

Relatório do Segundo Trabalho Prático

Grupo	13
38866	Manuel Dias

Docente	Artur Ferreira
---------	----------------

Relatório final realizado no âmbito da Unidade Curricular de Comunicações,
do curso de licenciatura em Engenharia Informática e de Computadores
Semestre de Verão 2019/2020

Junho de 2020

Índice

Índice de Figuras	2
1. Objetivos	4
2. Grupo 1	5
2.1 Alínea A	5
2.1.1 Códigos de Linha.....	6
2.1.1.1 Ausência de Código de Linha	6
2.1.1.2 NRZ Bipolar Absoluto	7
2.1.1.3 NRZ Unipolar Mark	7
2.1.2 Modulação Digital	8
2.1.2.1 Ausência de Modulação Digital.....	8
2.1.2.2 Modulação OOK.....	9
2.1.2.3 Modulação FSK	9
2.1.3 Canal Físico	10
2.2 Alínea B	11
2.3 Alínea C	16
2.3.1 Transmissão de um ficheiro de texto	16
2.3.1 Transmissão de um ficheiro de imagem	17
3. Grupo 2	20
3.1 Cálculo da curva de BER	20
3.2 Resultados experimentais	20
3.2.1 Transmissão de um ficheiro de texto com código de Hamming(7, 4).....	21
3.2.2 Transmissão de um ficheiro de texto sem código de Hamming(7, 4)	23
3.2.3 Transmissão de um ficheiro de imagem com código de Hamming(7, 4)	25
3.2.4 Transmissão de um ficheiro de imagem sem código de Hamming(7, 4).....	27
4. Grupo 3	29
4.1 Alínea (i)	29
4.2 Alínea (ii)	30

Índice de Figuras

Figura 1 - Representação da GUI totalmente modificada	5
Figura 2 - Representação do código de linha na GUI	6
Figura 3 - Representação da modulação digital na GUI	8
Figura 4 - Representação do Ruído de Canal na GUI	10
Figura 5 - Emissão de dados com NRZ Bipolar Absoluto	11
Figura 6 - Receção de dados com NRZ Bipolar Absoluto	11
Figura 7 - Ficheiro resultante de uma transmissão NRZ Bipolar Absoluto	11
Figura 8 - Emissão de dados com NRZ Unipolar Mark	12
Figura 9 - Receção de dados com NRZ Unipolar Mark	12
Figura 10 - Ficheiro resultante de uma transmissão NRZ Unipolar Mark	12
Figura 11 - Emissão de dados com modulação OOK	13
Figura 12 - Receção de dados com modulação OOK	13
Figura 13 - Ficheiro resultante de uma transmissão com modulação OOK	13
Figura 14 - Emissão de dados com modulação FSK	14
Figura 15 - Receção de dados com modulação FSK	14
Figura 16 - Ficheiro resultante de uma transmissão com modulação FSK	14
Figura 17 - Receção de dados com modulação FSK com valor de SNR alterado	15
Figura 18 - Receção de dados com modulação FSK com valor de atenuação do canal alterado	15
Figura 19 - Sinal de emissão do ficheiro de texto "a.txt" codificado	16
Figura 20 - Sinal de receção do ficheiro de texto "a.txt" codificado	17
Figura 21 - Conteúdo do ficheiro original	17
Figura 22 - Conteúdo do ficheiro transmitido	17
Figura 23 - Sinal de emissão do ficheiro de texto "bird.gif" codificado	18
Figura 24 - Sinal de receção do ficheiro de texto "bird.gif" codificado	18
Figura 25 - Resultado da tentativa de decodificar o sinal recebido	19
Figura 26 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=0,1	21
Figura 27 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=0,2	21
Figura 28 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=0,5	21
Figura 29 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=0,75	21
Figura 30 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=1	21
Figura 31 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=1,5	21
Figura 32 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=2	21
Figura 33 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=5	21

Figura 34 - Curva de BER da transmissão de um ficheiro de texto com código Hammin (7, 4).....	22
Figura 35 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=0,1	23
Figura 36 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=0,2.....	23
Figura 37 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=0,5.....	23
Figura 38 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=0,75.....	23
Figura 39 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=1.....	23
Figura 40 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=1,5.....	23
Figura 41 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=2.....	23
Figura 42 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=5.....	23
Figura 43 - Curva de BER da transmissão de um ficheiro de texto sem código Hammin (7, 4)	24
Figura 44 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=0,1	25
Figura 45 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=0,2	25
Figura 46 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=0,5	25
Figura 47 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=0,75	25
Figura 48 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=1	25
Figura 49 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=1,5	25
Figura 50 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=2	25
Figura 51 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=5	25
Figura 52 - Curva de BER da transmissão de um ficheiro de imagem com código Hammin (7, 4) ..	26
Figura 53 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=0,1	27
Figura 54 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=0,2.....	27
Figura 55 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=0,5.....	27
Figura 56 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=0,75.....	27
Figura 57 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=1	27
Figura 58 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=1,5.....	27
Figura 59 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=2.....	27
Figura 60 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=5.....	27
Figura 61 - Curva de BER da transmissão de um ficheiro de imagem sem código Hammin (7, 4) ..	28
Figura 62 - Resultados do script de testes das funções de BI	29
Figura 63 - Resultados do script de testes das funções de ISBN	30

1. Objetivos

Este trabalho tem como objetivos:

- Desenvolver programas e aplicações em MATLAB/OCTAVE;
- Estudo e aplicação de conceitos sobre codificação de fonte, cifra, correção de erros e desempenho de sistemas de transmissão digital, ao nível físico;

2. Grupo 1

No primeiro grupo do trabalho prático, é pretendido adicionar funcionalidades à GUI do trabalho anterior, adicionando 3 sub-blocos ao bloco CSD: emissor, canal e recetor, e testar as suas funcionalidades.

2.1 Alínea A

As funcionalidades adicionadas foram separadas em 3 partes:

1. Suporte de códigos de linha;
2. Suporte de modulações digitais;
3. Suporte de simulação de um canal físico;

Em todas as funcionalidades, existem 2 parâmetros genéricos que podem ser escolhidos pelo utilizador, o tempo de bit em segundos e o número de amostras por cada bit. Este último parâmetro não convém ser um número muito elevado, pois, dependendo do ficheiro a analisar, este vai indicar a resolução do sinal e quanto maior for a resolução, mais memória vai consumir. Estes parâmetros podem ser verificados na Figura 1 no canto inferior direito.

Opcionalmente, foi adicionado um modo janela que mostra 10 bits no gráfico da GUI, quando houver uma simulação e o utilizador carregar num dos 2 botões do sub-bloco de código de linha ou de modulação digital. Esta janela pode ser movível no eixo do tempo de transmissão, deslocando a barra por debaixo do gráfico.

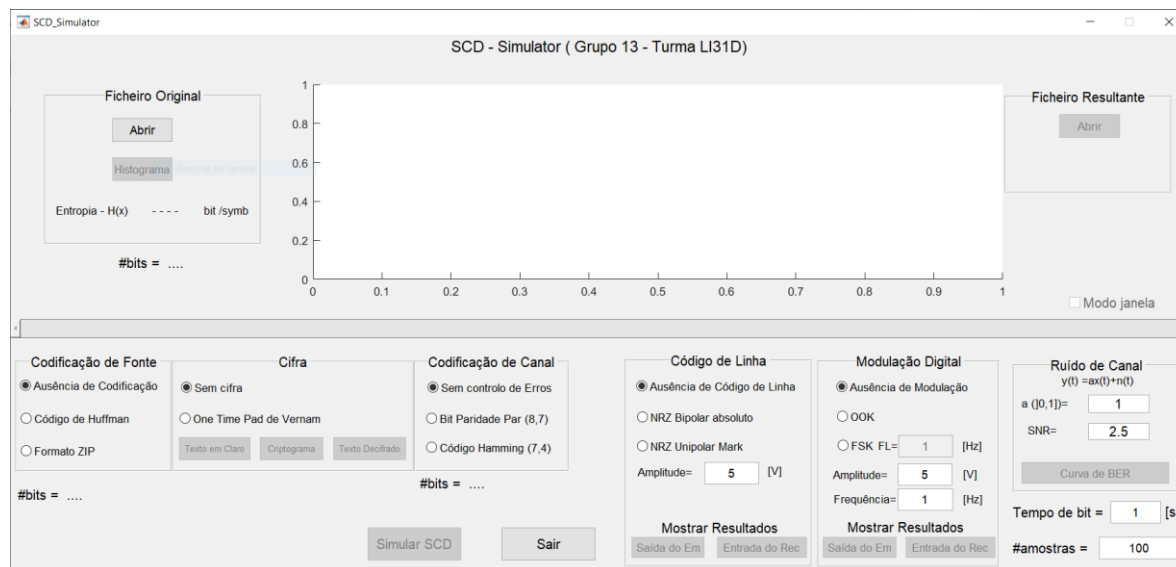


Figura 1 - Representação da GUI totalmente modificada

2.1.1 Códigos de Linha

O código de linha está dividido em 3 partes:

1. Ausência de código de linha;
2. NRZ Bipolar Absoluto;
3. NRZ Unipolar Mark;

Esta implementação permite ao utilizador escolher que tipo de código de linha quer e, para além disto, o utilizador pode escolher a amplitude de cada pulso em Volts, como pode ser verificado na Figura 1 e na Figura 2.

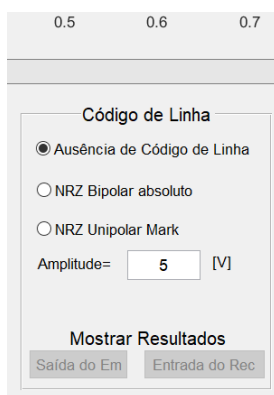


Figura 2 - Representação do código de linha na GUI

2.1.1.1 Ausência de Código de Linha

Na emissão de pulsos digitais não modulados, é necessário dos dados a transmitir a partir de um ficheiro, da amplitude de cada pulso, o número de amostras de cada pulso e o tempo de bit. A partir disto é possível transformar os dados do ficheiro num Stream de pulsos em que cada pulso pode ter entre 2 valores, ou 0, caso o bit seja 0, ou o valor de amplitude se o bit for 1.

Esta função, *Em_Aus*, retorna 3 vetores:

1. Vetor de pulsos;
2. Vetor de tempo;
3. Vetor de energia de cada bit;

Estes vetores são essenciais para representar graficamente os pulsos, no caso do vetor de pulsos e do tempo, como também para simular e calcular o ruído de canal, no caso do vetor de energia de cada bit.

Na receção de pulsos digitais não modulados, é feito o procedimento inverso, escrevendo, no final, num ficheiro cujo nome é a concatenação de “CLA_” com o nome do ficheiro passado na emissão.

2.1.1.2 NRZ Bipolar Absoluto

Na emissão de pulsos do tipo NRZ Bipolar Absoluto, com a técnica de bit-suffing com blocos de 6 bits, é necessário o mesmo número de parâmetros que na Ausência de Código de Linha. No entanto, antes de transformar os bits em pulsos, os bits são postos em blocos de 6 ou 7 bits. Para tal, foi feita a função *Generate_Bip_Block* que gera blocos de N bits, neste caso 6, se houver transições de bits dentro desse bloco, caso contrário adiciona um bit inverso ao primeiro bit do bloco para o recetor detetar uma transição. A transição destes blocos para pulsos é feita quase da mesma maneira que na funcionalidade de ausência. A diferença está quando um bit no bloco for 0, o pulso não terá o valor 0, mas sim o valor negativo da amplitude.

Na receção de pulsos do tipo NRZ Bipolar Absoluto, com a técnica de bit-suffing com blocos de 6 bits, é feito o procedimento inverso, no entanto, a cada 6 pulsos, se não detetar uma transição, então o bit a seguir é ignorado, pois este é o bit de deteção de transição. No final é gravado os dados num ficheiro cujo nome é a concatenação de “NRZBA_” com o nome do ficheiro recebido no emissor.

2.1.1.3 NRZ Unipolar Mark

Na emissão de pulsos do tipo NRZ Unipolar Mark, com a técnica de bit-suffing com blocos de 6 bits, é necessário o mesmo número de parâmetros que na Ausência de Código de Linha. No entanto, antes de transformar os bits em pulsos, os bits são postos em blocos de 6 ou 7 bits. Para tal, foi feita a função *Generate_Uni_Block* que gera blocos de N bits, neste caso 6, se houver transições de bits dentro desse bloco, caso contrário adiciona um bit inverso ao primeiro bit do bloco para o recetor detetar uma transição. Para fazer a transição dos blocos para pulsos, é preciso definir primeiro, o valor inicial, mark, do sinal que foi 0. Este valor, inicialmente, tanto pode ser 0 como 1, mas tem de ser o mesmo entre o emissor e o recetor. Depois, durante o ciclo da transformação, cada vez que é detetado o bit 1, o valor de mark altera para $mark = (mark + 1) \% 2$, para fazer a transição de sinal. Este sinal está compreendido entre a amplitude escolhida pelo utilizador e 0.

Na receção de pulsos do tipo NRZ Unipolar Mark, com a técnica de bit-suffing com blocos de 6 bits, é feito o procedimento inverso, no entanto, a cada 6 pulsos, se não detetar uma transição, então o bit

a seguir é ignorado, pois este é o bit de detecção de transição. No final é gravado os dados num ficheiro cujo nome é a concatenação de “NRZUM_” com o nome do ficheiro recebido no emissor.

2.1.2 Modulação Digital

A modulação digital está dividida em 3 partes:

1. Ausência de modulação;
2. OOK;
3. FSK;

Esta implementação permite ao utilizador escolher que tipo de modulação quer e, para além disto, o utilizador pode escolher a amplitude de sinal em Volts como a frequência em Hertz, como pode ser verificado na Figura 1 e na Figura 3. Caso o utilizador escolha a opção FSK, a frequência à frente da opção indica a frequência quando um bit for 0.

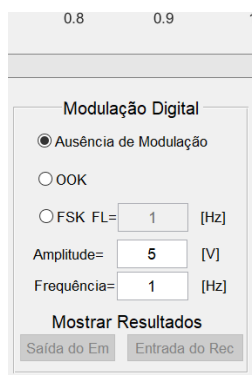


Figura 3 - Representação da modulação digital na GUI

2.1.2.1 Ausência de Modulação Digital

Neste modo, como não existe modulação, os dados à saída do emissor são iguais aos dados à saída do emissor de código de linha.

Na receção, é menos simples, pois o sinal pode vir com ruído. Para resolver isto, é calculado a energia de bit recebido em cada pulso. Se o código de linha escolhido for o NRZ Bipolar, então é verificado se esta energia é positiva ou negativa, devolvendo a amplitude de um pulso negativo caso a energia seja positiva. Caso contrário o pulso devolvido vem com amplitude negativa. Nos outros tipos de

código de linha, é comparada esta energia com a energia de um pulso sem ruído e caso que a energia calculada com ruído for superior a metade da energia de um pulso sem ruído, então devolve um pulso com amplitude escolhida pelo utilizador, caso contrário o pulso terá uma amplitude de 0.

2.1.2.2 Modulação OOK

Para fazer a emissão da modulação digital do tipo OOK, é necessário receber o vetor de dados a modular, a amplitude usada no código de linha, o tempo de bit, a amplitude de modulação escolhida pelo utilizador, a frequência e o número de amostras. Depois, num ciclo, é verificado pulso a pulso, se a amplitude do mesmo corresponde à amplitude do código de linha. Se for igual, então é gerado um sinal sinusoidal, caso contrário, o sinal é 0 no tempo de bit corrente.

Para a receção de sinais com modulação digital do tipo OOK é necessário calcular a energia absoluta do sinal, pois a partir disto é possível verificar se a amplitude de cada sinal de bit é igual ou superior à amplitude escolhida pelo utilizador. Se for superior ou igual, então o sinal representa um valor positivo, gerando um pulso com uma amplitude escolhida no código de linha. Caso contrário, o pulso terá ou uma amplitude com valor 0 ou com um valor negativo, dependendo do código de linha escolhido.

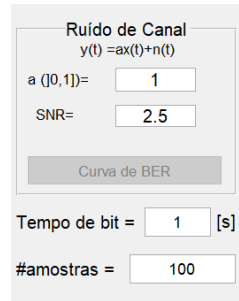
2.1.2.3 Modulação FSK

Para fazer a emissão da modulação digital do tipo FSK, é necessário receber o vetor de dados a modular, a amplitude usada no código de linha, o tempo de bit, a amplitude de modulação escolhida pelo utilizador, a frequência que representa o bit 1, a frequência que representa o bit 0 e o número de amostras. Depois, num ciclo, é verificado pulso a pulso, se a amplitude do mesmo corresponde à amplitude do código de linha. Se for igual, então é gerado um sinal sinusoidal com a frequência com bit 1, caso contrário, é gerado com a frequência com bit 0.

Para a receção de sinais com modulação digital do tipo FSK é necessário calcular a frequência fundamental do sinal, pois a partir disto é possível verificar a frequência de cada sinal de bit. Se esta for superior ou igual à frequência do bit 1 ao quadrado, então o sinal representa um bit 1, gerando um pulso com uma amplitude escolhida no código de linha. Caso contrário, o pulso terá ou uma amplitude com valor 0 ou com um valor negativo, dependendo do código de linha escolhido.

2.1.3 Canal Físico

A simulação do canal físico é feita seguindo a expressão $y(t) = \alpha x(t) + n(t)$. Isto significa que é necessário que o utilizador insira um valor de atenuação do canal, α , e a relação sinal-ruído, SNR, para adicionar ruído ao canal. Isto pode ser verificado na Figura 1 e 4.



Ruído de Canal
 $y(t) = ax(t) + n(t)$

a ([0,1])= 1

SNR= 2.5

Curva de BER

Tempo de bit = 1 [s]

#amostras = 100

Figura 4 - Representação do Ruído de Canal na GUI

Para calcular a adição de ruído, é preciso saber o SNR e a energia de sinal. A partir destes dois fatores, é possível calcular a energia do ruído e depois utiliza-se um gerador de números aleatórios normalmente distribuídos para simular o ruído. Por último é devolvido ao sinal de saída, os sinais de entrada atenuados com o ruído.

2.2 Alínea B

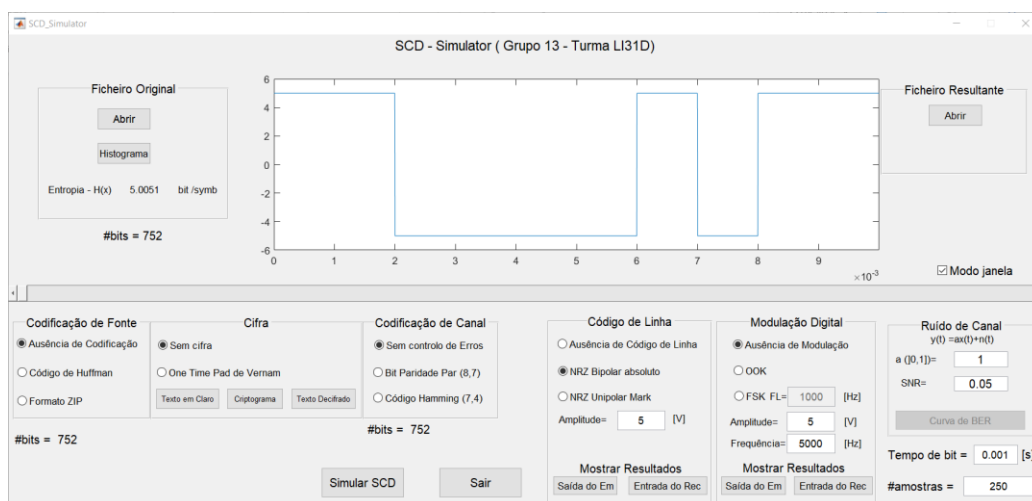


Figura 5 - Emissão de dados com NRZ Bipolar Absoluto

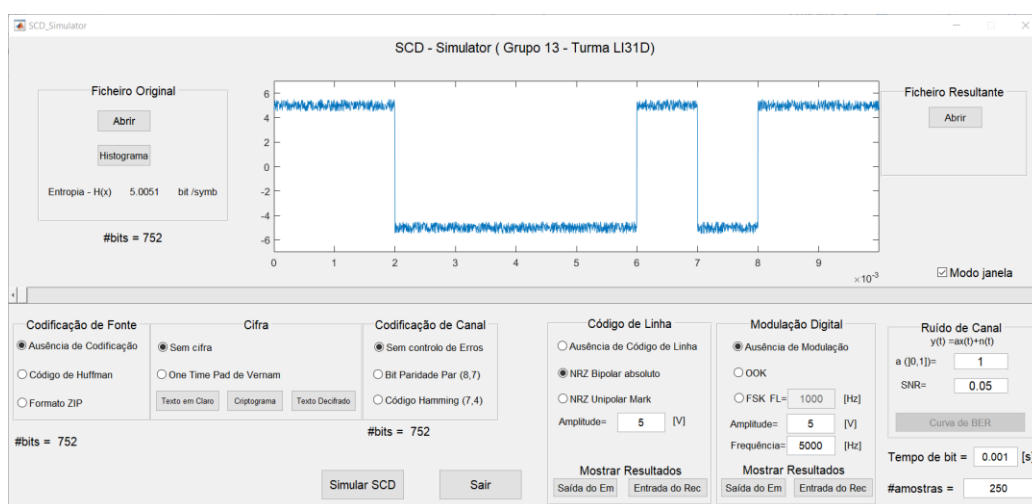


Figura 6 - Receção de dados com NRZ Bipolar Absoluto

Na Figura 5 é possível verificar a emissão de dados com código de linha NRZ Bipolar com uma amplitude de 5 a -5 Volts e um tempo de bit de 1 milissegundo. Na Figura 6 é possível verificar a receção dos mesmos dados, mas com uma adição de ruído na amplitude, no entanto, é visivelmente possível perceber os bits a 0 e a 1. Na Figura 7 está representado o ficheiro resultante da transmissão.

```
NRZBA_a.txt x +
1 Comunicações - ficheiro de teste.
2 1234567890
3 The quick brown fox jumps over the lazy dog.
```

Figura 7 - Ficheiro resultante de uma transmissão NRZ Bipolar Absoluto

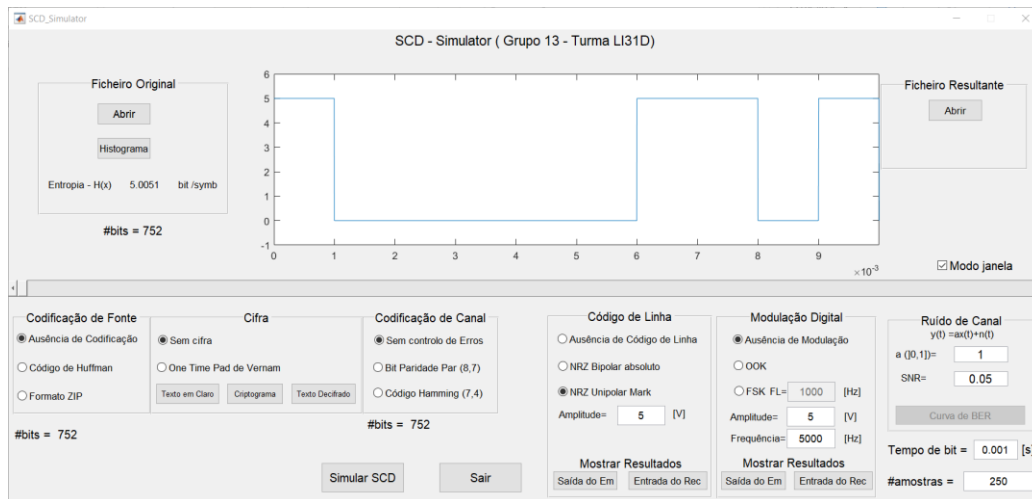


Figura 8 - Emissão de dados com NRZ Unipolar Mark

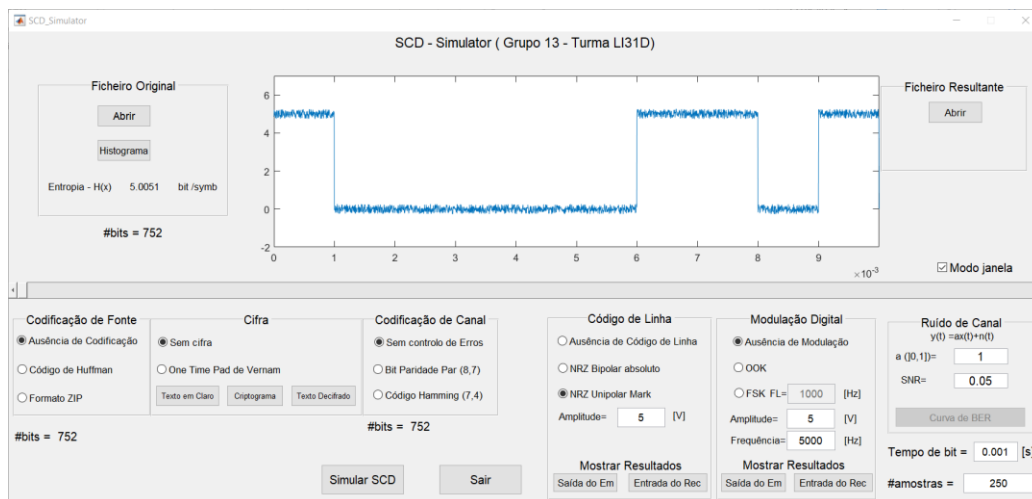


Figura 9 - Receção de dados com NRZ Unipolar Mark

Na Figura 8 é possível verificar a emissão de dados com código de linha NRZ Unipolar com uma amplitude de 5 a 0 Volts e um tempo de bit de 1 milissegundo. Na Figura 9 é possível verificar a receção dos mesmos dados, mas com uma adição de ruído na amplitude, no entanto, é visivelmente possível perceber os bits a 0 e a 1. Na Figura 10 está representado o ficheiro resultante da transmissão.

```
NRZBA_a.txt x NRZUM_a.txt x +
1 Comunicações - ficheiro de teste.
2 1234567890
3 The quick brown fox jumps over the lazy dog.
```

Figura 10 - Ficheiro resultante de uma transmissão NRZ Unipolar Mark

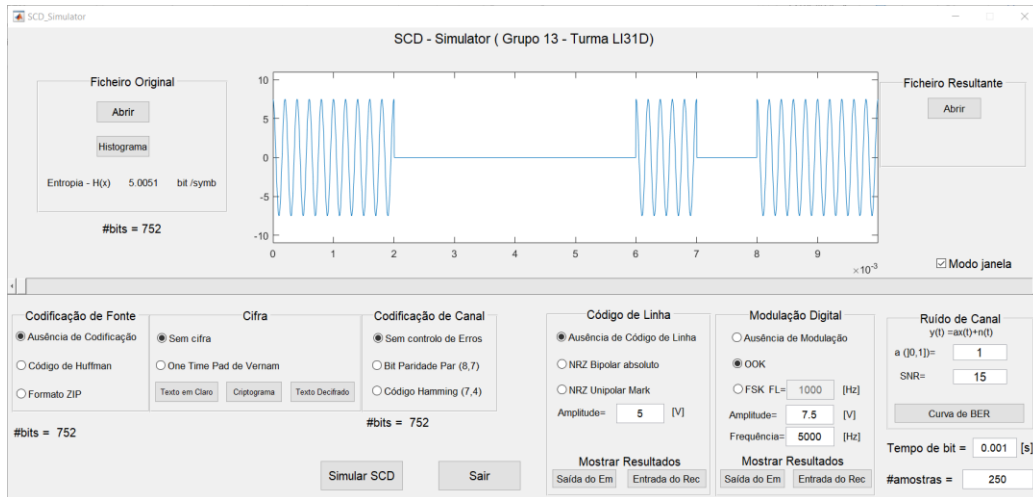


Figura 11 - Emissão de dados com modulação OOK

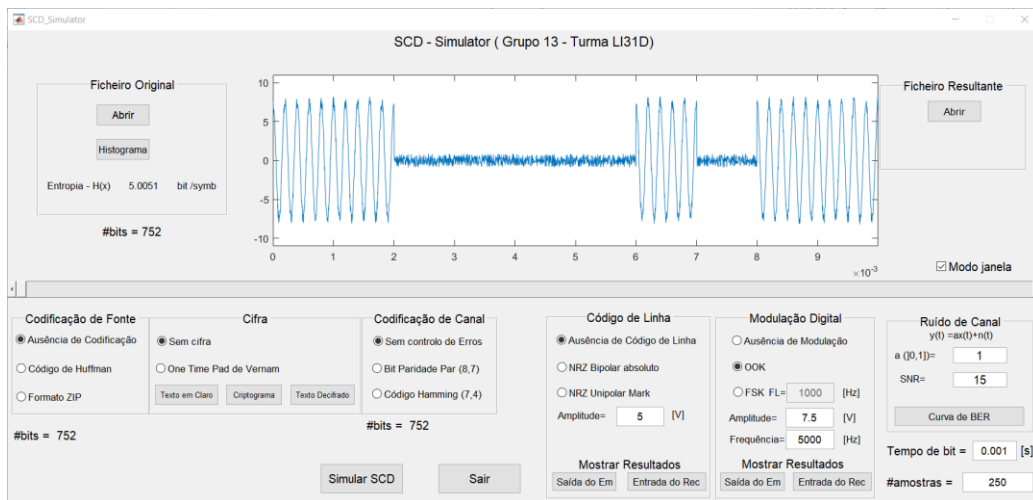


Figura 12 - Receção de dados com modulação OOK

Na Figura 11 é possível verificar a emissão de dados com modulação OOK com uma amplitude de 7,5 a -7,5 Volts, com um tempo de bit de 1 milissegundo e uma frequência de 5KHz. Na Figura 12 é possível verificar a receção dos mesmos dados, mas com uma adição de ruído na amplitude, no entanto, é visivelmente possível perceber os bits a 0 e a 1. Na Figura 13 está representado o ficheiro resultante da transmissão.

```
CLA_a.txt x +
1 Comunicações - ficheiro de teste.
2 1234567890
3 The quick brown fox jumps over the lazy dog.
```

Figura 13 - Ficheiro resultante de uma transmissão com modulação OOK

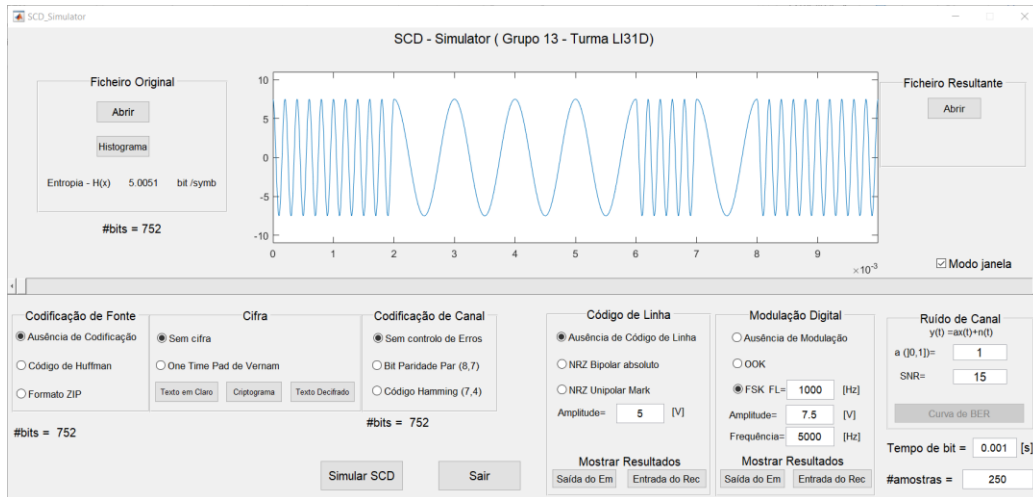


Figura 14 - Emissão de dados com modulação FSK

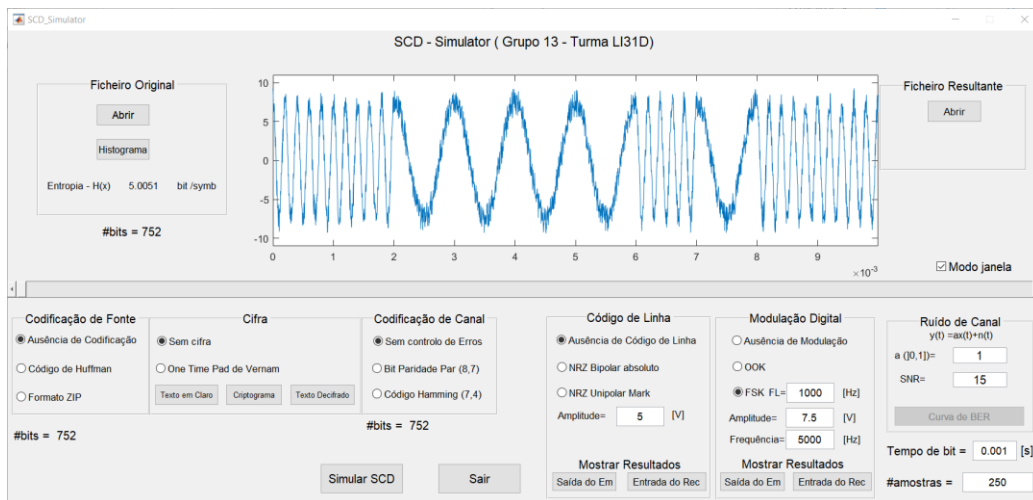


Figura 15 - Receção de dados com modulação FSK

Na Figura 14 é possível verificar a emissão de dados com modulação OOK com uma amplitude de 7,5 a -7,5 Volts, com um tempo de bit de 1 milissegundo e uma frequência de 5KHz com bits a 1 e 1KHz com bits a 0. Na Figura 15 é possível verificar a receção dos mesmos dados, mas com uma adição de ruído na amplitude, no entanto, é visivelmente possível perceber os bits a 0 e a 1. Na Figura 16 está representado o ficheiro resultante da transmissão.

```
CLA_a.txt
1 | Comunicações - ficheiro de teste.
2 | 1234567890
3 | The quick brown fox jumps over the lazy dog.
```

Figura 16 - Ficheiro resultante de uma transmissão com modulação FSK

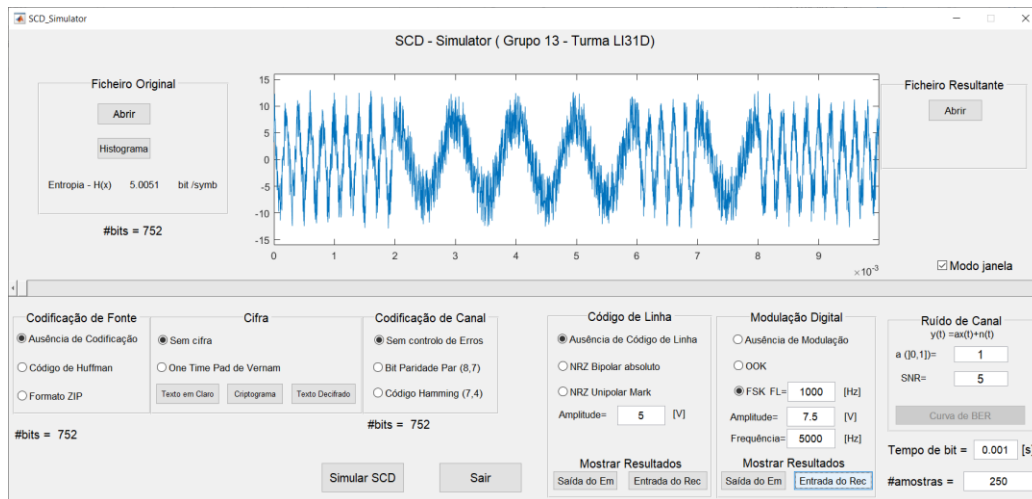


Figura 17 - Receção de dados com modulação FSK com valor de SNR alterado

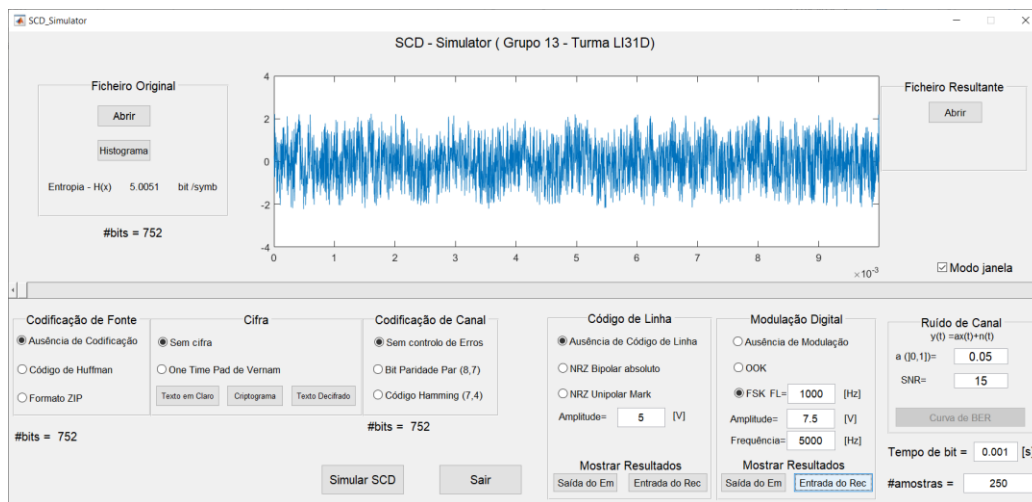


Figura 18 - Receção de dados com modulação FSK com valor de atenuação do canal alterado

As Figuras 17 e 18 representam a funcionalidade dos parâmetros do ruído de canal com o mesmo sinal usado da experiência das Figuras 15 e 16. Na Figura 17, é possível verificar que a amplitude do ruído adicionada é maior, baixando a relação de sinal-ruído, no entanto ainda é possível distinguir os bits 0 e 1. Na Figura 18, o valor da atenuação foi alterado para um valor muito baixo que é impossível distinguir os bits 0 dos bits 1.

2.3 Alínea C

2.3.1 Transmissão de um ficheiro de texto

A primeira experiência da simulação é uma transmissão de ficheiro de texto é aplicada ao ficheiro “a.txt”. Este vai sofrer uma codificação com código de Huffman, com a cifra One Time Pad de Vernam e com o código de Hamming (7, 4). Como explicado no trabalho anterior, cada uma destas funcionalidades lê de um ficheiro e escreve num novo, ou seja, na função de aplicar o código de linha, o nome do ficheiro a passar é diferente do que no início da codificação. Neste caso, o ficheiro é “H74C_OTPC_HufC_a.txt”. A emissão dos dados deste ficheiro está representada, uma parte, na Figura 19. Na Figura 20, é demonstrado o mesmo sinal no momento da receção após passar por um canal ruidoso. É possível verificar que a olho é muito difícil distinguir os bits 0 dos bits 1.

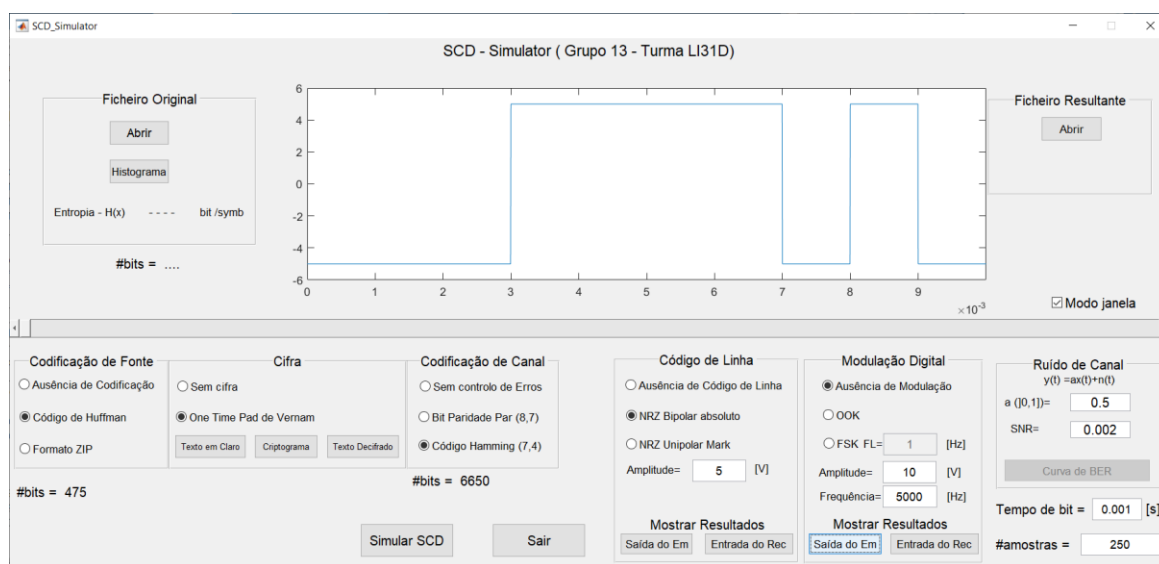


Figura 19 - Sinal de emissão do ficheiro de texto "a.txt" codificado

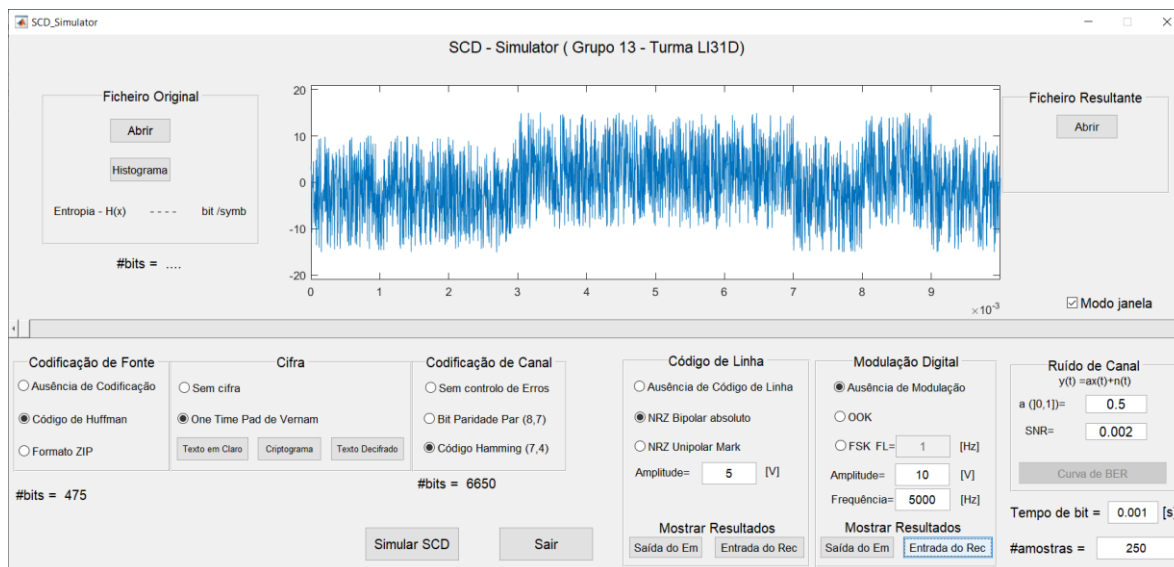


Figura 20 - Sinal de receção do ficheiro de texto "a.txt" codificado

Foi verificado e comparado o ficheiro original com o ficheiro transmitido, e conclui-se que o ficheiro foi transmitido com sucesso e sem erros. Na Figura 21 está representada o conteúdo do ficheiro original e na Figura 22 o ficheiro transmitido e como se pode verificar, o conteúdo é o mesmo.

```

HufD_OTPD_H74D_NRZBA_H74C_OTPC_HufC_a.txt a.txt
1 Comunicações - ficheiro de teste.
2 1234567890
3 The quick brown fox jumps over the lazy dog.

```

Figura 21 - Conteúdo do ficheiro original

```

HufD_OTPD_H74D_NRZBA_H74C_OTPC_HufC_a.txt +
1 Comunicações - ficheiro de teste.
2 1234567890
3 The quick brown fox jumps over the lazy dog.

```

Figura 22 - Conteúdo do ficheiro transmitido

2.3.1 Transmissão de um ficheiro de imagem

A segunda experiência da simulação é uma transmissão de ficheiro de imagem é aplicada ao ficheiro “bird.gif”, pois esta imagem não ocupa muito espaço. Este vai sofrer uma codificação com código de Huffman, com a cifra One Time Pad de Vernam e com o código de Hamming (7, 4). A emissão dos dados deste ficheiro está representada, uma parte, na Figura 23 e em comparação com a experiência anterior, o número de amostras teve que ser menor, para ocupar menos espaço na memória. Na Figura 24, é demonstrado o mesmo sinal no momento da receção após passar por um canal ruidoso. É possível verificar que a olho é muito difícil distinguir os bits 0 dos bits 1.

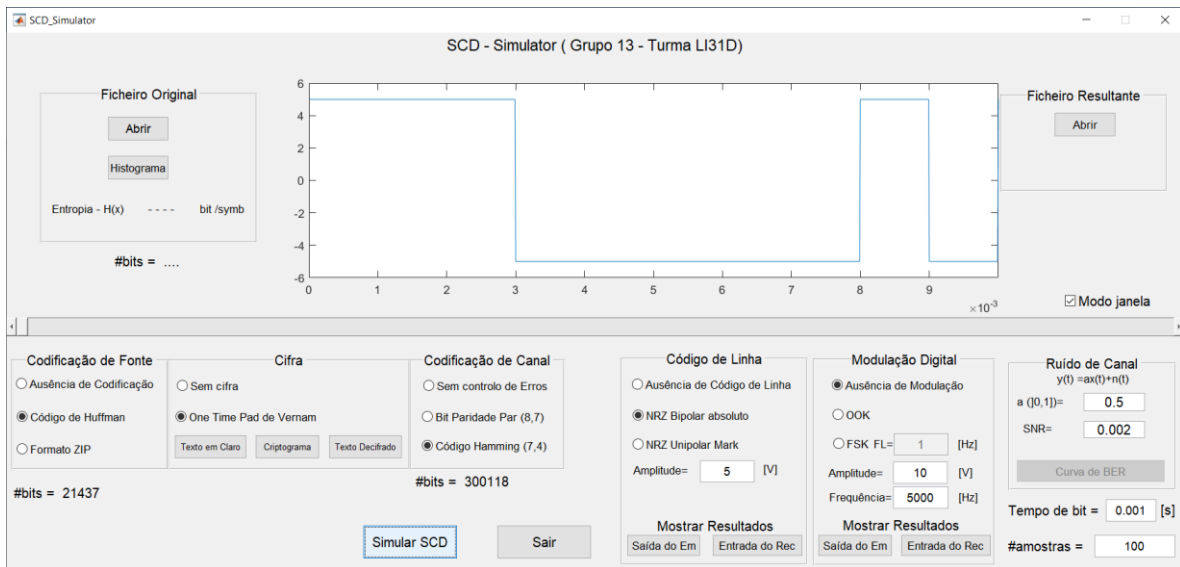


Figura 23 - Sinal de emissão do ficheiro de texto "bird.gif" codificado

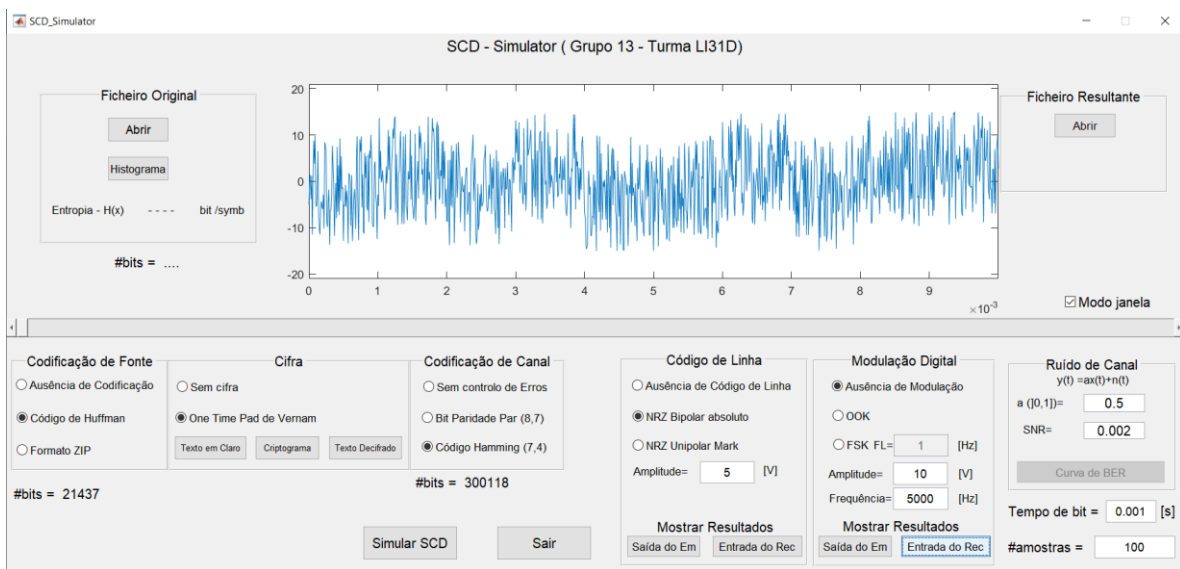


Figura 24 - Sinal de receção do ficheiro de texto "bird.gif" codificado

Como este ficheiro é maior que o ficheiro de texto, implica que seja necessário um número maior de bits a enviar e também um maior tempo de transmissão. Com isto, a probabilidade de haver 1 bit errado durante a transmissão é maior e, por acaso, nesta transmissão houve um erro que influenciou na tentativa de decodificação de Huffman encontrou um símbolo que não constava no dicionário que por consequência gerou um erro, como pode ser verificado na Figura 25, não sendo possível gerar um ficheiro resultante.

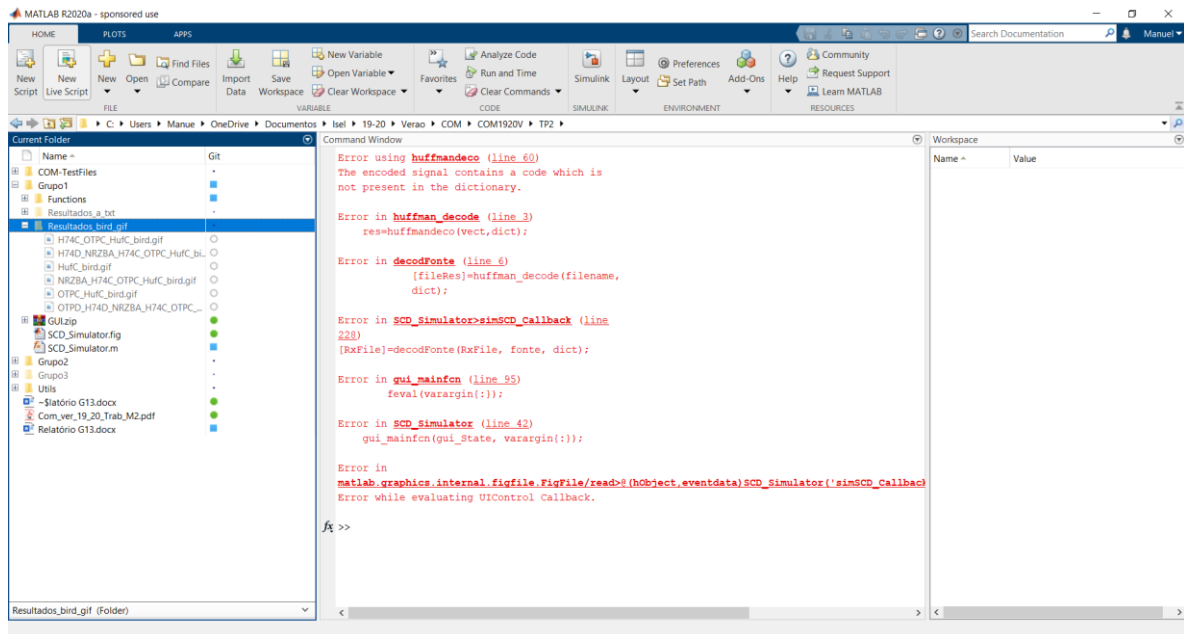


Figura 25 - Resultado da tentativa de decodificar o sinal recebido

3. Grupo 2

Neste grupo é pretendido avaliar a taxa de erro de várias transmissões do mesmo género, com valores de SNR e atenuação diferentes.

3.1 Cálculo da curva de BER

Para calcular a curva de BER, foram adicionados à estrutura da GUI 3 vetores:

1. Vetor que define o tipo de transmissão a ser feita;
2. Os valores da SNR de várias transmissões;
3. Os valores de BER de várias transmissões;

O primeiro vetor é definido sempre que se faz uma simulação. Um tipo de transmissão é definido pelas opções de código de linha e modulação digital. Sempre que for detetado uma simulação com um tipo de transmissão diferente à anterior, os vetores pares de BER e SNR são reiniciados.

O segundo vetor é definido a partir da concatenação dos valores de SNR que o utilizador usar em todas as transmissões do mesmo tipo.

O terceiro vetor é definido a partir do resultado da chamada da função *calculate_transmission_BER* que recebe o nome do ficheiro à entrada do emissor, o nome do ficheiro à saída do recetor e o vetor da BER. Esta função conta o número de bits trocados e calcula a taxa de erro em relação ao número de bits do primeiro ficheiro. No entanto, existe a possibilidade de haver mais ou menos bits no ficheiro recebido, nos casos onde supostamente devia de haver o bit de transição e houve uma troca de bits num bloco com os bits todos iguais e, conseqüentemente, não é ignorado o bit de transição, por isso, a diferença do número de bits dos dois ficheiros é contabilizado para a taxa de erro.

3.2 Resultados experimentais

Todas as experiências que se vão efetuar neste grupo irão ter sempre o mesmo tipo de transmissão e com valores de SNR [0,1; 0,2; 0,5; 0,75; 1; 1,5; 2; 5] e sem atenuação, NRZ Bipolar Absoluto com uma amplitude de 5 Volts, uma modulação digital OOK com amplitude de 7,5 Volts e uma frequência de 5KHz, 1 milissegundo de tempo de bit e 100 número de amostras. O ficheiro de texto e de imagem, são os mesmos que foram usados na experiência anterior. Os resultados experimentais apenas demonstram 10 bits, e não os bits todos da transmissão.

3.2.1 Transmissão de um ficheiro de texto com código de Hamming(7, 4)

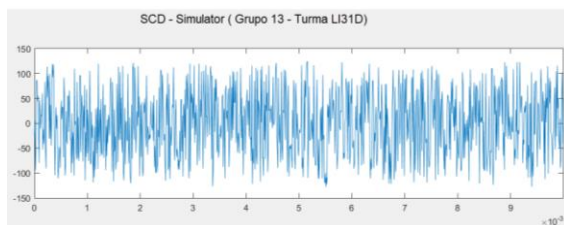


Figura 26 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=0,1

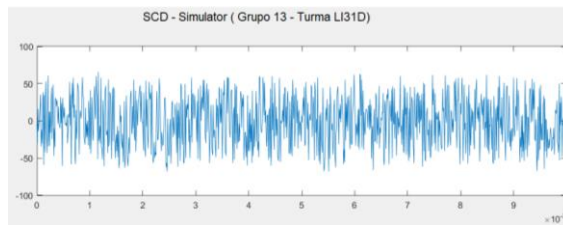


Figura 27 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=0,2

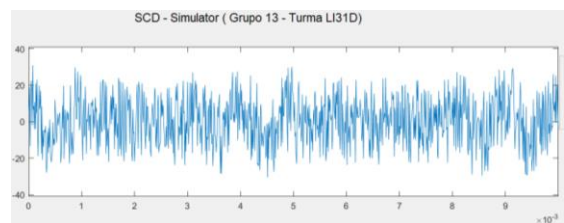


Figura 28 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=0,5

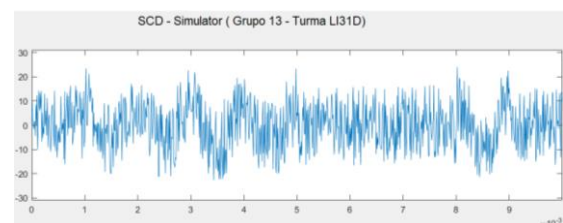


Figura 29 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=0,75

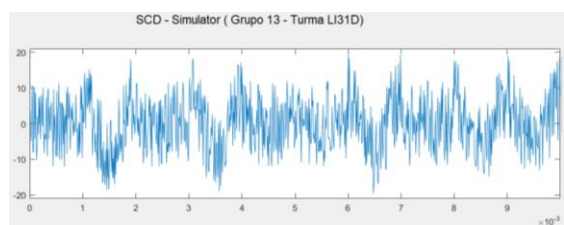


Figura 30 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=1

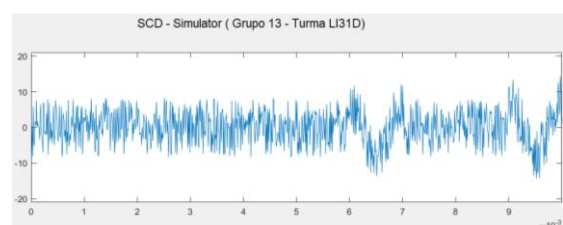


Figura 31 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=1,5

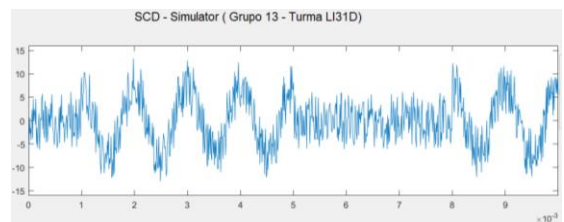


Figura 32 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=2

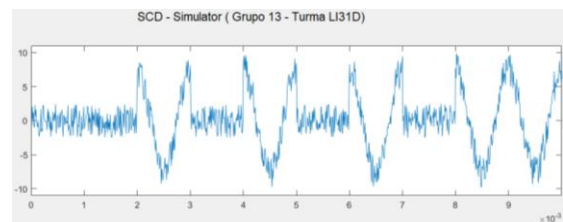


Figura 33 - Sinal do Recetor de texto com código Hamming(7, 4) e com SNR=5

Como se pode verificar, das Figuras 26 a 30, o valor de SNR é tão baixo que provoca um canal muito ruidoso que resulta na impossibilidade de perceber que bits estão a ser transmitidos. A partir da Figura 31, já começa a ser possível perceber e visualizar alguns bits, no entanto continua a ser muito ruidoso. Na Figura 32, já é muito perceptível e diferenciar os bits 0 e os bits a 1, no entanto o sinal ainda tem muito ruído, podendo provocar mesmo assim alguma troca de bits. Na Figura 33, o valor de SNR é tão alto, que muito dificilmente existe troca de bits.

Estes resultados podem ser verificados também na Figura 34, onde a curva de BER demonstra que entre os valores de SNR 0,1 e 0,75, a taxa de troca de bits é superior ou igual a 50%. Quando os valores de SNR estão entre 1 e 2, a taxa é entre os 20 e os 40%. No final, quando o SNR tem um valor de 5 é possível verificar que não houve troca de bits na transmissão. Foi possível completar a transmissão com sucesso com SNR a 2 e a 5, no entanto com o SNR a 2 o ficheiro veio diferente ao original enquanto que com a transmissão com SNR a 5, o ficheiro recebido veio com o mesmo conteúdo que o da fonte.

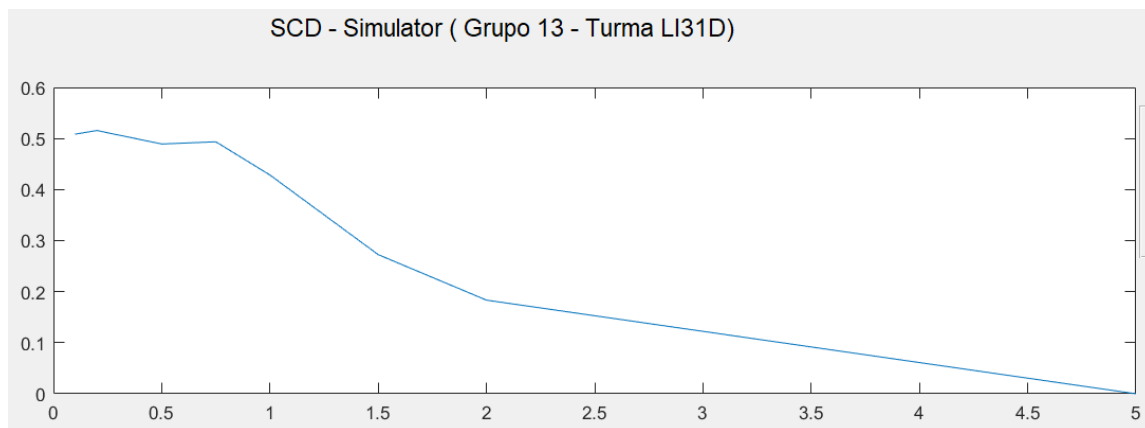


Figura 34 - Curva de BER da transmissão de um ficheiro de texto com código Hammin (7, 4)

3.2.2 Transmissão de um ficheiro de texto sem código de Hamming(7, 4)

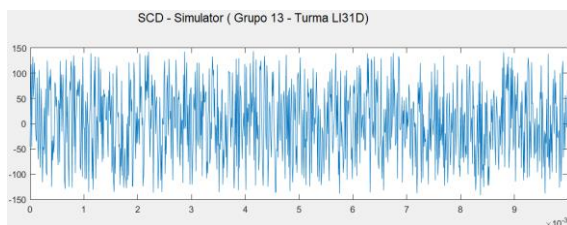


Figura 35 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=0,1

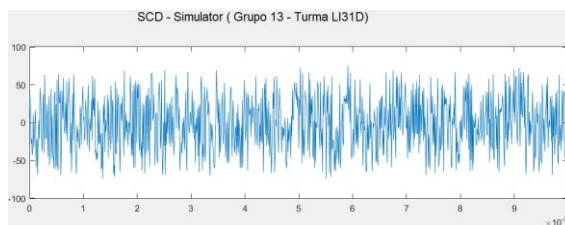


Figura 36 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=0,2

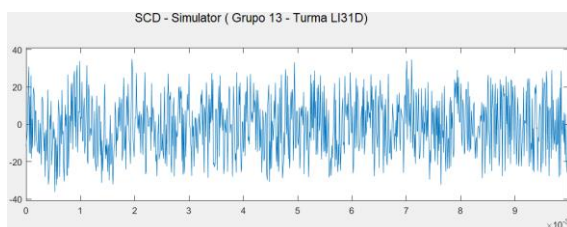


Figura 37 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=0,5

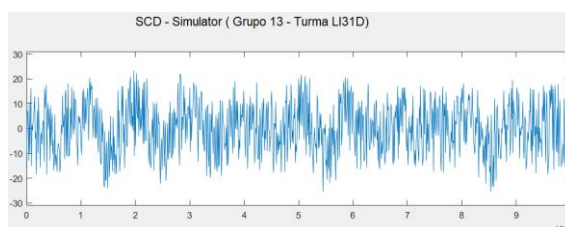


Figura 38 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=0,75

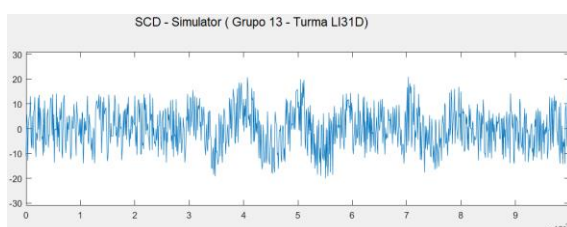


Figura 39 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=1

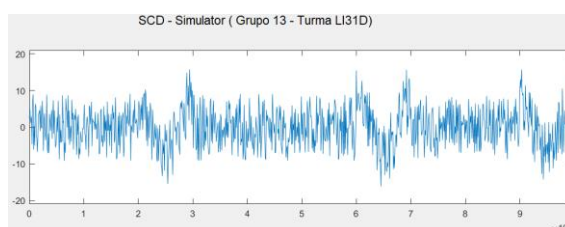


Figura 40 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=1,5

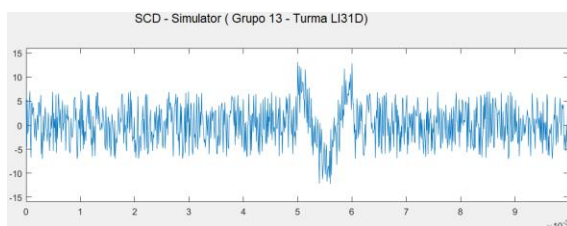


Figura 41 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=2

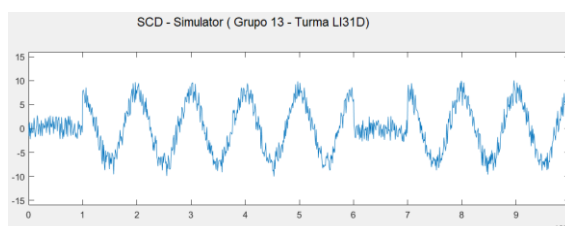


Figura 42 - Sinal do Recetor de texto sem código Hamming(7, 4) e com SNR=5

Como na experiência anterior, os primeiros quatro resultados têm um SNR muito baixo que provoca um canal muito ruidoso e impossibilita perceber que bits estão a ser transmitidos. As Figuras 39 e 40, já permitem perceber e visualizar alguns bits, no entanto, o sinal continua a ter muito ruído. Na Figura 41, já é muito perceptível e diferenciar os bits 0 e os bits a 1, no entanto o sinal ainda tem muito ruído, podendo provocar mesmo assim alguma troca de bits. Na Figura 42, o valor de SNR é tão alto, que muito dificilmente existe troca de bits.

Estes resultados podem ser verificados também na Figura 43, onde a curva de BER demonstra que entre os valores de SNR 0,1 e 1, a taxa de troca de bits é superior ou igual a 40%. Quando o valor de SNR for 2, a taxa é menor que 10%, no entanto, como não existe nem controlo nem correção, na altura da descodificação de Huffman, houve um símbolo que não pertencia ao dicionário, não sendo possível completar a transmissão. No final, quando o SNR tem um valor de 5 é possível verificar que não houve troca de bits na transmissão e o ficheiro recebido tinha o mesmo conteúdo que o original.

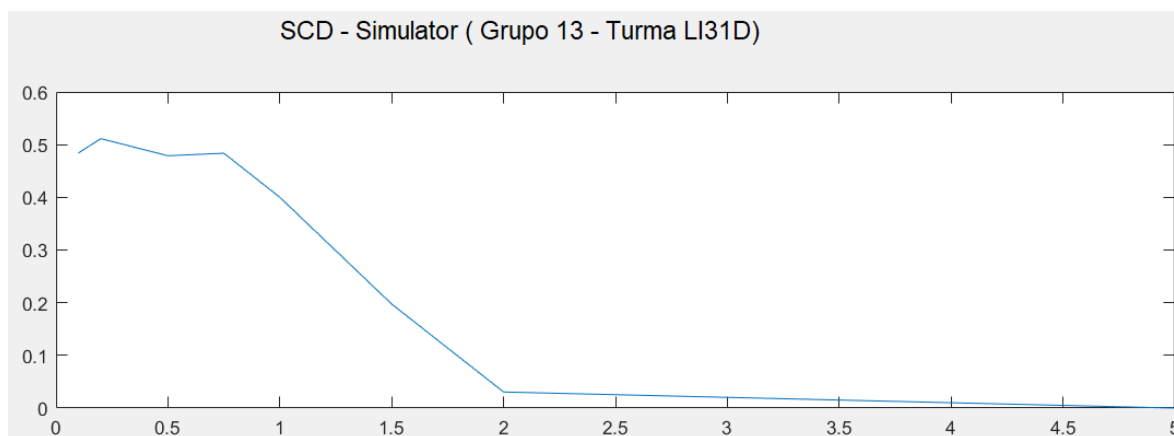


Figura 43 - Curva de BER da transmissão de um ficheiro de texto sem código Hammin (7, 4)

3.2.3 Transmissão de um ficheiro de imagem com código de Hamming(7, 4)

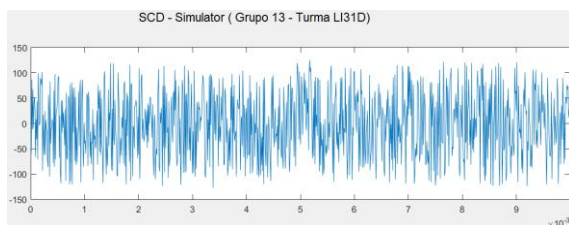


Figura 44 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=0,1

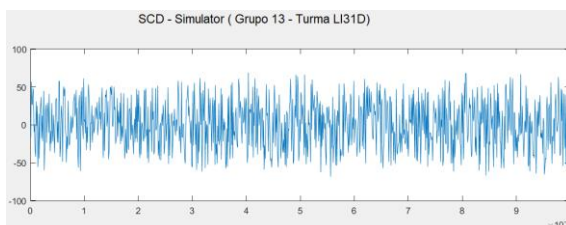


Figura 45 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=0,2

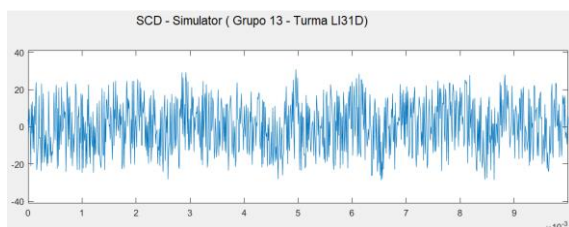


Figura 46 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=0,5

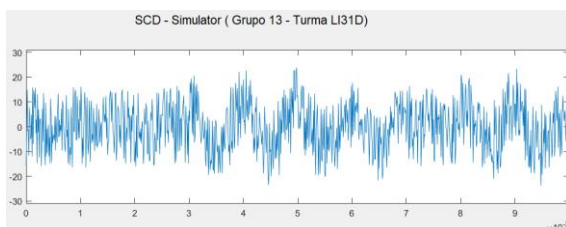


Figura 47 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=0,75

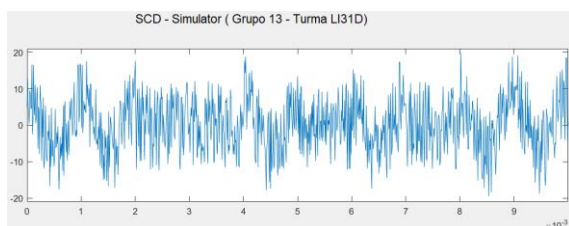


Figura 48 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=1

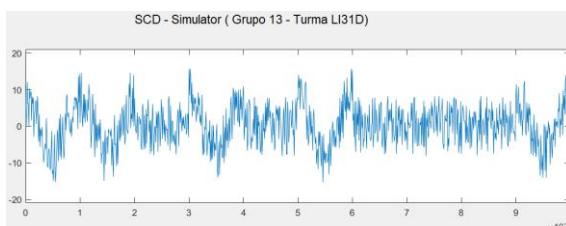


Figura 49 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=1,5

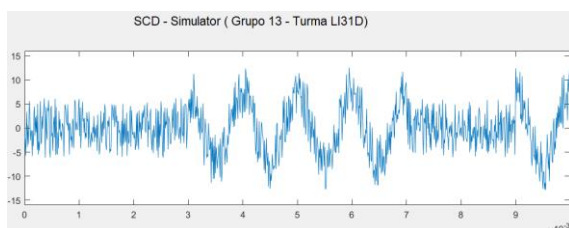


Figura 50 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=2

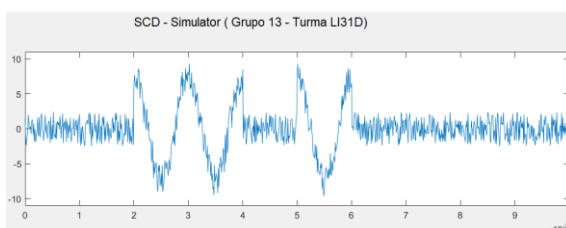


Figura 51 - Sinal do Recetor de imagem com código Hamming(7, 4) e com SNR=5

Nesta experiência, como a imagem contém mais bits que o ficheiro de texto, é expectável que haja mais trocas de bits. Os primeiros quatro resultados têm um comportamento idêntico às duas últimas experiências, com o valor de SNR muito baixo. As Figuras 48 e 49, já permitem perceber e visualizar alguns bits, no entanto, o sinal continua a ter muito ruído. Na Figura 50, já é mais perceptível e é

possível diferenciar com mais facilidade os bits 0 e os bits a 1, no entanto o sinal ainda tem muito ruído, podendo provocar mesmo assim alguma troca de bits. Na Figura 51, o valor de SNR é tão alto, que muito dificilmente existe troca de bits.

Estes resultados podem ser verificados também na Figura 52, onde a curva de BER demonstra que entre os valores de SNR 0,1 e 2, a taxa de troca de bits é superior ou igual a 50%. Isto deve-se ao fato de haver a possibilidade de trocar mais bits, pois este ficheiro tem 3KBytes enquanto que o outro tem 94 bytes. Só com valores grandes de SNR, como o valor 5, é possível transmitir sem problemas o ficheiro.

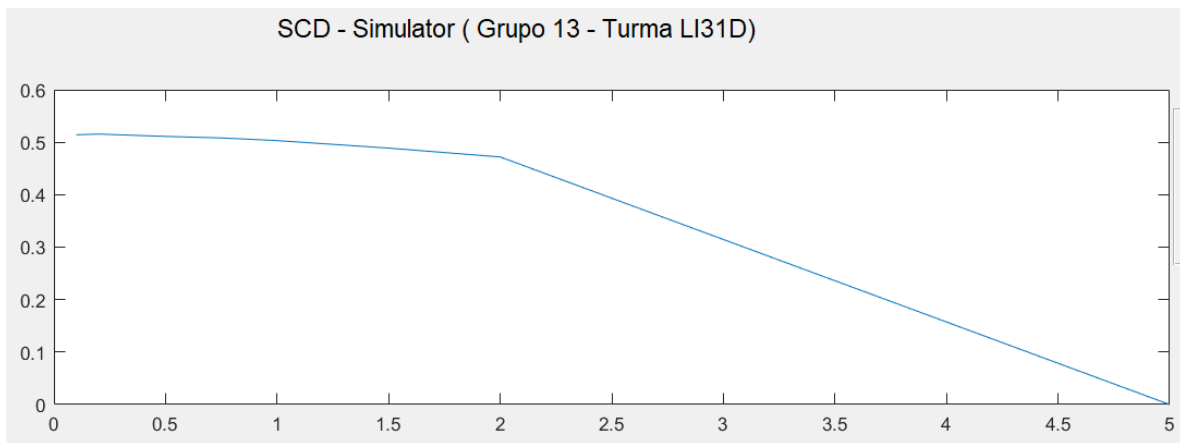


Figura 52 - Curva de BER da transmissão de um ficheiro de imagem com código Hammin (7, 4)

3.2.4 Transmissão de um ficheiro de imagem sem código de Hamming(7, 4)

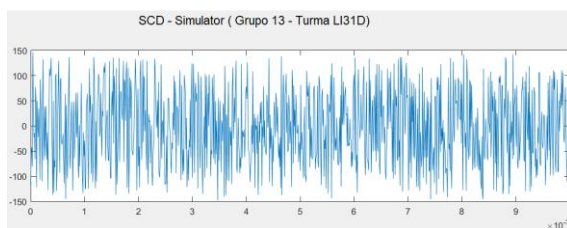


Figura 53 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=0,1

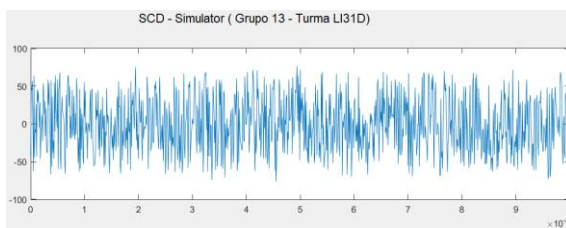


Figura 54 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=0,2

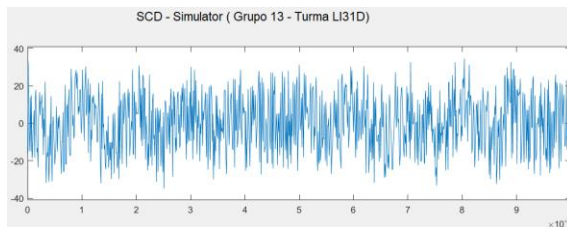


Figura 55 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=0,5

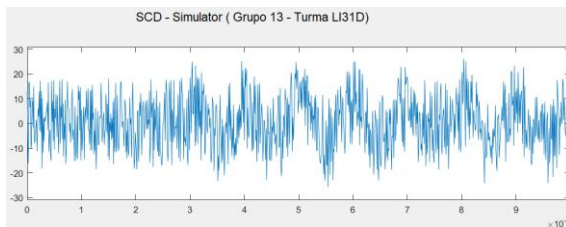


Figura 56 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=0,75

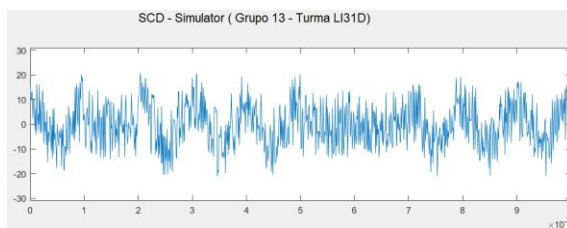


Figura 57 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=1

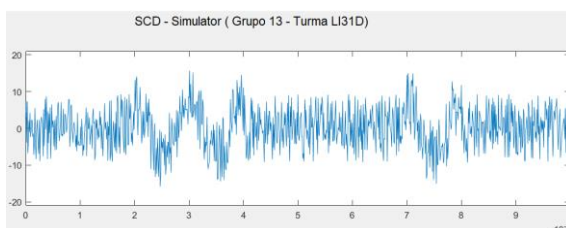


Figura 58 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=1,5

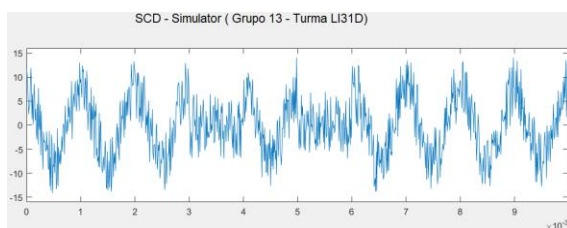


Figura 59 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=2

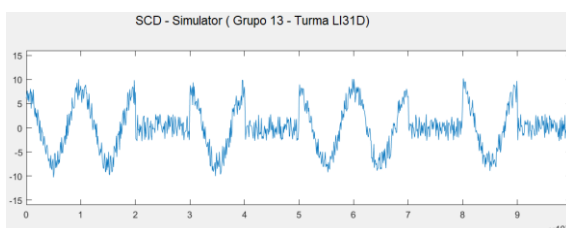


Figura 60 - Sinal do Recetor de imagem sem código Hamming(7, 4) e com SNR=5

Como foi visto nas três experiências anteriores, os primeiros quatros resultados demonstram um sinal impercetível, não sendo possível distinguir o bit 0 do 1. As Figuras 57 e 58, já permitem perceber e visualizar alguns bits, no entanto, o sinal continua a ter muito ruído. Na Figura 59, já é possível distinguir com mais facilidade os bits 0 e os bits a 1, no entanto o sinal ainda tem muito ruído, podendo provocar mesmo assim alguma troca de bits. Na Figura 60, o valor de SNR é tão alto, que muito dificilmente existe troca de bits.

Estes resultados podem ser verificados também na Figura 61, onde a curva de BER demonstra que entre os valores de SNR 0,1 e 2, a taxa de troca de bits é superior ou igual a 50%. Quando o valor de SNR for 5, é possível transmitir sem problemas o ficheiro, não havendo trocas de bits.

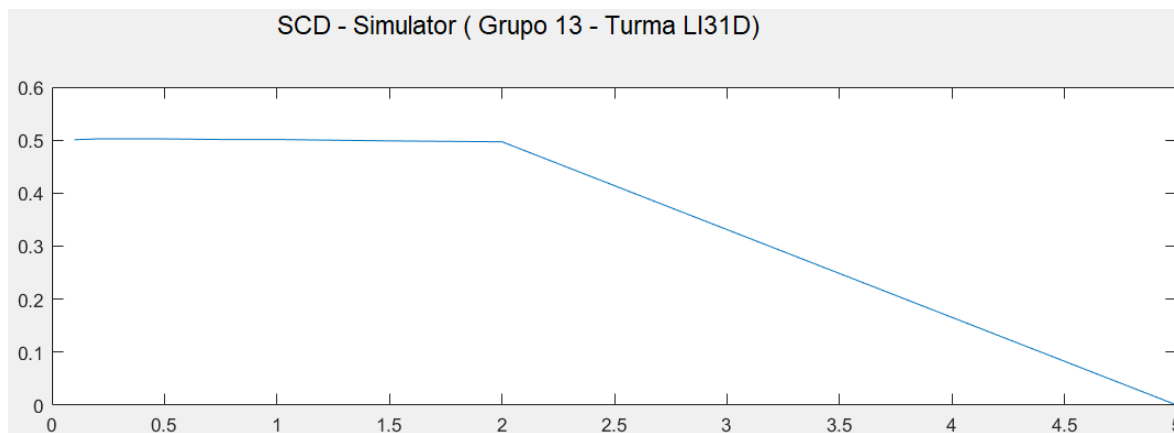


Figura 61 - Curva de BER da transmissão de um ficheiro de imagem sem código Hammin (7, 4)

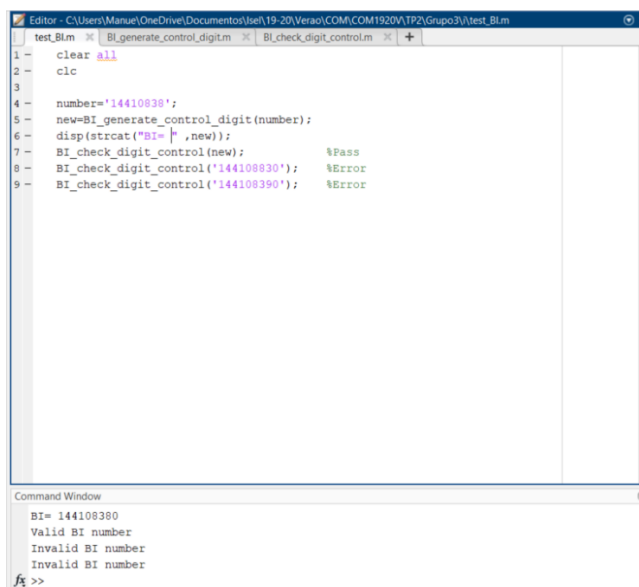
4. Grupo 3

Neste grupo é pretendido fazer funções que contenham algoritmos de verificação de erros, contextualizados na validação do dígito de controlo do Bilhete de Identidade e do ISBN. Apesar de se poder usar qualquer linguagem de programação, foi utilizado à mesma o MATLAB.

4.1 Alínea (i)

Para fazer o algoritmo de cálculo do dígito de controlo, foi criada a função *BI_generate_control_digit*. Esta recebe como parâmetro de entrada uma string com 8 caracteres, sendo estes todos números. Para obter o dígito de controlo, foi utilizada a fórmula $b_0 = x + t$, onde x é calculado a partir de um ciclo e t foi calculado usando esta fórmula $t = 11 - (x \% 11) + x$.

Para fazer o algoritmo de verificação, foi criada a função *BI_check_digit_control*, que tem a capacidade de verificar se o dígito de controlo está correto e se todos os algarismos do BI estão bem e na ordem correta. Esta função recebe uma string com 9 caracteres, sendo estes todos números. Para fazer a verificação é usado o algoritmo do enunciado, recorrendo a uma instrução cíclica e no final é verificado se o resto da divisão do resultado com 11 é 0. Se for 0, então o número do BI é válido, caso contrário é inválido. A Figura 62 demonstra a funcionalidade das duas funções através de um script de teste.



```
Editor - C:\Users\Manuel\OneDrive\Documents\Isef\19-20\Verap\COM\COM1920\TP2\Grupo3\test_BI.m
test_BI.m  BI_generate_control_digit.m  BI_check_digit_control.m  +
1 - clear all
2 - clc
3
4 - number='14410838';
5 - new=BI_generate_control_digit(number);
6 - disp(strcat('BI= ',new));
7 - BI_check_digit_control(new);           %Pass
8 - BI_check_digit_control('144108830'); %Error
9 - BI_check_digit_control('144108390'); %Error

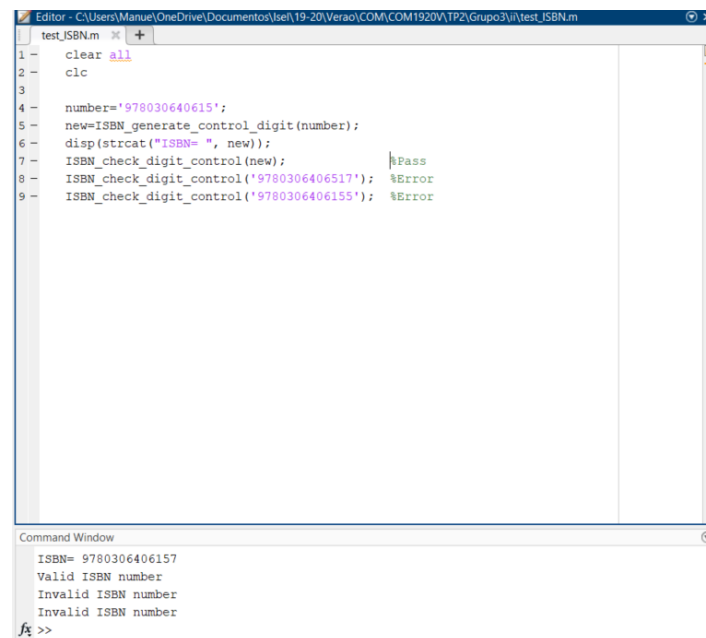
Command Window
BI= 144108380
Valid BI number
Invalid BI number
Invalid BI number
f6 >>
```

Figura 62 - Resultados do script de testes das funções de BI

4.2 Alínea (ii)

Para fazer o algoritmo de cálculo do dígito de controlo, foi criada a função *ISBN_generate_control_digit*. Esta recebe como parâmetro de entrada uma string com 12 caracteres, sendo estes todos números. Para obter o dígito de controlo, foi utilizada a fórmula $b_0 = 10 - (x \% 10)$, onde x é calculado a partir de um ciclo onde é verificado a paridade de cada algarismo e se o algarismo for par, soma-se o seu valor ao resultado, caso contrário é o triplo.

Para fazer o algoritmo de verificação, foi criada a função *ISBN_check_digit_control*, que tem a capacidade de verificar se o dígito de controlo está correto e se todos os algarismos do ISBN estão bem e na ordem correta. Esta função recebe uma string com 13 caracteres, sendo estes todos números. Para fazer a verificação é usado o mesmo algoritmo que a função de geração do dígito de controlo, recorrendo à instrução cíclica. Depois é utilizada a fórmula $d = (10 - r) \% 10$, onde r é o resultado da instrução cíclica. Se o valor de d for diferente que 0, quer dizer que o valor de ISBN é inválido, caso contrário então é válido. A Figura 63 demonstra um script de teste onde é gerado o dígito de controlo e são feitas várias verificações a diferentes ISBN, que incluem um ISBN correto, uma troca de ordem entre 2 dígitos e um dígito errado.



```
Editor - C:\Users\Manue\OneDrive\Documentos\Ise\19-20\Verão\COM\COM1920\TP2\Grupo3\ii\test_ISBN.m
test_ISBN.m
1 - clear all
2 - clc
3
4 - number='978030640615';
5 - new=ISBN_generate_control_digit(number);
6 - disp(strcat("ISBN= ", new));
7 - ISBN_check_digit_control(new); %Pass
8 - ISBN_check_digit_control('9780306406517'); %Error
9 - ISBN_check_digit_control('9780306406155'); %Error

Command Window
ISBN= 9780306406157
Valid ISBN number
Invalid ISBN number
Invalid ISBN number
fx >>
```

Figura 63 - Resultados do script de testes das funções de ISBN