Instituto Superior Técnico

Licenciatura em Engenharia Eletrotécnica e de Computadores

Telecomunicações

Relatório de Projecto Laboratorial

Emissor e Recetor BPSK e QPSK

Análise da taxa de erro de bit com o incremento de Ruído Gaussiano Branco Aditivo (AWGN)

Inês Cardoso Paiva, nº 99961
João Barreiros C. Rodrigues, nº 99968
Maria Teresa Ramos Nogueira, nº 100029

Outubro-Novembro 2022

Índice

1 | Contextualização Teórica

Admitindo que o delineamento deste projeto recai sobre a projeção e análise de dois sistemas de modulação (BPSK e QPSK), é relevante fornecer uma noção sucinta do seu desenvolvimento tradicional, de modo a garantir uma base de conhecimento para a compreensão do modulador de constelação[1], utilizado em regime laboratorial:

1.1 | Modulação PSK

> "M-ary phase-shift keying (M-PSK): (...) the carrier amplitude and carrier frequency are both maintained constant, while the carrier phase is keyed between M possible values (...) used to represent [M] symbols."[1]

Procuramos analisar as modulações para qual M toma os valores de 2 e 4, respetivamente, ***B**inary* e ***Q**uadrature Phase Shift Keying*:

1. A modulação BPSK ambiciona codificar os símbolos 0 e 1, consoante a função definida por ramos explicitada infra:

$$s(t) = \begin{cases} \sqrt{\frac{2\mathrm{E}_b}{\mathrm{T}_b}}\cos(2\pi \mathrm{f}_c \mathrm{t}) & \text{for symbol 1} \\ \sqrt{\frac{2\mathrm{E}_b}{\mathrm{T}_b}}\cos(2\pi \mathrm{f}_c \mathrm{t} + \pi) = -\sqrt{\frac{2\mathrm{E}_b}{\mathrm{T}_b}}\cos(2\pi \mathrm{f}_c \mathrm{t}) & \text{for symbol 0} \end{cases}$$

2. A modulação QPSK ambiciona codificar um conjunto de dibits, nomeadamente 00, 01, 10, 11, consoante a função definida por ramos explicitada infra:

$$s(t)_i = \begin{cases} \sqrt{\frac{2\mathrm{E}_b}{\mathrm{T}}}\cos\left(2\pi \mathrm{f}_c \mathrm{t} + \left[(2i-1)\frac{\pi}{4} + \Phi\right]\right) & 0 < t < \mathrm{T} \\ 0 & c.c. \end{cases}$$

Onde $i = 1, 2, 3, 4$; E_b = Energia de bit; $\mathrm{T} = 2\mathrm{T}_b$ = Duração do bit;
e Φ uma constante de fase.

Por observação direta da fórmula matemática dos sinais modulados é de fácil elação que ambos possuem um formato DBS-SC (Sendo QPSK um caso alterado de *Quadrature-Carrier Multiplexing*, produz duas BPSK que subsequentemente apresentam também um formato DBS-SC). O processo de descodificação está então subjugado ao processo de deteção coerente, tal como exemplificado nos esquema seguinte[2]:

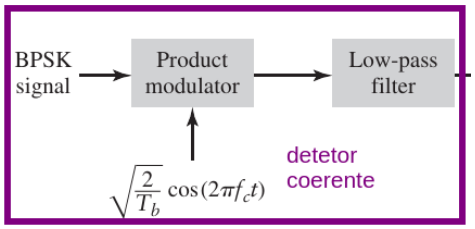


Fig. 1: Detetor coerente anterior ao sistema de decisão de BPSK

[1] Que utiliza uma representação em *Complex baseband*[2].

[2] Por motivos de brevidade não mencionamos o detetor coerente à entrada do desmodulador QPSK, O processo é análogo, realizado agora em dois ramos diferentes com auxílio de um *phase-shifter*.

1.2 | *Constellation Modulator*

Por ventura, o bloco que realiza a modulação em regime laboratorial, omite a existência da portadora piloto dos sinais de banda passante através de um equivalente de *Complex Baseband*:

> "In the passband model, also called as waveform simulation model, the transmitted signal, channel noise and the received signal are all represented by samples of waveforms. Since every detail of the RF carrier gets simulated, it consumes more memory and time. In the case of discrete-time equivalent baseband model, only the value of a symbol at the symbol-sampling time instant is considered. Therefore, it consumes less memory and yields results in a very short span of time when compared to the passband models. Such models operate near zero frequency, suppressing the RF carrier and hence the number of samples required for simulation is greatly reduced."[2]

A forma geral de um sinal de banda passante pode ser rescrita da seguinte forma:

$$\tilde{s}(t) = a(t)cos\left[2\pi f_c t + \phi(t)\right] = s_I(t)cos(2\pi f_c t) - s_Q(t)sin(2\pi f_c t)$$

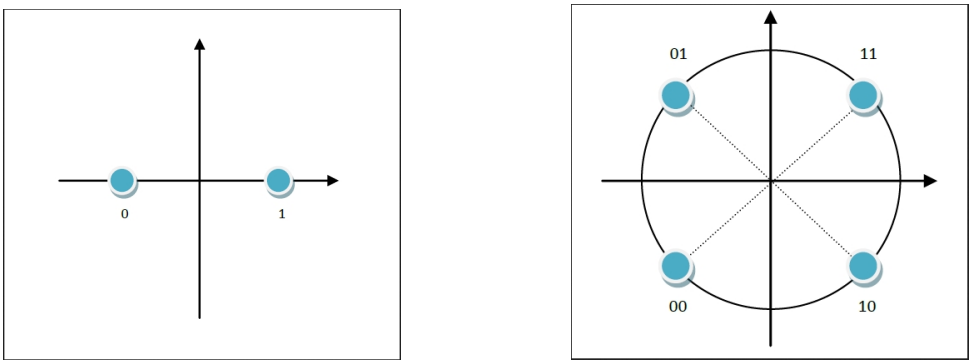
E reconhecendo que as componentes supramencionadas são ortogonais entre si, o sinal pode ser representado em formato complexo:

$$s(t) = s_I(t) + js_Q(t) \qquad \phi(t) = tan^{-1}\left(\frac{s_Q(t)}{s_I(t)}\right)$$

A análise direta das fórmulas deduzidas revelam a colocação no plano complexo dos sinais modulados BPSK e QPSK:

- **BPSK:** Suporta apenas pontos no eixo real, já que $\phi(t)$ toma apenas dois valores distintos, respetivamente 0 e π.
- **QPSK:** Suporta os quatro quadrantes do plano complexo, possui, portanto componente imaginária e $\phi(t)$ garante 4 valores, para cada dibit codificado, já como mencionado anteriormente $\rightarrow$ $\pi/4$, $3\pi/4$, $5\pi/4$, $7\pi/4$.[3]

Neste sentido, os sinais são passíveis de uma abordagem vetorial. Podemos então invocar a representação em constelação:



(a) BPSK (b) QPSK

Fig. 2: Visualização das constelações dos sinais BPSK e QPSK

Onde a distância da origem aos pontos de constelação é dada por:

$$\sqrt{\mathrm{E}_b} = \sqrt{s_I(t) + s_Q(t)}$$

1.2.1 | *Constellation Object*

O *Constellation Modulator* é o responsável pela codificação supramencionada: codifica a informação da modulação a partir de um mapeamento complexo com base nos diagramas de constelação. De forma sucinta, chamamos apenas atenção para os parâmetros de *symbol map* $\mapsto$ *constellation points* que reproduzem o mapeamento símbólico para o plano complexo[3]

- **BPSK**: $[0, 1] \longmapsto [-1 + 0j, 1 + 0j]$
- **QPSK**: $[0, 1, 3, 2] \longmapsto [-1 - 1j, -1 + 1j, 1 + 1j, 1 - 1j]$

Assim, sendo o sinal modulado livre de portadora e possuindo a informação codificada na respetiva constelação, o sinal resultante (em fase para PSK e em fase e quadratura para QPSK), à saída do bloco de constelação é um vetor de coeficientes (representantes dos respetivos bits ou dibits a transmitir), modulados segundo um pulso (RRC) (*summed together*[4]), cujo o número de amostras é estipulado *a priori*.[4].

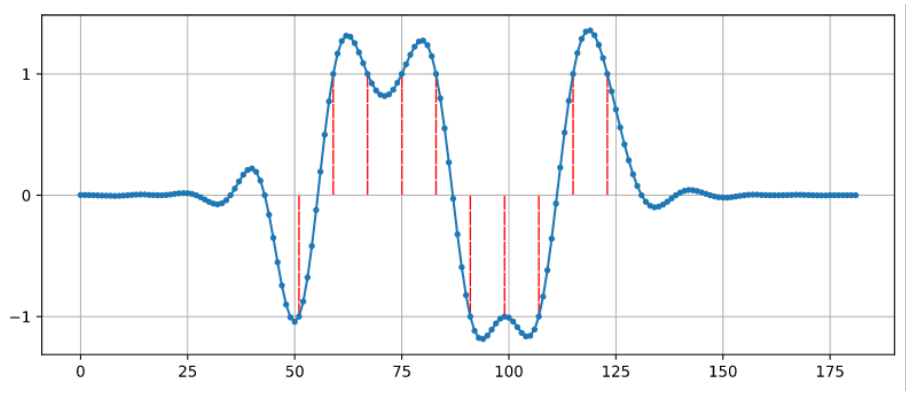


Fig. 3: Soma de pulsos dos coeficientes normalizados BPSK [-1, 1, 1, 1, 1, -1, -1, -1, 1, 1] assinalados a vermelho.

Mediante o parágrafo superior, é relevante mencionar os processos de *upsampling* e *matched filtering*:

1.2.2 | *Upsampling*

Para ilustrar o processo de *upsampling* no bloco de modulação, tomaremos a representação matemática da soma de pulsos ilustrada na figura 3 (BPSK):

$$s(t) = \sum_{n}^{\infty} a_k \mathrm{P}(t - n\mathrm{T}_s)$$

[3]Para cada ângulo, respetivamente, 11, 01, 00, 10.

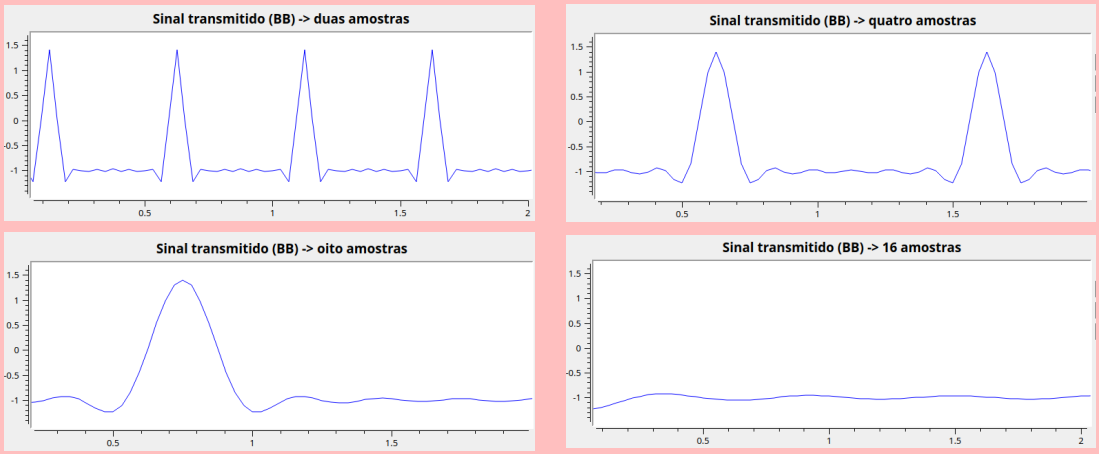
[4]"Por cada byte recebido (...), o módulo *Constellation Modulator* vai extrair e separar os respectivos 8 bits, usando para a emissão de cada bit um pulso com espectro RRC, com [N] amostras por pulso dentro do intervalo de bit de duração T (...)"[5]

Onde a_k são os coeficientes do sinal modulado e $T_s = N/F_m$ onde N é o número de amostras e F_m é a frequência de amostragem. Para efeitos de demonstração iremos assumir que $N = 4$ e $F_m = 32K$, tal como realizado na secção 5.1. do guia de laboratório.

Deste modo $T_s = 0.125ms$, tal como especificado no guia de laboratório, o que implica que o número de amostras irá afetará do pulso de modulação. Poderíamos também realizar o processo de modulação com somente 2 amostras por segundo (apesar de ser desvantajoso para o processo de pulse shaping subsequente[6]):

> "(...) typically we like filtered PSK, and the block does this with the RRC filter. However - a key point is that when filtering the signal, you have bandwidth expansion. So if you use a 35% excess bandwidth, your sample rate must be at least 1.35x larger , or another way to say it is that **you need at least 1.35 samples per symbol to satisfy nyquist. Similarly for 50% bandwidth you'd need 1.5 samples/symbol. The easiest way to deal with this is to just upsample by 2** (...)[7]"

Em regime prático → A alteração do número de amostras por símbolo (subsequentemente por pulso: "(...) separar os respectivos 8 bits, usando para a emissão de cada bit um pulso com espectro RRC" [5]) verifica a contextualização teórica: A utilização de duas amostras produz pulsos de curta duração com aspeto triangular enquanto que para um número de amostras >8 o pulso desenhado é de mais fácil visualização, mas progressivamente de maior duração, como é verificado nas imagens infra.



Admitimos assim um valor de 8 amostras por pulso (valor recomendado no guia laboratorial), para melhor visionamento e obtenção de maior largura de banda do pulso, já que em contexto real:

> "The advantage of (...) more samples per symbol is in the ability to have more relaxed filtering in subsequent stages[6]"

[5] Para fácil visualização dos pulso foi utilizado um vetor de 1 byte (00000001) em repetição

1.2.3 | *Matched Filtering* → *RRC Filter*

Sucintamente, o processo de matched filtering pode ser descrito mediante a seguite citação:

> "We want a low-pass filter in our transmitter to reduce the amount of spectrum our signal uses. But the receiver also needs a low-pass filter to eliminate as much **noise/interference** next to the signal as possible. As a result, we have a low-pass filter at the transmitter (...) and another at the receiver (...).[4]"

Assim, o processo de *matched* filtering procura maximizar o SNR (ruído incutido pelo noise channel face ao sinal transmitido) e eliminar a interferência intersimbólica. Para tal é utilizado um RRC, integrado no modulador de constelação e um FIR filter localizado na pré deteção, com o mesmo formato, de forma a emular o seguinte:

$$\mathbf{P_{RRC}(f)^{1/2} \rightarrow pós\ modulação} \qquad \mathbf{P_{RRC}(f)^{1/2} \rightarrow pré\ deteção}$$

Obtendo o ilustre ***Raised-Cosine Filter*** cuja forma no domínio do tempo é:

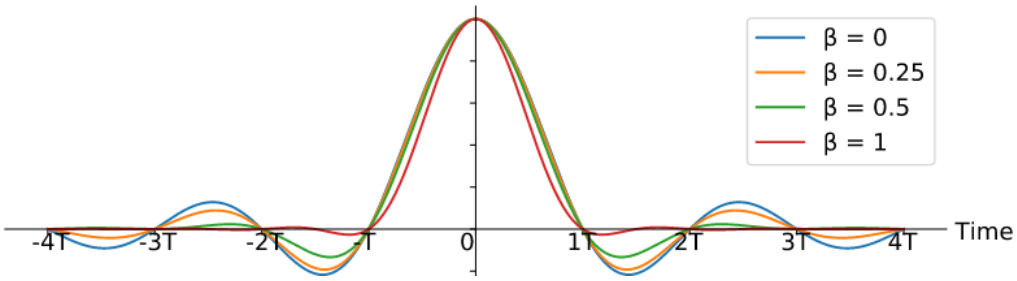


Fig. 4: Forma do pulso RC no domínio do tempo, notar a semelhança com os pulsos observados na secção 1.2.2

1.3 | *Symbol Sync*: *Loop Bandwidth & Ted Gain*

O processo de desmodulação do sinal emitido é realizado em duas etapas, uma de sincronização de símbolo, realizada pelo bloco *Symbol Sink*[6]:

> "Performs the timing synchronization needed so that the signal is sampled at exactly the right moment in time, which is when each symbol/pulse is at its max value (...)"[8]

e outra de conversão para binário, realizada pelo *Binary Slicer*:

> "Slice a float value producing 1 bit output. Positive input produces a binary 1 and negative input produces a binary zero."[9]

A deteção no seio do ruído nas etapas subsequentes do projeto é da inteira responsabilidade do bloco *Symbol Sync*, sendo a sua parametrização de extrema relevância para que sejam omitidos fidedignamente e na sua grande maioria o número de erros por patamar de ruído, alvo do projeto em questão.[6]

1.3.1 | *TED gain*

Embora omitido no guia de laboratório, a parametrização **correta** do *TED gain* é **fulcral** ao bom funcionamento do bloco, já que, caso incorreto, a parametrização do *Loop Bandwidth* deixa de ser simbólica:

> "This value [TED gain] is normally computed by the user analytically or by simulation in a tool outside of GNURadio. This value must be correct for the loop filter gains to be computed properly from the desired input loop bandwidth and damping factor"[8].

Tal como explicitado na documentação do bloco, procurámos calcular analiticamente o ganho correto, com recurso a um programa externo ao ambiente laboratorial *GNU Radio*[7]. O cálculo do ganho é dependente do *rolloff factor* do RRC e do *timing error detector* utilizado, *Maximum likelihood Timing Error Detector (ML-TED):*

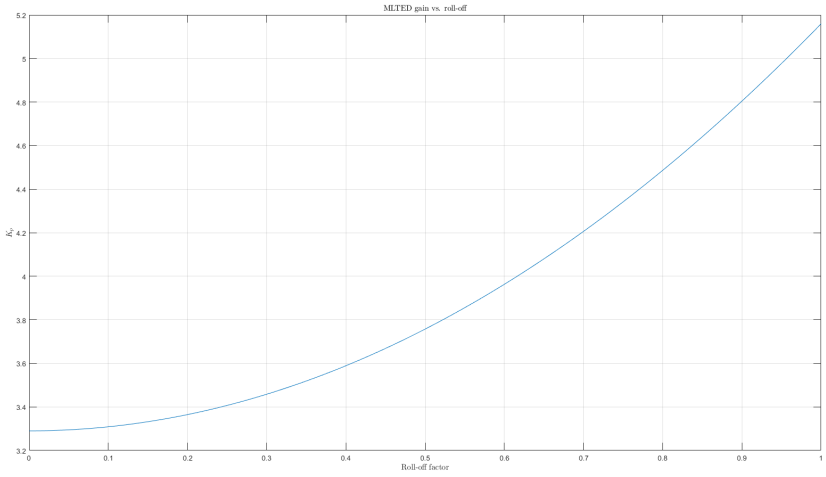


Fig. 5: curva de relação entre *TED gain* e *rolloff factor* para ML-TED. Computado aqui

[6] Esta é a visão sucinta de ambas as desmodulações, o QPSK necessita ainda de um bloco de interpolação dos dados em fase e quadratura, dada a natureza da modulação, como será explicitado na secção xx

Para o valor de *rolloff* especificado no guia de laboratório, 0.75, o valor ótimo encontrado foi de 4.3415. Comparando a aplicação deste ganho com o ganho *default* de 1, a diferença é estapafúrdia.

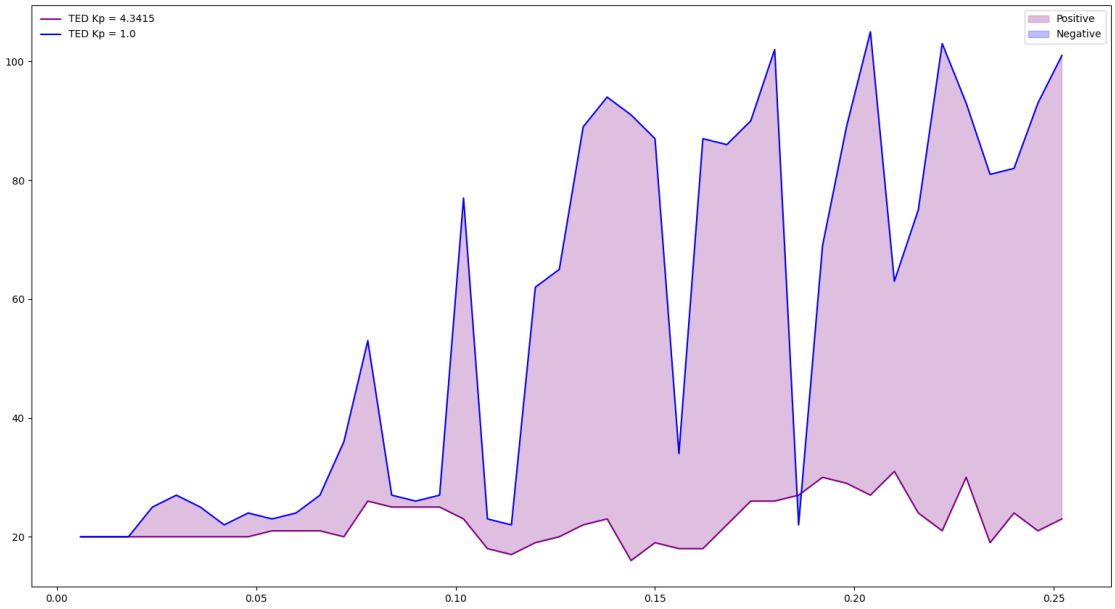


Fig. 6: Comparação da reação do número de erros para um patamar de *noise voltage* = 1 mediante o incremento do Loop Bandwidth para dois ganhos TED diferentes num sistema BPSK.

A curva a roxo, detentora do ganho de 4.3415, para além de garantir maior estabilidade para a gama de valores de *Loop Bandwidth* testados (cujo intervalo se encontra dentro da gama ótima de utilização estipulada pelo *GNU Radio*, como veremos de seguida), garante também maior mitigação do erro face ao ruído, que é exatamente o que pretendemos obter, mediante a alteração do *Loop Bandwidth.*

Usamos, portanto, um TED gain de 4.3415 para toda a obtenção de dados subsequente, já que o observado para o patamar de ruído 1 é generalizado para os restantes.

1.3.2 | *Loop Bandwidth*

Especificado o *TED gain*, podemos abordar livremente a gama de valores avaliados de *Loop Bandwidth* para melhor mitigar o número de erros. Segundo a documentação do *Symbol Sync*:

> "(...) It should nominally be close to 0, but greater than 0. If unsure, start with a number around 2*pi*0.04, and experiment to find the value that works best for your situation."[8]

Admitindo a situação acima, decidimos contabilizar um intervalo de *Loop Bandwidth* de]0;0.25], de onde será possível retirar uma gama de valores ótimos para cada patamar de ruído.

[7] calcTedKp.m

Devemos ainda ter em consideração o tradeoff especificado na seguinte citação:

> "The faster the loop, the faster the acquisition time, and recovery time (...) **However as the BW for the tracking loops increases (...) tracking errors will start to dominate the noise in SNR** (...).
>
> To explain further from the perspective of a trade of Loop BW and SNR (...) Timing noise has components at all frequencies (...), and the lower frequency components of the timing noise (...) will be tracked by the timing loop, and suppressed. The higher frequency components will not be tracked by the timing loop and therefore remain and contribute to noise as part of SNR. (...) **Thus you see the motivation to make the timing loop as fast as possible** (...). However, some of the signal energy of interest will be part of the "noise" being supressed by the timing loop (...) **As the loop BW increases, signal energy is also removed** (...)"[10]

Portanto, procuramos um *Loop Bandwidth* que não reduza a relação sinal-ruído mas que suprima as componentes de alta e baixa frequência do *timing noise*. Em regime laboratorial, o supracitado verifica-se: o número de erros tende a aumentar juntamente com o incremento do *Loop Bandwidth*, para cada patamar de ruído.

2 | Exposição de Dados

2.1 | Comparador de ficheiros

A obtenção de dados para análise foi realizada com recurso a um pequeno *script* em *Python*, que pode ser observado na imagem abaixo:

```
#!/usr/bin/env python3
from subprocess import run
def FileDiff(WhichFile):
    Length = 27*8
    OutpuOffset = 2
    BitError = 0

    #BPSK ou QPSK -> ajustar offset
    if WhichFile == True:
        BufferOffset = 49
    else:
        BufferOffset = 98

    #ler dump binario diretamente do comando xxd sem formatacao
    cmd1 = "xxd -b -c1 bpsk_vector_output.dat |
    cut -d\" \" -f2 | tr -d \"\\n\""

    cmd2 = "xxd -b -c1 bpsk_vector_output_receptor.dat |
    cut -d\" \" -f2 | tr -d \"\\n\""

    data1 = run(cmd1,capture_output=True,shell=True)
    TransmissionData = str(data1.stdout)

    data2 = run(cmd2,capture_output=True,shell=True)
    ReceptionData = str(data2.stdout)

    #contabilizar o numero de diferencas de cada dump binario
    for i in range(0, Length):
        if(TransmissionData[i + OutpuOffset] !=
                    ReceptionData[i+OutpuOffset + BufferOffset]):
            BitError+=1

    return BitError
```

Código 1: Comparador de ficheiros

O comparador é automatizado para ambas as modulações BPSK e QPSK através de dois *scripts* adjacentes, que, por brevidade não foram apresentados no relatório.

O *script* compara, para cada patamar de ruído, o ficheiro de input e o ficheiro de output, consoante o aumento do *Loop Bandwidth*, que será incrementado de 0.006 em 0.006 valores, até atingir o valor máximo de 0.252, como já explicitado na secção 1.3.2.

Os dados obtidos serão visualizados nas secções subsequentes.

2.2 | Dados da modulação BPSK

2.2.1 | Desmodulador BPSK

Antes de expor os dados obtidos na simulação BPSK, é de interesse apresentar o desmodulador projetado em ambiente GNU rádio. Como já referido, a sequência binária dá entrada no modulador, é agrupada segundo *bits* e mapeada para o plano complexo. Neste sentido, é proeminente aplicar um exemplo prático.

Iremos observar a constelação do *byte* 170 → 1 0 1 0 1 0 1 0 em repetição:

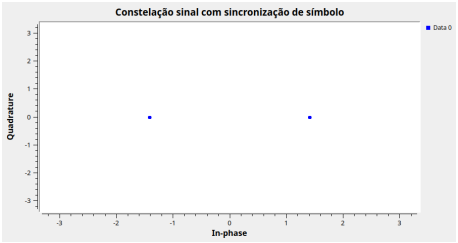


Fig. 7: Constelação após a sincronização de símbolo do *byte* 170

O *byte* é mapeado para as duas regiões do diagrama de constelação da seguinte forma:

$$\text{"1"} \mapsto [+1{+}0j] \rightarrow \begin{cases} \text{coef}_{real} = +1 \\ \text{coef}_{imag} = \ 0 \end{cases} \qquad \text{"0"} \mapsto [-1{+}0j] \rightarrow \begin{cases} \text{coef}_{real} = -1 \\ \text{coef}_{imag} = \ 0 \end{cases}$$

Obtemos assim um vetor que reproduz um sinal análogo ao da figura 3, que se pode verificar na figura infra:

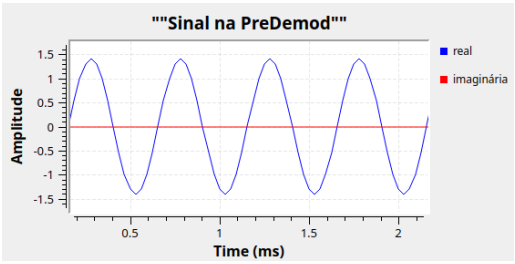


Fig. 8: Sinal modulado emitido:(1, -1, 1, -1]

Após a sincronização de símbolo (consequente encontro do intervalo ótimo de amostragem) é apenas necessário recorrer a um bloco de decisão, como já referido na secção 1.3. O bloco *Binary Slicer* efetua a conversão para *bits* que são posteriormente agrupados em *bytes* (realizado pelo bloco *Pack K Bits*). Tal como visualizado no diagrama:

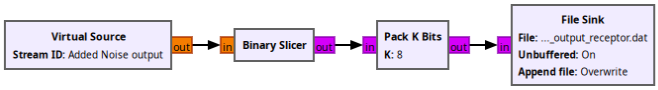


Fig. 9: Etapa final da desmodulação do sinal BPSK

Em regime prático → O ficheiro bpsk_output_receptor.dat possui um *offset* de 49 bits em relação ao vetor de transmissão, tal é explicado no guia de laboratório → "(...) os *bits* correspondentes aos *bits* enviados aparecem a partir da 50a posição sendo precedidos de 49 bits correspondentes ao *buffer* interno do módulo "Symbol Sync""[5].

2.2.2 | Exposição de dados BPSK[8]

Tab. 1: Taxa de erro do *bit* em função do *Loop Bandwidth* para cada *Noise Voltage*

Loop BW/Noise	0.0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0
0.006	0.0%	1.4%	9.3%	19.4%	26.9%	31.9%	36.1%	38.4%	39.4%
0.012	0.0%	1.4%	9.3%	19.4%	25.5%	31.0%	35.6%	37.0%	40.3%
0.018	0.0%	1.4%	9.3%	19.9%	25.5%	31.0%	35.6%	38.0%	38.9%
0.024	0.0%	1.4%	9.3%	19.9%	25.0%	31.5%	37.0%	40.7%	35.2%
0.03	0.0%	1.4%	9.3%	19.4%	25.5%	32.4%	36.6%	36.1%	36.6%
0.036	0.0%	0.9%	9.3%	19.4%	25.5%	33.8%	31.5%	31.0%	40.3%
0.042	0.0%	0.9%	9.3%	19.4%	25.5%	38.0%	31.5%	32.9%	39.4%
0.048	0.0%	0.9%	9.3%	18.5%	25.0%	29.2%	29.6%	35.2%	42.1%
0.054	0.0%	0.5%	9.7%	19.4%	32.9%	31.5%	31.0%	37.0%	42.1%
0.06	0.0%	0.5%	9.7%	22.2%	25.5%	29.6%	32.4%	42.6%	40.3%
0.066	0.0%	0.5%	9.7%	21.3%	24.5%	29.2%	36.6%	44.9%	35.6%
0.072	0.0%	0.5%	9.3%	20.8%	25.9%	29.6%	31.9%	46.8%	51.4%
0.078	0.0%	0.5%	12.0%	18.5%	25.9%	28.2%	39.4%	38.0%	47.2%
0.084	0.0%	0.5%	11.6%	21.3%	26.9%	31.0%	42.1%	39.8%	48.1%
0.09	0.0%	0.5%	11.6%	17.1%	28.2%	38.4%	38.9%	56.5%	50.0%
0.096	0.0%	0.5%	11.6%	16.2%	28.7%	38.9%	43.5%	50.0%	38.9%
0.102	0.0%	0.5%	10.6%	19.0%	27.8%	38.9%	39.4%	40.3%	45.4%
0.108	0.0%	0.5%	8.3%	19.0%	26.9%	37.5%	52.3%	47.7%	39.8%
0.114	0.0%	0.5%	7.9%	22.2%	29.2%	36.6%	45.8%	41.2%	48.6%
0.12	0.0%	0.5%	8.8%	23.6%	41.2%	41.2%	49.5%	47.2%	57.9%
0.126	0.0%	0.5%	9.3%	19.4%	42.1%	38.0%	48.6%	46.3%	43.5%
0.132	0.0%	0.5%	10.2%	20.4%	42.1%	35.2%	45.8%	53.2%	48.1%
0.138	0.0%	0.5%	10.6%	22.2%	38.9%	37.5%	49.1%	48.6%	48.6%
0.144	0.0%	0.5%	7.4%	24.5%	35.6%	39.4%	47.2%	47.2%	53.7%
0.15	0.0%	0.5%	8.8%	25.5%	36.6%	39.4%	46.8%	48.6%	50.5%
0.156	0.0%	0.5%	8.3%	23.1%	28.7%	37.0%	46.3%	48.6%	53.7%
0.162	0.0%	0.9%	8.3%	20.8%	42.1%	44.0%	41.2%	51.9%	51.9%
0.168	0.0%	0.5%	10.2%	20.4%	31.9%	48.1%	42.6%	38.9%	53.2%
0.174	0.0%	1.4%	12.0%	31.5%	39.4%	53.2%	53.2%	46.3%	48.6%
0.18	0.0%	1.4%	12.0%	29.6%	47.2%	54.6%	56.5%	49.1%	47.7%
0.186	0.0%	1.4%	12.5%	28.7%	44.9%	50.5%	45.8%	54.2%	46.3%
0.192	0.0%	1.4%	13.9%	25.0%	43.1%	50.5%	41.7%	46.3%	49.5%
0.198	0.0%	1.4%	13.4%	20.8%	28.2%	59.7%	47.7%	41.2%	50.9%
0.204	0.0%	1.4%	12.5%	44.0%	51.4%	53.7%	49.5%	47.2%	51.4%
0.21	0.0%	0.9%	14.4%	29.2%	31.5%	44.0%	47.2%	46.3%	50.0%
0.216	0.0%	1.4%	11.1%	31.0%	29.2%	36.6%	40.7%	40.3%	51.9%
0.222	0.0%	1.4%	9.7%	48.1%	42.6%	50.0%	47.2%	46.8%	43.5%
0.228	0.0%	1.4%	13.9%	14.4%	36.6%	44.9%	49.5%	50.9%	46.3%
0.234	0.0%	0.5%	8.8%	27.8%	29.6%	51.4%	50.0%	47.7%	52.3%
0.24	0.0%	0.9%	11.1%	28.7%	27.3%	40.3%	53.7%	50.0%	49.5%
0.246	0.0%	1.4%	9.7%	41.2%	45.4%	61.6%	50.9%	50.9%	45.4%

2.2.3 | Exposição Gráfica dos dados BPSK[9]

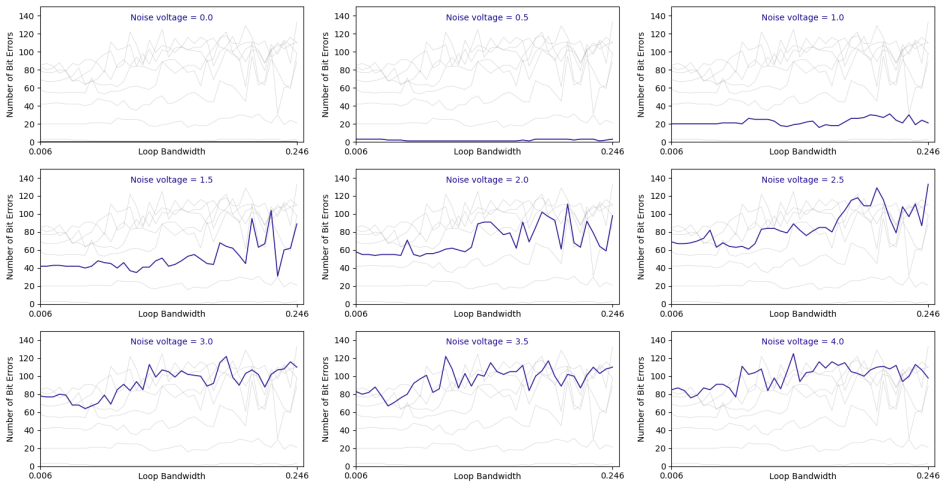


Fig. 10: Progressão do erro para cada patamar de ruído, consoante o incremento do *Loop Bandwidth*

Nota 1 → A taxa de erro mediante o incremento do *Loop Bandwidth* segue a *trendline* esperada (tal como já discutido na secção 1.3.2), i.e, o número de erros verificou-se cada vez maior e mais oscilante à medida que o valor do *Loop Bandwidth* se afasta de valores nominalmente perto de zero (i.e., afastar-se da sua gama de valores ótimos [8]). Para cada gráfico é notável uma zona de estabilidade inicial que denotamos como intervalo ótimo de *Loop Bandwidth* para o patamar de erro em questão, **observamos aqui o valor ótimo que mitiga o erro**.

Nota 2 → É também relevante mencionar que o número de erros é proporcional ao patamar de ruído, como esperado: O incremento do patamar de ruído oculta o pulso e torna difícil a sua deteção → maior taxa de erro.

Nota 3 → Como já referido na nota 1, a zona de estabilidade visível nos gráficos verifica o valor ótimo de *Loop Bandwidth*. Por outro lado,verifica-se que o comprimento desta zona contrai-se para cada patamar de ruído, pelo que o valor ótimo de *Loop Bandwidth* torna-se cada vez menor, facto corroborado pelos valores *highlighted* a vermelho na tabela da secção anterior.

[8] A vermelho encontram-se os valores de Loop Bandwidth que minimizam a taxa de erro

[9] Relevante mencionar que o gráfico são realizados com o número de erros por *Loop Bandwidth* e não percentualmente, como apresentado na tabela da secção anterior, o mesmo se verificará para a modulação QPSK

2.3 | Dados da modulação QPSK

2.3.1 | Desmodulador QPSK

Antes de expor os dados obtidos da simulação QPSK, é do nosso interesse, tal como na secção do BPSK, apresentar o desmodulador projetado em ambiente *GNU Radio*. Como já referido, a sequência binária dá entrada no modulador, é agrupada segundo *dibits* e mapeada para o plano complexo. Neste sentido, é proeminente aplicar um exemplo prático.

Iremos observar a constelação do *byte* 27 $\rightarrow$ 00 01 10 11 em repetição:

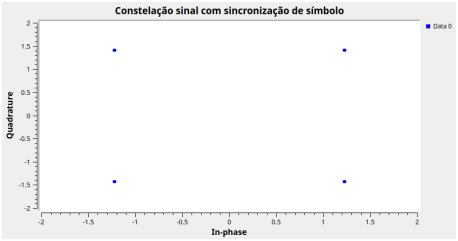


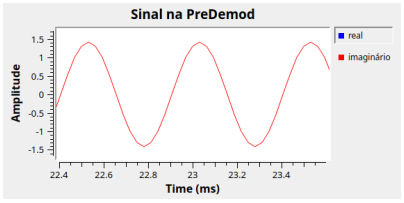
Fig. 11: Constelação após a sincronização de símbolo do *byte* 27

O *byte* é mapeado para as quatro regiões do diagrama de constelação da seguinte forma:

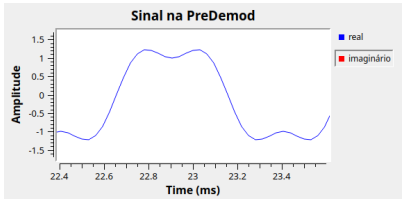
$$\text{"11"} \mapsto [+1+1j] \rightarrow \begin{cases} \text{coef}_{real} = +1 \\ \text{coef}_{imag} = +1 \end{cases} \quad \text{"01"} \mapsto [-1+1j] \rightarrow \begin{cases} \text{coef}_{real} = -1 \\ \text{coef}_{imag} = +1 \end{cases}$$

$$\text{"00"} \mapsto [-1-1j] \rightarrow \begin{cases} \text{coef}_{real} = -1 \\ \text{coef}_{imag} = -1 \end{cases} \quad \text{"10"} \mapsto [+1-1j] \rightarrow \begin{cases} \text{coef}_{real} = +1 \\ \text{coef}_{imag} = -1 \end{cases}$$

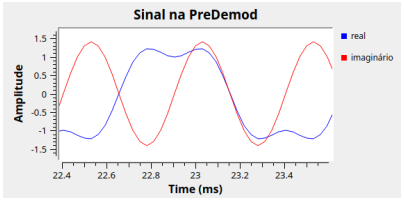
Obtemos assim dois vetores paralelos que reproduzem um sinal análogo ao da figura 3 em fase e quadratura que se podem verificar, sincronamente, nas figuras infra:



(a) componente imaginária: [-1,1,-1,1]



(b) componente real: [-1,-1,1,1]



(c) componentes sobrepostas

Fig. 12: Componentes em fase e quadratura do sinal transmitido

Nota: As modulações em fase e quadratura decorrem síncronamente: para o mesmo intervalo de tempo a parte imaginária e real codifica as componentes do mesmo *dibit*.

Após a sincronização de símbolo procuramos recuperar o sinal original, que se encontra, como já verificado acima, dividido, paralelamente, entre dois ramos de modulação, um em fase e outro em quadratura. Para tal, devemos intercalar a parte real (bit mais significativo do dibit) e a parte imaginária (bit menos significativo no dibit) de modo a reconstruir o vetor de símbolos transmitidos. Este processo é realizado através do bloco *Interleave*:

> "This block interleaves blocks of samples. For each input connection, the samples are interleaved successively to the output connection."[11]

Cuja disposição pode ser encontrada na imagem seguinte:

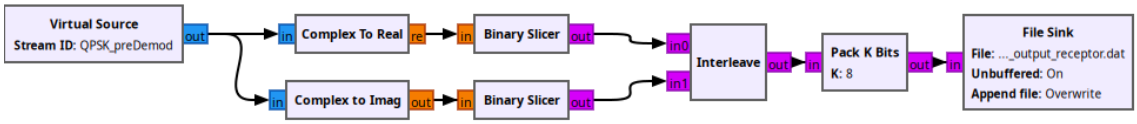


Fig. 13: Etapa final da desmodulação do sinal QPSK

Os blocos *Binary Slicer* convertem o pico no intervalo de decisão ótimo das componentes em fase e quadratura (obtido no *Symbol Sync*) num *bit* como já referido na secção 1.3, e o bloco *Interleave* intercala sequencialmente as componentes, *bits* real e imaginário respetivamente, de forma a recuperar o *stream* de *bits* inicial.

Em regime prático → Verificámos uma duplicação do *buffer* inicial incutido no *Symbol Sync* (como referido anteriormente na secção 2.3.1). Este fenómeno é esperado, visto que a modulação QPSK compreende duas codificações BPSK ortogonais em paralelo (componente real e imaginária). O *buffer* é aplicado duas vezes para a sincronização das duas componentes resultando na duplicação supramencionada. $49 + 49 = 98$, que é considerado no *script* de obtenção de dados.

2.3.2 | Exposição de dados QPSK

Tab. 2: Taxa de erro do bit em função do *Loop Bandwidth* para cada *Noise Voltage*

Loop BW/Noise	0.0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0
0.006	0.0%	1.9%	10.6%	19.0%	27.3%	30.1%	33.3%	36.1%	37.5%
0.012	0.0%	1.9%	10.6%	19.0%	26.9%	29.6%	32.9%	34.7%	36.6%
0.018	0.0%	1.9%	10.2%	18.5%	25.9%	29.2%	32.9%	35.6%	34.7%
0.024	0.0%	1.9%	9.7%	17.6%	25.9%	29.2%	35.2%	37.0%	39.4%
0.03	0.0%	1.4%	9.3%	17.6%	25.9%	28.2%	35.2%	39.4%	39.4%
0.036	0.0%	1.4%	9.3%	17.6%	24.1%	30.1%	38.0%	37.5%	39.4%
0.042	0.0%	1.4%	9.3%	16.7%	25.0%	31.9%	36.1%	38.0%	42.6%
0.048	0.0%	1.4%	9.7%	16.7%	23.1%	32.4%	36.1%	37.5%	45.4%
0.054	0.0%	1.4%	9.3%	16.7%	27.3%	32.9%	36.6%	40.3%	41.7%
0.06	0.0%	1.4%	9.7%	16.7%	27.3%	31.9%	41.7%	34.7%	46.3%
0.066	0.0%	1.4%	9.7%	16.7%	29.2%	36.6%	35.2%	46.8%	33.3%
0.072	0.0%	1.4%	9.7%	24.1%	28.2%	34.3%	48.1%	45.8%	44.4%
0.078	0.0%	0.5%	9.7%	20.8%	29.2%	43.5%	33.8%	40.3%	47.2%
0.084	0.0%	0.0%	9.7%	22.7%	29.2%	44.9%	59.3%	41.2%	49.1%
0.09	0.0%	0.5%	9.7%	22.7%	28.7%	40.7%	45.4%	48.6%	41.7%
0.096	0.0%	0.0%	9.7%	22.2%	27.3%	33.3%	38.9%	41.2%	51.9%
0.102	0.0%	0.0%	8.3%	22.7%	39.8%	36.1%	44.4%	46.3%	48.1%
0.108	0.0%	0.0%	7.4%	21.3%	43.5%	38.9%	45.8%	52.3%	44.9%
0.114	0.0%	0.0%	8.8%	20.4%	47.2%	37.0%	32.9%	48.1%	47.2%
0.12	0.0%	0.0%	14.8%	32.4%	30.1%	49.1%	50.5%	52.3%	53.2%
0.126	0.0%	0.0%	16.2%	21.8%	29.2%	46.3%	53.7%	54.2%	47.2%
0.132	0.0%	0.0%	15.7%	26.9%	29.6%	38.0%	46.3%	55.1%	46.8%
0.138	0.0%	0.0%	7.9%	21.8%	28.7%	51.9%	49.5%	48.1%	46.3%
0.144	0.0%	0.0%	16.7%	22.7%	32.9%	50.9%	46.8%	50.9%	48.1%
0.15	0.0%	0.5%	16.2%	24.5%	28.7%	54.2%	45.8%	53.2%	49.1%
0.156	0.0%	0.5%	29.6%	47.7%	38.4%	39.4%	50.9%	49.5%	47.2%
0.162	0.0%	0.5%	16.2%	49.5%	34.3%	49.5%	46.3%	48.1%	53.2%
0.168	0.0%	0.5%	17.1%	46.8%	30.6%	40.3%	47.7%	50.9%	55.1%
0.174	0.0%	0.5%	16.7%	17.1%	35.2%	52.8%	56.9%	57.4%	48.6%
0.18	0.0%	0.5%	16.7%	19.4%	35.6%	36.1%	50.9%	53.7%	39.8%
0.186	0.0%	0.0%	7.4%	25.5%	27.8%	45.8%	53.2%	51.9%	51.9%
0.192	0.0%	0.5%	16.2%	23.1%	33.3%	49.1%	54.2%	50.9%	50.9%
0.198	0.0%	8.8%	16.7%	24.1%	40.3%	55.6%	54.6%	48.1%	56.9%
0.204	0.0%	8.8%	16.7%	25.0%	50.0%	51.4%	56.5%	43.1%	44.4%
0.21	0.0%	6.9%	18.1%	22.7%	49.5%	44.0%	52.8%	53.7%	42.6%
0.216	0.0%	7.4%	17.1%	33.3%	44.4%	50.5%	44.0%	48.1%	46.8%
0.222	0.0%	8.8%	8.3%	31.5%	36.1%	43.5%	46.3%	43.5%	45.4%
0.228	0.0%	8.3%	6.9%	38.0%	40.7%	40.7%	47.2%	45.8%	56.5%
0.234	0.0%	8.3%	12.0%	33.8%	38.9%	44.9%	46.3%	46.3%	53.2%
0.24	0.0%	8.8%	16.7%	33.8%	38.4%	49.1%	41.2%	44.4%	45.4%
0.246	0.0%	5.6%	14.4%	34.7%	48.6%	41.2%	45.4%	49.5%	43.5%

2.3.3 | Exposição Gráfica dos dados QPSK

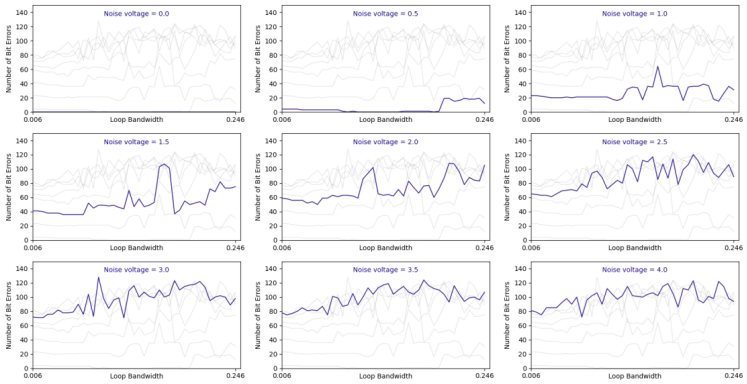
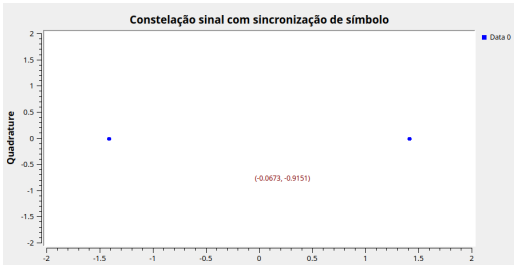


Fig. 14: Progressão do erro para cada patamar de ruído, consoante o incremento do *Loop Bandwidth*

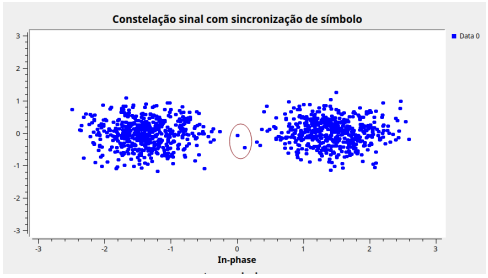
Nota → Os dados QPSK são análogos aos BPSK já apresentados e discutidos brevemente em cada nota. Devemos apenas realizar um reparo relevante: o valor médio da taxa de erro, para cada patamar de ruído é em tudo semelhante ao verificado para a modulação anterior (vide figura 14). Fenómeno este subsequentemente discutido.

3 | Análise de Resultados

Antes de proceder às etapas de comparação e análise de resultados, é conveniente abordar a taxa de erro de bit (BER) e a probabilidade de erro de ambas as modulações, para tal, é relevante a visualização dos diagramas de constelação no seio do ruído. Iremos supor, tal como nas secções 2.2.1 e 2.3.1, os bytes 170 e 27, respetivamente para as modulações BPSK e QPSK, respetivamente:



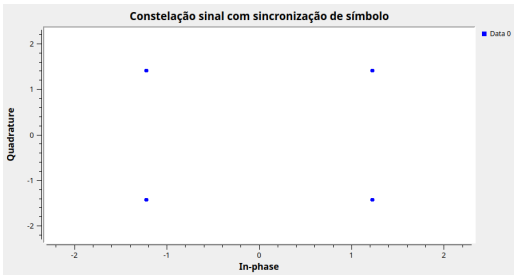
(a) BPSK: noise voltage = 0



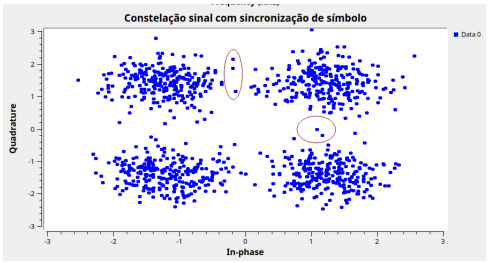
(b) QPSK: noise voltage = 0.5

Fig. 15: Comparação das constelações BPSK com e sem presença de ruído

A roxo encontramos duas amostras situadas numa região de indecisão, o pulso poderá ser identificado como *bit* 0 ou *bit* 1, pulso positivo ou pulso negativo. Admitimos então, que, na presença de ruído, a modulação BPSK possui uma única região de erro adjacente à região de correta identificação do pulso, i.e., escolher zero quando o *bit* emitido é 1 e vice versa.
Por sua vez, QPSK codifica *dibits*:



(a) QPSK: noise voltage = 0



(b) QPSK: noise voltage = 0.5

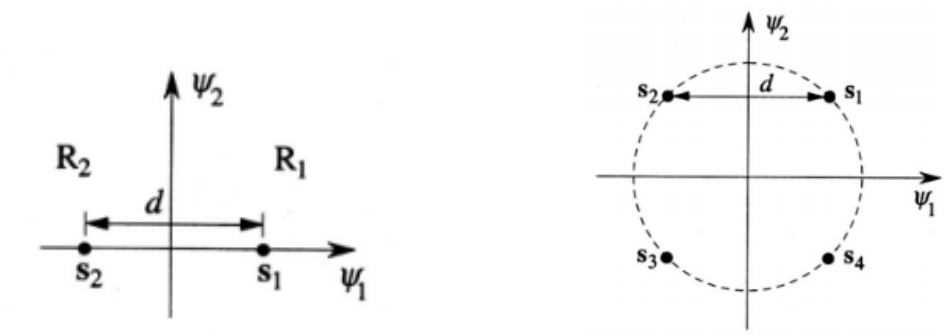
Fig. 16: Comparação das constelações BPSK com e sem presença de ruído

Novamente, a roxo encontramos pulsos situados em regiões de indecisão, no entanto, ao contrário da modulação BPSK, a modulação QPSK possui duas regiões de erro adjacentes à de correta identificação (dada a natureza da modulação, já previamente discutida em secções anteriores). Neste sentido, podemos admitir que a probabilidade de erro de símbolo da modulação QPSK é o dobro da BPSK:

$$\text{BPSK: } P_e = \frac{1}{2} erfc\left(\frac{d}{2\sqrt{N_0}}\right) \qquad \text{QPSK: } P_e \cong erfc\left(\frac{d}{2\sqrt{N_0}}\right)$$

Onde $d/2$ é a distância à origem e N_0 o patamar de ruído.

Embora as probabilidades de erro de símbolo sejam discrepantes entre modulações, o mesmo não pode ser dito da probabilidade de erro de *bit*:



(a) BPSK: Regiões de decisão

(b) QPSK: regiões de decisão

Fig. 17: Visualização da distância d nos diagramas de constelações

> "QPSK can be regarded as a pair of orthogonal BPSK systems, i.e. the real component is one BPSK system, the imaginary component is the second BPSK system. Because they are orthogonal, they don't interfere (to a good approximation), hence the BER curves are largely equivalent."[12]

Dada a natureza ortogonal da codificação, a etapa de decisão das componentes em fase e quadratura opera de forma totalmente independente (vide figura 17). O mapeamento de cada bit em quadratura é independente do de fase e portanto admitimos uma taxa de probabilidade de erro de bit análoga à do sistema BPSK, tal como corroborado pela regressão das curvas BER teórica, BPSK e QPSK:

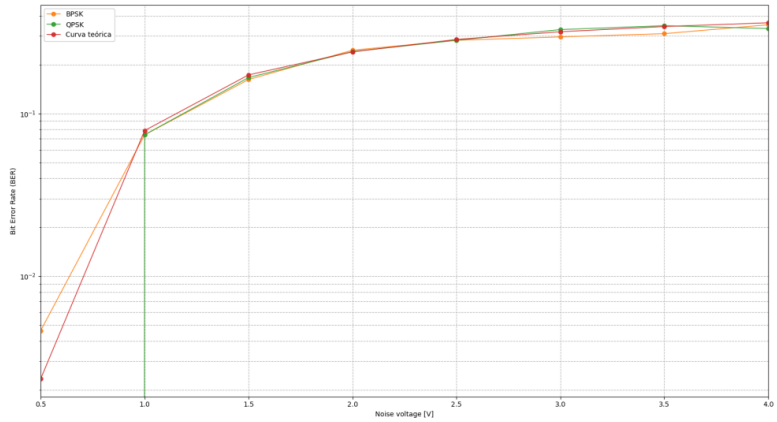


Fig. 18: Curvas BER teórica a vermelho, QPSK a verde e BPSK a laranja em escala semilogarítmica

Onde a curva teórica é computada com base na expressão já devidamente mencionada[10]:

$$\mathrm{BER}_{\mathrm{BPSK:}} = \mathrm{BER}_{\mathrm{QPSK}} = \frac{1}{2} erfc\left(\frac{d}{2\sqrt{N_0}}\right)$$

As curva experimentais são computadas mediante a menor taxa de erro de bit observada (apresentada a vermelho nas duas tabelas anteriores) por patamar de ruído.

A sobreposição das curvas demonstra a similaridade entre as taxas de erro de bit experimentais e verifica o conteúdo teórico.

[10] Onde $d/2 = \sqrt{2}$ e $N_0 = [0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0]$

5 | Referências

[1] Simon Haykin. Introduction to analog and digital communications. *2º Edition*, pages 270–277.

[2] Mathuranathan. Complex baseband equivalent models, Nov 2020. URL https://www.gaussianwaves.com/2017/10/complex-baseband-equivalent-models/.

[3] Constellation object. URL https://wiki.gnuradio.org/index.php/Constellation_Object.

[4] A guide to SDR and DSP using python. URL https://pysdr.org/content/pulse_shaping.html.

[5] Guia de laboratório, 2022.

[6] Dan Boschen. "sample rates, samples per symbol, and digital pulse shaping", Sep 2020. URL https://dsp.stackexchange.com/questions/68627/sample-rates-samples-per-symbol-and-digital-pulse-shaping?rq=1.

[7] Safe Cracker. "why samples per symbols ≥ 2 in the gnuradio constellation modulator block?", Apr 2021. URL https://dsp.stackexchange.com/questions/72648/why-samples-per-symbols-geq-2-in-the-gnuradio-constellation-modulator-block?noredirect=1&lq=1.

[8] Symbol sync. URL https://wiki.gnuradio.org/index.php/Symbol_Sync.

[9] Binary slicer. URL https://wiki.gnuradio.org/index.php/Binary_Slicer.

[10] avi1987. Loop bandwidth for symbol timing recovery, Aug 2016. URL https://dsp.stackexchange.com/questions/31170/loop-bandwidth-for-symbol-timing-recovery/31186#31186.

[11] Interleave. URL https://wiki.gnuradio.org/index.php/Interleave.

[12] "bpsk and qpsk have identical ber curve?", . URL https://www.dsprelated.com/showthread/comp.dsp/53352-1.php.

[13] Symbol timing synchronization: A tutorial - Igor Freire: EE ph.D.., Sep 2021. URL https://igorfreire.com.br/2016/10/15/symbol-timing-synchronization-tutorial/.

[14] Mwakyanjala. Gnu radio loop bandwidth normalization, Feb 2020. URL https://dsp.stackexchange.com/questions/53795/gnu-radio-loop-bandwidth-normalization.

[15] Ber performance for bpsk and qpsk, . URL https://www.dsprelated.com/showthread/comp.dsp/65309-1.php.

[16] Sergio Benedetto and Ezio Biglieri. *Waveform transmission over the Gaussian channel*, page 179–183. Springer US, 2002.

[17] jstraughjstraugh. Do bpsk vs qpsk have the same performance?, Feb 2016. URL https://dsp.stackexchange.com/questions/79490/do-bpsk-vs-qpsk-have-the-same-performance.

Apêndice

Performance: Eficiência Espetral

> "The biggest and primary advantage of QPSK over BPSK is spectral efficiency: for the same BER performance and data rate we can use half the bandwidth! This is also intuitively explained that we are sending two BPSK signals independently in the same bandwidth."[17]

Por ser, *in a nutshell*, duas cadeias de modulação BPSK simultâneas, uma em fase e outra em quadratura, a modulação QPSK possui o dobro da capacidade de transmissão de dados face a BPSK para a mesma taxa de erro.

Admitindo que:

$$\text{Eficiência espetral} = \frac{\text{Taxa de } \textit{bits}}{\text{Largura de banda}}$$

E considerando que a taxa de transmissão simulada é idêntica para ambos os sistemas simulados, mantemos o débito de *bits* e ficamos com metade da largura de banda.

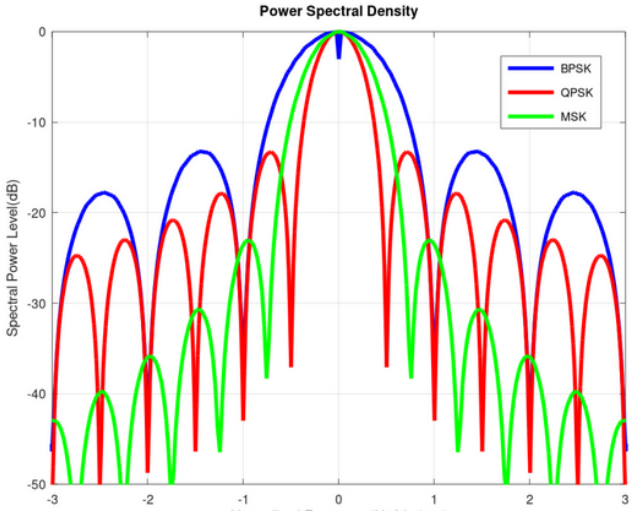


Fig. A1: Comparação entre os espetro BPSK e QPSK

É facilmente depreendido que a eficiência espetral da modulação QPSK é o dobro da BPSK.