



**Universidade Federal do Ceará
Centro de Tecnologia
Departamento de Engenharia de Teleinformática
Introdução a Teoria da Informação – TI0056**

Avaliação Parcial 02 - Simulação

Aluno	Diego Cartaxo Colaço - 385450 Filipe Tavares Sores - 389240 Kenneth Brenner dos Anjos Benício – 385468 Lucas de Souza Abdalah - 385472
Professor	Walter da Cruz Freitas Júnior

Fortaleza, 03 de dezembro de 2018

Sumário

1 Simulação	1
1.1 Questão 5	1
Referências	8

1 Simulação

1.1 Questão 5

O canal de comunicação faz a ponte entre transmissor e receptor no envio da informação, mas por imperfeições e/ou limitações nos meios de transmissão o canal introduz no sinal transmitido erros que devem ser mitigados. A mitigação desses erros se dá, por exemplo, por meio da codificação de canal que introduz redundância de forma controlada com o objetivo de detectar e corrigir os erros introduzidos pelo canal de comunicação. Nesse trabalho pretende-se transmitir a imagem *lena.jpg* com resolução 512×512 em escala de cinza (grayscale 0-255) em diversas situações de um canal de comunicação com e sem a utilização de um codificador de canal. Usando um codificador de comprimento fixo codificar a imagem *lena.jpg*. O seu programa deve:

1. Acrescentar o canal BSC em função do erro binário p_e . Comparar as imagens, original e a decodificada, para os seguintes valores de $p_e = \{0\%, 25\%, 50\%, 75\% \text{ e } 100\%\}$. Comente os resultados obtidos e suas conclusões.

Solução:

```

1 lena = imread('lenaTest2.jpg'); %adquire imagem Lena como matriz
2 tam = size(lena); %obter tamanho da matriz da imagem
3 lena_v = double(reshape(lena,tam(1)*tam(2),1)); %reordena a matriz como um vetor
4 lena_b = de2bi(lena_v,8); %converte de decimal para binário
5 tam_b = size(lena_b); %obtem tamanho da nova matriz com dados em binário
6 pe = 0:.25:1; %vetor com probabilidades de erro de canal pe
7 bsc_im = zeros([tam_b length(pe)]); %pré alocando array com imagens pós passagem pelo
    %canal
8 for p = 1:length(pe) %laço para passagens do canal aumentando gradativamente a
    %probabilidade de erro de transmissão
    bsc_im (:,:,p) = bsc(lena_b,pe(p)); %função que simula canal bsc com para um erro
    %pe
9 end
10 for p = 1:length(pe) %laço para vizualizar resultados
11     im = uint8(reshape(bi2de(bsc_im (:,:,p)),tam)); %transformando imagens para seu
        %formato otiginal
12     figure %plotando resultados
13     imshow(im)
14     xlabel(['$p_e = ' num2str(pe(p)*100) '\%' ], 'Interpreter', 'Latex', 'FontSize',
        ,18) %legenda
15     saveas(gcf,[num2str(pe(p)*100) 'errolena.pdf']) %salvando arquivo
16 end
17

```

A medida que a probabilidade de erro p_e aumenta é possível ver existe uma tendência da inversão das cores da imagem, porém a noção de ruído não aumenta linearmente com o erro. Visto que, uma simples “inversão” dos bits da ultima imagem nos forneceria a primeira, pode-se afirmar que o canal não prejudicaria a qualidade da transmissão. Mas, quando o erro se aproxima da faixa de 50% é possível identificar a imagem mais ruidosa. Esse comportamento revela que quanto “mais aleatório” for o canal ou mais equiprováveis forem as chances de erro mais ruído teremos. Assim, também quando visualizamos a entropia e a capacidade de um canal BSC, figuras (2) e (3), vemos que a medida que a probabilidade p_e se aproxima de 50% temos uma capacidade muito reduzida do canal e um valor máximo de entropia, o que nos permite concluir que para uma qualidade satisfatória de transmissão o canal deve ser de baixa entropia, ou haver uma codificação de canal eficiente para “contornar” o ruído.

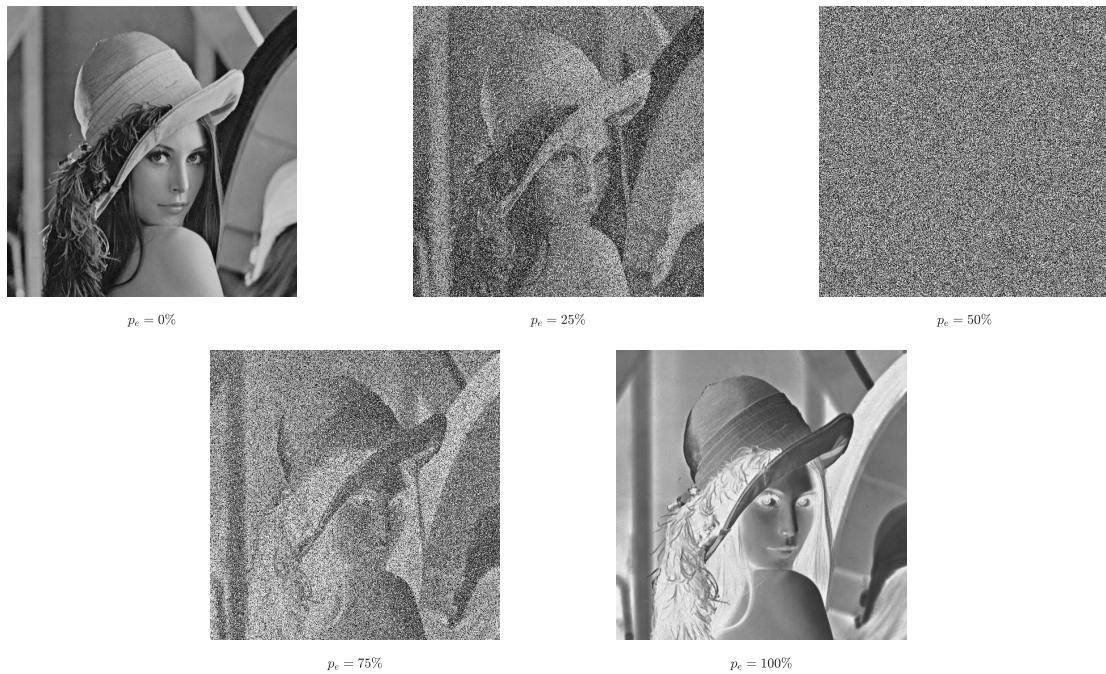
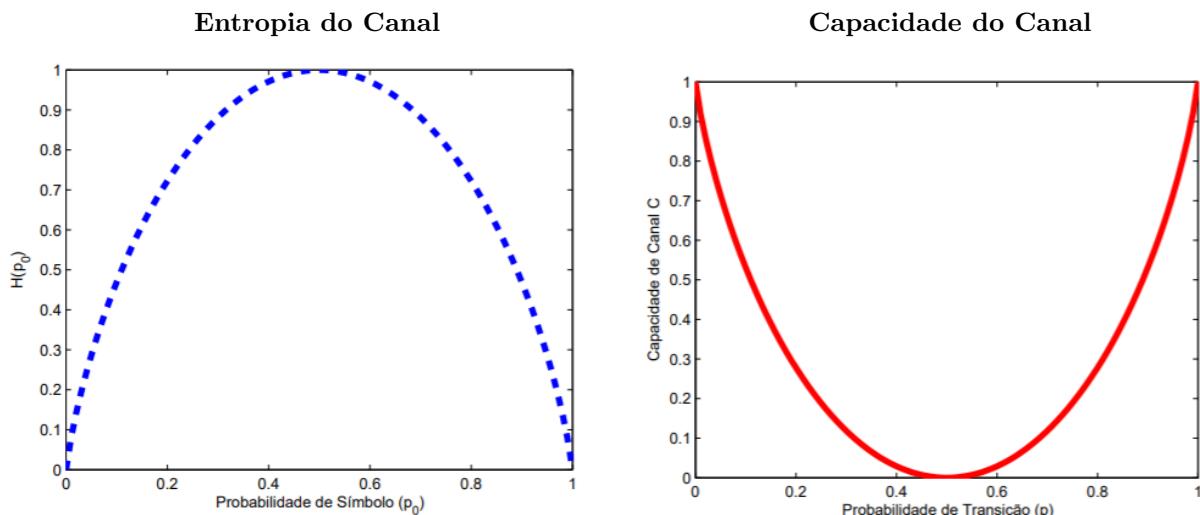
Figura 1: Transmissão da Imagem pelo canal BSC com probabilidade de erro p_e .

Figura 2: Notas de Aula (TI0056)

Figura 3: Notas de Aula (TI0056)

2. Com a finalidade de melhorar o desempenho do sistema de comunicação proposto, implemente o código de repetição de taxa $R_c = \frac{1}{n+1}$. Para uma taxa de erro igual a 25%, monte a curva da taxa de erro de bit (BER) no receptor em escala logarítmica versus o tamanho da palavra código ($n = \{2, 4, 8, 10, 12, 14, 16\}$). Comente os resultados obtidos e suas conclusões.

Solução:

```

1 lena = imread('lenaTest2.jpg'); %adquire imagem Lena como matriz
2 tam = size(lena); %obter tamanho da matriz da imagem
3 lena_v = double(reshape(lena,tam(1)*tam(2),1)); %reordena a matriz como um vetor
4 lena_b = de2bi(lena_v,8)'; %converte de decimal para binário
5 tam_b = size(lena_b); %obtem tamanho da nova matriz com dados em binário
6 lena_b = reshape(lena_b,[tam_b(1)*tam_b(2) 1]); %ajusta os bits para ficarem todos em
    uma coluna
7 tam_b = size(lena_b); %adquire as novas dimensões do vetor binário
8 pe = .25; %probabilidade de erro
9 ber = []; %pré alocando vetor com resultados ber
10 for n = 2:2:16 %laço para simulações aumentando gradualmente o número de bits
    repetidos
11     lena_n = repmat(lena_b,[1 n+1]); %repete os bits
12     tam_n = size(lena_n); %adquire novas dimensões
13     bsc_im = bsc(lena_n,pe); %simula canal bsc para um erro pe
14     v_erro = sum(bsc_im,2); %soma cada bloco de bits repetidos
15     resp = lena_n(:,1); %pré alocando resposta
16     for p = 1:tam_n(1) %laço para corrigir os bits
17         if v_erro(p) > (n+1)/2 %se a maioria dos bits de cada bloco for 1, resposta
            igual a 1
18             resp(p,1) = 1;
19         else %se a maioria dos bits de cada bloco for 0, resposta igual a 0
20             resp(p,1) = 0;
21         end
22     end
23     ber = [ber sum(abs(lena_b-resp),1)/tam_b(1)*100]; %calcula a ber
24     figure %plotando imagem resultante
25     im = uint8(bi2de(reshape(resp,[8 (tam_n(1))/8]))'); %colocando no formato
        original
26     im = reshape(im,tam); %colocando no formato original
27     imshow(im)
28     xlabel(['n = ' num2str(n) ', ' ' BER = ' num2str(ber(end)) '$\\%$' ' e $p_e=25\\$'
        ' ],'Interpreter','Latex','FontSize',14) %legenda
29     saveas(gcf,['pe25 ' 'n' num2str(n) 'lena.pdf']) %salvando arquivo
30 end
31 figure %plotando ber
32 loglog(2:2:16,ber./100)
33 ylabel('BER','Interpreter','Latex','FontSize',14)
34 xlabel('$n$', 'Interpreter', 'Latex', 'FontSize', 14)
35 grid on
36 saveas(gcf,'berxvsn.pdf')

```

Como se pode perceber ao observar as imagens, a medida que aumentamos o número de repetições o ruído começa a diminuir significativamente. Assim, menos poluída fica a imagem da mulher a medida que o ruído diminui. Isso é obviamente uma consequência da diminuição da taxa de erro de bit, a BER. O que indica que a medida que se aumenta o número de repetições para cada bit menores as chances de se obter uma interpretação errada.

Porém, como ilustrado na figura (6), para uma taxa de erro de 1,25% ainda ficam claramente perceptíveis as falhas de reprodução, sendo que foram necessárias 16 repetições para esse resultado. Portanto, esse tipo de solução só seria interessante para situações onde não houvesse a necessidade de grande qualidade de imagem e que os dados não fossem muito grandes, pois caso o contrário o sistema seria lento e facilmente sobrecarregado.

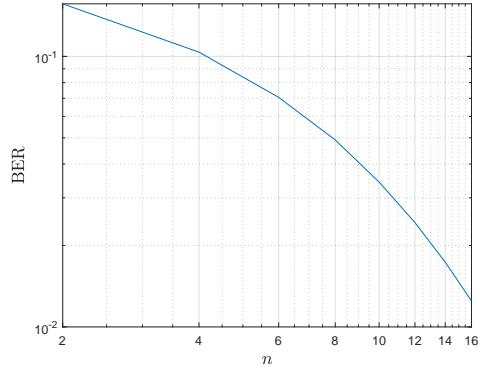


Figura 4: Variação da BER a com o crescimento de repetições do código



Figura 5: Transmissão da Imagem por Canal BSC com $p_e = 25\%$, Utilizando Códigos de Repetição.

Figura 6: Transmissão da Imagem por Canal BSC com $p_e = 25\%$, Utilizando Códigos de Repetição.

3. Considerando uma probabilidade de erro de $p_e = 45\%$ e usando um código de repetição, qual deve ser o máximo valor para a taxa do código de tal forma que se tenha uma probabilidade de erro desprezível no receptor. De acordo com a taxa encontrada, compare as imagens original, a não codificada e a codificada. Comente os resultados obtidos e suas conclusões.

Solução:

```

1 lena = imread('lenaTest2.jpg'); %adquire imagem Lena como matriz
2 tam = size(lena); %obter tamanho da matriz da imagem
3 lena_v = double(reshape(lena,tam(1)*tam(2),1)); %reordena a matriz como um vetor
4 lena_b = de2bi(lena_v,8)'; %converte de decimal para binário
5 tam_b = size(lena_b); %obtem tamanho da nova matriz com dados em binário
6 lena_b = reshape(lena_b,[tam_b(1)*tam_b(2) 1]); %ajusta os bits para ficarem todos em
    %uma coluna
7 tam_b = size(lena_b); %adquire as novas dimensões do vetor binário
8 pe = .25; %probabilidade de erro
9 C = 1 + pe*log(pe)/log(2) + (1-pe)*log(1-pe)/log(2); %calcula a capacidade do canal
    %BSC
10 n = ceil(1/C); %determina n para R_c = 1/n
11 tax = 1/n; %determina R_c máxima código ideal
12 lena_n = repmat(lena_b,[1 n+1]); %repetindo bits
13 tam_n = size(lena_n); %adquire novo tamanho dos dados
14 bsc_im = bsc(lena_n,pe); %obtem resultado após passar pelo canal BSC com erro pe =
    %25%

```

```

15 v_erro = sum(bsc_im,2); %soma cada bloco de bits repetidos
16 resp = lena_n(:,1); %pré alocando resposta
17 for p = 1:tam_n(1) %laço para corrigir os bits
18     if v_erro(p) > (n+1)/2 %se a maioria dos bits de cada bloco for 1, resposta igual
19         a 1
20         resp(p,1) = 1;
21     else %se a maioria dos bits de cada bloco for 0, resposta igual a 0
22         resp(p,1) = 0;
23     end
24 end
25 ber = sum(abs(lena_b-resp),1)/tam_b(1)*100; %calcula ber
26 figure %gera imagem
27 im = uint8(bi2de(reshape(resp,[8 (tam_n(1))/8]))); %colaca imagem no formato
28          original
29 im = reshape(im,tam); %colaca imagem no formato original
30 imshow(im)
31 xlabel(['n = ' num2str(n) ', ' ' BER = ' num2str(ber(end)) '$\%' ' e $p_e = 25\%''],
32       'Interpreter','Latex','FontSize',18) %legenda
33 saveas(gcf,['nidealpe25lena.pdf']); %salvando arquivo
34 ber_op = 999; %pré alocando ber para verificação manual
35 n = 2; %realocando valor de repetições
36 while ber_op > 10^(-5) %laço para determinar solução com R_c = 1/n com BER < 10^(-5)
37     lena_n = repmat(lena_b,[1 n+1]); %repetindo bits
38     tam_n = size(lena_n); %adquire novo tamanho dos dados
39     bsc_im = bsc(lena_n,pe); %obtem resultado após passar pelo canal BSC com erro pe
40             = 25%
41     v_erro = sum(bsc_im,2); %soma cada bloco de bits repetidos
42     resp = lena_n(:,1); %pré alocando resposta
43     for p = 1:tam_n(1) %laço para corrigir os bits
44         if v_erro(p) > (n+1)/2 %se a maioria dos bits de cada bloco for 1,
45             resposta igual a 1
46             resp(p,1) = 1;
47         else %se a maioria dos bits de cada bloco for 0, resposta igual a 0
48             resp(p,1) = 0;
49         end
50     end
51     ber_op = sum(abs(lena_b-resp),1)/tam_b(1) %calcula ber
52     n = n + 2 %incrementa número de repetições
53 end
54 figure %gera imagem
55 im = uint8(bi2de(reshape(resp,[8 (tam_n(1))/8]))); %colaca imagem no formato
56          original
57 im = reshape(im,tam); %colaca imagem no formato original
58 imshow(im)
59 xlabel(['n = ' num2str(n-2) ', ' ' BER = ' num2str(ber_op*100) '$\%' ' e $p_e =
60         25\%'], 'Interpreter','Latex','FontSize',18) %legenda
61 saveas(gcf,['n' num2str(n-2) 'pe25lena.pdf']); %salvando arquivo

```

Para determinar a taxa máxima do código é aquela que tem valor igual ou menor a capacidade do canal. Assim, assumindo que os bits de entrada são equiprováveis, a capacidade do canal é de:

$$C = 1 + p_e \ln(p_e) + (1 - p_e) \ln(1 - p_e) \approx 0,1887$$

Portanto a taxa máxima de um código que possa reduzir arbitrariamente teria uma taxa máxima $R_C = 0,1887$, o que se fosse aplicado a um código de repetição simples seria a taxa de $1/6$. Porém, essa taxa indica

que algum código com essa taxa é capaz de reduzir arbitrariamente o erro, não que essa seja a ideal para o código de repetição.

Para determinar a taxa em um código de repetição foi feito um laço até que a BER atingisse um valor mínimo ideal para transmissão de imagens. Assim, chegou-se que para um canal BSC com probabilidade de erro de 25%, o número mínimo ideal de repetições é de 62.

Se aplicarmos ambas as taxas ao código de repetição teremos sim melhorias em relação a imagem não codificada, porém fica claro nas figuras (7) e (8), porém com 6 repetições o resultado está longe de ter um ruído desprezível, o que não ocorre para 62 repetições onde o ruído é imperceptível. Portanto fica nítido mais uma vez que o código de repetição não é eficiente para esse tipo de problema, pois demanda muitos recursos para resultados não proporcionalmente eficientes.



$p_e = 0\%$



$p_e = 25\%$

Figura 7: Transmissão da Imagem pelo canal BSC com probabilidade de erro $p_e = 25\%$.



$p_e = 0\%$



$n = 6$, $BER = 7.0674\%$ e $p_e = 25\%$



$n = 62$, $BER = 0.00076294\%$ e $p_e = 25\%$

Figura 8: Transmissão da Imagem pelo canal BSC com probabilidade de erro $p_e = 25\%$, $n = 6$ e $n = 62$.

Referências

- [1] John G. Proakis and Masoud Salehi *Digital Communications*, 5th edition, New York (2008).
- [2] Simon Haykin *Communication Systems* , 4th edition, New York (2014).
- [3] Freitas Jr., W. C. *Notas de Aula de TI0056* (2018)
- [4] Bouda, J. *Lecture 9 - Channel Capacity*. (2010) Disponível em: <https://www.fi.muni.cz/~xbouda1/teaching/2010/IV111/lecture9.pdf>
- [5] Devarney, Bob *et al.* *A Tutorial on the Decibel*. (Desconhecido) Disponível em: <http://www.arrl.org/files/file/Instructor%20resources/A%20Tutorial%20on%20the%20Dec-N0AX.pdf>
- [6] Ricardo, M. *Transmissão de Dados* (2002) Disponível em: https://web.fe.up.pt/~mricardo/02_03/cdrc1/teoricas/transmissao_v5.pdf