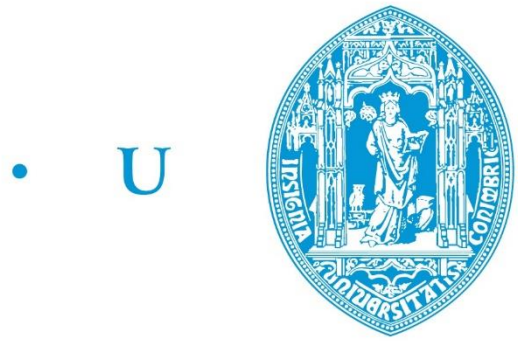


FCTUC FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Teoria de Informação

2017/2016

Trabalho pratico 1: Entropia, Redundância e Informação Mútua

Realizado por:

Ana Rita Ferreira Alfaro nº2013150362
Rui Miguel Freitas Rocha nº2010135453

Conteúdo

Objetivo

O objetivo do trabalho pratico era termos contacto com conceitos teóricos de entropia, redundância e informação mutua, aplicando-os na pratica em casos reais.

Resultados dos dados experimentais propostos no trabalho

Pergunta 3

Resultados da Entropia:

"Kid.bmp": 6.954143 bits por símbolo
"homer.bmp": 3.465865 bits por símbolo
"homerbin.bmp": 0.644781 bits por símbolo
"guitarSolo.wav": 7.35802 bits por símbolo
"english.txt": 4.227968 bits por símbolo

Será possível comprimir cada uma das fontes de forma não destrutiva? Se Sim, qual a compressão máxima que se consegue alcançar? Justifique.

Resposta: sim é possível comprimir cada uma das fontes de forma não destrutiva.
Através da formula $1 - \frac{H(x)}{logSx}$, onde H(x) e a entropia da fonte de estudo, e Sx o tamanho do conjunto de símbolos que x pode assumir, chegamos aos seguintes resultados em que podemos economizar as nossas fontes de informação:

"Kid.bmp": 13%
"homer.bmp": 57%
"homerbin.bmp": 92%
"guitarSolo.wav": 18%
"english.txt": 26%

Pergunta 4:

Resultados da media de huffman:
“Kid.bmp”: 6.983223 bits por símbolo
“homer.bmp”: 3.548316 bits por símbolo
“homerbin.bmp”: 1.000000 bits por símbolo
“guitarSolo.wav”: 7.379079 bits por símbolo
“english.txt”: 4.251830 bits por símbolo

Resultados da variância:

“Kid.bmp”: 2.098441
“homer.bmp”: 13.196838 bits por símbolo
“homerbin.bmp”: 0.000000 bits por símbolo
“guitarSolo.wav”: 0.756291 bits por símbolo
“english.txt”: 1.108861 bits por símbolo

Será possível reduzir-se a variância? Se sim, como pode ser feito em que circunstância será útil?

Resposta: sim, é possível reduzir a variância, colocando os símbolos na lista, usando a ordem mais elevada possível. Isto pode ser útil na construção do buffer, que pode permitir poupança de espaço.

Pergunta 5:

Resultados teóricos da entropia com agrupamento de 2 bits:

“Kid.bmp”: 4.9 bits/2 símbolos do alfabeto original
“homer.bmp”: 2.41 bits/2 símbolos do alfabeto original
“homerbin.bmp”: 0.39 bits/2 símbolos do alfabeto original
“guitarSolo.wav”: 3.23 bits/2 símbolos do alfabeto original
“english.txt”: 4.46 bits/2 símbolos do alfabeto original

Com o agrupamento dos símbolos consegue-se diminuir a entropia e, como consequência, também existe aumento de informação.

Pergunta 6:

b) No primeiro cenário, o gráfico mostra-nos que o som da query é muito semelhante ao do target01 no primeiro step, mas ao longo da simulação, os valores diferem até reencontrarem o mesmo "som".
No segundo cenário, devido ao ruido, as semelhanças entre a query e o target são menores. No entanto consegue se notar algumas semelhanças entre a query e o target quando o step é igual ao primeiro step.

c) Na visualização dos gráficos pode-se observar que a informação útil dos ficheiros "Song01.wav", "Song02.wav", "Song03.wav" e "Song04.wav" é baixa, e ao longo do tempo vai baixando. Nos ficheiros, "Song05.wav", "Song06.wav" e "Song07.wav", a informação útil é alta e depois vai diminuindo. Com isto podemos concluir que os inícios destes ficheiros são muito semelhantes ao ficheiro "guitarSolo.wav". Num programa de identificação de musica, o inicio destes últimos ficheiros tem alta probabilidade de serem semelhantes ao "guitarSolo.wav".

Resultados dos valores da informação mútua:

"Song01.wav":
0.255197 ;0.257843; 0.000000

"Song02.wav":
0.377678; 0.000000

"Song03.wav":
0.304454; 0.302177; 0.000000

"Song04.wav":
0.409696; 0.406710 ;0.000000

"Song05.wav":
3.961751; 0.327806; 0.328180; 0.329000; 0.343165; 0.335888; 0.330289; 0.337192; 0.334828; 0.334499; 0.333738; 0.340693; 0.342853; 0.333688; 0.344169; 0.329154; 0.332203; 0.328019; 0.335540; 0.334998; 0.3420666; 0.332047; 0.340686; 0.352540; 0.334384; 0.333538; 0.345864; 0.329612; 0.321956; 0.000000

"Song06.wav":

7.338379; 0.326351; 0.330051; 0.328104; 0.343682; 0.334521; 0.330718; 0.339093; 0.330755; 0.335016; 0.336113; 0.337875; 0.339655; 0.337028; 0.344040; 0.331813; 0.329351; 0.325077; 0.334794; 0.335297; 0.341349; 0.333826; 0.342509; 0.352747; 0.332806; 0.333731; 0.343368; 0.331489; 0.323445; 0.000000;

“Song07.wav”:

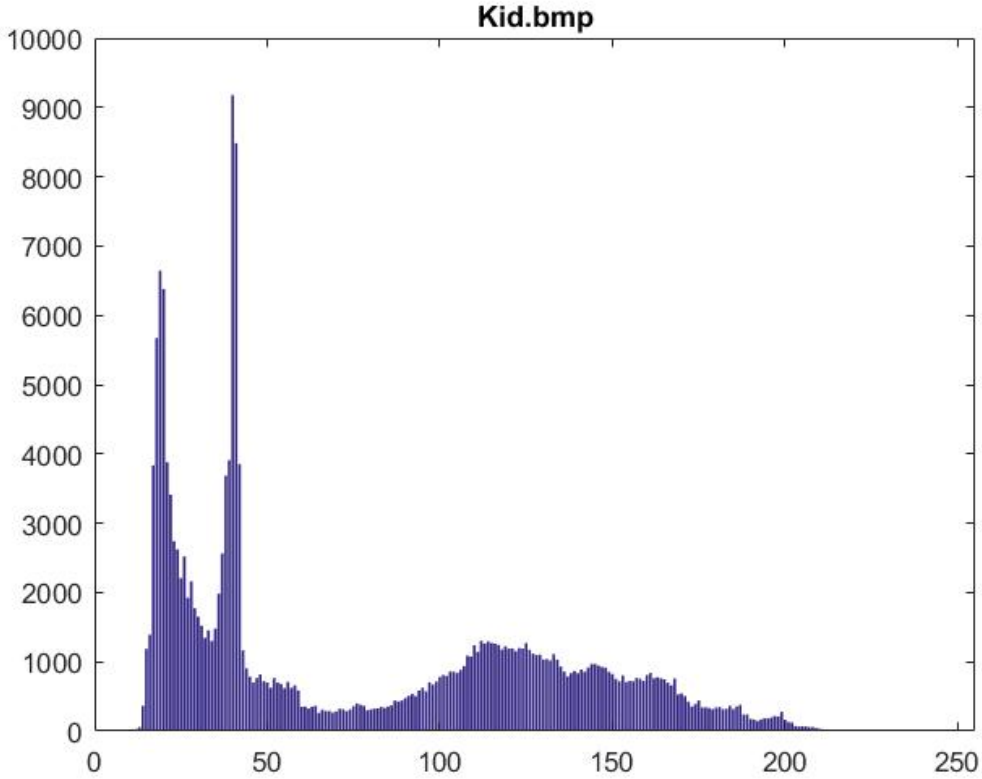
6.313053; 0.177815; 0.181806; 0.179524; 0.192036; 0.185465; 0.179658; 0.193703; 0.184813; 0.185570; 0.184349; 0.187897; 0.187683; 0.185237; 0.197092; 0.185362; 0.183628; 0.183176; 0.188352; 0.182436; 0.189472; 0.183889; 0.193629; 0.209979; 0.187701; 0.184365; 0.194029; 0.183228; 0.183251; 0.000000;

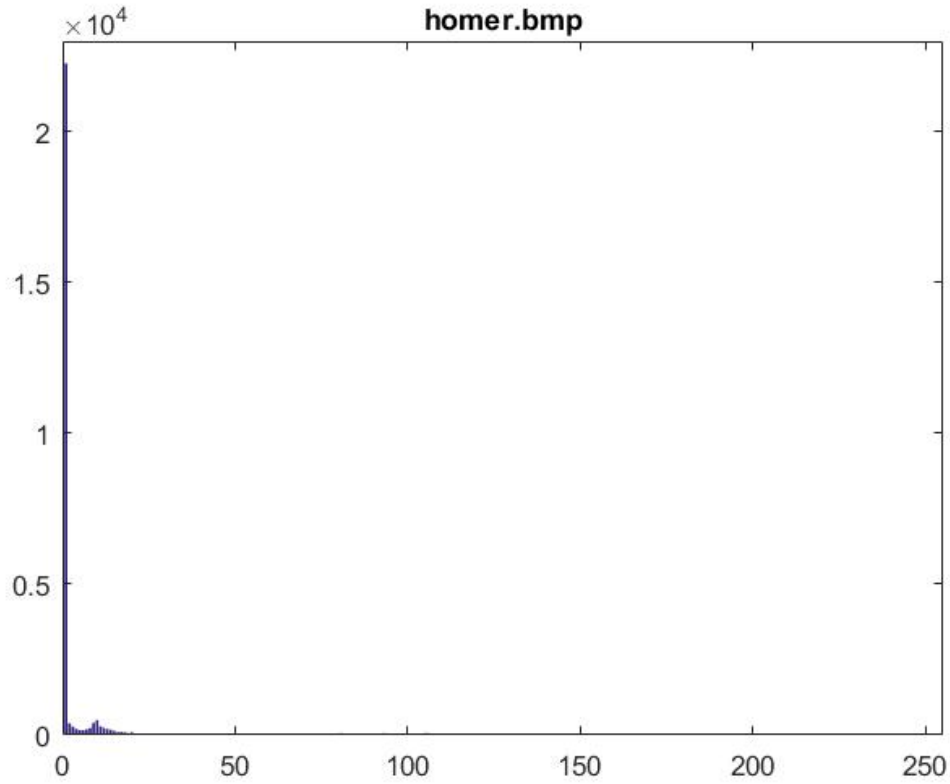
Resultados dos máximos da informação mútua máxima:

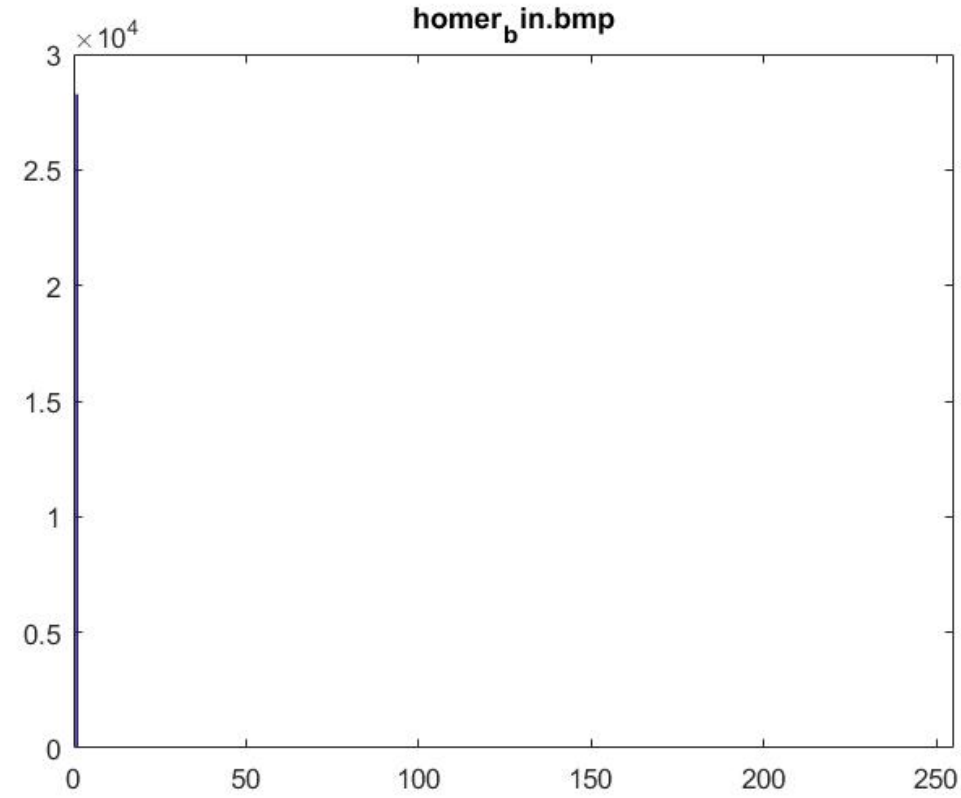
“Song06.wav”: 7.338379e+00
“Song07.wav”: 6.313053e+00
“Song05.wav”: 3.961751e+00
“Song04.wav”: 4.096962e-01
“Song02.wav”: 3.776776e-01
“Song03.wav”: 3.044541e-01
“Song01.wav”: 2.578426e-01

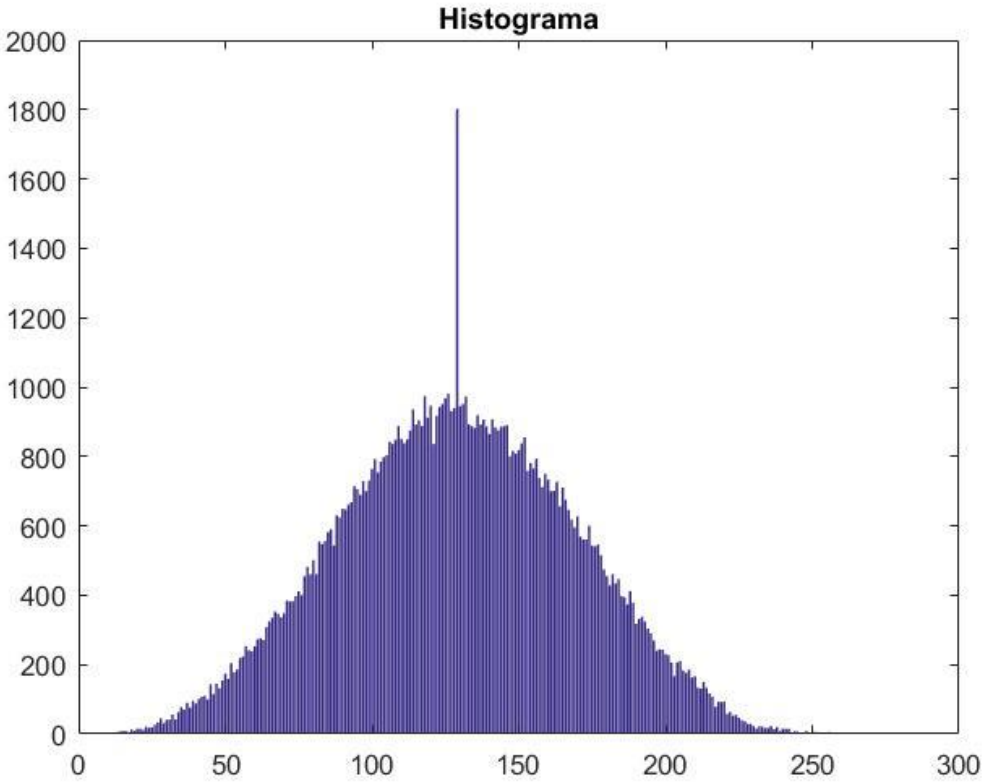
Anexos

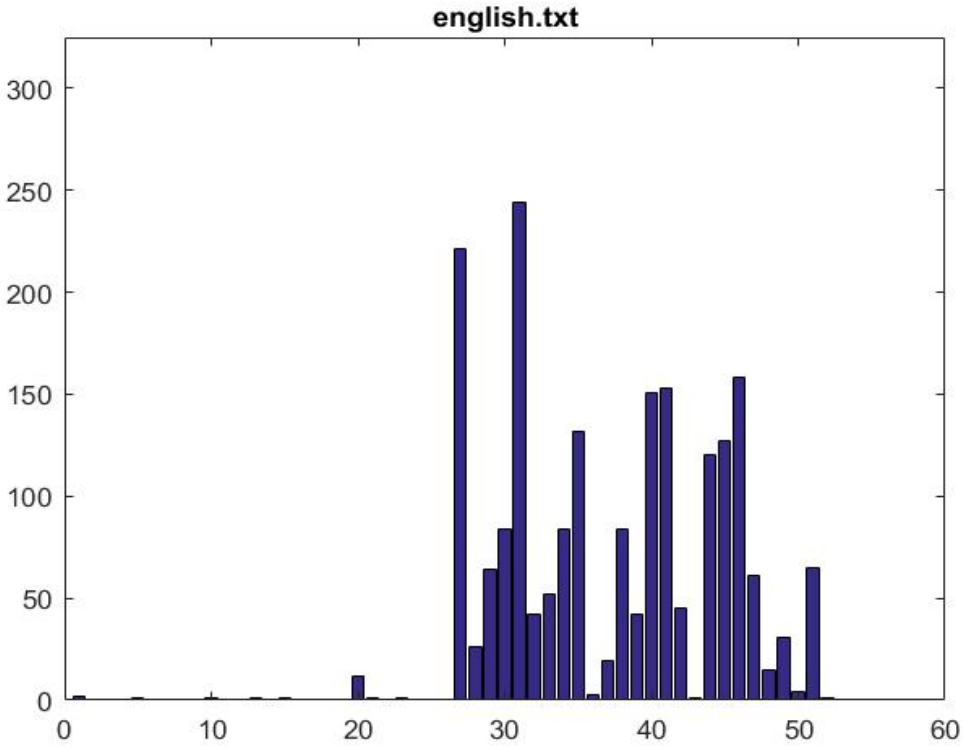
Pergunta3: Histogramas





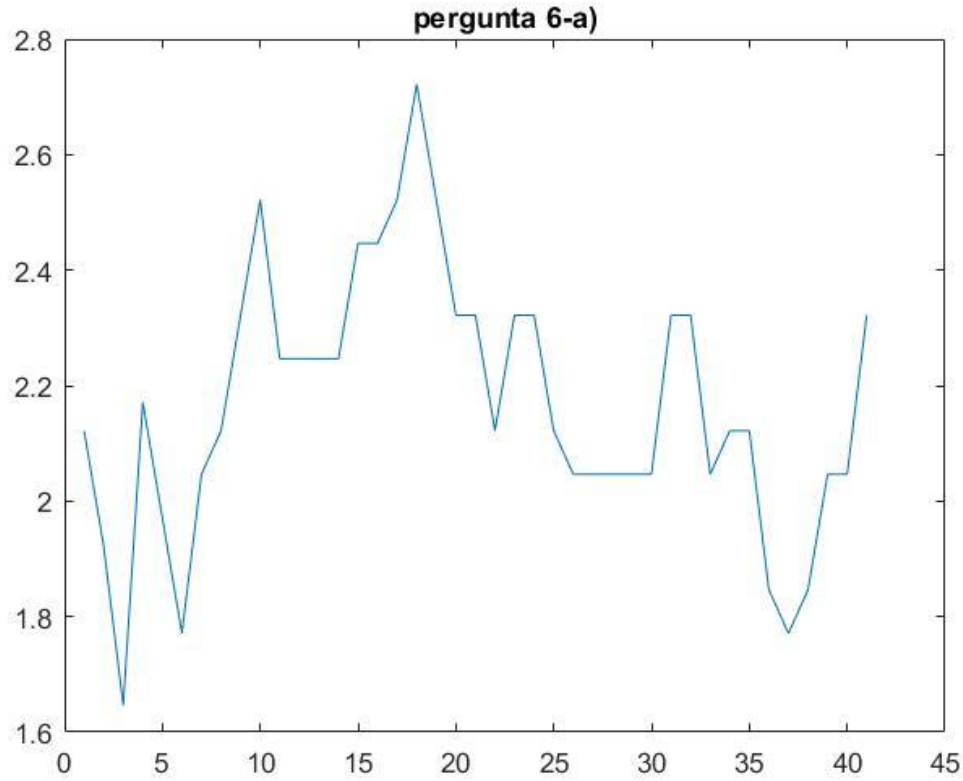

homer_bin.bmp
×10^4
3
2.5
2
1.5
1
0.5
0
0
50
100
150
200
250


Histograma
2000
1800
1600
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250
300


english.txt
300
250
200
150
100
50
0
0
10
20
30
40
50
60

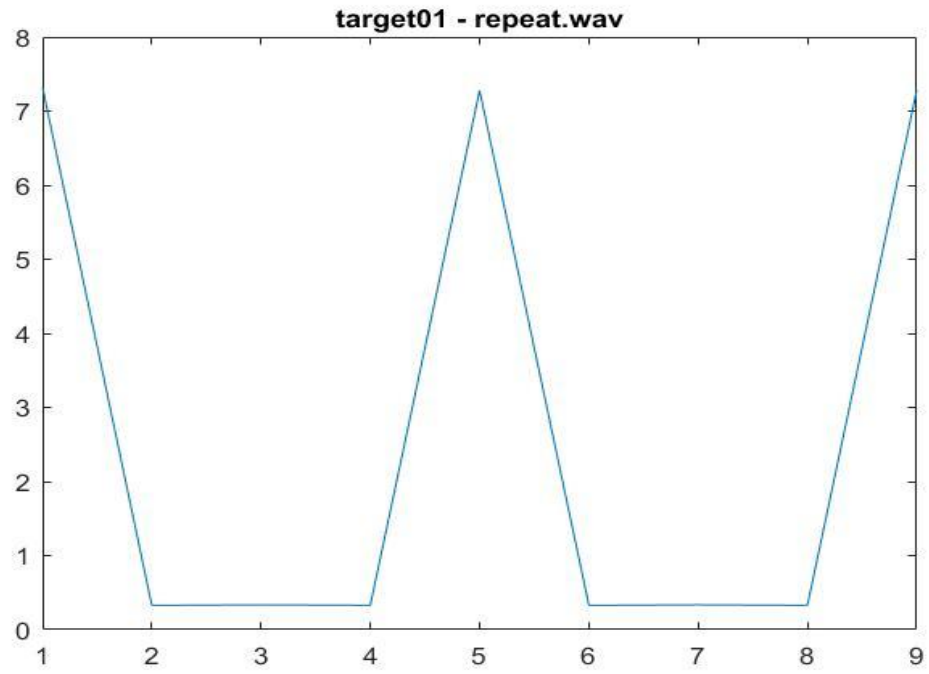
Pergunta 6:

a)

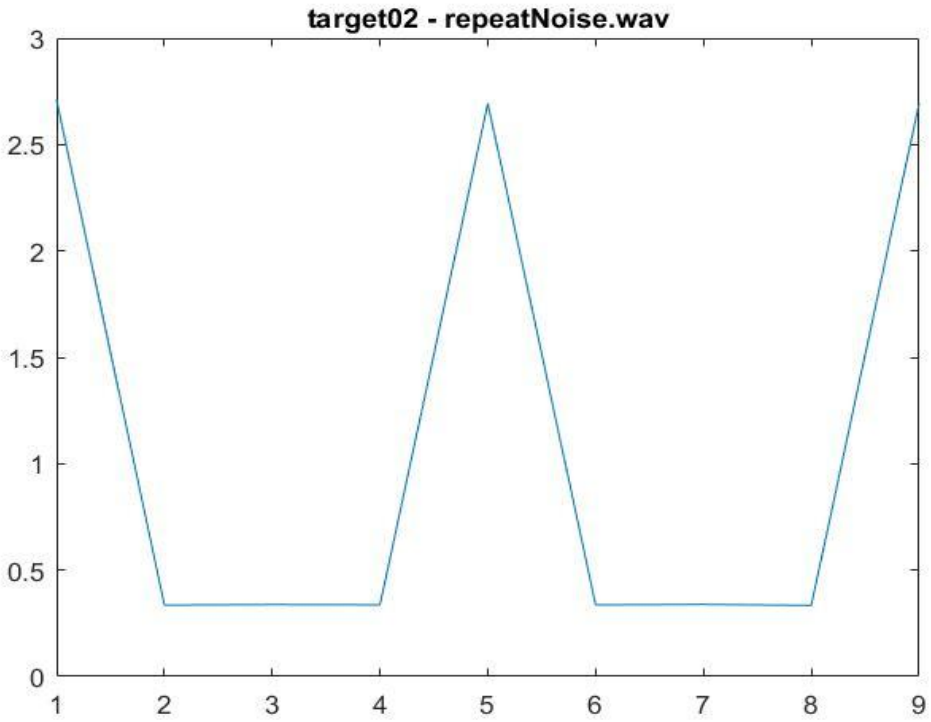

pergunta 6-a)
2.8
2.6
2.4
2.2
2
1.8
1.6
0
5
10
15
20
25
30
35
40
45

b) query: “guitarSolo.wav”

target: “repeat.wav”

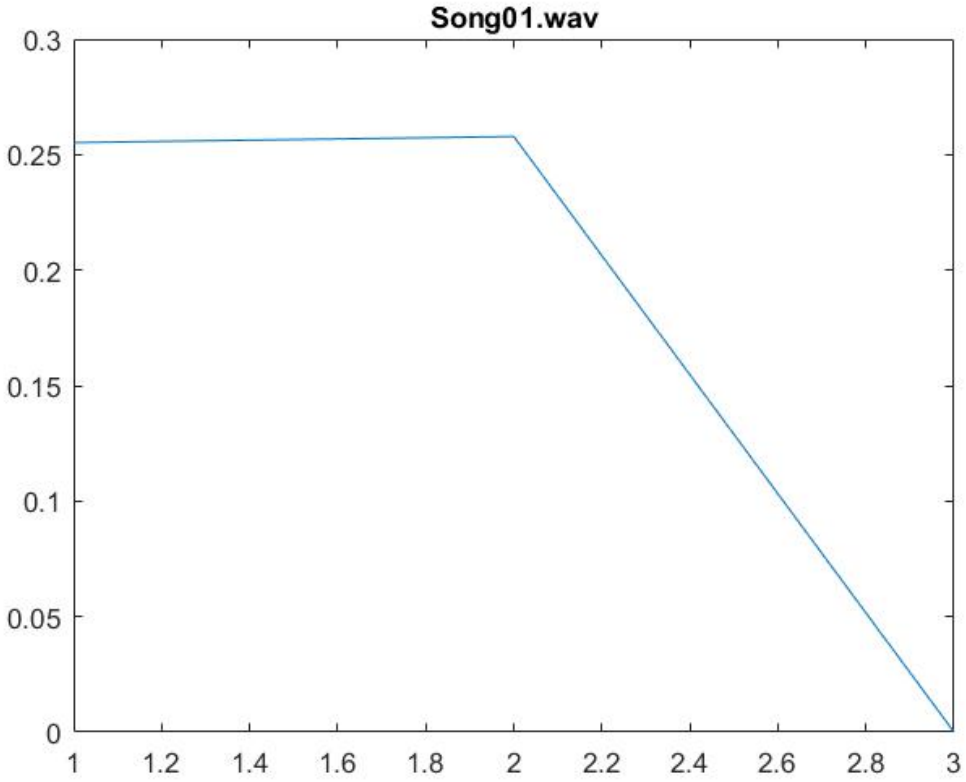


target: “repeat.wav”

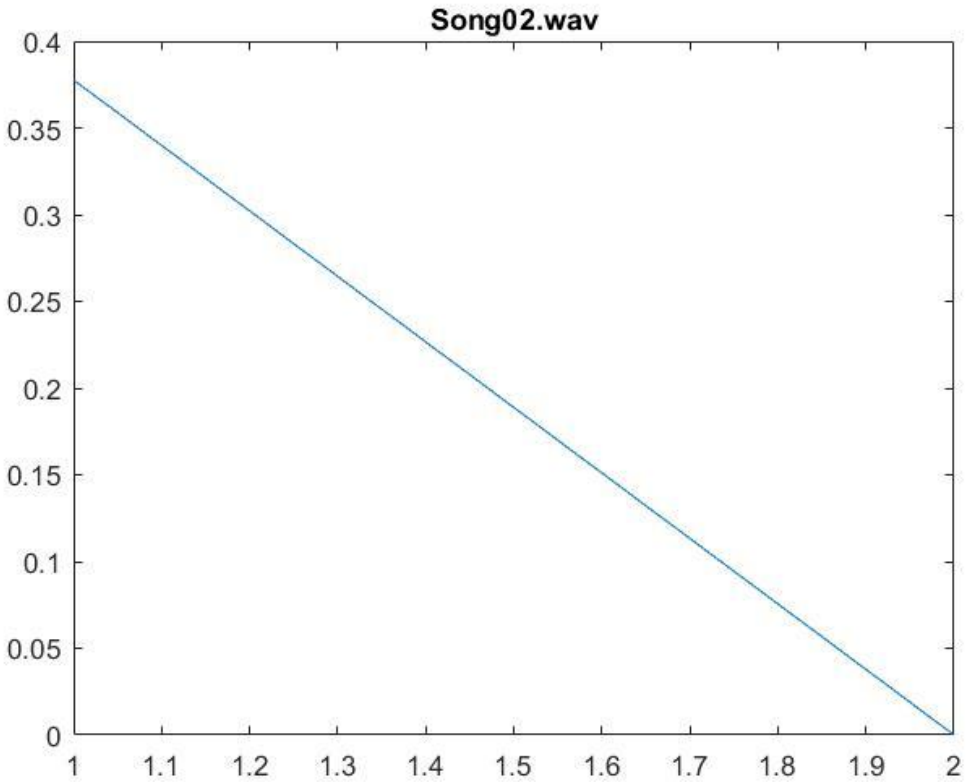


c)

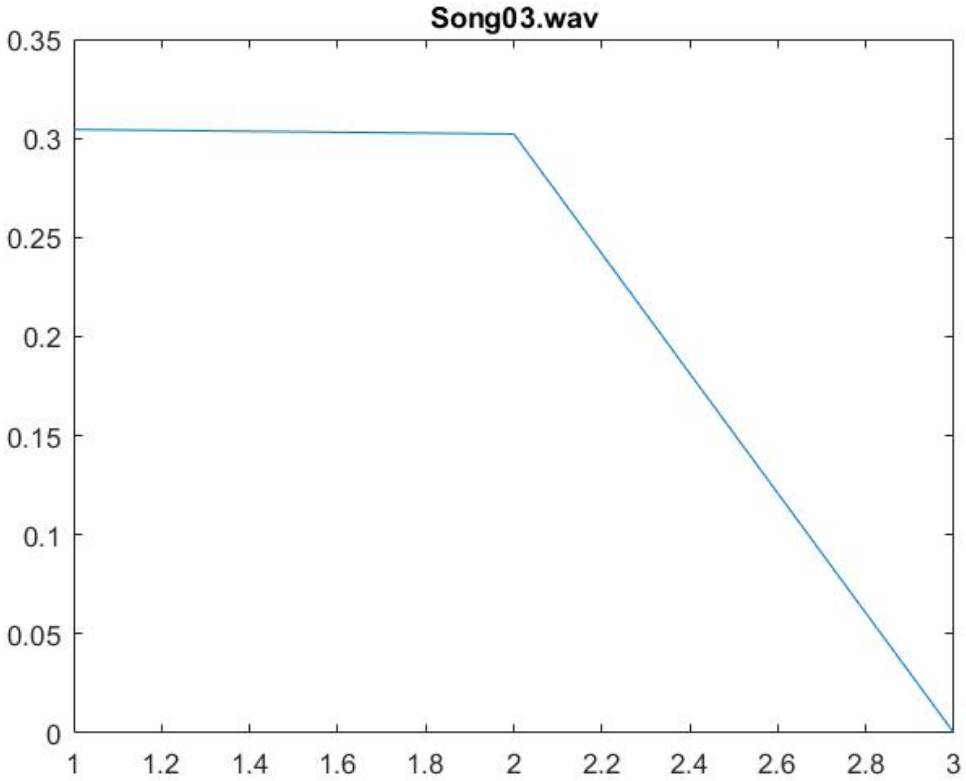
query: “guitarSolo.wav”
target: “Song01.wav”



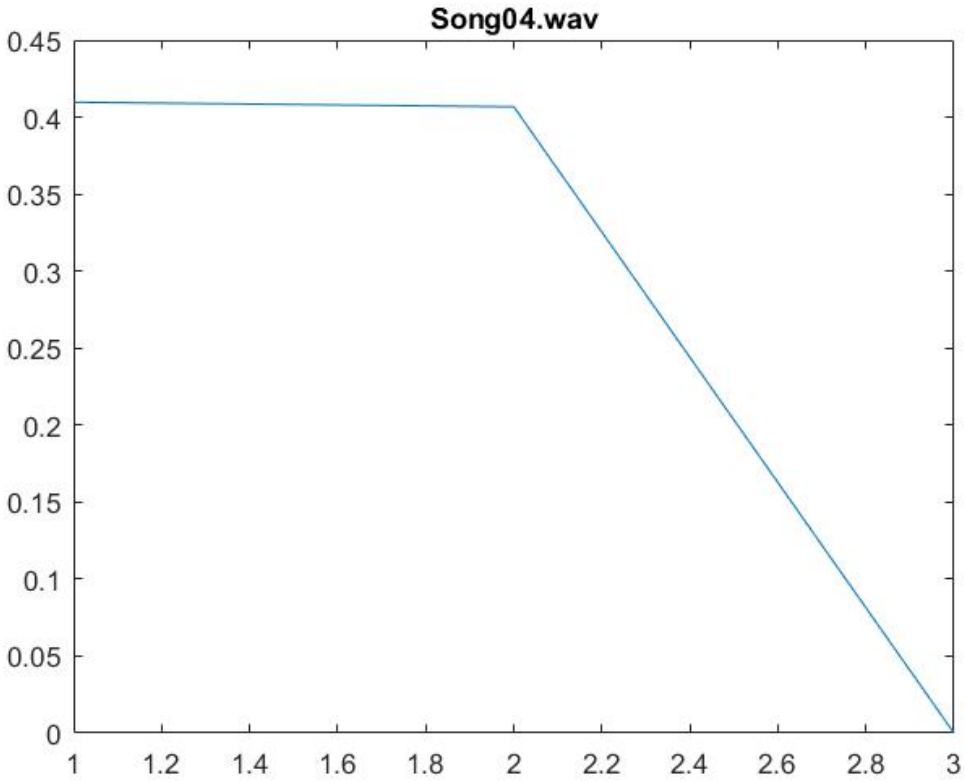
target: “Song02.wav”



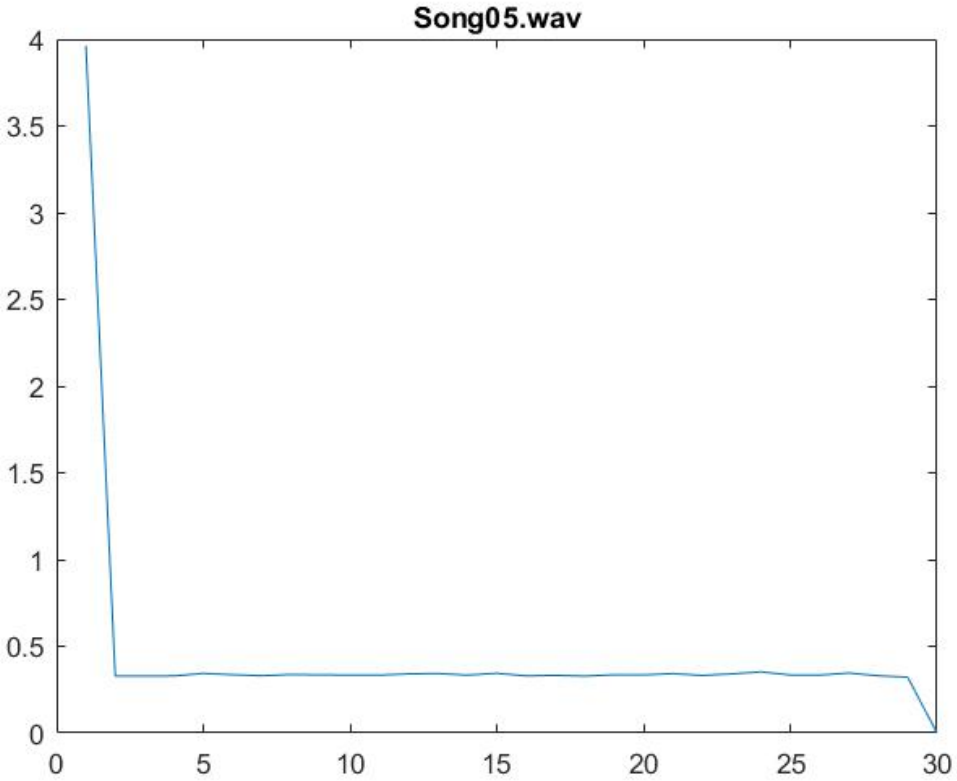
target: “Song03.wav”



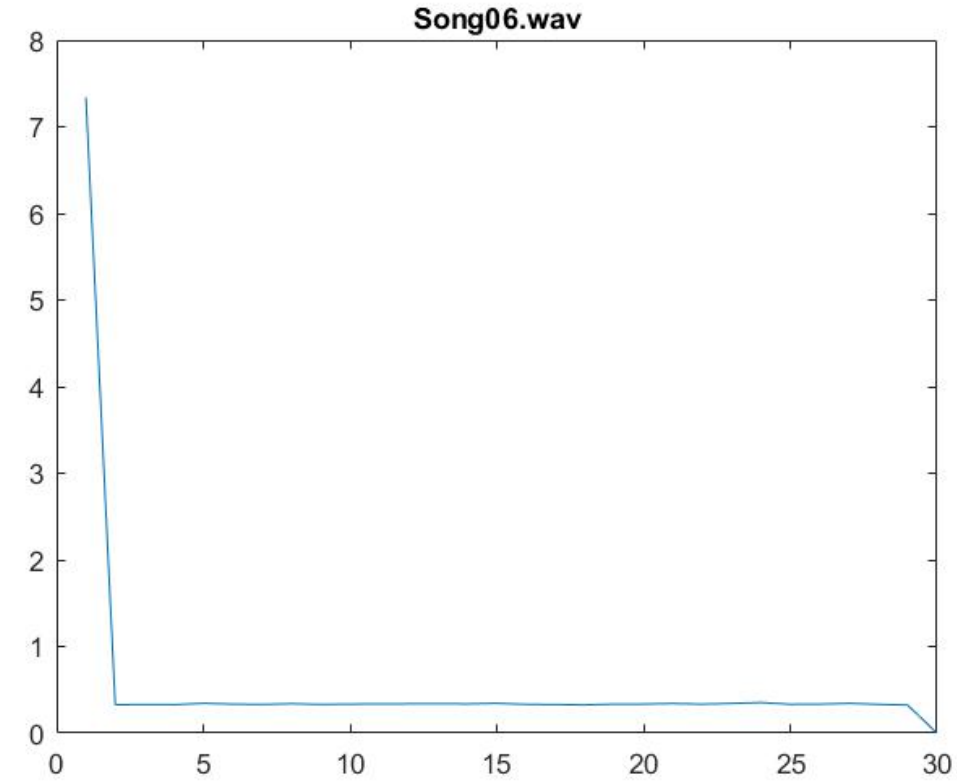
target: “Song04.wav”



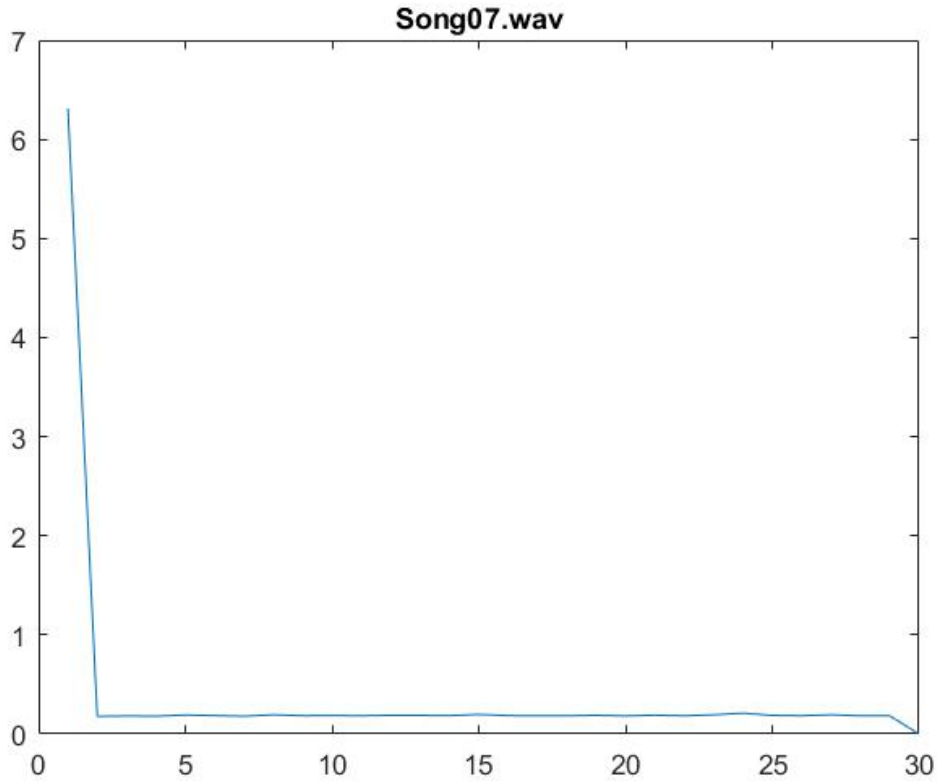
target: “Song05.wav”



target: “Song06.wav”



target: “Song07.wav”



Código fonte do trabalho

kid.m pergunta 1, 2 e 3

```
clc, clear, close all
caminho_kid = '.\dados\Kid.bmp';
imagem_kid = imread(caminho_kid);
info = imfinfo(caminho_kid);
nrBits = info.BitDepth;
alfabeto=zeros(2^nrBits,1);
for i=0:2^nrBits
   alfabeto(i+1)=i;
end
alfabeto = 0:255;

figure(1);

histograma_kid=histograma(imagem_kid,alfabeto);
title('Kid.bmp');
axis([0, 255, 0, 10000]);

disp(sprintf('pergunta 3'));
entropia_kid = entropia(histograma_kid);
format long;
disp(sprintf('Entropia: %f bits por simbolo\n',entropia_kid));


disp(sprintf('pergunta 4'));
hufflen_kid = hufflen(histograma_kid);
valor_medio_kid = valor_medio_hufflen(hufflen_kid,histograma_kid);
disp(sprintf('valor medio de hufflen: %f bits por
simbolo',valor_medio_kid));
%variancia
varKid = var(hufflen_kid,histograma_kid);
disp(sprintf('variancia da media de hufflen: %f bits por simbolo\n',varKid));

disp(sprintf('pergunta 5'));
agrupa_2_simbolos=agrupamento_de_2_simbolos(histograma_kid);
final = entropia(agrupa_2_simbolos);
disp(sprintf('agrupamento de 2 simbolos: %f bits/2 simbolos do alfabeto
original \n',final));
```

homer.m pergunta 1, 2 e 3

```
clc, clear, close all
caminho_homer = '.\dados\homer.bmp';
imagem_homer = imread(caminho_homer);
info = imfinfo(caminho_homer);
nrBits = info.BitDepth;
alfabeto=zeros(2^nrBits,1);
for i=0:2^nrBits
   alfabeto(i+1)=i;
end
alfabeto = 0:255;

figure(2);
histograma_homer = histograma(imagem_homer,alfabeto);
title('homer.bmp');
axis([0, 255, 0, 23000]);

entropia_homer = entropia(histograma_homer);
disp(sprintf('pergunta 3'));
disp(sprintf('Entropia: %f bits por simbolo\n', entropia_homer));

%pergunta 4
disp(sprintf('pergunta 4'));
hufflen_homer = hufflen(histograma_homer);
valor_medio_homer = valor_medio_hufflen(hufflen_homer,histograma_homer);
disp(sprintf('Valor medio de hufflen: %f bits por simbolo', valor_medio_homer));
%variancia
varHomer = var(hufflen_homer,histograma_homer);
disp(sprintf('variancia de hufflen: %f bits por simbolo\n',varHomer));

%pergunta 5
disp(sprintf('pergunta 5'));
agrupa_2_simbolos=agrupamento_de_2_simbolos(histograma_homer,size(alfa
beto));
final = entropia(agrupa_2_simbolos);
disp(sprintf('agrupamento de 2 simbolos: %f bits/2 simbolos do alfabeto original
\n',final));
```

homer_bin.m pergunta 1, 2 e 3

```
clc, clear, close all

caminho_homerBin = '.\dados\homerBin.bmp';

imagem_kid = imread(caminho_homerBin);
info = imfinfo(caminho_homerBin);
nrBits = info.BitDepth;
alfabeto=zeros(2^nrBits,1);
for i=0:2^nrBits
   alfabeto(i+1)=i;
end
alfabeto = 0:255;

disp(sprintf('pergunta 3'));
figure(3);
imagem_homer_bin = imread(caminho_homerBin);
histograma_homer_bin = histograma(imagem_homer_bin,alfabeto);
title('homer_bin.bmp');
axis([0, 255, 0, 30000]);

entropia_homer_bin = entropia(histograma_homer_bin);



disp(sprintf('Entropia: %f bits por simbolo\n', entropia_homer_bin));

%pergunta 4

disp(sprintf('pergunta 4'));
hufflen_homer_bin = hufflen(histograma_homer_bin);
valor_medio_homer_bin =
valor_medio_hufflen(hufflen_homer_bin,histograma_homer_bin);
disp(sprintf('Valor medio de hufflen: %f bits por simbolo',
valor_medio_homer_bin));
%variancia

varBin = var(hufflen_homer_bin,histograma_homer_bin);
disp(sprintf('variancia da media de hufflen: %f bits por simbolo\n',varBin));

% %pergunta 5
disp(sprintf('pergunta 5'));
agrupa_2_simbolos =
agrupamento_de_2_simbolos(histograma_homer_bin,length(alfabeto));
final = entropia(agrupa_2_simbolos);
disp(sprintf('agrupamento de 2 simbolos: %f bits/2 simbolos do alfabeto original
\n',final));
```

guitarSolo.m pergunta 1, 2 e 3

```
clc, clear, close all

caminho_guitarSolo = '.\dados\guitarSolo.wav';
figure(4);
[som_guitarSolo,fs] = audioread(caminho_guitarSolo);
info = audioinfo(caminho_guitarSolo);
nrBits = info.BitsPerSample;

d=2/(2^nrBits);
alfabeto_audio = -1:d:1-d;

disp(sprintf('pergunta 3'));
histograma_guitarSolo = histograma(som_guitarSolo,alfabeto_audio);
entropia_guitarSolo = entropia(histograma_guitarSolo);
disp(sprintf('Entropia: %4.5f bits por simbolo\n', entropia_guitarSolo));

disp(sprintf('pergunta 4'));
hufflen_guitarSolo = hufflen(histograma_guitarSolo);
valor_medio_hufflen_guitarSolo =
valor_medio_hufflen(hufflen_guitarSolo,histograma_guitarSolo);
disp(sprintf('Valor medio de hufflen: %f bits por simbolo',
valor_medio_hufflen_guitarSolo));
%variancia
varGuitar = var(hufflen_guitarSolo,histograma_guitarSolo);
disp(sprintf('variancia da media de hufflen: %f bits por simbolo\n',varGuitar));


disp(sprintf('pergunta 5'));
agrupa_2_simbolos = agrupamento_de_2_simbolos(histograma_guitarSolo);
final = entropia(agrupa_2_simbolos);
disp(sprintf('agrupamento de 2 simbolos: %f bits/2 simbolos do alfabeto original
\n',final));
```

english.m pergunta 1, 2 e 3

```
clc, clear, close all

disp(sprintf('pergunta 3'));
caminho_english_txt = '.\dados\english.txt';
figure(5);
ficheiro = fopen(caminho_english_txt,'r');
alfabeto_texto=double(['A':'Z' 'a':'z' ]);

leitura_ficheiro = fscanf(ficheiro,'%s');
texto_sem_pontos = retira_pontos(leitura_ficheiro);
histograma_texto = histograma(texto_sem_pontos,alfabeto_texto);
title('english.txt');
axis([0, 60, 0, 325]);


entropia_texto = entropia(histograma_texto);
disp(sprintf('Entropia: %f bits por simbolo\n', entropia_texto));

disp(sprintf('pergunta 4'));
hufflen_texto = hufflen(histograma_texto);
valor_medio_hufflen_texto =
valor_medio_hufflen(hufflen_texto,histograma_texto);
disp(sprintf('Valor medio de hufflen: %f bits por simbolo',
valor_medio_hufflen_texto));
%variancia
varTexto = var(hufflen_texto,histograma_texto);
disp(sprintf('variancia da media de hufflen: %f bits por simbolo\n',varTexto));

%pergunta 5
disp(sprintf('pergunta 5'));
agrupa_2_simbolos = agrupamento_de_2_simbolos(histograma_texto);
final = entropia(agrupa_2_simbolos);
disp(sprintf('agrupamento de 2 simbolos: %f bits/2 simbolos do alfabeto original
\n',final));
```

histograma.m

```
function[hist] = histograma(fonte,alfabeto)
%funçao dada uma fonte de informaçao e um alfabeto, devolve o histograma
da
%fonte de informaçao
%histcounts

   edges = [alfabeto alfabeto(end)+(alfabeto(2)-alfabeto(1))];
   [hist] = histcounts(fonte,edges);

%     for i=1:m
%           procura = ocorrencias(fonte,alfabeto(i));
%           hist(i)=procura;
%
%     end

   %plot(hist,'blue');
   bar(hist);

   title('Histograma');
end
```

entropia.m

```
function[res] = entropia(fonte)
%funçao para calcular a entropia de uma fonte de informaçao

   res =0;
   tamanho = sum(fonte);
   for i=1:length(fonte)
      if(fonte(i)~=0)
         prob = fonte(i)/tamanho;
         res = res + (prob*log2(prob));
      end
   end
   res=res*-1;
end
```

valor_medio_hufflen.m

```
function[media]  = valor_medio_hufflen(fonte_hufflen,fonte_histograma)

    tamanho_hufflen=length(fonte_hufflen);
    soma=0;
    nr=0;
    for i=1:tamanho_hufflen
        if(fonte_hufflen(i)~=0)
            soma=soma+fonte_hufflen(i)*fonte_histograma(i);
            nr=nr+fonte_histograma(i);
        end
    end;
    media=soma/nr;
end
```

pergunta6.m

```
clc, clear, close all
query = [2 6 4 10 5 9 5 8 0 8];
target = [6 8 9 7 2 4 9 9 4 9 1 4 8 0 1 2 2 6 3 2 0 7 4 9 5 4 8 5 2 7 8 0 7 4 8 5 7 4
3 2 2 7 3 5 2 7 4 9 9 6];
alfabeto = 0 : 10;
step = 1;

figure(1);
informacaoMutua=  vetorInforMutua(query,target,alfabeto,step);

disp('   Informaçao Mutua: ');
disp(informacaoMutua);
plot(informacaoMutua);
title('pergunta 6-a)');
```

pergunta6_b.m

```
clc, clear, close all

caminho_guitarSolo  = '.\dados\guitarSolo.wav';
caminho_repeat  = '.\dados\target01 - repeat.wav';
caminho_repeatNoise  = '.\dados\target02 - repeatNoise.wav';

[query, fs1] = audioread (caminho_guitarSolo);
info = audioinfo(caminho_guitarSolo);
nrBitsQuant1  = info.BitsPerSample;
d=2/(2^nrBitsQuant1);
alfabeto= -1:d:1-d;
query=query(:,1);
step=floor(length(query)/4);

%----------ficheiro target01 - repeat
figure(1);
[target1, fs1] = audioread (caminho_repeat);
info_repeat = audioinfo(caminho_repeat);
nrBitsQuant2  = info_repeat.BitsPerSample;
target1=target1(:,1);
informacaoMutua=vetorInforMutua(query,target1,alfabeto,step);
plot(informacaoMutua);
title('target01 - repeat.wav');

%----------ficheiro target02 - repeatNoise
figure(2);
[target2, fs1] = audioread (caminho_repeatNoise);
info_Noise  = audioinfo(caminho_repeatNoise);
nrBitsQuant3  = info_Noise.BitsPerSample;
target2=target2(:,1);
infoMutuaNoise=vetorInforMutua(query,target2,alfabeto,step);
plot(infoMutuaNoise);
title('target02 - repeatNoise.wav');
```

pergunta6_c.m

```
clc, clear, close all

caminho_guitarSolo  = '.\dados\guitarSolo.wav';
caminho_SONG1  = '.\dados\Song01.wav';
caminho_SONG2  = '.\dados\Song02.wav';
caminho_SONG3  = '.\dados\Song03.wav';
caminho_SONG4  = '.\dados\Song04.wav';
caminho_SONG5  = '.\dados\Song05.wav';
caminho_SONG6  = '.\dados\Song06.wav';
caminho_SONG7  = '.\dados\Song07.wav';

maximos=zeros(7,1);
id=1:7;

%----------ficheiro guitarSolo
[query, fs1] = audioread (caminho_guitarSolo);
info = audioinfo(caminho_guitarSolo);
nrBitsQuant1  = info.BitsPerSample;
d=2/(2^nrBitsQuant1);
alfabeto= -1:d:1-d;
query=query(:,1);
step=floor(length(query)/4);

%--------------------ficheiro Song1
figure(1);
[song1, fs1] = audioread (caminho_SONG1);
song1=song1(:,1);
SONG1=vetorInforMutua(query,song1,alfabeto,step);
plot(SONG1);
title('Song01.wav');
maximos(1)=max(SONG1);

%--------------------ficheiro Song2
figure(2);
[song2, fs1] = audioread (caminho_SONG2);
song2=song2(:,1);
SONG2=vetorInforMutua(query,song2,alfabeto,step);
plot(SONG2);
title('Song02.wav');
maximos(2)=max(SONG2);

%--------------------ficheiro Song3
figure(3);
[song3, fs1] = audioread (caminho_SONG3);
song3=song3(:,1);
SONG3=vetorInforMutua(query,song3,alfabeto,step);
plot(SONG3);
title('Song03.wav');
```

```
maximos(3)=max(SONG3);

%--------------------ficheiro Song4
figure(4);
[song4, fs1] = audioread (caminho_SONG4);
song4=song4(:,1);
SONG4=vetorInforMutua(query,song4,alfabeto,step);
plot(SONG4);
title('Song04.wav');
maximos(4)=max(SONG4);

%--------------------ficheiro Song5
figure(5);
[song5, fs1] = audioread (caminho_SONG5);
song5=song5(:,1);
SONG5=vetorInforMutua(query,song5,alfabeto,step);
plot(SONG5);
title('Song05.wav');
maximos(5)=max(SONG5);

%--------------------ficheiro Song6
figure(6);
[song6, fs1] = audioread (caminho_SONG6);
song6=song6(:,1);
SONG6=vetorInforMutua(query,song6,alfabeto,step);
plot(SONG6);
title('Song06.wav');
maximos(6)=max(SONG6);

%--------------------ficheiro Song7
figure(7);
[song7, fs1] = audioread (caminho_SONG7);
song7=song7(:,1);
SONG7=vetorInforMutua(query,song7,alfabeto,step);
plot(SONG7);
title('Song07.wav');
maximos(7)=max(SONG7);

%-------------resultados de maximos
disp('Maximos:');
[maximos,id]=ordenar_maximos(maximos,id);
for i=1:length(maximos)
   disp(sprintf('Song0%d - %d',id(i),maximos(i)));
end
```

vetorInforMutua.m

```
function[IM]  = vetorInforMutua(query,target,alfabeto,step)
   tam=1;
   pos=1;
   for i=1:step:length(target)
      if(i+length(query)<=length(target))
         tam=tam+1;
      else
         break;
      end
   end
   IM=zeros(tam,1);

   for i=1:step:length(target)
      if(i+length(query)-1>length(target))
         break;
      end
      calculo = calculoIM(target(i:i+length(query)-1),query,alfabeto);

      IM(pos)=calculo;
      pos=pos+1;
   end
end
```

probabilidadeX.m

```
function[probX] =  probabilidadeX(sinal,i,alfabeto,tamanho)
   probX=0;
   for y=i:i+tamanho-1
      if(y>length(sinal))
         break;
      end
      if(sinal(y)==alfabeto)
         probX=probX+1;
      end
   end
   probX=probX/tamanho;
end
```

calculoIM.m

```
function[res]  = calculoIM(target,query,alfabeto)
   prob_target=zeros(length(alfabeto),1);
   prob_query=zeros(length(alfabeto),1);

   for i=1:length(alfabeto)
      for y=1:length(query)
         if(target(y)==alfabeto(i))
            prob_target(i)=prob_target(i)+1;
         end
         if(query(y)==alfabeto(i))
            prob_query(i)=prob_query(i)+1;
         end
      end
      prob_query(i)=prob_query(i)/length(query);
      prob_target(i)=prob_target(i)/length(query);
   end

   for i=1:length(alfabeto)
   res=0;
      for y=1:length(alfabeto)
          if(prob_target(i)~=0 && prob_query(y)~=0)
             p=probabilidade_x_y(target,query,alfabeto(i),alfabeto(y));
             if(p~=0)
                res=res+p*log2(p/(prob_target(i)*prob_query(y)));
             end
          end
       end
   end
end
```

probabilidade_x_y.m

```
function[prob]  = probabilidade_x_y(target,query,alfabeto_target,alfabeto_query)
   prob=0;
   for i=1:length(query)
       if(target(i)== alfabeto_target)
          if(query(i)==alfabeto_query)
             prob=prob+1;
          end
       end
   end
   prob=prob/length(query);
end
```

retira_pontos.m

```
function [texto_final]= retira_pontos(texto_original)
%funçao para retirar pontos "." de um texto
%no final devolve o mesmo sem os pontos de final
   a=1;
   texto_final=zeros(0,1);
   for i=1:length(texto_original)
      if(texto_original(i)~='.')
         texto_final(a)=texto_original(i);
         a=a+1;
      end
   end
end
```

ordenar_maximos.m

```
function [maximos,id] = ordenar_maximos(maximos,id)
   for i=1:length(maximos)
      aux=maximos(i);
      for y=i:length(maximos)
         if(aux<maximos(y))
            aux2=id(i);
            id(i)=id(y);
            id(y)=aux2;

            aux=maximos(y);
            maximos(y)=maximos(i);
            maximos(i)=aux;
         end
      end
   end
end
```

agrupamento_de_2_simbolos.m

```
function [prob] = agrupamento_de_2_simbolos(fonte)
%funçao para agrupar simbolos e retorna uma sequência de dois símbolos contíguos
   pos =1;
   prob = 0;
   [nl, nc] = size(fonte);
   [elem] =  nl * nc;
   agrupa = zeros(length(elem),1);
   for i=1:1:size(nl)
      for j = 1:1:size(nc)

            id = (length(nc)*nc(j)) +nc(j);
            agrupa(pos) = id;
            pos=pos+1;

       end
   end

   for k=1:1:length(elem)
      if  elem(k)== agrupa(k)
        prob=prob+1;
      end
   end
   prob=prob/length(elem);
end
```