note that the maximum a posteriori is equal to maximum likelihood if the probability of a transmitted signal is equal to $\frac{1}{M}$ with M the number of possible transmitted signals.

3 Part 2: Low-density parity check code

This section discusses the implementation and performance of LDPC encoding and, soft and hard decoding.

3.1 Coding gain and Bit error rate for soft and hard decoding

The bit error rate for soft and hard decoding are given in figure 4 and 5. One can see on the bit error rate for Hard Decoding that increasing the number of iterations does not affect the bit error rate in a major way. So using only one iteration would not result in a way lower bit error rate then for example 5 iterations. One can also see that the bit error rate of the hard decoding is worse for lower E_b/N_0. This is because the algorithm cannot correct the received symbols since too much noise is present. This causes the algorithm to "correct" errors that are in fact not errors and thus causes more errors to be present. For an E_b/N_0 ratio of around 2 dB the hard decoding algorithm starts to have a better bit error rate then the BPSK without hard decoding. One can see on the bit error rate for soft decoding that the number of iterations do have a large impact on the bit error rate. The more iterations, the higher the bit error rate for low E_b/N_0. Here again the same effects as for hard decoding occur. For higher E_b/N_0, one would expect a decrease in bit error rate with an increasing amount of iterations. But due to the limited computational power (Matlab runs out of memory) a high enough E_b/N_0 ratio where the bit error rate is not zero, could not be achieved. Thus the point where the bit error rate of the algorithm with more iterations crosses the bit error rate of the algorithm with less iterations, could not be obtained. Note that soft decoding leads to a lower bit error rate compared to hard decoding if the E_b/N_0 ratio is high enough. But it comes of course with a larger computational cost compared to hard decoding.

3.2 Questions

- When building the new BER curves, do you consider the uncoded or coded bit energy on the x-axis?

  One should consider the coded bit energy for the Bit Error Rate since the noise is added to the coded bits.

- How do you limit the number of decoder iterations?

  The decoder iterations are limited by two elements: a hard limit and the condition that there are no detectable more errors. The hard limit will limit the maximum amount of iterations the decoder will have since not every error can be corrected by the decoder. The second limit uses the property that $\hat{u}H^T = 0$. If this condition is true, then there are no detectable errors left.

- Why is it much simpler to implement the soft decoder for BPSK or QPSK than for 16-QAM or 64-QAM?

  It is easier for BPSK and QPSK since it is easier to compute the euclidean distance between a received bit and the most probable send bit. One has to only look at the sign and distance from the origin of the real and imaginary (only for QPSK) part of the received bit. This is more complicated for 16-QAM and 64-QAM.

- Demonstrate analytically that the parity check matrix is easily deduced from the generator matrix when the code is systematic.

  If the code is systematic that means that the input symbols of the encoded block is embedded in the output symbols. In this case, the input bits can be found at the end of the encoded block. This is possible by constructing the generator matrix G with an identity matrix I as following:

$$G = \begin{bmatrix} P & | & I \end{bmatrix} \tag{14}$$

  Where I has size $K \times K$ with K the amount of input symbols in the encoded block. P is then the parity portion of G and has size $K \times K$ since for this project the amount of parity bits of a block is equal to the amount of input bits. This means that for an input block of 10 bits, one has a encoded block of 20 bits.

  Now one can deduce the parity-check matrix by using its definition. This definition states that the rows of the parity-check matrix should be orthogonal to the rows of the generator matrix or equivalent: $GH^T = 0$. So this means that the rows of H form a basis of the subspace complementary to the code subspace. Using this property and the fact that systematic code is used, one can find that

$$H = \begin{bmatrix} I & | & P^T \end{bmatrix} \tag{15}$$

  With I a $K \times K$ matrix, since the amount of parity bits of a block is equal to the amount of input bits. P^T is also $K \times K$ for the same reason.

- Explain why we can apply linear combinations on the rows of the parity check matrix to produce an equivalent systematic code.

  Since the basis vectors of the parity-check matrix are orthogonal to the code subspace and the rows of this matrix form the basis, any linear combination of the rows of the parity-check matrix will still lead to a basis of that same subspace and will produce equivalent systematic code.

- Why is it especially important to have a sparse parity check matrix (even more important than having a sparse generator matrix)?

  A sparse parity-check matrix will lead to a simpler implementation of the decoding algorithms. This is because the decoding algorithms are constructed using check nodes and variable nodes. The amount of check nodes is controlled by the amount of ones in each row of the parity-check matrix. Reducing these ones will lead to less complex algorithms since less check nodes have to be created and evaluated.

- Explain why the check nodes only use the information received from the other variable nodes when they reply to a variable node.

  The check nodes calculate what the variable node, from which they did not use the information from, should have received based on the information from the other variable nodes. It does not use the information from that node such that the response of the check node to that variable node is as statistically independent as possible for the received symbol at the variable node.

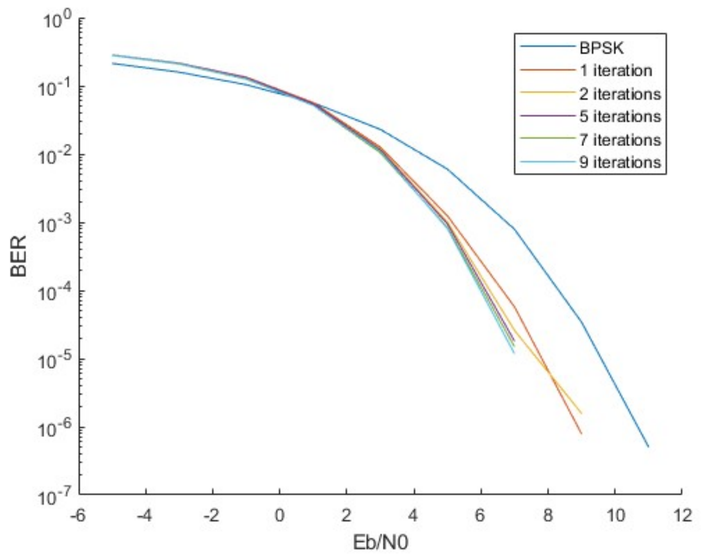


Figure 4: BER for Hard Decoding

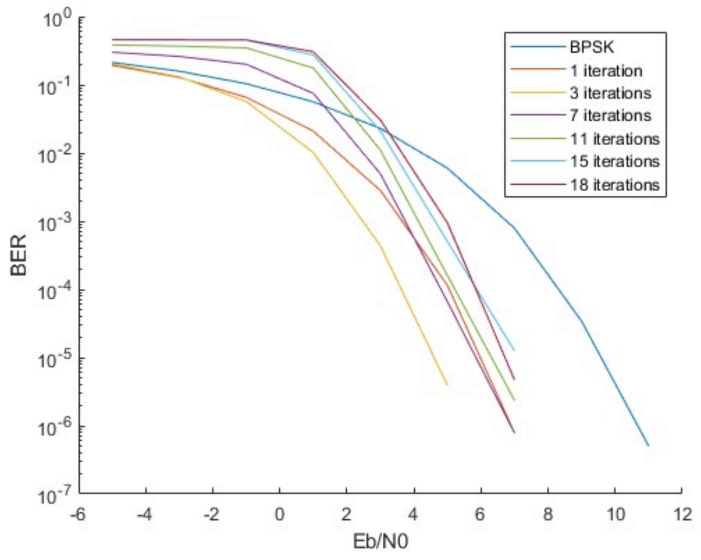


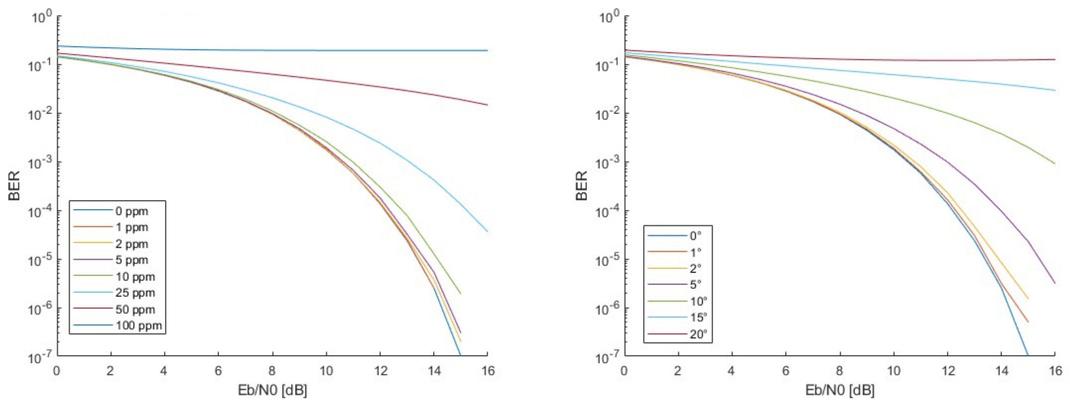
Figure 5: BER for Soft Decoding

4 Part 3: Time and frequency synchronisation

4.1 Bit error rate degradation caused by CFO and carrier phase error

In figure 6a and 6b one can see the influence of the carrier frequency offset and carrier phase error. For low carrier frequency offset, up to 10 ppm, the bit error rate does not degrade, but for higher ppm, the degradation is significant. This is because the CFO causes that the matched filters shift in frequency domain compared to each other. This causes bad matching filters and becomes worse for higher CFO. Note that for this simulation, the phase drift is compensated at the receiver after downsampling. If this is not compensated, the bit error rate would stay around 0.5 for any E_b/N_0 value.

The carrier phase error causes large degradation of the bit error rate for low values. The degradation is noticeable for a phase shift of around 5°. For a phase shift of around 15°, the bit error rate stays almost constant and cannot be influenced by the E_b/N_0 ratio. So one should carefully compensate the carrier phase error in order to avoid a large bit error rate.



(a) CFO (b) carrier phase error

Figure 6: Bit error rate degradation caused by CFO and carrier phase error

4.2 Bit error rate degradation caused by sample time shift

The bit error rate degradation caused by sample time shift can be seen in figure 7. The sample time shift is expressed as a fraction of the symbol period. When the time shift is higher then 0.02 times the symbol period, the bit error rate degradation is noticeable. If the time shift becomes even larger, then the degradation increases significantly. The bit error rate stays constant for any E_b/N_0 value for a sample time offset bigger or equal to 0.3 times the symbol period.

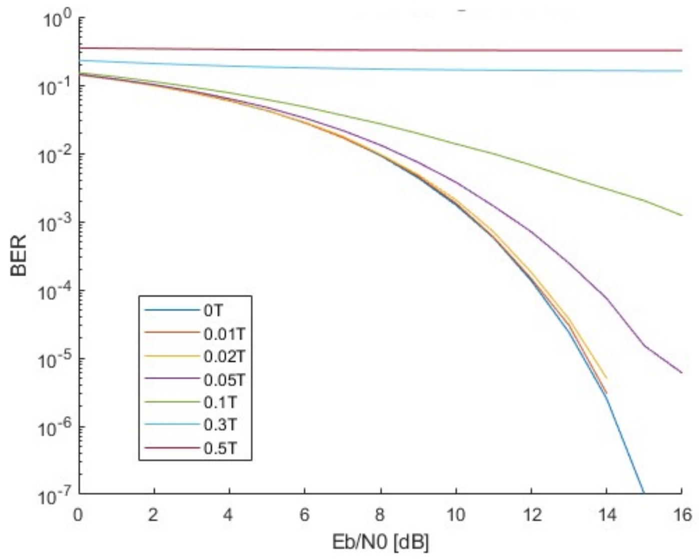


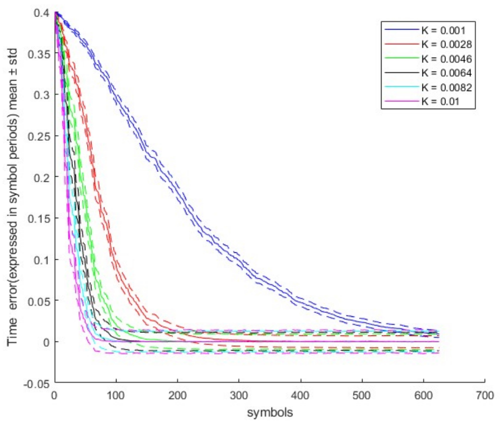
Figure 7: BER for 16QAM with a sample time shift

4.3 Gardner algorithm

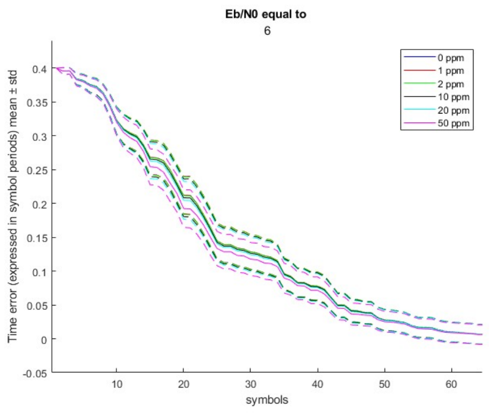
The Gardner algorithm is an algorithm that aims to estimate and compensate the sample time shift error. This estimation happens iterative with the received symbols. This iterative process can be seen in figure 8a. The time error converges to zero as the amount of symbols increase.

The figure 8a also shows the effect of the error weight K on the speed of the convergence and the standard deviation of this convergence. It is clear that for a small K, the error converges slower to zero then for a bigger K. But the standard deviation will also be larger for a big K and smaller for a small K. So in practice a balance should be struck between the convergence speed and standard deviation of the error. Note that for a good working frame and frequency acquisition, the time error should be less then 2% of the symbol period. So an error weight K of around 0.05 should work fine if the E_b/N_0 ratio is bigger than 5-6 dB.

Figure 8b shows the robustness of the Gardner algorithm to CFO. The algorithm is hardly affected by CFO values between 0 and 50 ppm. CFO values lie realistically between 0 and 10 ppm, so the algorithm can easily handle the CFO without degrading performance. Note that for higher CFO values, the algorithm might not behave as wanted.



(a) Influence of the error weight K



(b) Influence of the CFO

Figure 8: Influence of the error weight K and CFO on the convergence of the Gardner Algorithm for 16QAM.

4.4 Frame and frequency acquisition

Each simulation, a random (but known by the receiver) pilot sequence is generated and inserted at a fixed interval in the main bitstream. This pilot sequence marks where a frame starts. In practice, the symbols are received in real time, so the receiver does not know where the frame boundaries are. To simulate this, a small prefix of random length is inserted before the first frame. The introduction of this pilot sequence allows to correct most of the CFO. A differential cross-correlator is used to not only find the frame boundaries (marked by the pilot sequences), but also the CFO associated with the sequence. The received signal is then corrected using this recovered CFO.

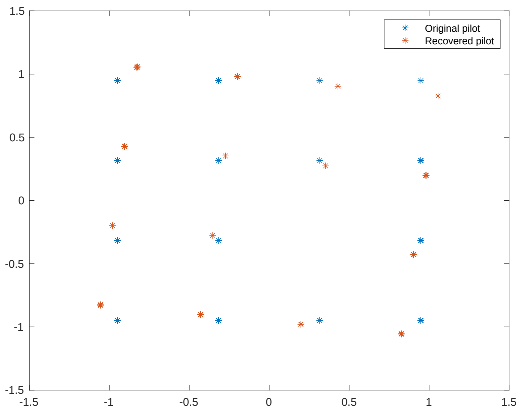
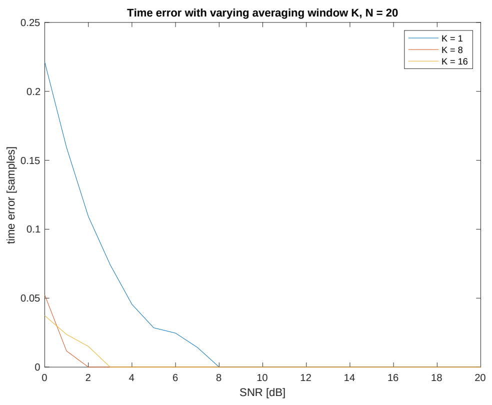
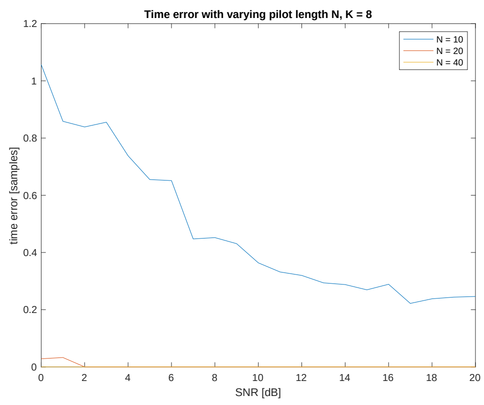


Figure 9: The residual phase shift of one of the pilot sequences.

Since the recovered CFO is not exactly the original CFO, a residual phase error remains (illustrated by figure 9). To correct this, the phase error is calculated for each pilot sequence in a frame and the phase is then interpolated for the symbols in between pilot sequences. This phase shift is then also corrected. The factors that determine the performance of this algorithm are the pilot length N and the averaging window length K, used in the differential cross-correlator. Figures 10 & 11 show the influence of these parameters on the performance of the algorithm. The improvements offered by increasing K from 8 to 15 are minimal, so it doesn't seem beneficial to increase K any further above 8. For values of N above 20, the improvements in CFO estimate don't seem to outweigh the cost of having a lower usable throughput, so again the best value seems to be $N = 20$. Having a residual CFO of about 2 ppm is easily compensated using the phase interpolation.

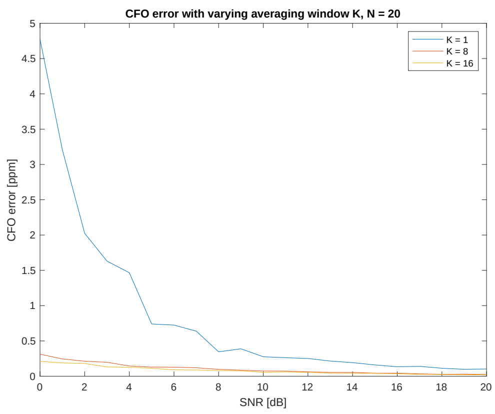


(a) K

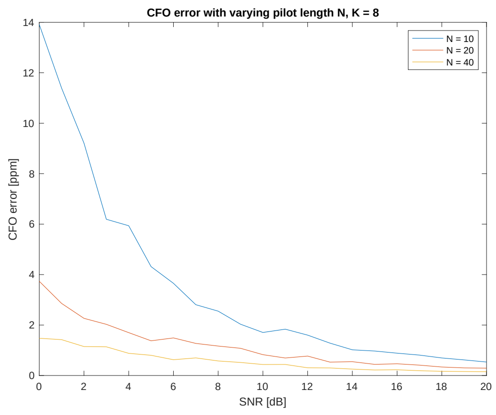


(b) N

Figure 10: Influence of parameters K & N on ToA estimate



(a) K



(b) N

Figure 11: Influence of parameters K & N on CFO estimate

4.5 Questions

- Derive analytically the baseband model of the channel including the synchronisation errors.

  The baseband equivalent model of a received signal of an ideal channel using the passband representation can be written as:

  $$e_r(t) = r(t)\cos(\omega_c t + \phi) - jr(t)\sin(\omega_c t + \phi) \tag{16}$$

  Where r(t) is the received passband signal and $e_r(t)$ the equivalent received baseband signal. This equation can be rewritten as

  $$e_r(t) = r(t)e^{j(\omega_c t + \phi)} \tag{17}$$

  The synchronisation errors are:

  - an offset in the carrier frequency $\Delta\omega$

    $$\omega_c^{receiver} = \omega_c^{transmitter} + \Delta\omega \tag{18}$$

  - phase offset ϕ_0

    $$\phi^{receiver} = \phi^{transmitter} + \phi_0 \tag{19}$$

  Introducing these errors into the baseband model, one obtains:

  $$\begin{aligned} e_r^{sync}(t) &= r(t)e^{j((\omega_c+\Delta\omega)t+\phi+\phi_0)} \\ &= r(t)e^{j(\omega_c t+\phi)}e^{j(\Delta\omega_c t+\phi_0)} \\ &= e_r(t)e^{j(\Delta\omega_c t+\phi_0)} \end{aligned} \tag{20}$$

  So the synchronisation errors can be calculated in the baseband model by multiplying the ideal baseband model with an exponential that contains the synchronisation errors.

- How do you separate the impact of the carrier phase drift and ISI due to the CFO in your simulation?

  To evaluate the ISI due to CFO, the carrier phase drift must be compensated. This is done by multiplying the received signal after filtering at the receiver with $e^{-j(\Delta\omega_c t+\phi_0)}$. This method ensures that the CFO still causes ISI by compensating the carrier phase drift after filtering at the receiver. If it was done before filtering at the receiver, the whole CFO will be compensated and will not cause ISI.

- How do you simulate the sampling time shift in practise?

  The sampling time shift is simulated in two ways. The first way is used for determining the bit error rate degradation and the second way is used for the implementation of the Gardner algorithm.

  The first method consists of choosing the upsampling rate such that the sampling time shift is equal to a integer number of the sample time. When downsampling occurs, the sample selected for decoding is now selected later in time compared to when no sampling time shift is present. Here is an example to clarify this: If the sample time shift is equal to $0.1T$ (with T the symbol duration) and the upsample rate is equal to 50, then the selected sample when downsampling will now be chosen 5 samples later compared to when no sampling time shift is present.

The second method builds on the first method but now includes interpolation, since it is almost impossible to get an upsample rate such that the sampling time shift compensated with Gardner is equal to an integer number of the sample time. This interpolation is done linearly.

- How do you select the simulated E_b/N_0 ratio?

  The E_b/N_0 ratio should be chosen such that the sampling time shift error is less than 2% of the symbol duration after converging. The time shift error variance after convergence depends on the E_b/N_0 ratio and the error weight. So for a weighted error of 0.005, the E_b/N_0 ratio should be bigger than 5 dB. If the error weight increases, the E_b/N_0 ratio should be bigger to achieve a maximum error of 2%. If the error weight decreases, the E_b/N_0 ratio can be lower to achieve the 2% error.

- How do you select the lengths of the pilot and data sequences?

  The lengths of the pilot sequences should be determined taking three things into account: The pilot should be short compared to the length of the actual data sequence to maximise actual data throughput, it should be long enough such that a good synchronisation can be obtained and the data sequences should be not be too long such that no phase ambiguities can occur (carrier phase error bigger or equal to 2π). For a E_b/N_0 ratio bigger than 5 dB, a pilot length of 20 symbols should be long enough to estimate the channel properties while still having a high data throughput. The phase drift caused by the remaining CFO error is compensated using interpolation.

- In which order are the synchronisation effects estimated and compensated. Why?

  First the sampling time shift error is estimated using the Gardner algorithm, since it is robust against CFO. Then the CFO and time of arrival are estimated. Finally the phase drift is estimated using interpolation. This phase drift is caused by the residual CFO error.

- Explain intuitively how the error is computed in the Gardner algorithm. Why is the Gardner algorithm robust to CFO?

  The Gardner algorithm is a recursive algorithm that computes the sampling time shift error by multiplying the difference between the sampled value of two consecutive symbols with the value of the signal halfway between those symbols. If there is a symbol transition, then the sampled value of the signal halfway between the symbols should be ideally to zero. In this case, the Gardner error stay the same and the sampling time shift error is reduced to zero. If this is not the case, then a sampling time shift error is still present. This error is estimated by the difference between the sampled value of two consecutive symbols with the value of the signal halfway between those symbols. The sign of the halfway value is compensated by the sign of the difference of the two consecutive symbols. This ensures that the correction happens in the right direction of the time axis. If there is no symbol transition, then the difference between the sampled value of the two consecutive symbols will be zero and the Gardner error does not change.

The Garner algorithm is robust to CFO since the algorithm multiplies the the halfway sample with the complex conjugate of the current and previous sample. This compensates most of the CFO.

- Explain intuitively why the differential cross-correlator is better suited than the usual cross-correlator? Isn't interesting to start the summation at k = 0 (no time shift)?

  The usual cross-correlator has a very high complexity due to its exhaustive 2D search by cross-correlating the received sequence with multiple frequency shifted pilot copies. The differential cross-correlator also cross-correlates the received sequence with pilot copies. But now the pilot copies consist of pieces of different sizes of the pilot, where the piece size ranges from 1 to K. This reduces the complexity. k = 0 is not used since the CFO cannot be taken into account.

- Are the frame and frequency acquisition algorithms optimal? If yes, give the optimisation criterion.

  The usual cross-correlator algorithm is optimal since it uses the maximum likelyhood criterion:

$$(\hat{n}, \hat{\Delta\omega}) = arg \max_{n,\Delta\omega} p(\underline{y}[n]|\underline{a}, \Delta\omega) \tag{21}$$

  The differential cross-correlator is not optimal since it omits a part of the above equation to simplify the algorithm.

5 Conclusion

After implementing the three parts, one obtains a complete baseband simulation of a DVB-S2 communication link. The focus lies here on the baseband model with noise, the channel coding to improve the error rate and the synchronisation algorithms to correct for timing discrepancies between transmitter & receiver.