DCS final assignment

Rui L. V. Oliveira

January 28, 2020

Abstract

In this study we analyse the performance characteristics of different modulations in the presence of additive white gaussian noise or multipath channels. We analyse the best modulations and modulation orders for each scenario and demonstrate orthogonal frequency-division multiplexing's capability ti resist to intersymbol interference up to the guard interval.

1 Introduction

It is rather intuitive that different (de)modulation schemes have different performance when subject to similar channel conditions. In this study, we compare the performance of M-PSK, M-QAM and M-FSK modulations in the presence of additive white Gaussian noise (AWGN), with and without using error correction techniques, and over multipath channels. But first, let us define and explore all these things.

1.1 PSK modulation

The first modulation scheme studied is phase-shift keying modulation, usually shortened to PSK modulation. Formally, it works by encoding one bit or groups of bits, called symbols, in phase-shifts of a carrier wave. In essence, a PSK signal can be described by the following formulation:

$$s_{\text{PSK}}(t) = \cos\left(2\pi f t + 2\pi n/M\right) \tag{1}$$

where M is the number of different symbols being keyed, called the modulation order, and $n = \{0, \ldots, M-1\}$ is the symbol being keyed, which would be itself a function of time.

While this formulation describes well the behaviour in RF signals, at baseband we have to be more careful, and resort to a complex exponential notation. In this case, we can describe PSK as:

$$s_{\text{PSK}}(t) = e^{2\pi j n/M} \tag{2}$$

This leads to an interesting interpretation of PSK: what we are doing is choosing one out of a series of discrete points in the unit circle in the complex plane. To these points we call a constellation. The modulation order, M is equal to the number of points in this circle.

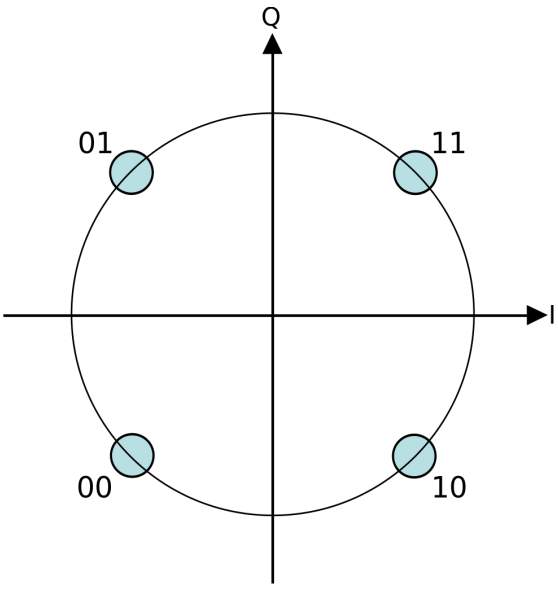


Figure 1: Constellation diagram for QPSK (4-PSK) with Gray coding [1]. Licensed under the Creative Commons Attribution-Share Alike 3.0 Unported.

It is also important to discuss how to assign digital information to these aforementioned symbols. The natural thing to do is to assign each symbol to a particular bit pattern. This leads that modulation orders should be chosen as powers of two and, for a modulation order M each symbol carries $\log_2(M)$ bits. In assigning each symbol to a particular bit pattern, it is usual to use Gray coding, meaning that each symbol differs from it's neighbours by only one bit. Figure 1 illustrates these concepts.

Finally, it is worth discussing the process of getting the symbols back from the transmitted signal, at the receiving end. If the signal is contaminated with noise (and it will generally be), the phase of

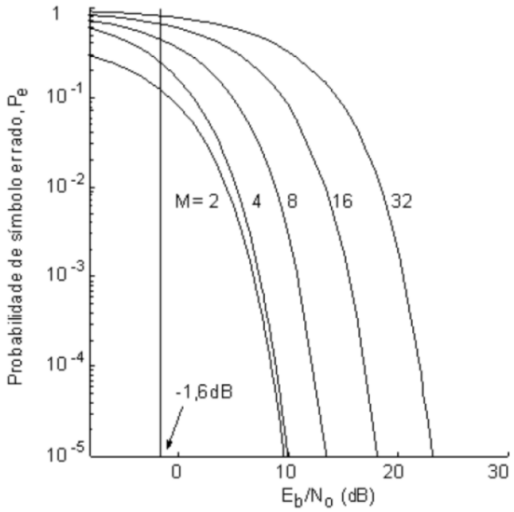


Figure 2: SER for PSK with different modulation orders. Adapted from the class documents [2].

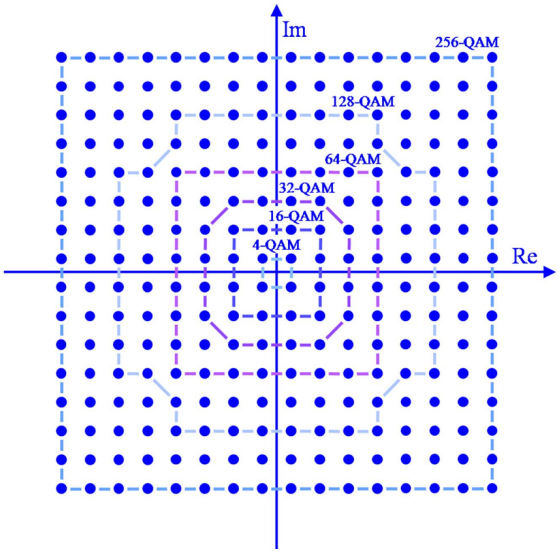


Figure 3: Typical constellations for QAM signals. Adapted from [3].

the signal gets harder to determine. In particular, it's not expected that the phase matches exactly the one of any of the symbols. A simple method to solve this is to choose a symbol whose phase is closest to the one measured.

1.1.1 Error rate vs noise for PSK

Figure 2 describes the probability of symbol error (SER) for PSK modulation. These curves are fairly easy to understand. If we look back into the constellation, we'll notice that the frequency step between adjacent symbols is smaller for higher modulation orders. If the modulator decides on the symbol by choosing whichever is closest, a certain phase shift caused by noise has a higher probability to cross a decision boundary when these decision regions are smaller. For completion sake, the approximate analytical expression for these curves is given by [2]:

$$P_e = 2Q\left(\sqrt{\frac{2E_s}{N_0}}\sin\frac{\pi}{M}\right) \tag{3}$$

1.2 QAM modulation

Quadrature amplitude modulation, or QAM, can be seen as an extension to PSK (or the latter as a particular case of the former). Instead of selecting points in the unit circle for our constellation, we select our points "arbitrarily" in the complex plane, leading to both phase and amplitude modulation. Let S_{QAM} be the set of symbols (points in the complex plane) that define the constellation. A QAM signal in baseband would be described as:

$$s_{\text{QAM}}(t) = \left|S_{\text{QAM}}[n]\right| e^{2\pi j \angle S_{\text{QAM}}[n]} \tag{4}$$

where $n = \{1, \ldots, M-1\}$, is the index of a symbol in the set. It is usual to prefer square QAM shapes, leading to modulation indexes that are powers of 4, albeit any constellation can be defined. Figure 3 illustrates typical QAM constellations.

Similarly to what is done with PSK, in a QAM demodulator the decision on the symbol during demodulation can be done by choosing whatever is closest to the measured signal. Since both amplitude and phase are at play in QAM, the use of IQ demodulators is of great help, transforming this problem into a problem of two phases.

1.2.1 Error rate vs noise for QAM

The behaviour of bit and symbol error rated for QAM is very similar to the one for PSK, for the same reasons. The bit error probability for QAM modulation is given by:

$$P_{b,\text{QAM}} = \frac{4}{\log_2 M}\left(1-\frac{1}{\sqrt{M}}\right)Q\sqrt{\frac{3\log_2 M}{M-1}\frac{E_b}{N_0}} \tag{5}$$

1.3 FSK Modulation

Frequency-shift keying, or FSK, works a bit different from the previous ones. Instead of encoding information into multiplying a carrier signal by a

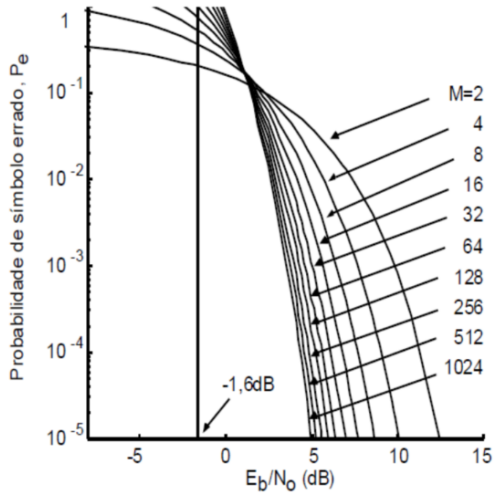


Figure 4: SER for FSK with different modulation orders. Adapted from the class documents [2].

complex-valued factor, we instead frequency-shift the carrier signal altogether. At baseband level, an FSK signal can be described as:

$$s_{\text{FSK}}(t) = e^{2\pi j \times n/M \times \Delta f_{\text{MAX}}} \tag{6}$$

where $n = \{1, \ldots, M-1\}$ is a number corresponding to the symbol to the transmitted, M is the modulation index (and maximum number of symbols) and Δf_{MAX} is the maximum frequency shift of the signal.

1.3.1 Error rate vs noise for FSK

Figure 4 describes the probability of symbol error for FSK modulation. It's interesting to see that with FSK the probability of symbol error is actually smaller for higher order modulation orders, given a positive SNR. While this may seem surprising at first, there is an explanation. We can imagine that, similarly to a Fast Fourier Transform (FFT), the frequencies between the carrier frequency and the maximum signal frequency signal are divided into N frequency"bins". For high modulation order, these bins are ever smaller. This means that they will "integrate" a smaller bandwidth of noise, resulting in less noise energy in the bin. At the same time, the power of the signal transmitting the symbol itself is the same, resulting in higher "contrast". The drawback to use high modulation orders is the need to very accurate frequency matching between the transmitter and the receiver. The approximate analytical expression for these curves is given by [2]:

$$P_e \leqslant (M-1)Q\left(\sqrt{\frac{E_s}{N_0}}\right) \tag{7}$$

1.4 A note on pulse shaping filters

Doing simple PSK or QAM modulation per the equations given results in sharp transitions of the carrier wave. These sharp transitions yield high (theoretically infinite) bandwidth of the carrier signal in these transients. For the multipath simulations especially, this is very important, as time delays were implemented with an FFT-based algorithm (these is discussed in the appropriate section). Therefore, for these simulations, it is important to apply a bandwidth limiting filter. It is usual to use a pair of matched raised-cosine filters for these modulations. This is what I did. MATLAB's own function to implement the raised-cosine filter also has the advantage of oversampling the modulated symbols, which was convenient.

For FSK, there are problems with the frequency shifts between symbols are not phase-continuous. Fortunately, MATLAB's own FSK modulation function provides a phase continuity option, which was used.

1.5 OFDM modulation

Orthogonal Frequency-Division Multiplexing modulation, or OFDM modulation for short, works differently from the modulation schemes seen previously. As the name implies, OFDM is not a modulation scheme *per se* but rather a multiplexing technique. On OFDM, digital data that was previously modulated by other modulation schemes is multiplexed in frequency in a way that ensures orthogonality. This means that subcarriers of the OFDM signal do not suffer from cross-talk, and therefore there is no intersymbol interference. The orthogonality condition is ensured if the subcarrier spacing is given by:

$$\Delta f = \frac{k}{T_S} \tag{8}$$

where T_S is the symbol duration, and k is a positive integer. $k = 1$ yields the lowest positive bandwidth.

Using subcarriers allows a system to achieve data throughputs equivalent to a very high bandwidth single carrier system of other modulation types (like PSK and QAM) but using subcarriers that themselves have relatively low baud rates. This is the key for making OFDM resistant to multipath channels (to be discussed in a section below). With the use of a guard interval, i.e., an extra time delay between symbols, all the interference problems due to multipath effect that generate time delays up to that interval are simply ignored. This trade-off is not important in OFDM

modulation since, as discussed, each subcarrier's baud rate is relatively small so the effect of adding a guard interval doesn't result in dramatic changes of data throughput.

1.6 AWGN

We will start by justifying Additive White Gaussian Noise as a good model for a single path communication channel. The additive characteristic of AWGN models the radiation that is present at the receiving end that does not relate to the signal being transmitted. Without any further information, there is no reason to assume that this noise would dependant in the frequency in any way. This is what is said to be "white" noise. Finally, the source of this model is modelled as a random source. By the central limit theorem, it's justifiable that the distribution for this noise would be a Gaussian distribution. In fact, instead of saying that there is a single noise source, the noise at a receiver can be modelled as an arbitrarily large number of individual noise sources, giving strength to the central limit theorem argument. In the context of this work, this AWGN will be generated in a way that produces a determined signal-to-noise (SRN) ratio at the receiver's "input". To compute that, we first compute the average sample energy of the discrete-time transmitted signal, $x_{\text{TX}}[n]$, of length Ns samples:

$$E_{\text{s}} = \frac{1}{N\text{s}} \sum_{n}^{N\text{s}} \left| x_{\text{TX}}[n] \right|^2 \tag{9}$$

We then use this value and our target SNR to compute a noise spectral density, N_0:

$$N_0 = \frac{E_{\text{s}}}{\text{SNR}} \tag{10}$$

White Gaussian noise becomes fully characterized by it's variance, σ_{AWGN}:

$$\sigma_{\text{AWGN}} = \sqrt{N_0} \tag{11}$$

We'll be performing only baseband simulations, using complex-valued signals. With this in mind, our signal at the receiver, $x_{\text{RX}}[n]$, becomes:

$$x_{\text{RX}}[n] = x_{\text{TX}}[n] + \frac{\sigma_{\text{AWGN}}}{\sqrt{2}} \left(N[n] + jN[n] \right) \tag{12}$$

where $N[n]$ describes a concretization of a random variable with normal distribution and j is the imaginary unit.

The work presented relies heavily on MATLAB's "Communications Toolbox" [4], which happens to include a function for this purpose, `awgn` [5]. However, I did notice this function only adds real-valued noise, leaving the imaginary part of signals intact. For the sake of achieved a more correct behaviour, I used a custom function to add AWGN [6], included as the `add_awgn_noise.m` file.

1.7 Multipath channels

Radio communication channel is said to be a multipath channel where the signal transmitted reaches the receiver by two, or more, path. Effects of multipath channels include coherent interference, that when destructive, can cause fading of radio channels. Other effect is intersymbol interference, since the secondary channels with be delayed relative to the dominant channel, the receiver will have instants where it will receive conflicting information.

While models for OFDM channels can have some complexity, the scope of this work focus mainly on the effect of channel delays. The simulation model will be discussed in the results section, but the note is that it is admittedly simplistic, but it allows for the analysis of the effects intended.

1.8 Cyclic codes

The first of the two error correction techniques studied in this work is the use of cyclic codes. Cyclic codes are a particular type of linear block codes and the characteristic of cyclic means that all code vectors of the block code are obtained from each other by circular shifts.

The two important parameters for the context of these simulations are n, the code length (meaning the length of the "output" of the encoder) and k, the message length (meaning the length of the "input" of the encoder). Let then d be the minimum distance of a cyclic code. That code can then detect up to $d-1$ errors and correct $(d-1)/2$ errors.

1.9 Convolutional codes

Convolutional codes work differently from cyclic codes. In convolutional codes error correction, we add memory into the system, meaning that the output for each input sequence depends not only on the present input but on the past of the system.

This means that convolutional codes are described by three parameters: n and k as before, and now K, meaning the memory of the system. The convolutional encoder then works basically as a state machine.

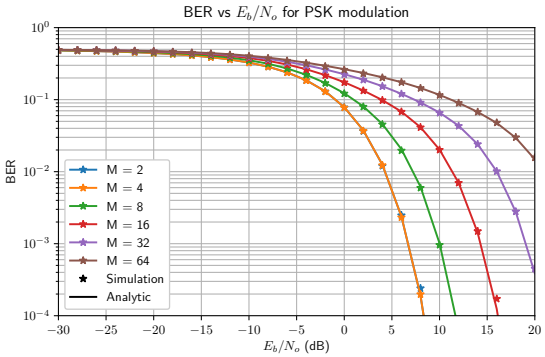


Figure 5: BER for M-PSK modulation.

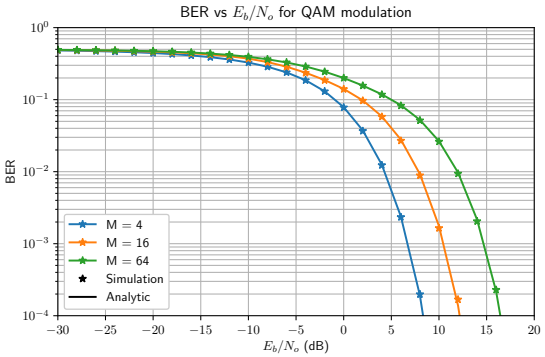


Figure 6: BER for M-QAM modulation.

2 Simulation and results

2.1 Exercise 1

In question 1 we are asked to explore different modulation schemes, at different modulation orders, analysing the bit error rate in relation to the power of AWGN. This simulation is on the script `simulation_awgn.m` and makes heavy use of the function defined on `modem_awgn.m`. For PSK, QAM and FSK modems, the function modulates a random binary stream, upsamples the data using a Raised Cosine filter for PSK and QAM, and for each target E_b/N_0, it applies the proper noise power, demodulates the signal and calculates the final BER.

For the sake of comparability, the independent variable in the simulation is the relation E_b/N_0. However, the function `add_awgn_noise` takes a target SNR as the argument. The equation that transforms between these quantities is given by:

$$\text{SNR dB} = \frac{E_b}{N_0}\ \text{dB} + 10\log_{10}\left(\log_2(M)\right) - 10\log_{10}(N_s) \quad (13)$$

where N_s is the number of samples per symbol.

Figure 5 shows the results for PSK modulation. The theoretical curves were computed using the MALTAB function `berawgn`. We see good agreement between the curves. We see that 2-PSK (also called BPSK for binary) and 4-PSK have pretty much the same BER curve. For higher order modulations, we see the BER get progressively worse for the same E_b/N_0, as expected and discussed in the introduction.

Figure 6 shows the results for QAM modulation. The conclusions are very similar to the case for PSK modulation.

Finally, figure 7 shows the results for FSK modulation. Here, we see a good agreement for lower modulation orders, but higher order modulations, especially for $M = 32$ gets worse. For the lower modulation orders we see the behaviour discussed in the introduction: the BER is actually better for higher modulation orders. This trend continues in the theoretical curves, but for some reason that I couldn't figure out, practical results do not confirm this.

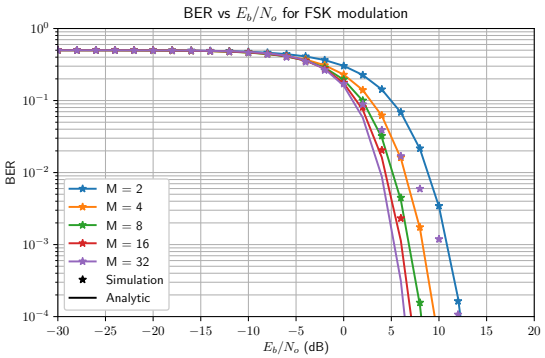


Figure 7: BER for M-FSK modulation.

Now that we have seen how each modulation scheme behaves in according to its modulation index, it becomes interesting also to see how do they compare amongst themselves. Figure 8 shows exactly this. Keep in mind that not all modulation orders where used for all modulation schemes. In the way that the simulation was made, the same modulation order will yield the same bit rate, but not necessarily the same bandwidth. If the latter is not a factor, this plot can help us choose the best modulation technique, depending on the modulation order one intends to use. We can see that for low modulation orders, like $N = 2$ and $N = 4$, where QAM and PSK are nominally the same thing, they perform better than FSK. However, for high modulation orders, FSK takes the lead as a best modulation strategy.

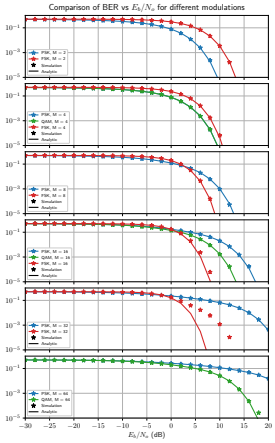
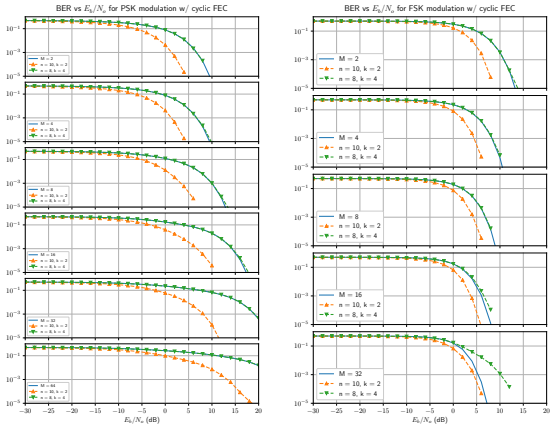
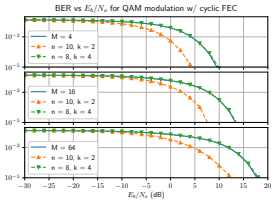


Figure 8: BER for different modulations.



(a) BER for PSK modulation w/ cyclic coding.

(b) BER for FSK modulation w/ cyclic coding.



(c) BER for QAM modulation w/ cyclic coding.

Figure 9: Results for the cyclic encoder.

2.2 Exercise 2

In exercise 2, we are prompted to study the effect of both cyclical and error correction codes. We are to apply this correction methods to the previously studied modulation schemes and see the variation in BER.

2.2.1 Cyclic codes

For the cyclic code study, I used two pairs of (k, n) parameters: $k = 8$, $n = 4$, and $k = 10$, $n = 2$. The polynomials for these parameters were generated using the `cyclpoly` function. Then, a parity-check matrix is generated using the function `cyclgen`. Finally, this parity-check matrix is fed into the function `syndtable` to produce the syndrome decoding table. MATLAB's own `encode` and `decode` functions provide the interface to use these elements into encoding the message and decoding the received bits.

For the $k = 8$, $n = 4$ pair of parameters, the minimum distance is 2, so the maximum number of detectable errors is 1 and there is not the possibility of correction, so we don't expect any improvements in relation to the previous case. For $k = 10$ and $n = 2$, we do expect improvements: the minimum distance is 5, so it's possible to detect up to 4 errors and decode up to two errors, so we do expect some improvements.

Figures 9a, 9c, and 9b, show the results for the different modulations.

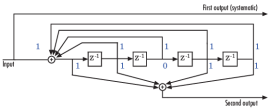
Exactly as predicted, the $k = 8$, $n = 4$ parameters do not provide any benefit, while $k = 10$ and $n = 2$ do provide significantly better error rates at the same SNR due to the error correction.

2.2.2 Convolutional codes

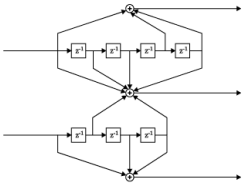
To use convolutional codes for encoding messages in MATLAB, it's easy to use the `poly2trellis` function. It converts convolutional code polynomials to trellis description. The the function `convenc` encodes the data. Since there is no trivial convolutional decoder, the function `vitdec` provides a solution to decode this data using the Viterbi algorithm.

Since there is no trivial way to design a specific convolutional encoder, I settled on MATLAB's own suggestions [7]. These are structures for a 1/2 Feedback Convolutional Encoder, a 2/3 Feedforward Convolutional Encoder and a 1/2 Feedforward Convolutional Encoder. The diagrams of the first of the two are in figure 10.

The results are presented in figure 11. We see that in general we get improved performance regarding the BER by using these encoders. However, it's worth noting how the 2/3 feedforward convolutional encoder lags behind the theoretical curve for no correction. This means that for some particular values of SNR, the presence of this encoder makes thing worse. Also, funnily enough, we for some modulation orders in FSK, this same encoder can't reach a state of zero correction. I was not able to figure out the reasons behind this behaviour.



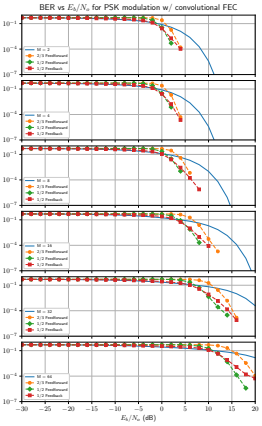
(a) 1/2 Feedback Convolutional Encoder.



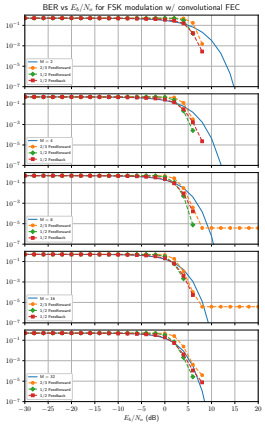
(b) 2/3 Feedforward Convolutional Encoder

.

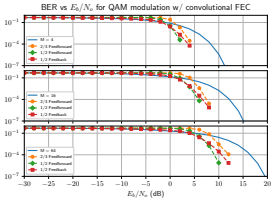
Figure 10: Two of the convolutional encoders studied. Adapted from [7].



(a) BER for PSK modulation w/ convolutional coding.

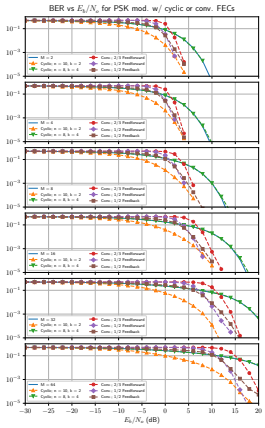


(b) BER for FSK modulation w/ convolutional coding.

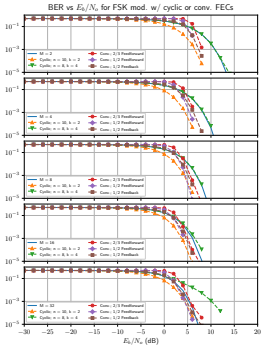


(c) BER for QAM modulation w/ convolutional coding.

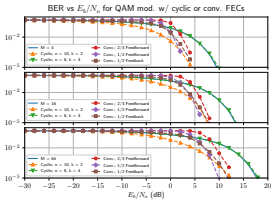
Figure 11: Results for the convolutional encoder.



(a) BER for PSK modulation w/ different coding schemes.



(b) BER for FSK modulation w/ different coding schemes.



(c) BER for QAM modulation w/ different coding schemes.

Figure 12: Combined results for the different modulations.

2.2.3 All together

For the sake of comparison, figure 12 shows the combined results of both cyclic and convolutional encoding for each modulation. Results show how careful design is important: cyclic encoding can be both useless of better than the convolutional encoders studied. It all depends on the careful design and analysis of the systems, demonstrating the importance of this kind of exploratory work.

2.3 Exercise 3

In exercise 3, we shift the focus from AWGN channels to multipath channels. In particular, we are to look for the effect of intersymbol interference due to interference between paths on a multipath channel. For the sake of having a controlled experiment, I did not use any randomness in these simulations. What I did was to add a delayed version of the signal to itself before decoding, with a power adjustment. This power adjust is made enough for the effect of ISI to be visible but not enough to drown the receiver's SNR. The results are presented in figure 13. For scale, a symbol time is 4 µs. As a hint for the study of ODM, it's

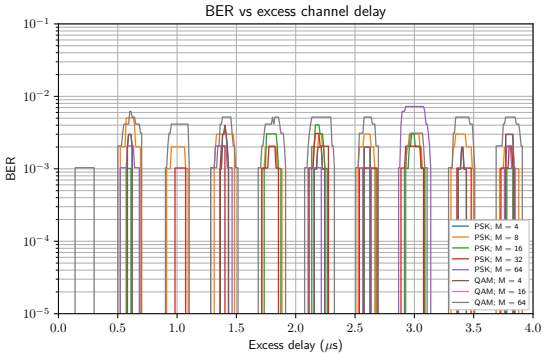


Figure 13: BER in relation to excess time delay.

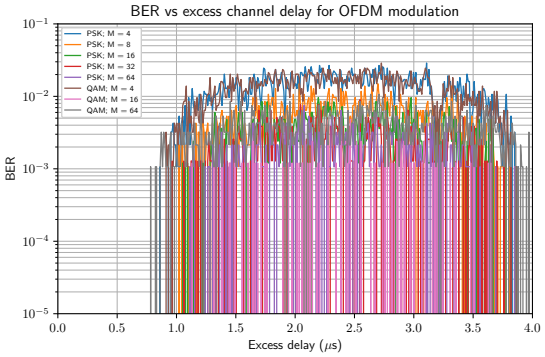


Figure 14: BER in relation to excess time delay for OFDM modulation.

important to mention the 0.8 µs mark, as this coincides with the guard interval we are going to use. We see that all modulations suffer from intersymbol interference before this mark. It's also worth noting that this simulation only computes two paths. If one was to have the superposition of N paths, the plot would not have just those small zones of interference but basically show a flat erratic behaviour.

2.4 Exercise 4

Now, in exercise 4, we shift the focus to OFDM modulation. The assignment give us parameters to implement this modem. However, MATLAB's OFDM function, like `ofdmmod` do not take into themselves the notion of time, or a sampling frequency. So, I'll explain how this was implemented. I selected the number of subcarriers to be 64, with the first 6 and last 6 left as empty subcarriers. This complies with the requirement of 52 used subcarriers. At the same time, and referring to equation 8, if we take the symbol time to be 3.2 µs (this being the OFDM symbol duration minus the guard interval), it will immediately yield a subcarrier spacing of 312.5 kHz, and the required 20 MHz of total bandwidth. Finally, since the FFT size of the `ofdmmod` function will be 64, then, the required extra 0.8 µs of guard interval correspond to 16 extra samples, which is the parameter that the function takes. Finally, the relation between 64 samples and 3.2 µs indicates the need for a sampling rate of exactly $20\,\mathrm{Msa\,s^{-1}}$. This makes the function that otherwise doesn't use any kind of frequency information specifically to comply with the requested parameters.

The results for the excess time delay can be seen in figure 14. Notice that we can clearly observe that there is no errors up to 0.8 µs exactly, as predicted. For the sake of comparison, figure 15 shows a comparison between using the modulations from previous exercises and using them on subcarriers of an OFDM signal. We can see the improvements in BER up to the guard interval period. Keep in mind the previous discussion on how the traditional plots would admittedly look worse for a channel with more than two paths being compared.

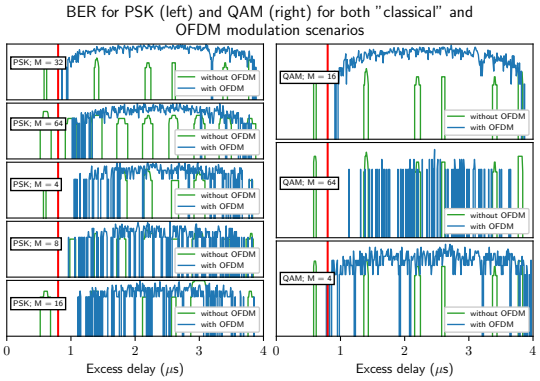


Figure 15: BER in relation to excess time delay, a comparison.

3 Conclusion

Different modulation schemes have different behaviours. It is very important to do a proper analysis of the channel conditions of the application and the necessary data throughput before making any decisions. OFDM modulation provides a good way to protect against channels that suffer from a multipath condition.

References

[1] Splash using w:Inkscape for Wikipedia. (Mar. 5, 2006). Constellation diagram for gray-coded 4psk., [Online]. Available: https://en.wikipedia.org/wiki/File:QPSK_Gray_Coded.svg (cited on page 1).

[2] A. Moura, M. Violas, and J. Cabral, *Documentos das aulas teóricas*, U. do Porto, U. de Aveiro, and U. do Minho, Eds., Sep. 1, 2019 (cited on pages 2, 3).

[3] T. Pfau, S. Hoffmann, and R. Noe, "Hardware-efficient coherent digital receiver concept with feedforward carrier recovery for m-qam constellations", *Journal of Lightwave Technology*, vol. 27, no. 8, pp. 989–999, Apr. 2009 (cited on page 2).

[4] MathWorks. (Jan. 16, 2019). Communications toolbox, [Online]. Available: https://www.mathworks.com/products/communications.html (cited on page 4).

[5] ——, (Jan. 16, 2019). Awgn, [Online]. Available: https://www.mathworks.com/help/comm/ref/awgn.html (cited on page 4).

[6] M. Viswanathan. (Jan. 16, 2019). How to generate awgn noise in matlab or octave (without using in-built awgn function), [Online]. Available: https://www.gaussianwaves.com/2015/06/how-to-generate-awgn-noise-in-matlaboctave-without-using-in-built-awgn-function/ (cited on page 4).

[7] MathWorks. (Jan. 16, 2019). Poly2trellis documentation, [Online]. Available: https://www.mathworks.com/help/comm/ref/poly2trellis.html (cited on pages 6, 7).