

Processamento Digital de Sinal - Optimização e redimensionamento de sinais de áudio usando o Matlab

Realizado por: Patrícia Margarida Silva de Castro Neves Barbosa

Nº: 62012

Docente: Professor Carlos Manuel Gregório Santos Lima

Curso: Mestrado em Engenharia Eletrónica Industrial e Computadores

Ano: 3º

Unidade Curricular: Processamento Digital de Sinal

Índice

Introdução1

Contextualização2

**Distribuição de Gauss ou Gaussiana 3**

Ruído Branco4

Matlab6

**Primeira solução encontrada6**

**Segunda solução encontrada8**

**Conclusões14**

**Bibliografia15**

Introdução

No âmbito da unidade curricular de Processamento Digital Sinal, foi proposto um trabalho final, este trabalho tem como principal objectivo a optimização/redimensionamento de sinais de áudio provenientes da fala humana. A pergunta óbvia está no motivo/força que leva a este redimensionamento, ora o principal motivo é a optimização de recursos, isto é, garantir que apenas informação realmente uteis (fala) são transmitidos, ou seja, impedir que dados inúteis/parasitas (ruídos de fundo ou silêncios) sejam perpetuados. Aliás esta meta de eliminação de ruídos é uma das grandes ambições por exemplo da área das comunicações. Para proceder a esta optimização terá de ser desenvolvido um algoritmo que permita em primeiro lugar detectar ruído, quer seja no início, meio ou no fim de um dado sinal áudio e em segundo lugar proceder à eliminação desse mesmo ruído, de forma que o sinal resultante seja apenas o sinal de áudio correspondente à fala humana, sem qualquer ruído/silêncio associado. O algoritmo a ser projectado deve inspirar-se substancialmente no protocolo de Shewart (“*Shewart Protocol*”) mas com threshold adaptado às condições do ruído.

Para a realização deste trabalho recorre-se à utilização da plataforma do programa Matlab, quer para a projecção das funções, quer para o teste das mesmas, sendo que os dados terão de ser recolhidos e estudados/comentados.

Contextualização

Distribuição de Gauss ou Gaussiana

A distribuição de Gauss de probabilidades é um modelo que assenta na incerteza/grau de variabilidade associada ao desempenho de uma dada variável monitorizada, ou seja, é o grau de incerteza associado a um dado fenómeno a ser estudado, sendo que acaba por depender sempre do comportamento de uma variável de monitorização ou variação. No caso da distribuição de Gauss, esta caracteriza-se por uma curva de distribuição de probabilidades simétrica, e acaba directamente por assentar na própria lei dos grandes números de Bernoulli (uma aplicação interessante neste campo é por exemplo o lançamento de um dado, teoricamente apenas temos um sexto de probabilidade para cada face, mas isto só acontece para infinitos testes e não para reduzidos testes, ou seja, para mais testes temos menos erro), e na ideia desenvolvida a partir deste modelo por Abraham de Moivre, que é a ideia de que os erros de alto valor são muito pouco frequentes, isto é, é percentualmente mais comum a existência de erros de baixo ou muito baixo grau que alto grau, por isso a distribuição de Gauss assume um aspecto semelhante ao mostrado abaixo.



Figura : modelo/distribuição de Gauss

Ou seja, com

µ=media

 σ=desvio padrão

X = valor da variável de monitorização:

P ( |X-µ| < σ) = 68,26 = 68.3 Probabilidade associada ao ruído

P ( |X-µ| > σ) = 100-68.26 = 31.74 = 31.7 Probabilidade associada à fala

Outros valores importantes

P ( |X-µ| < 2xσ) = 95.44 = 95.4 (mais ou menos) P ( |X-µ| >2xσ) = 100 - P ( |X-µ| < 2xσ)

P ( |X-µ| < 3xσ) = 99.74 = 99.7 (mais ou menos) P ( |X-µ| >3xσ) = 100 - P ( |X-µ| < 3xσ)

Ruído Branco



O ruído branco é um tipo de ruído produzido pela combinação simultânea de sons de todas as frequências. O adjetivo branco é utilizado para descrever este tipo de ruído em analogia ao funcionamento da luz branca, dado que esta é obtida por meio da combinação simultânea de todas as frequências cromáticas. Por conter sons de todas as frequências, o ruído branco é frequentemente empregado para mascarar outros sons.

Por exemplo: numa conversação entre duas pessoas, seu cérebro consegue captar claramente a voz do interlocutor e compreendê-la. Mesmo com três ou quatro interlocutores isso ainda é possível. Entretanto, se 1000 pessoas falam simultaneamente, não há como seu cérebro captar uma voz isoladamente. O efeito de 1000 pessoas falando simultaneamente, assim como o de um ventilador que é ligado em seu quarto para mascarar a conversa entre duas pessoas no quarto ao lado, ajuda a ilustrar algumas das aplicações do ruído branco).

Matlab

Primeira solução encontrada:

function [ y ] = filtra(x,nseg,aux,multiplier)

med=mean(x(1:nseg));

v=var(x(1:nseg));

dp=sqrt(v);

j=0;

ciclo=1;

n\_amostras=0;

while(ciclo)

j=j+1;

if( abs(x(nseg+j)-med) > multiplier\*dp )

n\_amostras=n\_amostras+1;

if(n\_amostras==aux)

y=x( (nseg + j - amostras\_fala) : length(x) );

ciclo=0;

end

else

n\_amostras=0;

end

end

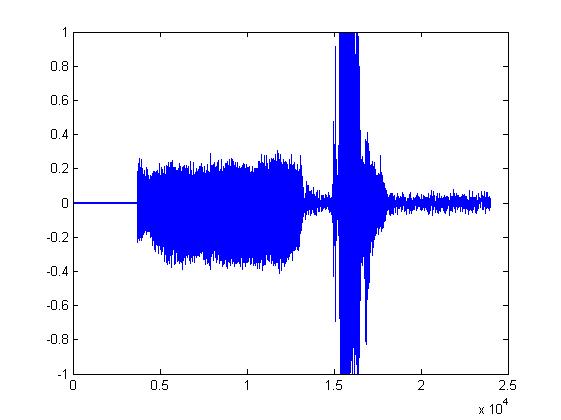
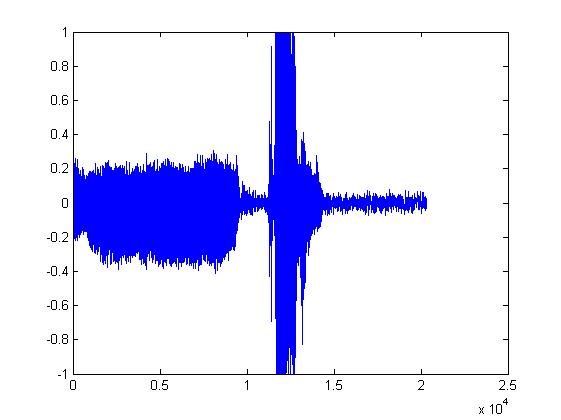
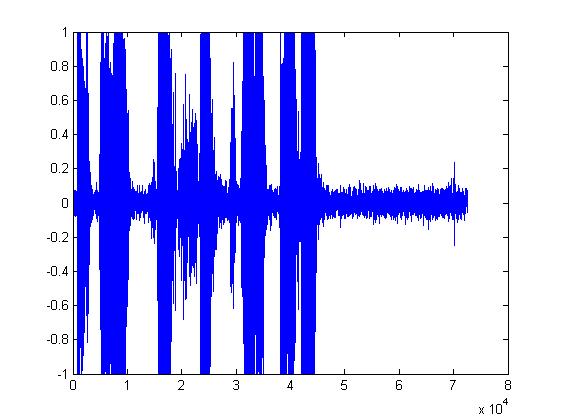
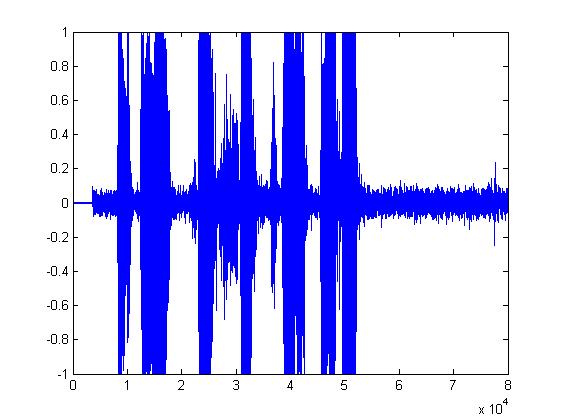
end

A função apresentada serve principalmente para eliminar o ruído inicial presente no sinal. Contudo, só consegue eliminar o ruído quando detecta que existiu alguém a falar, o que não vai de encontra com o pretendido.

Legenda da função:

* aux: assegurar uma maior consistência na identificação da fala sem ter qualquer impacto negativo no sinal resultante;
* multiplier: permite a expansão ou compressão do intervalo de valores entre a média e a fronteira/limite do modelo actual de ruído;
* ciclo: contador;
* nseg: serve para estabelecer o interval escolhido;

**Testes**:

* **Primeiro** teste: utilizando a palavra “**Sala**”
  + **Sinal original:**
  + **Sinal Original com filtro:**
* **Segundo** teste: utilizando a frase “O balão foi a Braga muito rápido”
  + **Sinal Original:**
  + **Sinal Original com filtro**

Segunda solução encontrada:

function [buff,noisebuff]

=filtra\_melhorada(x,nseg,wlen,step,frac,conta\_amostras\_fala,dp\_multiplier)

m = mean(x(1:nseg));

v = var(x(1:nseg));

dp = sqrt(v);

alfa = frac \* wlen;

conta=0;

for i=1 : length(x)

a(i)=0;

if ( abs(x(i) - m) > dp\_multiplier\*dp )

conta=conta+1;

if(conta==conta\_amostras\_fala)

while(conta~=0)

a(i-conta)=1;

conta=conta-1;

end

end

else

conta=0;

end

end

buff=0;

noisebuff=0;

n=0;

while( ( n \* step + wlen ) < length(x) )

w = a( ( n \* step + 1) : ( n \* step + wlen ) );

if(sum(w) >= alfa)

if(~buff)

buff = x( ( n \* step + 1) : ( n \* step + wlen) );

else

buff = cat(1, buff , x( ( n \* step +1) : ( n \* step +wlen) ) );

end

else

if(~noisebuff)

noisebuff=x( ( n \* step + 1) : ( n \* step + wlen) );

else

noisebuff = cat(1, noisebuff , x( ( n \* step +1) : (

n \* step + wlen) )

end

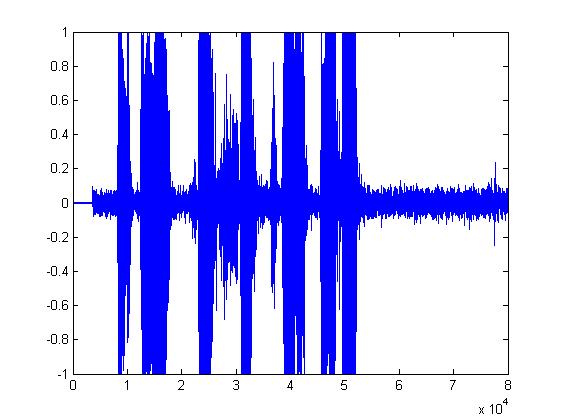
end

n=n+1;

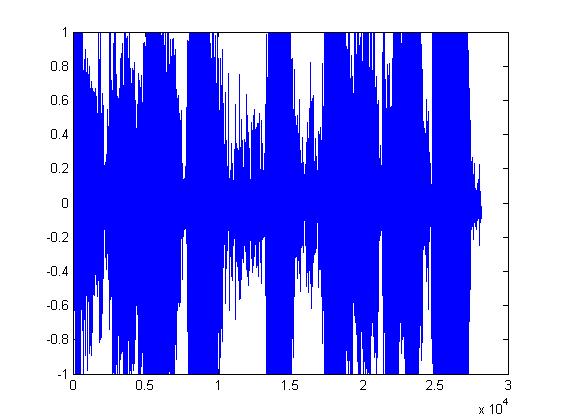
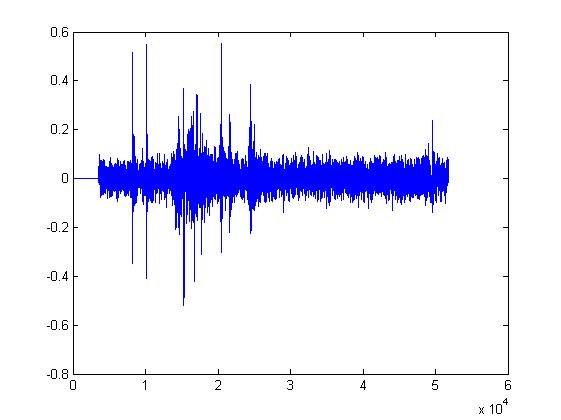
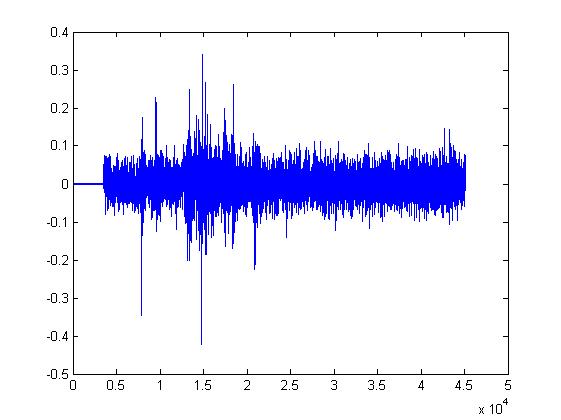
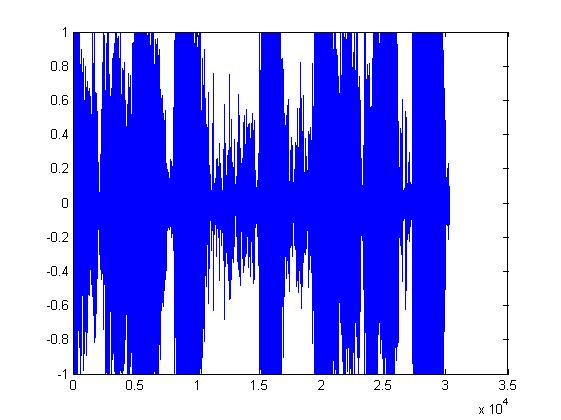
end

end

**Testes:**

* **Primeiro** teste: utilizando a frase “O balão foi a Braga muito rápido”
  + **Sinal Original:**

Ao contrário da primeira solução implementada, esta solução cumpre de forma satisfatória o pretendido. Contudo, é preciso ter em conta a parametrização que se aplica, pois podemos vir a ter resultados bastantes d´spares e até mesmo inesperados. Como não houve necesssidade de melhorar a análise do sinal foi colocado a variável contador\_amostras a um. Além disso, a variável multiplier foi usada de modo a permitir optimizer a análise de certos sinais, tendo por desvantagem a possibilidade de perda de components de fala. E por último, a variável wlen que vai tomar o valor passado ao step.

* + - multiplier = 4;
    - frac = 0.6;
    - step = 25;
    - wlen = 25;
    - contador\_amostras = 1:
    - multiplier = 2;
    - frac = 0.6;
    - step = 25;
    - wlen = 25;
    -  contador\_amostras = 1:

Conclusão

Ao longo da realização deste trabalho pode concluir que na primeira solução optada que existe a necessidade de por vezes, de palavra para palavra, variarmos o tipo de modelo de ruído a ser construído, uma vez que, por vezes (a maioria), é de melhor tom escolher um modelo de ruído mais centralizado (com desvio-padrão x 1) pois pela variação dos restantes parâmetros, se consegue colmatar as deficiências de alguns sinais, no entanto por vezes para maior optimização consegue-se melhores resultados com um modelo mais disperso (k x desvio-padrão). Sendo muito mais fácil perder components de fala válidas no modelo mais disperse, não desprezando por complete este modelo, pois temos de ter em conta sempre o objectivo qualidade/fibilidade final que desejamos.

No caso da escolha dos parâmetros wlen e nseg, o pretendido foi dar valores simples diminuendo o esforço computacional. No entanto, admito que step devia ter sido menor do que wlen, para existir uma avaliiação de amostras mais rigorosa.

Outros observações tiradas, ao longo deste trabalho foram por exemplo de que a falta de uma ferramenta já desenhada tornaria o trabalho mais minucioso. Além de que este método de avaliação pode não ser o melhor, mas está próximo do aceitável, tornando deste modo os resultados satisfatórios e até quase os pretendidos em certos casos.

Desta forma penso que o saldo é positivo, embora reforce a ideia de que o grau de exigência de execução do trabalho, fosse difícil de ser respeitado, devido à falta de tempo existente, e por isso os resultados não são devidamente fundamentados com mais testes e avaliações.

Bibliografia

http://en.wikipedia.org/wiki/Signal-to-noise\_ratio

<http://pt.wikipedia.org/wiki/Ru%C3%ADdo_branco>

Anexos