ELEC-H-401 - Modulation and Coding

DVB-S2 Communication Chain Implementation

MA1 - Electronics and Information Technology

Théo Lepoutte
John Antoun

Academic Year 2019-2020

Contents

1 Introduction

In order to apply the theoretical concepts learned in the Modulation and Coding course, students were asked to simulate the communication chain of the second generation Digital Video Broadcasting-Satellite (DVB-S2).
This first part of the report focuses on the simulation of the baseband equivalent model of the communication channel, while the second part will be devoted to the LDPC channel encoding and the third one to the simulation of the synchronisation algorithms.

2 Optimal communication chain over the ideal channel

2.1 Bit generation and symbol mapping

At the transmitter side, a chain of bits[1] is randomly generated then mapped on a sequence of complex symbols, accordingly with the number of bits per symbol specified in the parameters. The symbol sequence is then upsampled by adding a constant number of zero-valued samples between every symbol, which increases the sampling rate by an upsampling factor M. The resulting constellation diagram for a quadrature amplitude modulation (16-QAM) is shown on Figure 1.

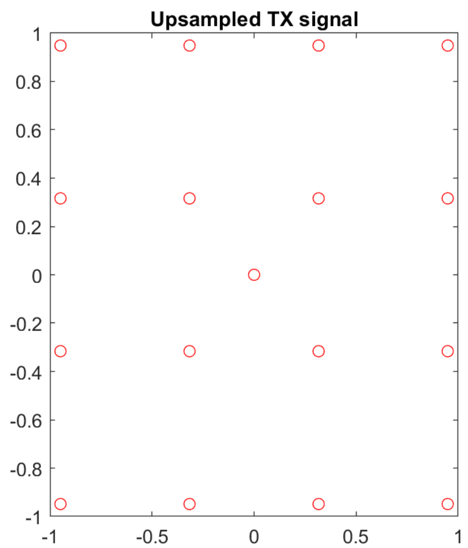


Figure 1: Constellation of an upsampled 16QAM signal before RRC filtering

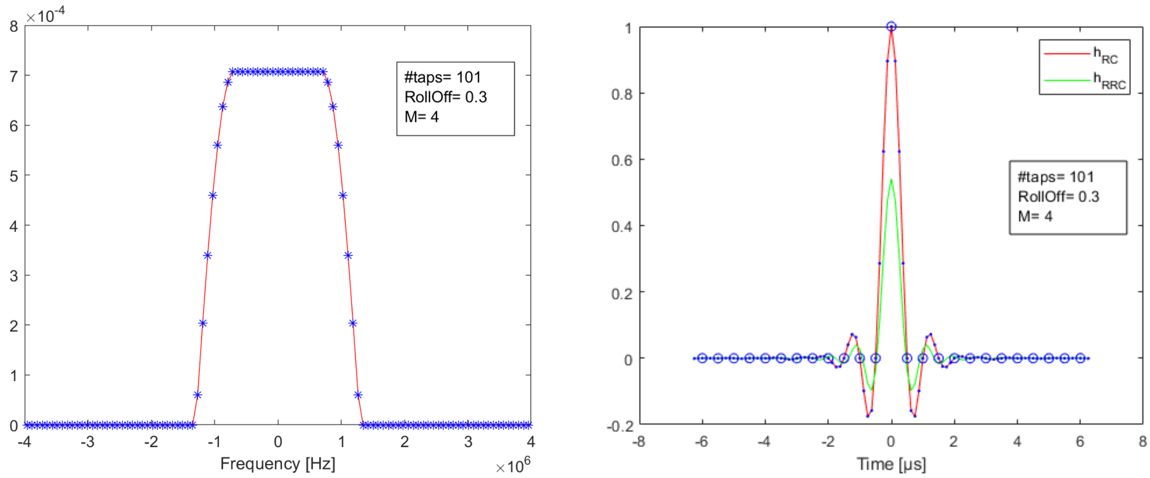
[1] N_b=number of bits in the sequence

2.2 Nyquist filter design

The step following the upsampling is the convolution of the sequence with a shaping pulse $g(t)$ in order to form the baseband equivalent model of the transmitted signal. However, as a rectangular pulse shape causes inter-symbol interference (ISI) in the received sequence, a raised-cosine filter, also known as Nyquist filter, is used. Indeed, when sampled at the symbol rate, the impulse response of this filter is equivalent to a Dirac pulse (Figure 2b). Its frequency response is described as following :

$$H(f) = \begin{cases} T_{sym} & (0 \leq |f| < \frac{1-\beta}{2T_{sym}}) \\ \frac{T_{sym}}{2}(1 + cos[\frac{\pi T_{sym}}{\beta}(|f| - \frac{1-\beta}{2T_{sym}})] & (\frac{1-\beta}{2T_{sym}} \leq |f| \leq \frac{1+\beta}{2T_{sym}}) \\ 0 & (\frac{1+\beta}{2T_{sym}} > |f|) \end{cases}$$

where β is the roll-off factor and T_{sym} is the symbol period.



(a) Frequency response

(b) Impulse responses

Figure 2: Raised-cosine filter

The raised-cosine pulse $h(t)$ is used to cancel inter-symbol interference and is split into $g(t)$ and $g(-t)$ as matched filters (to maximise the SNR). This is done by taking the square root of the frequency response : $G(f) = \sqrt{H(f)}$. It is also necessary to normalise the raised-cosine pulse to avoid the amplitude changes of the QAM symbols at the receiver. As the simulation requires the filter to be sampled (in the frequency domain in this case), a number of taps N high enough has to be chosen to increase the frequency resolution and extend the time vector. However, it is important not to choose a too high number of taps because it increases the computation time. For example, as shown in fig.3, a number of taps higher than $N = 23$ is not really necessary for the 64-QAM as the Bit Error Rate

(BER) does not change significantly.

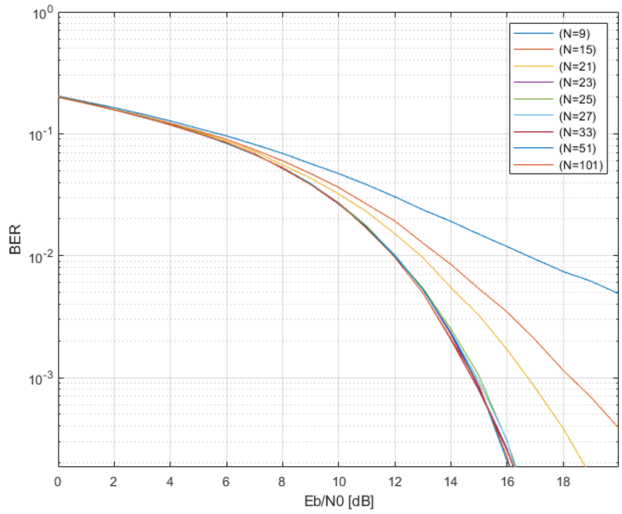


Figure 3: Bit Error Rate (BER) of the 64-QAM for different number of taps

This number of taps impacts the length of the filtered signal which becomes :

$$N_{TX} = M \cdot N_b + N - 1$$

2.3 Noise addition

A realistic simulation requires the addition of white Gaussian noise (AWGN). As the sent symbols remain in their bandpass form, the noise, which has a real nature in the real channel, has to be expressed in its complex equivalent form. It is therefore necessary to generate two distinct sequences of numbers following a standard normal distribution ($\mu = 0, \sigma = 1$) as the noise has to affect any transmitted sample independently :

$$n(t) = \sqrt{\frac{P_n}{2}}(\mathbf{X} + j \cdot \mathbf{Y}) \qquad \text{with} \begin{cases} \mathbf{X} = (X_1, ..., X_{N_{TX}}) \\ \mathbf{Y} = (Y_1, ..., Y_{N_{TX}}) \end{cases} \quad (X_i, Y_i \sim \mathcal{N}(0, 1))$$

The noise power P_n can be computed by multiplying the power spectral density of the noise by the sampling frequency F_s. It is important to notice that, as the real and imaginary part of the noise both have a PSD equal to N_0, a 2 factor is necessary :

$$P_n = 2 \cdot N_0 \cdot F_s$$

The value of N_0 can be found considering the desired ratio $\frac{E_b}{N_0}$ where E_b is the energy per bit. As the energy of the transmitted signal is calculated with :

$$E_s = \frac{1}{F_s}\int_0^{N_{TX}} |s(t)|^2 dt$$

where F_s it the sampling frequency, the bit energy is computed with :

$$E_b = \frac{E_s}{2 \cdot N_b}.$$

The factor 2 comes from the fact that the power of a bandpass signal is equal to half the power of its baseband representation.
Finally, the received sequence is obtained by adding the noise to the transmitted upsampled symbol sequence :

$$r(t) = s(t) + n(t).$$

The last step to retreive the initial bit sequence is to make the convolution between this received sequence and the matched filter, downsample it and then demap it.

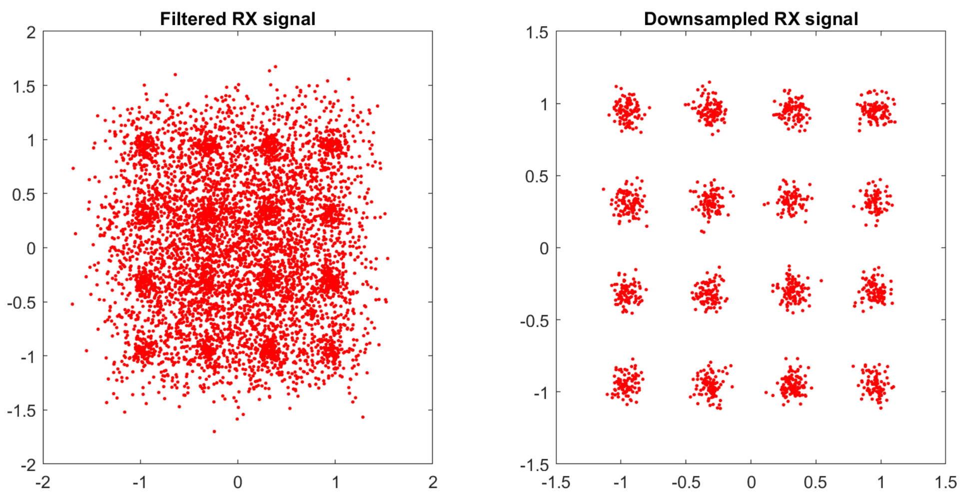


Figure 4: Constellation of the signal at the receiver with noise addition after RRC filtering and downsampling

It is interesting to evaluate the Bit Error Rate (BER) of every modulation depending on the bit energy to noise PSD ratio. As shown in fig.5, the modulations containing a higher number of bits per symbol (Nbps) are more sensitive to the noise. This is due to the different constellation points becoming closer as the number of bits per symbol increases.

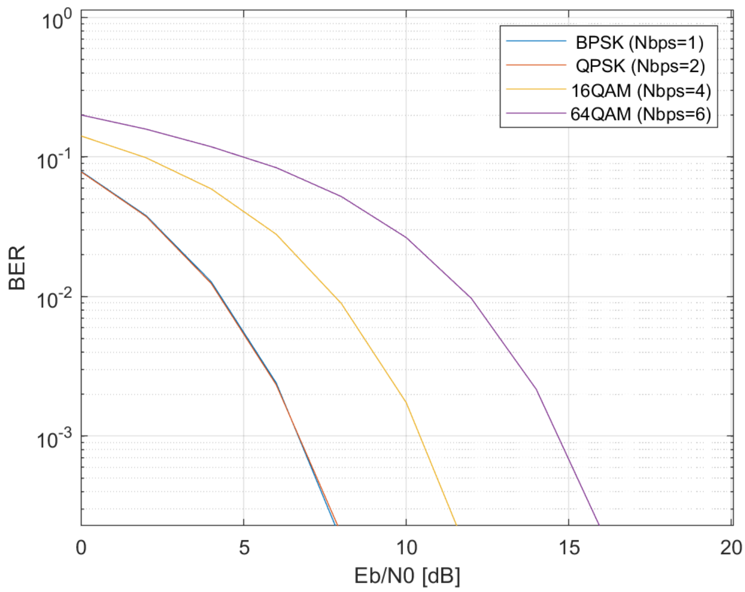


Figure 5: Bit Error Rate (BER) comparison for the different modulations as a function of E_b/N_0

2.4 Questions

2.4.1 Simulation

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

  Working with a baseband equivalent model at this stage allows to avoid the bandpass representation. A bandpass signal occupies bandwidth around a carrier frequency that is in general large. The baseband equivalent representation allows to work at lower sampling frequency (thus reduce complexity) and to develop modulation/demodulation independently which makes the design process more flexible.

- ***How do you choose the sample rate in Matlab?***

  The sampling rate has to be at least twice the highest frequency F_{max} of the signal in order to avoid aliasing. This requires the symbols sequence to be upsampled, so that the frequency is multiplied by the upsampling factor M. In fact, with the Nyquist filter we know the highest uncut frequency is $F_{max} = (1+\beta)/(2\,T_{symb})$. Since $0 < \beta \leq 1$, it can be shown that the signal needs to be upsampled ($M \geq 2$), so that $Fs = M/T_{symb} \geq 2 \cdot Fmax$.

- ***How do you make sure you simulate the desired E_b/N_0 ratio?***
  The method to simulate the noise is described in section 2.3. To sum up, the noise power has to be computed using the energy per bit of the signal and the E_b/N_0 ratio, then two sequences of samples following a normal distribution are generated in order to form the real and imaginary parts of the complex noise weighted by the noise power.

- ***How do you choose the number of transmitted data packets and their length?***
  The length of the packet (number of symbols) must be a multiple of the number of bits per symbol chosen. If this is not the case, some information will be lost. The number of bits sent should be high enough in order to observe a relevant BER estimation.

2.4.2 Communication system

- ***Determine the supported (uncoded) bit rate as a function of the physical bandwidth.***
  Considering the cutoff frequency F_c of the raised-cosine filter, the two-sided bandwidth is equal to $B = 2\ F_c = 2\ \frac{1}{2T_{sym}} = \frac{1}{T_{sym}}$, which is equal to the symbol rate. As every symbol contains $Nbps$ bits, the bit rate is equal to :

  $$R = \frac{Nbps}{T_{sym}} = 2\ Nbps\ \cdot\ F_c = B\ Nbps\ \text{[bits/s]}$$

  For example, by considering F_c = 1MHz, the 16-QAM modulation implies that the bit rate is 8Mbit/s.

- ***Explain the trade-off communication capacity/reliability achieved by varying the constellation size.***
  A higher constellation size means a higher probability of making errors at the demapping, since constellation points are closer to each other. As a consequence the BER increases for the same power transmitted and the reliability of communication is decreased. A higher constellation size means a higher bit rate but also increases the BER.

- ***Why do we choose the halfroot Nyquist filter to shape the complex symbols?***
  The use of a simple rectangular pulse causes spectral leaking, so loss of power. The frequency sampling method allows to define a well limited bandwidth in the frequency domain and sample it to make an adequate pulse in the time domain, satisfying cancellation of ISI if sampled at the symbol rate. Thus, a raised cosine filter $H(f)$ satisfying

this requirements has been designed and a root raised cosine (RRC) filter was computed by $G(f) = \sqrt{H(f)}$ and used at both transmitter and receiver as matched filters.

- ***How do we implement the optimal demodulator? Give the optimisation criterion.***

  A demodulator is optimal when the received signal $e_r = e_s + e_n$ (signal transmitted e_S corrupted by AWGN) is matched filtered and sampled at symbol rate. The optimisation criterion that is satisfied is the SNR, that is maximised when a matched filter is used at the receiver. More specifically, the transmitted signal is the shaping of the symbols by $g(t)$ and is matched filtered at the receiver (after being corrupted by noise) with $g(-t)$.

- ***How do we implement the optimal detector? Give the optimisation criterion.***

  The detector function is to make a decision on the transmitted symbol regarding the received corrupted symbol r. The *maximum a posteriori* (MAP) criterion maximises the probability of making a good decision, which can be computed by Bayes rule:

  $$s_{MAP} = \max_{s_m} p(s_m|r) = \max_{s_m} p(r|s_m)p(s_m)$$

  As all M symbols are equiprobable, $p(s_m) = 1/M$, the criterion above is equivalent to the maximum likelihood criterion (ML): $s_{ML} = \max_{s_m} p(r|s_m)$. As the received symbols follow a Gaussian distribution due to the noise, and they are independent, the probability $p(r|s_m)$ is equal to :

  $$p(r|s_m) = \frac{1}{\sqrt{(\pi N_0)^K}} \exp[-\frac{1}{N_0}\sum_{k=1}^{K}(r_k - s_{mk})^2]$$

  As the Napierian logarithmic function is monotonic, the maximisation of $\ln(p(r|s_m))$ is equivalent to minimise the Euclidian distance in the complex plan.

3 Low-density parity check code

3.1 Encoder

In order to detect and correct errors at the receiver, some redundancy has to be added to the message of bits. To do so, the message is divided into blocks of K bits which are encoded by mean of a generator matrix G of size $N \times K$. The code vector u resulting from this transformation belongs to a subspace such that the modulo-2 sum of any two vectors in this subspace also belongs to the subspace. As we are considering a systematic code, the generator matrix can be written as a horizontal concatenation of a parity array portion and an identity matrix $G = [P|I_K]$, therefore the codeword is composed of the parity bits $(d \cdot P)$ and the block of bits d. This codeword is then going through the physical channel.

3.2 Hard decoding

The received message is decoded by help of the parity-check matrix H formed by rows that are the basis of the subspace complementary to code space. These rows are therefore orthogonal to the rows of the generator matrix so the matrix H can be written as a horizontal concatenation of an identity matrix I_{N-K} and the transposed parity array portion : $H = [I_{N-K}|P^T]$. The particularity of a low-density parity check code is the sparseness of the parity-check matrix which reduces the number of computation and so the complexity. An error is detected when the product of the received codeword and the transposed parity-checked matrix, which is called syndrome, is not equal to the null vector. The corrupted codeword is then corrected using the hard decoding algorithm implementing the Tanner graph as shown in fig.6 in function of matrix H. Indeed, hard decoding works on a binary symmetric channel (BSC) and receives the transmitted bits as input. After being tested on a small scale case (H of low dimensions), it is then integrated to the project to decode the message received r that is a corrupted version of the sent message u. The H matrix for the project is of dimensions 128×256.

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

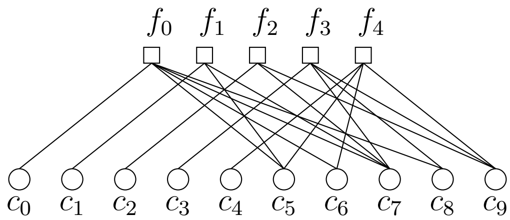
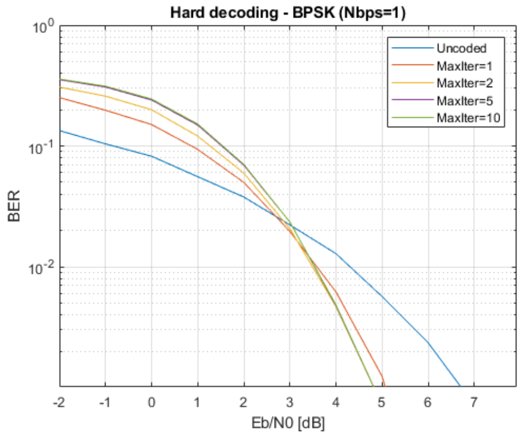
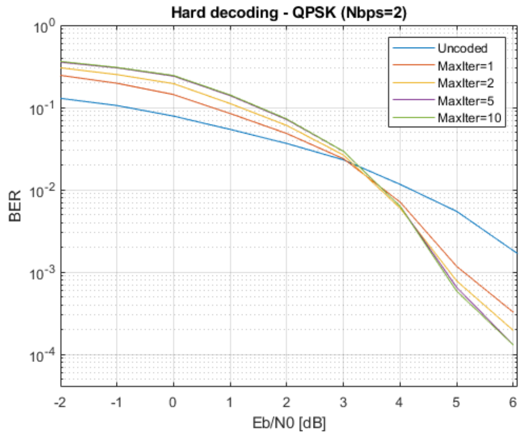


Figure 6: Parity-check matrix and its associated Tanner graph

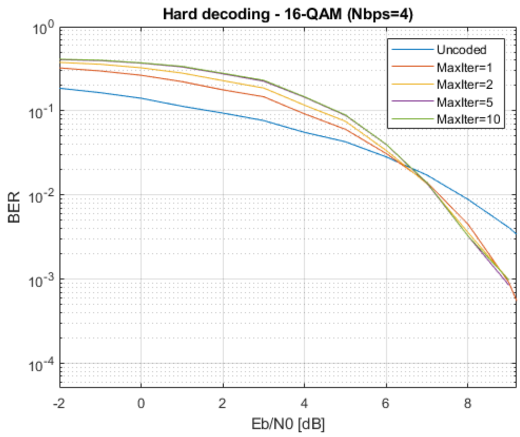
As shown in fig. 7, below a certain SNR the BER is worse than without any coding. The hard decoding algorithm is therefore useful only if the SNR is higher than this point. The coding gain defined by $\text{SNR}_{uncod} - \text{SNR}_{cod}$ can be computed for the different modulations. For example, if the uncoded QPSK has a BER of 10^{-2} at SNR level 4.2dB and the coded system has the same BER at a SNR of 3.5dB, then the coding gain is equal to $4.2 - 3.5 = 0.7$dB. It is also important to notice that increasing the number of maximum iterations for the hard decoding improves the BER after the intersection point and deteriorates it before that same point. Complex modulations also have an



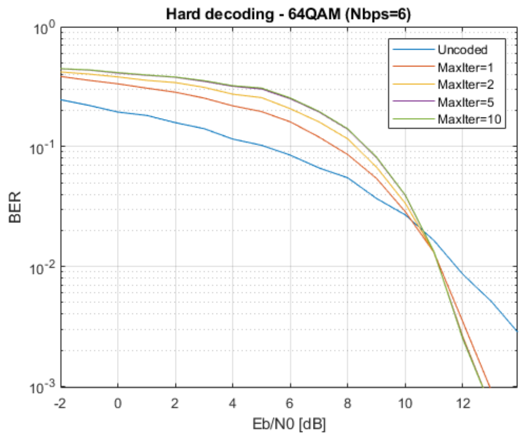
(a) BPSK



(b) QPSK



(c) 16QAM



(d) 64QAM

Figure 7: Comparison of BER curves for uncoded and hard decoded messages with different values of maximum iteration

intersection point which is located at a higher SNR, which shows that hard decoding should only be used for low-noise channels.

3.3 Soft decoding

The soft decoding algorithm is very similar to the hard decoding, except that real values are exchanged between nodes instead of binary values. The message sent by variable nodes are the probability that the received bit c_i is 0 given an observation r_i, $P(c_i = 0|r_i)$. The response from the check nodes are probability that $c_i = 0$ based on the message from other connected variable nodes. Another difference between soft decoding and hard decoding is that soft decoding is applied to downsampled symbols in the communication chain, while hard decoding was applied to demodulated bits. Soft decoding was only implemented for BPSK (fig. 8). The signal after downsampling has however a complex form and it required to take its real part to enter it in the soft decoding algorithm.

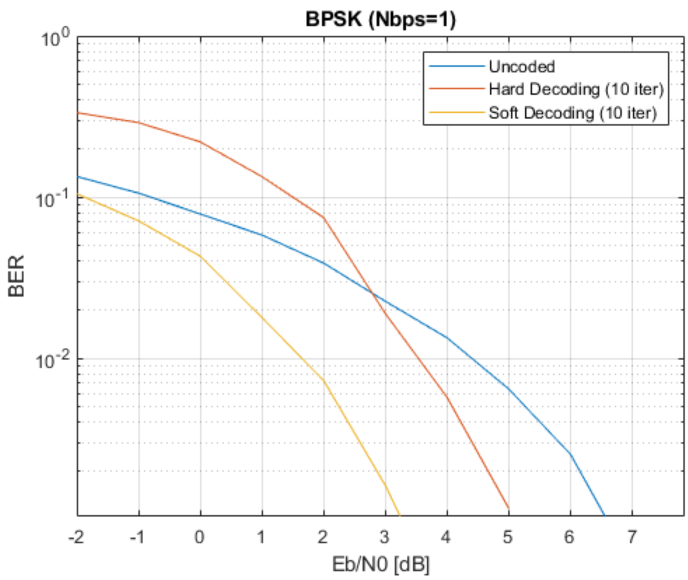


Figure 8: Comparison of BER curves for uncoded, hard decoded and soft decoded BPSK messages

As shown in fig.8, the soft decoded message has a lower BER than the hard decoded and uncoded messages, no matter the SNR value.

3.4 Questions

3.4.1 Simulation

- ***When building the new BER curves, do you consider the uncoded or coded bit energy on the x-axis?***
  The energy taken into account is the energy of the coded signal since this is the signal that will be affected by the noise through the channel.

- ***How do you limit the number of decoder iterations?***
  The algorithms of decoding (soft and hard) are running until the syndrome of the message that is being corrected u is equal to a null vector, so $s = \hat{u}.H = \bar{0}$. This means that the vector corrected has no more projection on the complementary subspace (to the code one) and that the error has been found (according to the algorithm). However, the algorithm could eventually never converge to a zero syndrome, in which case the algorithm must force a maximum number of iterations.

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

The soft decoding is based on the bit probabilities which are easier to compute for BPSK and QPSK because it only requires to observe the sign of the real and imaginary parts of the received symbols, while for higher modulation formats it requires to compute the euclidian distance.

3.4.2 Communication system

- ***Demonstrate analytically that the parity check matrix is easily deduced from the generator matrix when the code is systematic.***
  As explained in section 3.2, when the code is systematic, the generator matrix can written as a horizontal concatenation of the parity array portion and an identity matrix : $G = [P|I_K]$. As the rows of the parity-check matrix H are the basis of the subspace complementary to code space and are therefore orthogonal to the rows of the generator matrix, H can be written as a horizontal concatenation of an identity matrix and the transposed parity array portion :$H = [I_{N-K}|P^T]$.

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

  As the rows of the parity-check matrix H form a basis of the subspace complementary to the code subspace, any linear combination of these rows still belongs to the subspace and produces an equivalent systematic code.

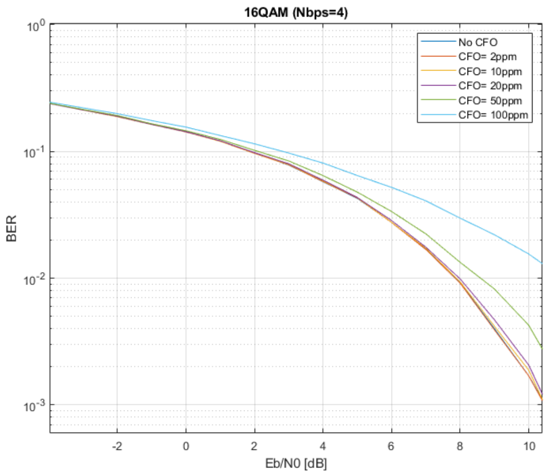
- ***Why is it especially important to have a sparse parity check matrix (even more important than having a sparse generator matrix) ?***
  The purpose of having a sparse parity-check matrix is to keep the complexity low. Indeed, every non-zero element creates new links between the different nodes which increases the number of computations. As the decoding algorithms are iterative, the complexity can be greatly affected by the sparseness of the parity check matrix.

- ***Explain why the check nodes only use the information received from the other variable nodes when the reply to a variable node.***
  The check nodes only use the information received from the other variable nodes in order to ensure that its response is as independent as possible from the received r_i at the concerned variable node.

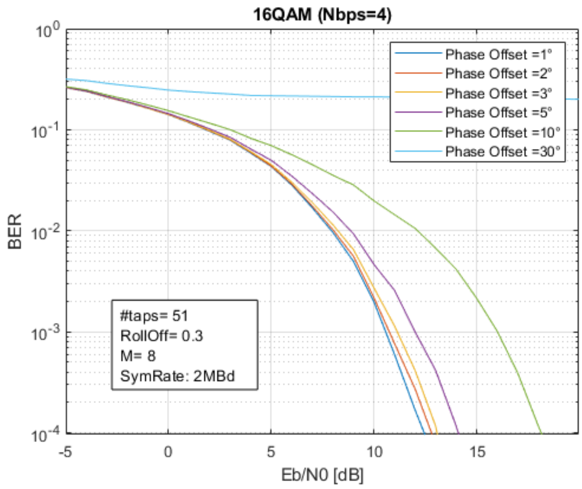
4 Time and frequency synchronisation

4.1 Impact of the CFO and the phase offset

In order to simulate hardware imperfections which cause inaccuracy of the local oscillators, synchronisation errors are implemented. The carrier frequency offset (CFO) and the phase error are implemented by multiplying the received signal by $exp(j(2\pi\Delta f + \phi))$. The BER obtained by only observing ISI after applying different values of CFO is shown in Figure 9a. The impact of the phase offset on the BER is shown in Figure 9b.



(a) Impact of the CFO on the BER



(b) Impact of the phase offset on the BER

Figure 9

4.2 Impact of the time shift

As the sample clock offset (SCO) is complex to implement, it is neglected in this project. The time shift is simulated by increasing the sampling rate and shifting the samples by n positions. As the sampling rate is equal to $\frac{M}{T_{symb}}$, the time shift is therefore equal to $t_0 = \frac{nT_{symb}}{M}$. The impact of different values of t_0 is shown in Figure 10.

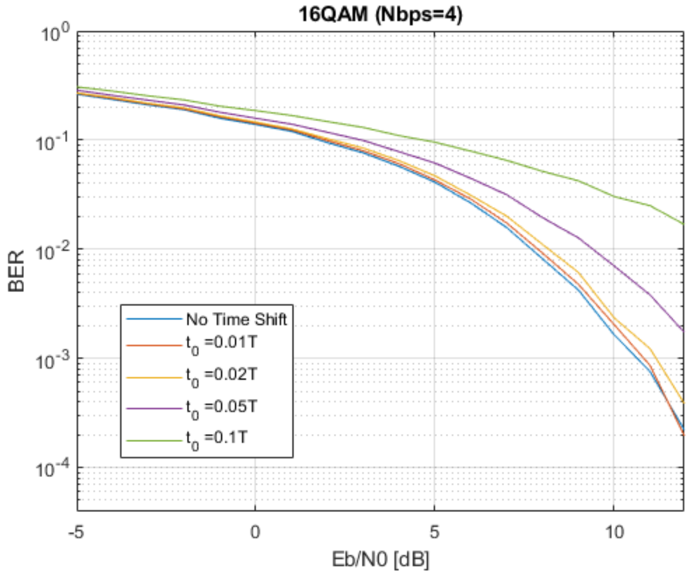


Figure 10: Impact of time shifting on the BER

4.3 Gardner algorithm

In order to estimate and compensate the effects of the synchronisation errors, we use synchronisation algorithms. The purpose of the Gardner algorithm, which is a non-data aided (NDA) feedback algorithm, is to correct the sampling time errors. The impact of the error weight κ is shown in Figure 11.

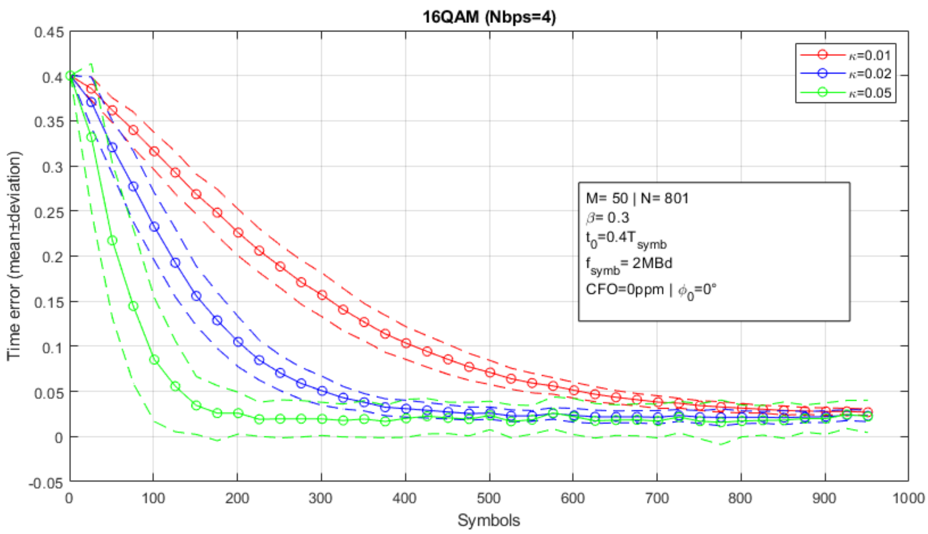
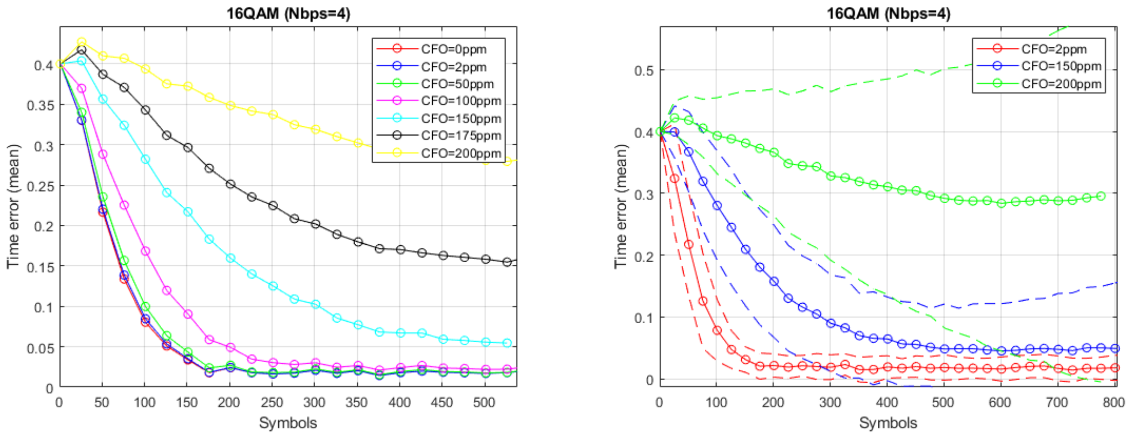


Figure 11: Convergence of the time error for different κ values

The robustness of the algorithm to the CFO is shown in Figure 12a. Increasing the CFO also impacts the standard deviation as shown in Figure 12b. In this case ($t_0 = 0.4T_{symb}$,

$\kappa = 0.05$), for CFO values up to 50 ppm, the convergence is not greatly affected by the CFO. A realistic case of CFO is of 10 ppm or less. Therefore the algorithm is robust enough to be applied before other corrections.



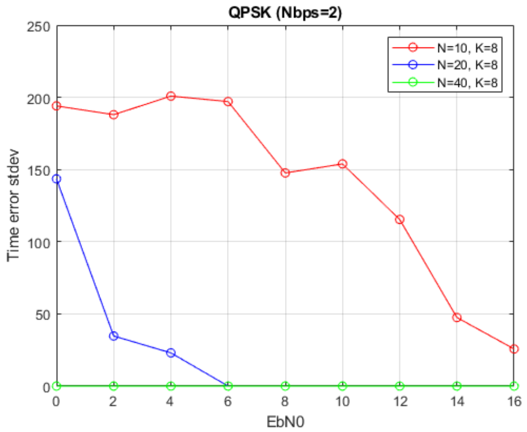
(a) (b)

Figure 12: Robustness of the Gardner algorithm to CFO

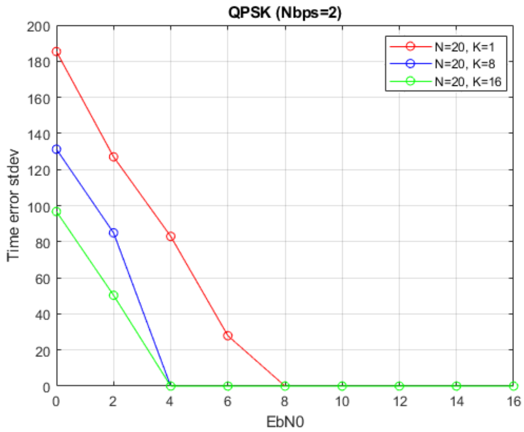
4.4 Data frame acquisition

In a real life communication, a device does not know when the useful transmitted information starts. Hence, a pilot is added in front of a message and its position can be determined by computing the differential cross-correlation between the message and the pilot. The index that maximises this correlation corresponds to the estimate of position of the pilot (time of arrival or ToA). The correlation also allows to estimate the CFO. The algorithm efficiency depends on the size N of the pilot and the maximum size K of subwindow used for the correlation average.

The influence of these parameters on the ToA estimate is illustrated in Figures 13a and 13b. The estimation accuracy increases with N and with K. The same conclusions can be obtained on the CFO estimate as observed in Figures 14a and 14b.

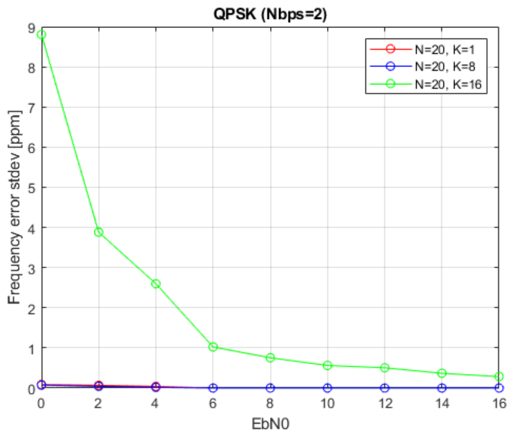


(a) Influence of pilot size N on ToA estimate, no CFO

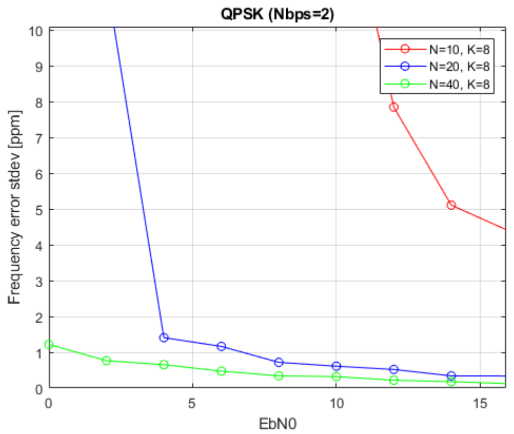


(b) Influence of K on ToA estimate, no CFO

Figure 13



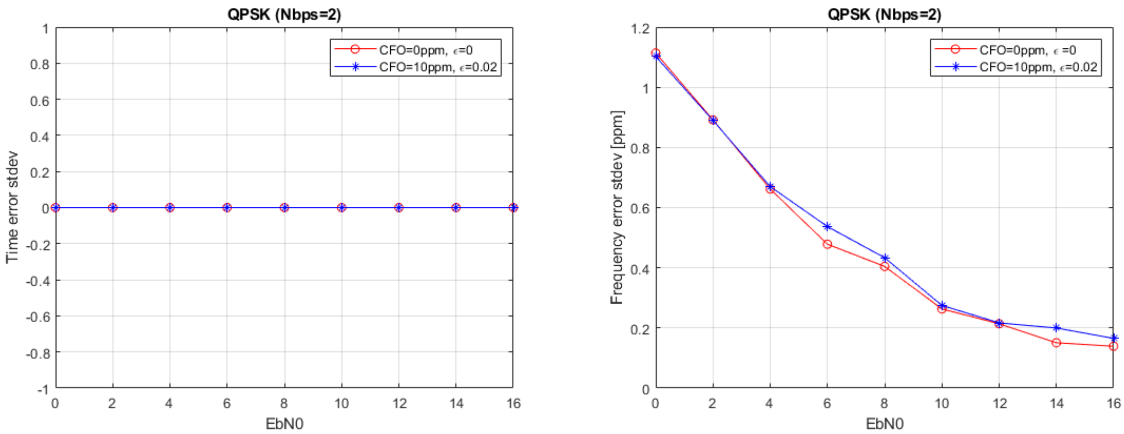
(a) Influence of K on CFO estimate, no CFO on received signal



(b) Influence of N on CFO estimate, no CFO on received signal

Figure 14

The algorithm efficiency is somewhat slightly influenced by the presence of a CFO and time shift on the received signal. No significant effect has been observed however on the ToA estimate (fig.15a) and a slight effect was observed on the CFO estimate (fig.15b).



(a) Robustness to CFO on the ToA estimate (N=40, K=8)

(b) Robustness to CFO on the CFO estimate (N=40, K=8)

Figure 15

4.5 Questions

4.5.1 Simulation

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

  The band-pass representation of the received signal into an ideal channel can be written :

  $$r(t) = \Re[\tilde{r}(t)\exp(j2\pi f_c t)] = \Re[\tilde{r}(t)]\cos(2\pi f_c t) - \Im[\tilde{r}(t)]\sin(2\pi f_c t)$$

  where $\tilde{r}(t)$ is the baseband representation of the signal and f_c is the carrier frequency. If the synchronisation errors are added, the band-pass signal is therefore :

  $$r_e(t) = \Re[\tilde{r}(t)\exp(j2\pi(f_c + \Delta f)t + \Phi_0)] = \Re[\underbrace{\tilde{r}(t)\exp(j2\pi\Delta f t + \Phi_0)}_{\tilde{r}_e(t)}\exp(j2\pi f_c t)]$$

  where $2\pi\Delta f = \Delta\omega$ is the carrier frequency offset(CFO) and Φ_0 is the phase offset. $\tilde{r}_e(t) = \tilde{r}(t)\exp(j2\pi\Delta f t + \Phi_0)$ is the baseband model of the received signal considering the synchronisation errors.

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

  As seen above, the CFO simulation consists in multiplying the received signal (symbols stream convolved with the Nyquist filter) with q complex exponential. Assuming no noise is present, the model is:

$$r(t)\, e^{j\Delta\omega t} = \sum_n I[n] g(t - nT)\, e^{j\Delta\omega t} = \sum_n I[n]\, e^{j\Delta\omega nT} g(t)\, e^{j\Delta\omega t}|_{t-nT}$$

The last equality shows that the CFO adds a linearly varying phase $\Delta\omega nT$ to the symbols of the sequence generated. It also shows with $g(t)\, e^{j\Delta\omega t}$ that it is as if the Nyquist filter was shifted in frequency by $\Delta\omega$ which makes filters mismatch and causes ISI. To observe the impact of this, the signal at the receiver after convolution with the second filter is multiplied again by $e^{-j\Delta\omega t}$ and the effects can be observed.

- ***How do you simulate the sampling time shift in practice?***

  After the convolution with the match filter, every sample is shifted using the $circshift()$ MATLAB function. As the sampling frequency is $F_s = MF_{symb} = \frac{M}{T_{symb}}$, shifting a sample by n elements positions is equivalent to time shifting it by $t_0 = \frac{n}{F_s} = \frac{nT_{symb}}{M} = \epsilon T_{symb}$. It is therefore necessary to choose the right $\frac{n}{M}$ ratio to simulate the desired t_0.

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

  The simulated E_b/N_0 ratio is chosen such that the time error is smaller than 2% of T_{symb} for the synchronisation algorithms to work properly. The Figure 14 and 15 show the impact of the noise on the time and frequency acquisition algorithms for different values of N and K. For example, by considering $N = 20$ and $K = 8$, the E_b/N_0 ratio should be greater than 4dB in order to avoid an algorithm malfunction. The impact of the noise on the Gardner algorithm is also measured and a value of E_b/N_0 greater than 4dB is also found.

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

  By observing results in Figures 13a and 13b, it is noticed that a longer pilot length leads to a better estimate of the ToA. This can be intuitively interpreted as the fact that if the pilot is not long enough, it has higher chances to be confused with a sequence of symbols at a random place in the message. However, for a E_b/N_0 ratio around $6dB$ or more, one can afford to take only a length of $N = 20$, since a longer pilot length increases processing time and the throughput.

  After data acquisition, there remains a residual CFO that still makes the phase vary linearly (added to the carrier phase offset induced in the channel). The phase drift can be tackled by linearly interpolating the phase between two pilot sequences. The length of the pilot is then appropriately chosen so that the phase can be estimated properly. At the same time the data sequence should not be too long so that there is no ambiguity in the phase estimation (if the drift has exceeded 2π during the sequence time). There comes a compromise between pilot and data sequence length.

4.5.2 Communication system

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

  As the Gardner's algorithm is robust to CFO, it should be applied first. Then the frame and frequency acquisition can be applied to correct the CFO and detect the time of arrival.

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

  The Gardner algorithm uses the fact that if there is no time shift, symbols are sampled at the right place and the midway sample between two symbols is at zero. This is not the case anymore when samples are shifted in time, and the algorithm uses the slope of the line joining two symbols and the sign of the midway sample. If the slope and the midway sample have the same sign, this means the error positive (samples are shifted on the right compared to the no time shift case). If they are of opposite sign, the error is negative. This observation is repeated through the whole signal and the error estimated is fedback to be updated each iteration. Each iteration, the midway sample value is then multiplied by the value of the slope with some gain κ and the result is added to the estimate computed at the previous step. If there is no symbol transition, the slope is close to zero and the estimate computed does not significantly update the estimate value. The total estimated shift is returned at the end of the loop. During the process the signal is progressively corrected and downsampled at the same time (rejecting midway values undesired anymore). It is shown that the longer the corrected sequence, the closer the algorithm gets to the real shift error.

- ***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 correlates the signal received to the pilot only. This is however sensitive for a CFO corrupted signal (because the pilot in the received signal has been impacted by the CFO as well). The differential cross-correlation takes into account the phase induced by the CFO. To do this, it correlates the received signal by multiple pieces of the pilot of different sizes k. If $k = 0$, the CFO is not taken into account and is not relevant to the algorithm output. So, $k \in [1, K]$, where K is called the cross-correlation averaging window.

- ***Are the frame and frequency acquisition algorithms optimal? If yes, give the optimisation criterion.***
  The original idea of cross-correlating the signal with the pilot and multiple replicas of it shifted in frequency comes from the maximum likelihood criterion, which is optimal:

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

  However, the implementation of the above criterion is very complex because it implies to cross-correlate the symbols with multiple replicas of the pilot sifted in frequency by possible values of the CFO. This implies a very high complexity 2D search of the ToA and CFO and therefore the differential cross-correlation was used instead, which is less complex. The differential cross-correlation algorithm is not optimal because it omits to compute the signal power term that would appear in the development of the above equation (it is ”near optimal”).