UNIVERSITÉ LIBRE DE BRUXELLES

Bruface Brussels Faculty of Engineering

VRIJE UNIVERSITEIT BRUSSEL

DVB-S2 Communication Chain

Modulation and coding

Denguir Anass
Noel Luca
Phan Trong Quy

Engineering Science

Contents

List of Figures

Chapter 1

Introduction

DVB-S2 communication

The Digital Video Broadcasting-Satellite (DVB-S) standard was released in 1993 and specifies the communication for the broadcasting of television to the homes. This standard is based on quaternary phase shift keying (QPSK) modulation and convolutional forward error correction (FEC). In order to have an idea of how this communication chain works, a implementation of this particular type of communication has been made in Matlab. The Figure 1.1 shows the main parts of the communication chain and will be fully detailed in the following chapters.

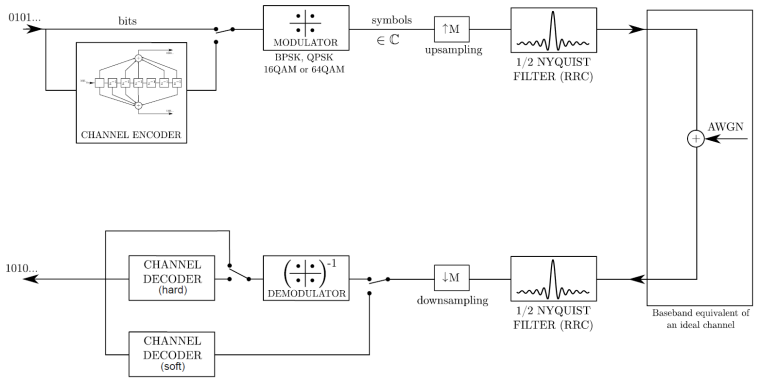


Figure 1.1: DVB-S2 communication chain

Chapter 2

Optimal communication chain over the ideal channel

2.1 Symbol mapping

The first step in order to implement the DVB-S2 communication system is to map the bit stream, that have to be sent to the receiver. This consists of decomposing the binary data in packets of fixed length (i.e the number of bits per symbol) and associate each chunk of bits to a given symbol in a constellation diagram using a specific modulation coding. The modulation codes supported by DVB-S2 systems are the following: BPSK, QPSK, 16QAM and 64QAM. To implement a communication chain, some trade-offs between data rate and noise sensitivity have to be made. For example, the higher the number of bits per symbol, the higher the information rate but the higher the bit error rate. Indeed the number of constellation symbols increases with the number of bits per symbol, however the constellations are closer one to each other, leading to a higher probability of corrupted symbol by noise. Indeed at the receiver side, the demapping bloc associates each noise-corrupted symbol to the closest constellation point following the maximum likelihood criterion. This last step allows to reconvert the binary data.

2.2 Nyquist filter

The interest of the Nyquist filter rises when one wants to increase the symbol rate on the communication chain. As the bandwidth of each symbol increases, an overlapping of their spectrum occurs. The main consequence of this is the observation of inter-symbol interference in the time domain. The Nyquist filter avoids inter-symbol interferences (ISI) by shaping the symbols with a raised cosine filter. The two main properties of this filter are : the limited communication bandwidth and the cancellation of the inter-symbol interferences. The ISI cancellation is illustrated in Figure 2.1. At each symbol period, the amplitude of the symbol currently generated is maximal while the other symbols vanish. Hence, the receiver has to sample the filtered signal at the symbol rate to perfectly recover the signal. The Nyquist filter implemented in this project has a 1 MHz cutoff frequency and a 0.3 roll-off factor.
The SNR in the communication chain can be maximized by using a matched filter. It consists of splitting the Nyquist filter into two root raised cosine filters (RRC filters). One is applied at the transmitter side $g(t)$ while its matched filter $g(-t)$ is applied at the receiver side. The

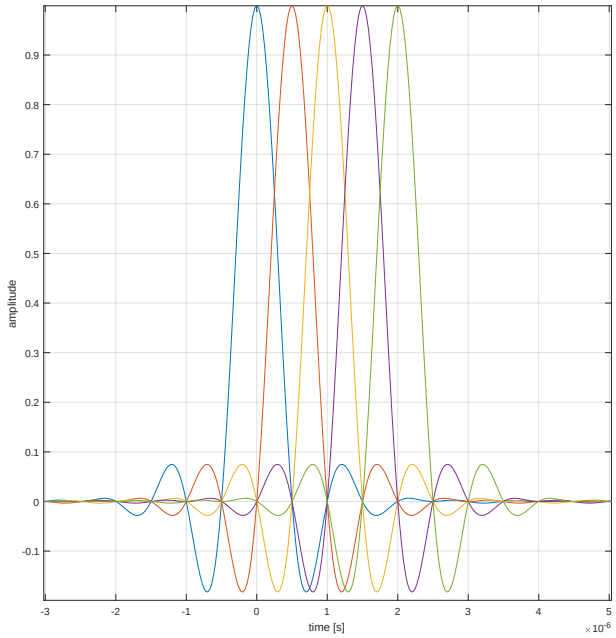


Figure 2.1: Cancellation of the inter-symbol interference ($Tsymb = 0.5\mu s$)

convolution between the two matched filters reconstructs the original Nyquist filter $h(t)$.

$$h(t) = g(t) \otimes g(-t) \tag{2.1}$$

Hence, starting from the frequency domain, the root raised cosine filter is defined as follows:

$$g(t) = ifft\ (\sqrt{H(f)}) \tag{2.2}$$

2.3 Noise addition

To implement the communication chain, the consideration of the AWGN is essential. The sent signals are expressed as their baseband equivalent. Therefore, the baseband model for the noise must be used. The complex envelope of the noise n(t) is defined as:

$$n(t) = n_I(t) + jn_Q(t) \tag{2.3}$$

Where $n_I(t)$ is the in-phase lowpass component and $n_Q(t)$ is the quadratic lowpass component. The white noise is supposed to have a zero mean and a PSD of value N_0. As a consequence, the real and imaginary part of the envelope have a zero mean and a PSD of $2N_0$. This is due to the independent nature of the real and imaginary part. The power of the noise is expressed as follows :

$$P_{N_0} = 2N_0F_s \tag{2.4}$$

Where F_s is the sampling frequency, which is at least equal to the double of the symbol frequency F_{symb}. In practice, the PSD of the noise is not known. However, it can be computed from the

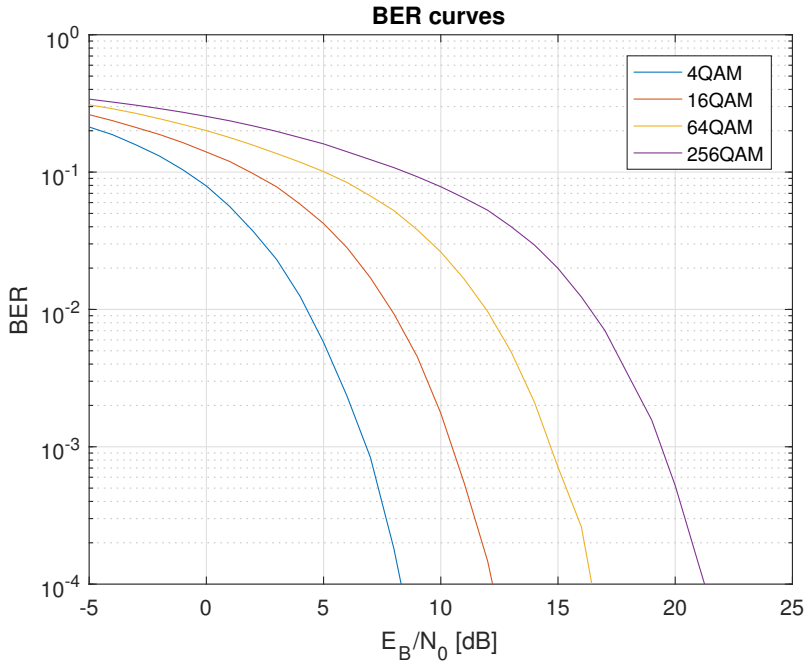


Figure 2.2: BER vs SNR for different modulation types

knowledge of the SNR and the bit energy E_b by evaluating equation 2.5.

$$N_0 = \frac{E_b}{SNR} \tag{2.5}$$

The bit energy is calculated by dividing the signal power P by the bit rate R (see equation 2.6). Therefore, the PSD of the noise N_0 can be deduced by fixing the value of the SNR on the channel.

$$E_b = \frac{P}{R} = \frac{\sum_{k=-\infty}^{\infty} |s(kT_s)|^2}{N_{bits}/T_s} \tag{2.6}$$

As the power of a bandpass signal is equal to the power of its complex envelope divided by two, the bit energy computed in equation 2.6 has to be divided by 2 in the simulation (which operates in baseband). As the noise power is known, the noise of the channel can be added to the signal s. It is generated following a Gaussian distribution of zero mean and variance $P_{N_0}/2$. The Bit Error Rate can be computed to measure the impact of E_b/N_0 on the communication. This is illustrated in Figure 2.2 where the evolution of the BER is plotted for different types of modulation. One can see that for low values of the SNR, the BER is almost 50% which is the worst case. This means that the noise is so dominant that there is no way to distinguish a 1 from a 0. As expected the BER decreases when the SNR increases. Notice that the BER decreases faster for small modulation orders. This is due to the fact that the constellation points are more spread in the complex plane, which reduces the probability to detect a wrong symbol.

2.4 Questions

1. **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 bandpass implementation of the system would require a much higher sampling frequency since we are dealing with narrow-band signals: the carrier frequency is much higher than the bandwidth.

2. **How do you choose the sample rate in Matlab?**
   The sampling rate is chosen following the Nyquist Criterion which ensures that aliasing effects can be avoided when the sampling rate is at least twice the symbol frequency. In practice, an oversampling factor is introduced in *MATLAB* in order to increase the filter resolution.

3. **How do you make sure you simulate the desired $\frac{E_b}{N_0}$ ratio?**
   The SNR factor $\frac{E_b}{N_0}$ is imposed in the *MATLAB* code and the PSD of the noise N_0 is deduced from the SNR and the bit energy E_b using the following relation:

$$N_0 = \frac{E_b}{E_b/N_0} \tag{2.7}$$

4. **How do you choose the number of transmitted data packets and their length?**
   The number of bits sent to the communication channel has to be a multiple of the number of bits per symbol (i.e the data packet's length).

5. **Determine the supported (uncoded) bit rate as a function of the physical bandwidth.**
   As the cut-off frequency F_{cut} of the RRC filter is 1MHz, the symbol frequency F_s is chosen as twice this frequency, namely 2 MHz. The bit rate R can then be deduced as follows:

$$R = F_s * Nbps \tag{2.8}$$

6. **Explain the trade-off communication capacity/reliability achieved by varying the constellation size.**
   One can intuitively understand that working with a higher constellation will lead to a higher information rate but with also a higher bit error rate as shown on the Figure 2.2. This is expected since the number of constellation symbols increases with the number of bits per symbol, however the constellations are closer one to each other, leading to a higher probability of corrupted symbol by noise.

7. **Why do we choose the halfroot Nyquist filter to shape the complex symbols?**
   In order to reduce the impact of the noise on the signal and maximize the SNR, matched filters are usually used. This is why the Nyquist filter has been split into two RRC. Also, the Nyquist filter allows to cancel out the inter-symbol interference and to limit the signal to the desired frequency spectrum.

8. **How do we implement the optimal demodulator? Give the optimisation criterion.**
   The optimal demodulator is implemented by using the RRC filter at the transmitter and its matched filter at the receiver. It maximizes the SNR of the received signal by choosing the folded transmitted signal $s(t)$ as a filter: $h(t) = s(-t)$ ($H(f) = S^*(f)$ in the frequency domain). This can be intuitively understood by verifying that the response of the matched filter convolved with the received signal corresponds to the auto-correlation of the filter without delay, which reaches its maximum value at t=0.

9. **How do we implement the optimal detector? Give the optimisation criterion.**
   The optimal detector is implemented using the maximum likelihood criterion:
   $\tilde{\underline{s}}_m^{ML} = \underset{s_m}{\text{maxp}}(\underline{r}|\underline{s}_m)$ which corresponds to minimizing the Euclidian distance between the received symbols and the constellation vectors of a given modulation.

Chapter 3

Low-density parity check code

3.1 Hard Decoder

In order to ensure a more reliable communication, parity bits are added in the transmitted messages. To do that, the data to be transmitted are divided in chunks of 128 bits. Each chunk of data is prefixed with 128 additional parity check bits to form a coded bit frame of 256 bits. These extra bits are generated with a sparse parity check matrix H of size 128×256. At the reception, the coded data are decoded using an algorithm based on the Tanner representation of the matrix H (see Figure 3.1). In the Tanner Graph, two types of nodes can be distinguished: the check nodes (c_{nodes}) and the variable nodes (v_{nodes}). The number of c_{nodes} corresponds to the number of parity check bits (here 128) while the v_{nodes} correspond to the received bits (the coded word of length 256). The connections between the c_{nodes} and the v_{nodes} correspond to a 1 in the parity check matrix. The LDPC hard decoder algorithm is an iterative algorithm that corrects the bits corrupted by noise through the channel. To do so, each v_{node} transmits to the linked c_{nodes} the value of the bit they believe to be correct. The c_{nodes} correct the value of each v_{node} by computing the parity check equation without taking into account the message previously transmitted by the targeted v_{node}. Afterwards, the v_{nodes} decide the correct received bits $\hat{u}$ based on a majority vote on received c_{nodes} messages. This process is repeated until the error is corrected (if equation 3.1 is satisfied) or the maximum number of iteration is reached.

$$\hat{u}H^T = 0 \tag{3.1}$$

One can see in Figure 3.2 that for small value of the SNR, the coded data have a higher bit error rate than the uncoded data. This is due to the fact that errors on the parity check bits involve a lot of bit errors because the hard decoder will converge to a completely wrong value. However, when the SNR is high enough, the coded words are less erroneous than the uncoded ones since the correction is more accurate. This pattern between coded and uncoded data is shown in Figure 3.2 for 3 different types of modulation. Naturally the BER decreases faster for "small" modulation orders as the euclidean distance between each symbol is bigger, which reduces the impact of the noise at the reception.

Figures 3.3 represent the evolution of the BER with the number of iteration allowed in the hard decoder algorithm for different types of modulation. As expected, for high SNR the BER decreases with the number of iterations. However, after a threshold number of iterations the BER will not decrease anymore because the algorithm will converge before the iteration counter ends.

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

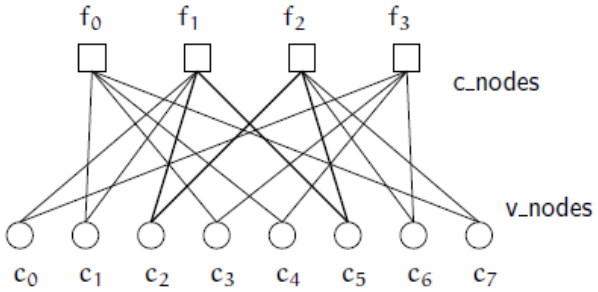


Figure 3.1: Tanner Graph of a parity check matrix H of size 4×8

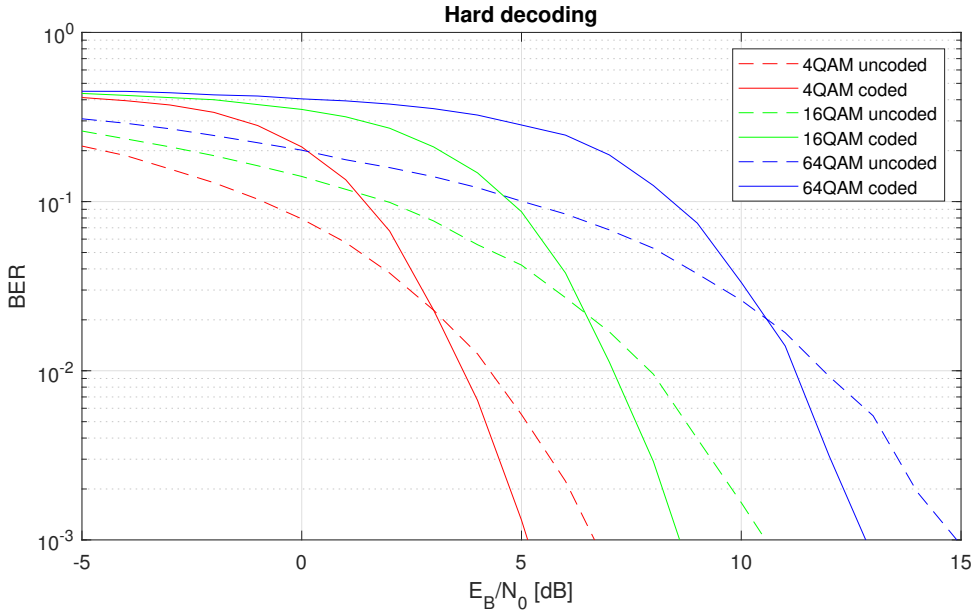
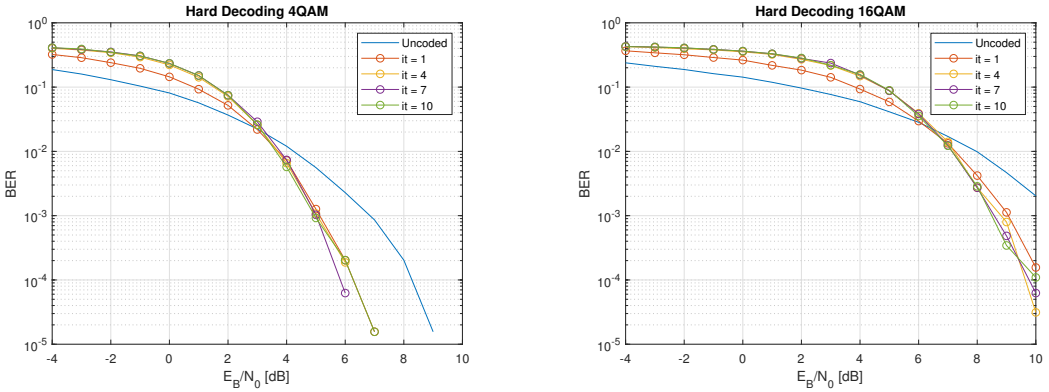


Figure 3.2: BER vs SNR : parameter = modulation



(a) 4QAM (b) 16QAM

Figure 3.3: BER vs SNR for 4 and 16QAM: parameter = iteration

3.2 Soft Decoder

The principle of the soft decoder algorithm is based on belief propagation. The idea of this algorithm is the same as the hard decoder one. But, instead of exchanging binary data, the nodes communicate a probability which depicts the reliability of the node. For the BPSK and the 4QAM cases, the initial message $q_{ij}(0)$ of each v_{node} c_i to a connected c_{node} f_j is simply the probability that c_i is 0 knowing the received symbol r_i. This probability 3.2 takes into account the Gaussian corruption of the noise around each symbol r_i in the complex plane.

$$q_{ij}(0) = P(c_i = 0|r_i) = \frac{1}{1 + e^{\frac{2r_i}{\sigma_w^2}}} \tag{3.2}$$

The response $r_{ji}(0)$ of c_{node} f_j to the v_{node} c_i is computed with the Galager's formula 3.3. It calculates the probability that there is an even number of 1 among the v_{nodes} except c_i (i.e. $C_{j\backslash i}$), which is the probability that the response $r_{ji}(0)$ returns 0. This corresponds to the parity check equation in the hard decoder case.

$$r_{ji}(0) = \frac{1}{2} + \frac{1}{2} \prod_{i' \in C_{j\backslash i}} (1 - 2q_{i'j}(1)) \tag{3.3}$$

At each step of the algorithm, each v_{node} c_i updates its message $q_{ij}(0)$ to c_{node} f_j according to the responses of the c_{nodes}, except the c_{node} f_j it is responding to (i.e. $F_{i\backslash j}$) (see equation 3.4). In order to determine the binary value of each v_{node} found by the algorithm, a soft decision $Q_i(0)$ is computed (see 3.5). If the soft decision is bigger than 0.5, the detected bit is 0 else it is a 1. As the hard decoder algorithm, this process is repeated until the bit errors are corrected (3.1) or the maximum number of iteration is reached.

$$q_{ij}(0) = K_{ij} P(c_i = 0|r_i) \prod_{j' \in F_{i\backslash j}} r_{j'i}(0) \tag{3.4}$$

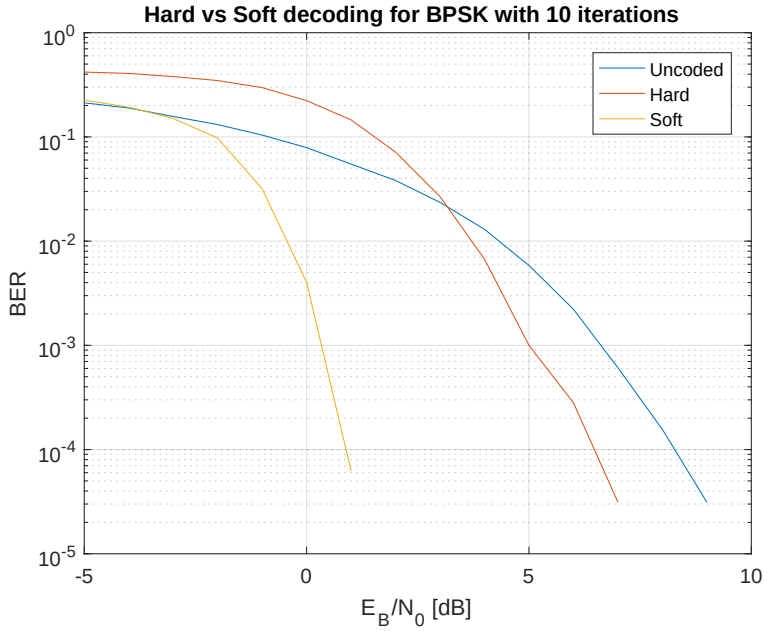


Figure 3.4: BER vs SNR for BPSK: Hard decoder vs Soft decoder

$$Q_i(0) = K_{ij} P(c_i = 0|r_i) \prod_{j \in F_i} r_{ji}(0) \tag{3.5}$$

Figure 3.4 compares the BER curves of data coded following the two presented algorithms. It appears that the soft decoder outperforms the hard decoder algorithm. The performance of the two algorithms are put into perspective with the uncoded case.

3.3 Questions

1. **When building the new BER curves, do you consider the uncoded or coded bit energy on the x-axis?**
   The coded bits are considered to compute the SNR since these are the ones corrupted in the communication channel.

2. **How do you limit the number of decoder iterations?**
   The decoder stops iterations when $uH^T = 0$. This means that the errors are fully corrected. A condition to stop the iteration is to chose a maximal number of iteration. This number is chosen depending on the BER and SNR.

3. **Why is it much simpler to implement the soft decoder for BPSK or QPSK than for 16- QAM or 64-QAM?**
   The soft decoder is based on probabilities. The probabilities for BPSK are based on the sign of the real part of the symbol. For the QPSK, the first bit is determined by the sign of the real part while the second bit is determined by the sign of the imaginary part. The maximum likelihood is simplified in this case. With QAM, implementing a decoder is more difficult as the constellations are bigger.

4. **Demonstrate analytically that the parity check matrix is easily deduced from the generator matrix when the code is systematic.**
   The generator matrix has the form :

   $$G = [P|I] \tag{3.6}$$

   Th parity-check matrix H is orthogonal to G :

   $$G.H^T = 0 \tag{3.7}$$

   The solution for H is given by :

   $$H = [I|P^T] \tag{3.8}$$

   $$[P|I].[I|P^T]^T = 0 \tag{3.9}$$

5. **Explain why we can apply linear combinations on the rows of the parity check matrix to produce an equivalent systematic code.**
   As the vectors forming the parity check matrix H span a subspace orthogonal to the one spanned by generator matrix G, the modulo-2 sum of any vectors in the parity check subspace remains in it. Therefore, applying linear combination on the rows of H guarantees that the parity check matrix spans the same subspace and thus produces an equivalent systematic code.

6. **Why is it especially important to have a sparse parity check matrix (even more important than having a sparse generator matrix)?**
   The more non-zero elements in the parity check matrix, the more connections between the nodes in the Tanner graph to decode the message. As the hard and soft decoding algorithm

are both based on successive computations through these connected nodes, increasing the number of zero elements in the parity check matrix lowers the complexity of the decoding algorithm (as it lowers the number of connected nodes).

7. **Explain why the check nodes only use the information received from the other variable nodes when they reply to a variable node.**
   The belief propagation algorithm is based on the hypothesis that exchanged messages between check nodes and variable nodes must be stochastically independent. This implies that the exchanged information to a given variable node V_j is done considering that the information provided by the other variable nodes $V_{i/j}$ are true (with a degree of reliability in the case of soft decoding).

Chapter 4

Time and frequency synchronization

4.1 Problems encountered

During transmission, synchronization errors occur. These errors must be taken into account during the simulations and before establishing a communication. Different types of synchronization error are identified and caused by different factors :

- the limited crystal accuracy at the receptor causes a carrier frequency offset and a sampling clock offset (respectively CFO and SCO);
- a random carrier phase and time shift occur between receiver and transmitter due to their different physical location.

All the impact of synchronization errors are studied in the following except the SCO. Figure 4.1 illustrates the synchronization errors mentioned above.

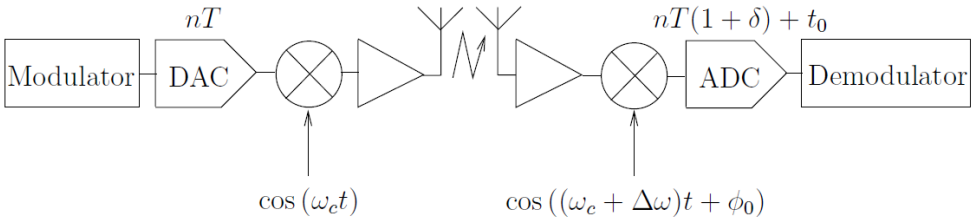


Figure 4.1: Synchronization errors

4.2 Synchronization errors

4.2.1 Phase Offset

The phase offset is introduced during the propagation of the symbols through the channel. It corresponds to a rotation of the constellation : $y[n] = e^{j\phi_0} I[n]$ where $I[n]$ is the sent symbol. This offset has been simulated and is illustrated in Figure 4.2.

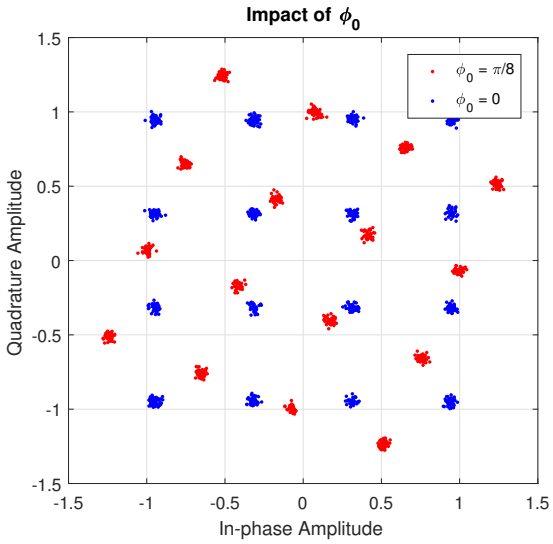


Figure 4.2: Impact of the phase offset ϕ_0

4.2.2 Carrier Frequency Offset

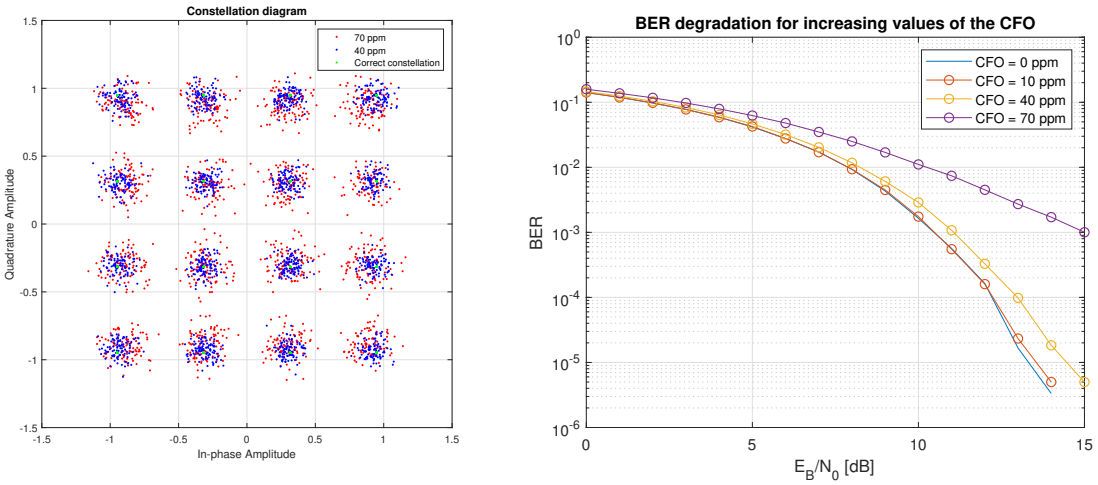
Two impacts can be shown after introducing the CFO in the simulation. Firstly, a linearly varying phase error is present. Secondly, a mismatch between the RRC at transmitter and receiver can be introduced and will cause ISI. The rotation and the clustering of symbols are present even without the presence of noise. Equation 4.1 models the impact of CFO :

$$r(t)e^{j\Delta\omega t} = \sum_n I[n]g(t-nT)e^{j\Delta\omega t} = \sum_n (I[n]e^{j\Delta\omega T})(g(t-nT)e^{j\Delta\omega t})_{|_{t-nT}} \tag{4.1}$$

During simulations, the linearly varying phase error was canceled by multiplying back the symbols by $e^{-j\Delta\omega t}$ in order to only observe ISI. Constellation diagram and BER curves are depicted in Figure 4.3

4.2.3 Sample time shift

A time shift is present during communication due to the difference in location of the receiver and transmitter. This synchronization error is due to the fact that the signal takes time to travel between the two edges of the channel. This problem leads to a wrong sampling time at the reception. The impact of the time shift on the BER curves is shown in Figure 4.4.



(a) Constellation diagram

(b) BER

Figure 4.3: Impact of CFO

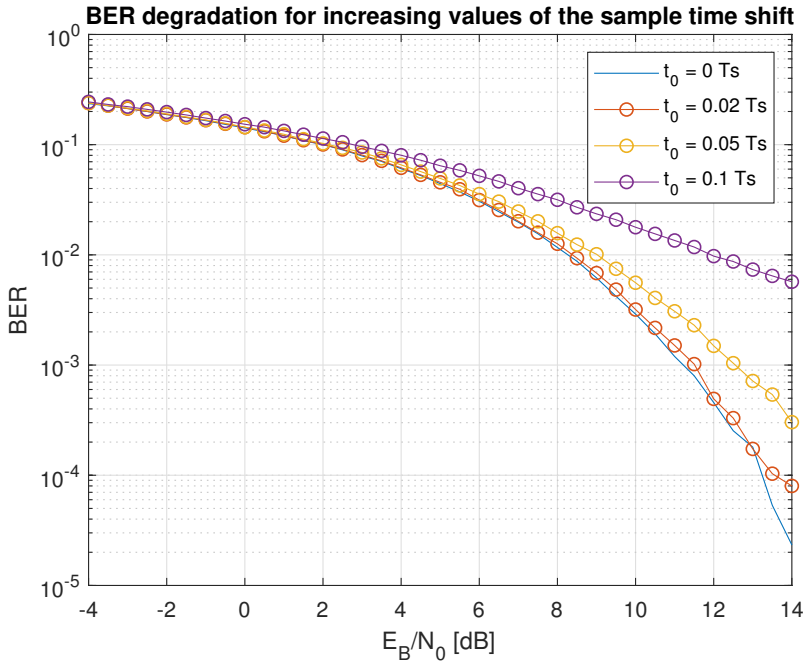


Figure 4.4: Impact of time shift on BER curves

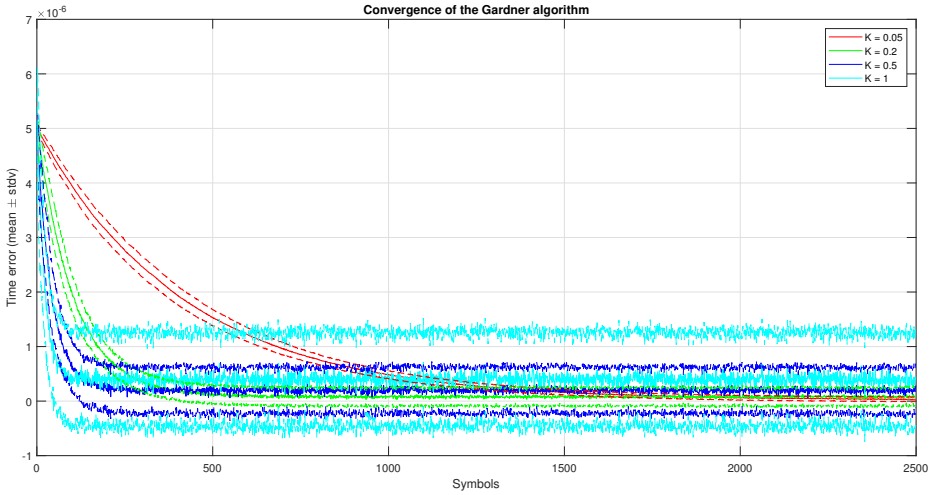


Figure 4.5: Convergence of Gardner algorithm for different values of K

4.3 Synchronization correction

4.3.1 Gardner algorithm

To decrease the effects of the sample time shift, the Gardner algorithm is used. This is a Non-Data Aided feedback algorithm which compensates time shift and track any time shift variation. This algorithm is robust to any other synchronization error so it is applied first. The algorithm estimates the time shift error by taking the difference between two symbols and multiplying this difference by an estimate of the midway symbol. The calculated factor is then weighted by a factor K and then added to the current time error.

$$\hat{\epsilon}[n+1] = \hat{\epsilon}[n] + K \cdot \Re\Big[y_{\hat{\epsilon}[n]}[n-0.5](y^*_{\hat{\epsilon}[n]}[n] - y^*_{\hat{\epsilon}[n-1]}[n-1])\Big] \tag{4.2}$$

The performances of this algorithm depends on the value of K parameter. This is illustrated in Figure 4.5. One can see that the higher the parameter K, the faster the algorithm finds the time shift. However, high values of K make the algorithm unstable. This is illustrated by a higher standard deviation around the tracked time shift. Hence, a trade off must be made for the choice of K. It must be high enough to have a fast convergence and small enough to avoid instability. The time error can then be compensated by using equation 4.2.

The influence of the CFO on the time-shift tracking algorithm is depicted in Figure 4.6. For a realistic CFO (between 2 and 10ppm), the Gardner algorithm undergoes low to no influence on the convergence of the time shift. If the simulation works with a high value of CFO, the algorithm makes the time error converge slower. The noise can also influence the convergence of the algorithm. High level of noise can disturb the computation of the time error. It can be concluded that the Gardner algorithm is robust to CFO regardless of the observation with high CFO values. Its robustness can be explained by the fact that the time shift error is calculated in small time intervals. Hence, the time shift correction is applied before the CFO correction.

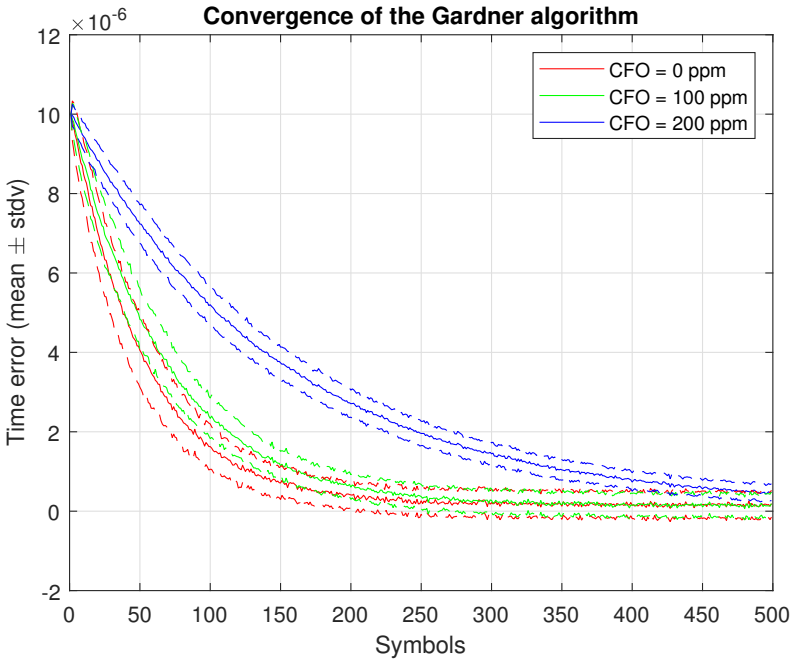


Figure 4.6: CFO Convergence

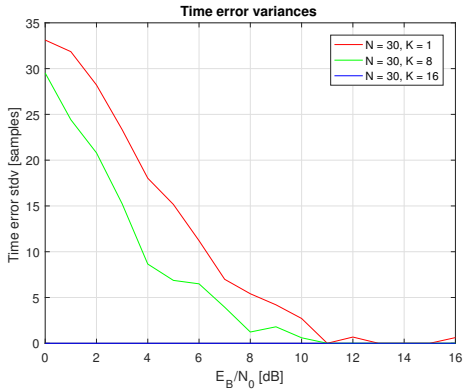
4.3.2 Frame and frequency acquisition

One problem encountered in practice is that the receiver is unaware of the moment when the transmitter sends a message. In addition, several sources of synchronization errors are present in the communication chain. The receiver must re-synchronize regularly.

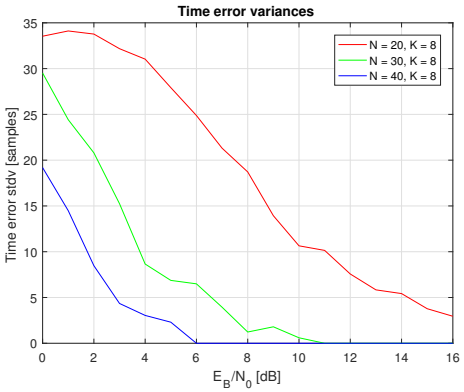
In order to perform a synchronization, a pilot of fixed length is placed at the beginning of each packet to inform the receiver where a sequence begins. The detection of this pilot is done by computing the differential cross-correlation. Note that a simple correlation is expensive in computation hence the choice of a differential cross-correlation. The time of arrival (ToA) is given by the differential cross-correlation. This corresponds to the index of the first bit of the pilot in the received sequence. The computation of the output metric is necessary to obtain the ToA :

$$D_k[n] = \frac{1}{N-k} \sum_{l=k}^{N-1} (y^*[n+l]a[l])(y^*[n+l-k]a[l-k])^* \tag{4.3}$$

In this equation $y[n]$ is the received sequence, $a[n]$ is the known pilot, k is the length of a window and N is the length of the pilot. The ToA $\hat{n}$ is computed by taking the argument that maximize the summation of output metric. The evolution of the estimation error of the ToA with SNR is shown in Figure 4.7 for different combinations of the pilot length N and the window size K. As expected, the algorithm converges faster for high SNR. Moreover the ToA estimation error is reduced when N is increased because bigger pilots are easier to identify (see 4.7(b)). Similarly, decreasing the window size K increases too much the probability of matching windows of the

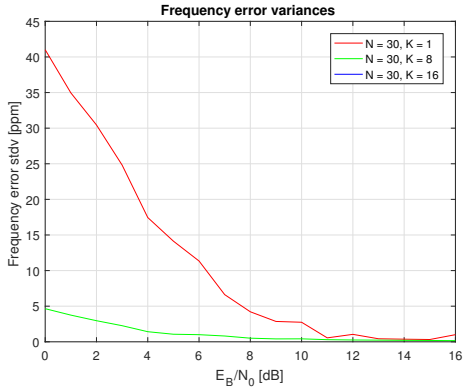


(a) K

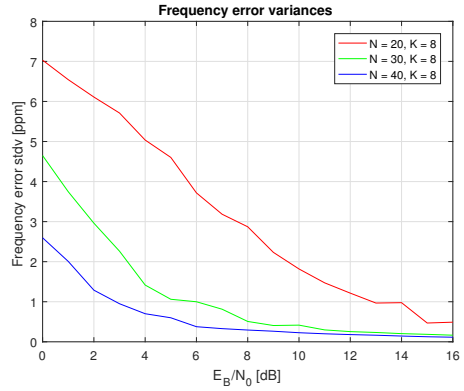


(b) N

Figure 4.7: Impact of parameter N and K on the ToA estimation



(a) K



(b) N

Figure 4.8: Impact of parameter N and K on the CFO estimation

signal which can lead to a bad localization of the pilot (see 4.7(a)).

$$\hat{n} = \arg\max_{n} \sum_{k=1}^{K} |D_k[n]| \tag{4.4}$$

The estimated CFO is presented below in equation 4.5, where T is the symbol duration. The evolution of the CFO is depicted in Figure 4.8 for different values of the parameters N and K. The evolution of the CFO has the same behavior as the ToA (for the same reasons). Therefore, The CFO can be compensated by its estimated value $\hat{\Delta f}$.

$$\hat{\Delta f} = -\frac{1}{K} \sum_{k=1}^{K} \frac{\angle D_k[\hat{n}]}{2\pi k T} \tag{4.5}$$

4.4 Questions

1. **Derive analytically the baseband model of the channel including the synchronization errors.**
   The received symbols $r(t)$ are demodulated at the receiver by multiplying the Rice components of the received signal by a sine and cosine at the carrier frequency. Then, a low pass filter is used in order to remove the signal part at frequency $2\omega_0$.

$$\begin{cases} r_1(t) = \Re[r(t)] = (s_1 cos(\omega_0 t) + s_2 sin(\omega_0 t)) \cdot cos((\omega_0 + \Delta\omega)t + \Phi_0) * LPF \\ r_2(t) = \Im[r(t)] = (s_1 cos(\omega_0 t) + s_2 sin(\omega_0 t)) \cdot sin((\omega_0 + \Delta\omega)t + \Phi_0) * LPF \end{cases} \tag{4.6}$$

   Where $s_1(t)$ and $s_2(t)$ are respectively the real and the imaginary part of the sent symbol $s(t)$. ω_0 is the carrier frequency, $\Delta\omega$ is the CFO and Φ_0 is the phase shift on the original signal. Since the low pass filter removes the high frequency components of the demodulation, one obtains the following equations:

$$\begin{cases} r_1(t) = \Re[r(t)] = \frac{1}{2} s_1 cos(\Delta\omega t + \Phi_0) - \frac{1}{2} s_2 sin(\Delta\omega t + \Phi_0) \\ r_2(t) = \Im[r(t)] = \frac{1}{2} s_1 sin(\Delta\omega t + \Phi_0) + \frac{1}{2} s_2 cos(\Delta\omega t + \Phi_0) \end{cases} \tag{4.7}$$

   After converting the last equations in the polar form, the synchronization errors are represented as follows:

$$r(t) = s(t) e^{j\Delta\omega t + \Phi_0} \tag{4.8}$$

2. **How do you separate the impact of the carrier phase drift and ISI due to the CFO in your simulation?**
   To keep the impact of ISI only, it is possible to multiply the received downsampled signal by $e^{-j\Delta\omega nT}$ after the downsampling to remove the effect of the CFO on the carrier phase drift.

3. **How do you simulate the sampling time shift in practice?**
   The oversampled signal with an oversampling factor of M is downsampled at the receiver. A time shift $t_0 = n.T_s$ is introduced by shifting indexes to simulate a sampling error.

4. **How do you select the simulated Eb=N0 ratio?**
   If no synchronization errors are present, depending on the type of modulation one can choose a E_b/N_0 to obtain a BER below a wanted threshold. If there are synchronization errors, the E_b/N_0 value must be chosen in order to ensure a good performance of the Gardner algorithm, frame and frequency acquisition. The error of the sampling time should be less than % 2 of symbol duration. This error corresponds to minimum 4 dB for E_b/N_0.

5. **How do you select the lengths of the pilot and data sequences?**
   The pilot's length should be long enough to not match a data in the transmitting sequence. The pilot length affects the accuracy during the computation of ToA and CFO. The frame acquisition is perfect and CFO remaining error is inferior to $2ppm$ if $N \geq 20$, $K \geq 8$. The length of the pilot, however, has to be optimized. The frequency error standard-deviation depends on the length of the window. The length is then chosen depending on the wanted BER and corresponding SNR.

6. **In which order are the synchronization effects estimated and compensated. Why?**

   - Gardner algorithm is first used in order to track the sampling time shift error because the CFO and phase shift errors can be corrected only if the input sequences are sampled correctly. Moreover, as Gardner algorithm is robust to CFO, it must be applied first;
   - Data acquisition to find the frame starting time using differential cross-correlation;
   - CFO estimation and compensation;
   - Interpolation of the phase drift between pilot sequences.

7. **Explain intuitively how the error is computed in the Gardner algorithm. Why is the Gardner algorithm robust to CFO?**
   In the Gardner algorithm, the output of the matched filter is sampled at the symbol frequency and the midway value between each samples is interpolated. If there is no time error and there is a symbol transition, the midway value should be zero. A time shift error will make this midway value non zero. The sign and the amplitude of the error will be updated by computing the weighted product of the midway sample value and the signal slope. Hence the error can be corrected by constantly estimating the new value of the time shift error after each sample. This algorithm is robust to CFO because it is computing local errors. Therefore the CFO does not corrupt that much these local evaluations.

8. **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 cross correlator uses as input the received sequence with multiple replicas of a pilot, each shifted in frequency by the possible CFO values. The differential cross-correlator is defined as the cross-correlation between the sequence and its time shifted homologous. Without noise, the CFO can be extracted. In noisy communication system, the CFO can be estimated. If k=0, the differential correlation is equal to :

$$D_0[n] = \frac{1}{N} \sum_{l=0}^{N-1} (y^*[n+l]a[l])(y^*[n+l]a[l])^* \tag{4.9}$$

   This represents the average power of the window. It does not give any useful information concerning the ToA.

9. **Are the frame and frequency acquisition algorithms optimal? If yes, give the optimisation criterion.**
   The frame and acquisition algorithms used for the project are based on the differential correlation optimization equation. This is not an optimal algorithm since this does not maximize the maximum likelihood criterion:

$$(\hat{n}, \hat{\Delta\omega}) = \arg\max_{n} p(\underline{y}[n]|\underline{a}, \Delta\omega) \tag{4.10}$$

   The solution that maximizes this criterion is computationally complex and necessitates a 2D exhaustive search. That is why a differential correlation was used.

Chapter 5

Conclusion

In this project, a DVB-S2 satellite communication has been simulated. In order to ensure a reliable communication, a Nyquist filter has been studied and designed. The latter allowed to avoid inter-symbol interferences. Afterwards, a parity check coding algorithm has been implemented to reduce the impact of the noise on the channel. Finally, synchronization algorithms have been applied on the data in order to correct synchronization errors that occur on realistic communication chains. In conclusion, this project allowed us to have a deeper understanding of how a communication chain works and the problematics that occur in its design.