ISEL
INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Trabalho 3
Comunicações e Processamento de Sinais

LICENCIATURA EM ENGENHARIA INFORMÁTICA E MULTIMÉDIA
ANO LETIVO 2017/2018

Turma LEIM 31D
Grupo:13
Data:27/12/2018
Docentes:
Eng. André Lourenço
Eng. Pedro Fazenda
Alunos:
Luis Fonseca (A45125)
Philipp Al-Badavi (A45138)

Índice

Introdução

O 3º trabalho da disciplina Comunicação e Processamento de Sinais consistia na finalização da implementação de blocos de esquema de envio e recepção de informação usando a modulação digital e por usa vez o teste do funcionamento da mesma. Desta vez foi nos solicitados a implementação de um sistema de emissão e recepção de códigos de linha e modulações digitais

Parte A

1. Este exercício consistia na construção de um emissor que, recebe um array de bits e retorna uma codificação de Manchester dessa sequência de bits. O código Manchester é um código polar, de transição entre níveis, ou seja, o nível lógico “1” é representado pela transição a meio do tempo de bit da tensão -A para +A, o nível lógico “0” tem representação contrária, ou seja transita de +A para -A a meio do tempo de bit. Na implementação foi nos silicitado que a função recebesse, a amplitude de código(A), o número de amostras por bit, ou dito por outras palavras ,tempo de bit(P) e uma trama de bits(n). Dentro de um lambda é percorrido o array de bits, e para cada posição de array é aplicado o código de Manchester , de forma como foi descrito o seu funcionamento.

   Na figura seguinte vem o exemplo do seu funcionamento com uma trama de [1,0,1,0,1,1,0,0] e o tempo de bit igual a 4.

```
[-1. -1.  1.  1.  1.  1. -1. -1. -1. -1.  1.  1.  1.  1. -1. -1. -1. -1.
  1.  1. -1. -1.  1.  1.  1.  1. -1. -1.  1.  1. -1. -1.]
```

Figura 1-Exempo de funcionamento do Código de Manchester

2. Neste exercício foi nos solicitado a construção de uma função que simulasse o filtro adaptado. Esta devia receber uma código de linha com ruído, um limiar de decisão e retornar um array de bits filtrado.No código, no início é feito um resize de forma que cada nível lógico do código de Manchester ocupasse uma só posição no array, a seguir é criado um array que simulasse a transição de nível lógico “1” do código de Manchester. De seguida no lambda é feita a multiplicação de arrays iguais, onde o “x” é códificação de Manchester de um nível lógico do array que é fornecido ao filtro e o “c” é o nível lógico de “1” no código de Manchester, assim é criado um novo array denominado “resultado” que vai ter o resultado da soma, das multiplicações dos códificações de Manchester dos níveis lógicos do array fornecido pelo nível lógico de “1” do código de Manchester. Por fim este array(resultado) é percorrido outra vez ,dentro de um lambda e é criado um novo array, este array terá o valor de 1 na posição onde o array – resultado- terá um valor maior que o limiar de decisão e, terá o valor de 0 na posição onde o mesmo array tera o valor menor ou igual a do limiar de decisão.

3. Para este exercício foi nos solicitado criar uma função que simulasse um canal AWGN,está função é meramente fácil na sua implementação pois consiste numa simples soma do array que sai do emissor , com um array de tamanho igual, que apresenta valores aleatórios entre 0 e 1 e é multiplicado pela potência fornecida, este array simula o ruído de um canal de transmição. A função devia receber como parâmtetros o array do sinal emitido e uma potência. Por fim o array retornava um sinal com ruído.

4. Neste exercício para o sinal previamente gravado, o grupo gerou o código de linha com P=8 e A=1, o sinal foi transmitido pelo canaAWGN onde obteve ruído e de seguida foi descodificado, este percurso foi feito para valores de potência de 0.5,1,2 e 4. Mas o resultado do SNR dava sempre um valor negativo, por esse motivo o grupo optou por repetir o exercício e apresentar os valor que se obteve com um sinal, sem fala, um sinal que representa uma simples melodia.

```
SNR
35.113228208728756
18.98764070342804
8.885052028861924
3.820247483238785
SNR inicial
45.89451988252491
BER
2.8658039535357653e-05
0.002327669655594049
0.022736970144691258
0.07869497656409212
BERP
3.167124183311986e-05
0.002338867490523633
0.022750131948179195
0.07864960352514258
BERL
3.064818116975749e-05
0.0023181169757489303
0.022788315798858774
0.07850949536376604
```

Figura 2- Valores de SNR e BER para as diferentes potências

É possível observar que o SNR diminui conforme o aumento da potência, isso tudo porque o SNR é um resultado de potência do sinal sobre a potência do erro, por isso quanto maior for a potência do erro, menor será o SNR. Também é possível observar que o BER prático(BER) não difere mais do que 20% do BER teórico(BERP). Também é possível observar que o Ber linha(BERL), em algumas das situações apresenta um valor superior ao Ber prático(BER), isso devido ao módulo de correção que só é capaz de corrigir um erro por cada trama, se virem dois erro numa trama ele é capaz de induzir erro, o que acontece nesse caso.Os gráficos dos BER é possível encontrar em anexo ou na pasta “prints”

Parte B

1. Este exercício consistia na criação de uma função para simular a modulação digital de ASK. A modulação ASK é obtida através de um produto de código de linha por uma sinusoidal, onde o nível lógico “1” irá ter uma dada amplitude A1 e o nível lógico “0” irá ter uma outra amplitude A0, construindo assim um sinal sinusoidal com diferentes amplitudes. No nosso código essa função recebe o número de pontos por cada bit (P) ,a trama e, também recebe as duas amplitudes A1 e A0, dentro da função é criado uma sinusoidal com aplitude de 1 e que tem P pontos, depois é criado um novo array(ASK), dependendo do nível lógico em cada posição irá ter a sinusoidal s0 com amplitude A1, caso o nível lógico na posição for “1” ou s0 com amplitude A0 caso o nível lógico na posição for “0”. A resolução deste exercício difere da maneira como é pedida no enunciado.

2. Neste exercício foi nos solicitado a criação de um filtro ASK que devia receber o sinal ASK com ruído e retornar uma sequência binária correspondente. Para o funcionamente desse receptor é necessário calcular um limiar de decisão que por sua vez será o ponto de referência de se o sinal naquele ponto tem o valor lógico de "1" ou de "0", para o cálculo deste limiar é necessário obter a energia do bit 1 e bit 0, de seguida somam-se as suas raízes e divide-se por 4, há várias formas de calcular o limiar, o grupo obtou por esta. No nosso recetor , a função recebe 4 parâmetros, o sinal com ruído, as amplitude A1 e A0 e, o número de pontos por bit(P), depois do cálculo do limiar é feito um recize do sinal para mais fácil avaliação dos seus níveis e por final é criado um array que terá o valor de 1 na posição se e so se , na sinusoidal que estiver em avaliação o primeiro valor for maior que o limiar de decisão e o quinto valor multiplicado por "-1" for maior que o limiar de decisão, este valor é multiplicado por "-1" pois o limiar é positivo e por isso é preciso forçar o valor a ser positivo, esse quinto valor é o suposto "minímo" da função.

3. Tanto este exercício como o exercício 4 consiste em por o sistema a correr com a modulação digital e apresentar o resultado desse sistema para diferentes valores de parâmetros, como por exemplo o nosso grupo optou em variar a potência, a amplitude, e os bits na quantização do sinal e, também avaliá-lo sem o módulo de correcção de erros. Infelizmente o grupo não foi capaz de fazer a parte da constelação do exercício 4 da parte B.

   De seguida são apresentados os resultados obtidos do exercício 4 da parte B após ter o sistema a funcionar:

```
BERS
0.0
0.0
0.0
SNR
28.429963647165163
28.429963647165163
28.429963647165163
```

Figura 3 - A1 = 3, pot = 0.02 , 5 bits

Os BERS são, BER , BER linha e BER sem módulo de correcção de erros, SNR são SNR inicial, SNR final e SNR final sem módulo de correcção de erros. Neste caso todo o ruído que foi possível induzir no sinal durante a sua transmição pelo canal AWGN, foi filtrado de maneira a fazer desaparecer o ruído fazendo como que não haja erros no sinal. O BER teórico calculado pela função 0.5 * special.erfc(np.sqrt(((1**2)*P)/(2 * 0.5))) deu o valor de 2.7536241186061556e-89 esse valor é muito parecido ao 0,o que nos fez considerar que a simulação está correta. Como não houve erro nenhum no sinal, o SNR permaneceu constante. Também é possível observar que neste caso o módulo de correcção de erros não erra necessário.

```
BERS
0.014219482372121454
0.015014265335235378
0.01580777460770328
SNR
28.429963647165163
9.174487402243855
8.912385544959129
```

Figura 4 - A1 = 3 , pot =0.2, 5bits

Aqui a ordem dos BERs e SNRs é a mesma e é possível observar que quando a potência for de 0.2, o módulo de correcção já não funciona tão bem como devia e em vez de corrigir o erro, este faz com que haja ainda mais erro, isso é possível observar na medida em que o BER é menor que o BER linha. O SNR diminui

pois como o BER não é 0, quer dizer que a potência do erro incrementa,pelo que faz diminuir o SNR.

```
BERS
0.00021493529651518258
0.00021453726818830243
0.00024239925106990013
SNR
45.89451988252491
26.820017864174087
26.456946705978133
```

Figura 5- A1 = 6 , pot = 0.5 , 8bits

Aqui é possível observer a eficiência da correcção do erro, para amplitude A1 = 6, potência = 0.5 e quantização a 8 bits, o módulo foi capaz de corrigir alguns erros, porém isso nem sempre se verificava, na maior parte dos casos foi possível observar que o módulo de correcção adicionava ainda mais ruído ao sinal.Aqui também é possivel observar a diminuição do valor do SNR.

Conclusões

No final do trabalho os objetivos foram quase todos alcançados, não foi possível fazer constelação do sinal e, o sinal de voz previamente gravado apresentava o SNR negativo, o que fez com que o grupo mudasse de som, apesar do SNR naquele som gravado dar negativo, era possível percorrer o ciclo todo desde a quantização , até a desquantização e ouvir o som sem qualquer ruído quando a potência era igual a 0. Com o novo som, foi possível chegar a conclusão de que,na parte B, o modulo de correcção de erros , na maior parte das vezes adicionava mais ruído e , por sua vez para certas potências até era desnecessária pois o ruído era filtrado no filtro ASK, porém havia certas potências , em certos casos onde a correção de erros diminuía o BER, o que significa que o ruído diminuia, só que como o sinal era muito grande, essa diminuição era pouco notável.

Bibliografía

Sebenta 1 - Carlos Eduardo De Meneses Ribeiro

III-1 Códigos detectores e correctors – André Lourenço

Anexos

Aqui são apresentados os gráficos dos BERS do exercício 4 grupo A.

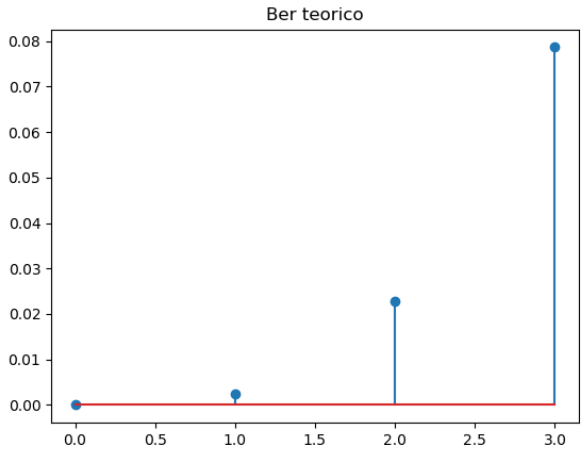

Figura 6 - BER teórico

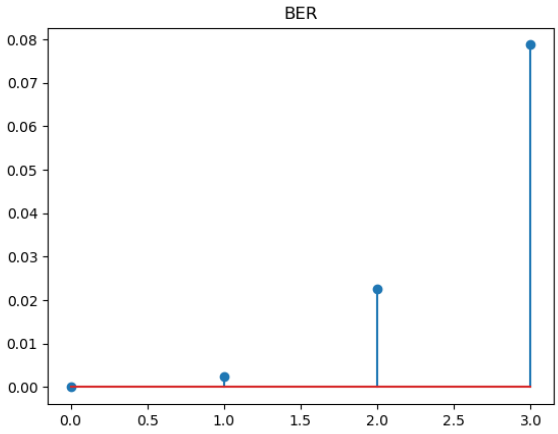

Figura 7 - BER prático

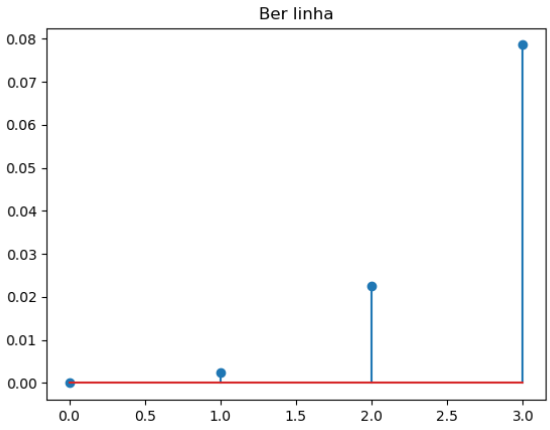

Figura 8 - BER linha

Também são apresentado os gráficos do sinais desquantizados em comparação com sinais inicialmente quantizados(exercício 4 parte B). O sinal desquantizado esta a azul e o inicial está a laranja.

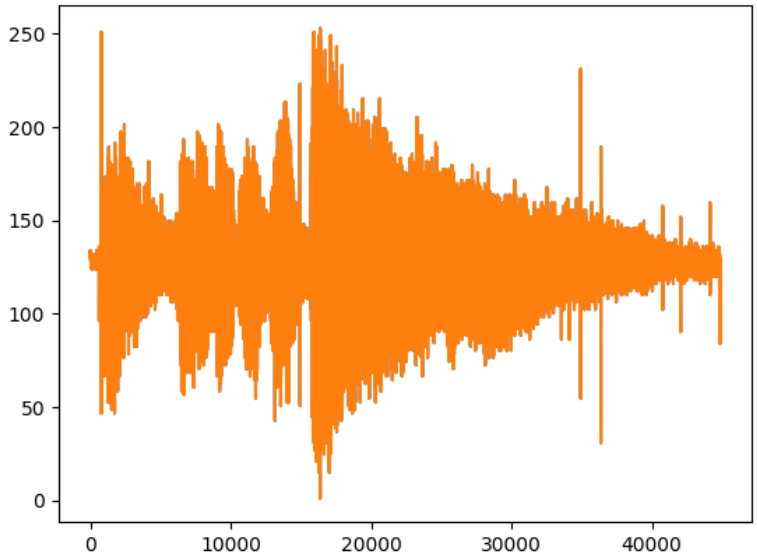

Figura 9 - A1 = 3 , pot = 0,02 , 8bits

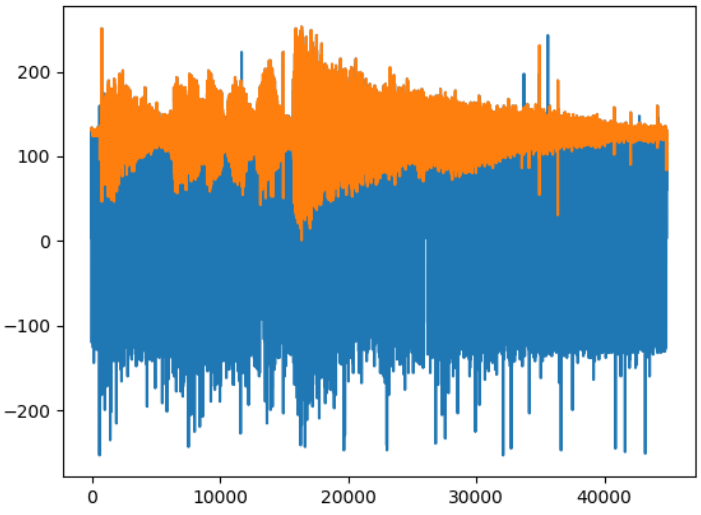

Figura 10 - A1 =3, pot = 0,2 , 8bits