



Centro Universitário SENAI CIMATEC  
Curso de Bacharelado em Engenharia Elétrica

# **Algoritmo para correção de tom por meio da análise em frequência**

**Cleber Couto Filho  
Ícaro Nascimento Queiroz**

**Salvador-BA, 19 de outubro de 2018**



Cleber Couto Filho  
Ícaro Nascimento Queiroz

## **Algoritmo para correção de tom por meio da análise em frequência**

Relatório apresentado como requisito parcial  
para obtenção de aprovação na disciplina  
Processamento Digital de Sinais, no centro  
universitário SENAI CIMATEC.

**Docente:** Leonardo Vasconsellos

Centro Universitário SENAI CIMATEC

Salvador-BA  
19 de outubro de 2018

# Lista de ilustrações

Figura 1 – Transformada de Fourier para sinais contínuos . . . . .	6
Figura 2 – Transformada Discreta de Fourier . . . . .	6
Figura 3 – Transformada de Fourier de Tempo Curto . . . . .	6
Figura 4 – Transformada de Fourier de Tempo Curto sendo aplicada em diversas frequências de um mesmo sinal . . . . .	7
Figura 5 – Efeitos da função de janelamento em um sinal no espectro de frequência	8
Figura 6 – Função da janela Hanning . . . . .	9
Figura 7 – Função da janela Hanning . . . . .	9
Figura 8 – Variáveis do projeto . . . . .	14



# Introdução

Este relatório descreve os procedimentos e códigos utilizados para a criação de um sistema de processamento de áudio para correção de tons musicais.

## Teoria Musical

Na música a notação utilizada, chamada diastemática, os sons são representados graficamente, de maneira que seja possível mensurar os intervalos de frequências, o que indica diferentes notas musicais.

Para representação de sons mais longos ou curtos surge então a notação da grandeza tempo, que diferentemente da convenção comum, não possui um valor fixo, portanto, cria-se uma referência com relação à duração da notas semibreve, sendo elas a mínima, semínima, colcheia, semicolcheia, fusa, semifusa.

Outro aspecto extremamente importante da teoria musical é a frequência, grandeza qual classifica o quão graves ou agudos são os sons. Sons que apresentam maiores frequências são mais agudos que os de frequências mais baixas

Diferentes instrumentos são capazes de gerar sons diferentes mesmo que as frequências fundamentais sejam idênticas, o que em outras palavras significa dizer que um Ré gerado por um instrumento de corda não tem o mesmo som de um Ré gerado por um instrumento de sopro ou da voz humana, Este fenômeno, que é chamado de timbre, se dá principalmente pela geometria do instrumento, que define quais serão as outras componentes senoidais adicionadas à fundamental.

## Proposta

Projetar um algoritmo capaz de realizar a afinação de um som emitido pela voz humana para uma frequência selecionada através das técnicas de processamento como transformadas e mudanças de tom. Para a realização dos algoritmos foram tomadas as seguintes decisões:

- Aplicação dos conceitos de Processamento digital de Sinais;
- Identificação das componentes de frequência da voz;
- Realizar a troca de tom através das técnicas de Pitch Shift



# 1 Fundamentação Teórica

## Processamento de Áudio

As gravações de áudio se dão por meio de microfones e captadores de áudio, os quais realizam a conversão por meio de um transdutor, o qual converte a variação da pressão acústica em uma variação de tensão elétrica correspondente. Este sinal é convertido em pequenas amostras individuais espaçadas no tempo de maneira regular, constituindo a aproximação da forma de onda original.

Este processo é conhecido como conversão analógico-digital. O número de amostras retiradas da onda original no período de um segundo, é chamada frequência de amostragem, e quanto mais elevado o seu valor, mais fiel será a representação do sinal no domínio digital. De acordo com o teorema de Nyquist, o limite mínimo para a frequência de amostragem de qualquer sinal é o dobro da sua frequência original. A representação do sinal de áudio no domínio digital é apresentada como uma sequência de palavras onde o número de bits determina a resolução em amplitude do sinal.

No processo de reprodução de áudio, acontece o inverso da situação original, onde o sinal digital é enviado a um conversor digital-analógico, responsável pela reconstrução do sinal para que ele possa ser reproduzido em alto-falantes, caixas de som, ou qualquer aparelho que possa reproduzir sinais de áudio.

Dentro do domínio digital, o sinal de áudio pode ser tratado utilizando todas as técnicas de processamento, tais como as FFTs, DFTs filtros digitais, técnicas de janelamento e filtros digitais. Sendo assim, torna-se extremamente interessante e necessário realizar diversos tratamentos nestes sinais no domínio digital para reduzir as interferências externas, ruídos e desafinação, mantendo assim uma afinação constante em uma música.

Uma das principais técnicas utilizadas para correção de afinação é a de pitch-shift, a qual consiste na mudança do tom de um sinal de áudio sem modificar o tamanho dele. A diferença de um deslocamento em frequência para o pitch-shift é justamente o fato de que num deslocamento em frequência, existe um deslocamento do espectro do som, enquanto um phase-shift dilata ou comprime o espectro de som.. O Pashe-Vocoder é uma técnica de pitch-shift muito utilizada por produtores musicais.

## STFT (Short Time Fourier Transform)

Uma das técnicas mais utilizadas em processamento digital de sinais é a Transformada Discreta de Fourier (DFT), a qual é uma variação da Transformada de Fourier para



sinais discretos. A transformada de Fourier define a redução de uma função periódica a um somatório de senos e cossenos. Este procedimento matemático gera uma representação de um sinal originalmente no domínio do tempo, em uma representação no domínio da frequência.

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(\omega) e^{i\omega x} d\omega.$$

Figura 1 – Transformada de Fourier para sinais contínuos

$$X(n) = \sum_{k=0}^{k=N-1} x(k) e^{-j(2\pi/N)nk}$$

Figura 2 – Transformada Discreta de Fourier

As transformadas de Fourier se aplicam somente a sinais de funções estacionárias, onde o espectro de frequência é fixo e não variam com o tempo.

Os sinais de áudio gerados pela voz humana se encontram dentro do espectro de frequências entre 50 a 3400 Hz, e a sua principal característica é a de que a sua frequência não é constante no tempo, o que dá ao sinal da voz humana a característica da não-ergodicidade (seu sinal não mantém as propriedades estatísticas ao longo do tempo), sendo assim, a utilização da STFT (Short Time Fourier Transform) se torna bastante eficaz em sinais dessa natureza.

A STFT é um algoritmo desenvolvido com base na transformada discreta de Fourier, diferenciando-se pela inclusão de uma função de janelamento  $w(t)$ . Sua principal aplicação é para funções cujo o espectro de frequência varia com o tempo.

$$\mathbf{STFT} \{x(t)\} \equiv X(\tau, \omega) = \int_{-\infty}^{\infty} x(t) w(t - \tau) e^{-j\omega t} dt$$

Figura 3 – Transformada de Fourier de Tempo Curto

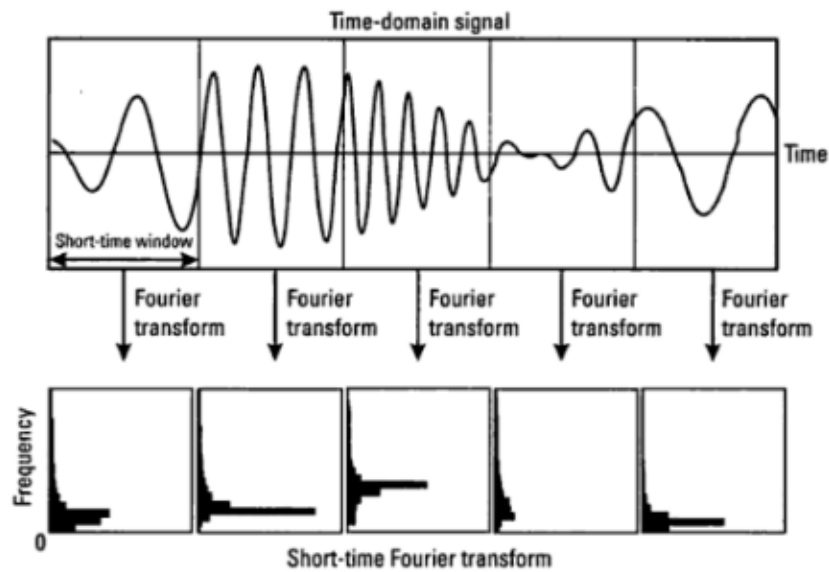


Figura 4 – Transformada de Fourier de Tempo Curto sendo aplicada em diversas frequências de um mesmo sinal

O principal propósito da utilização de uma STFT é separar o sinal em pequenos intervalos que possam ser tratados individualmente, obtidos através da janela que está inserida na transformada. Desta maneira a modificação de frequência se dá de forma independente, sem a alteração de tempo e vice versa.

## Funções de Janelamento

Para aplicações que consistem na amostragem de sinais, a amostragem, por ser finita, resulta em uma forma de onda truncada com características diferentes do sinal original, consequentemente a influência do vazamento espectral torna-se maior para uma situação como esta, gerando uma perda de informação do sinal original. Para reduzir os efeitos das imperfeições de amostragem, melhorando a qualidade da reconstrução do sinal é a aplicação de uma função de janelamento.

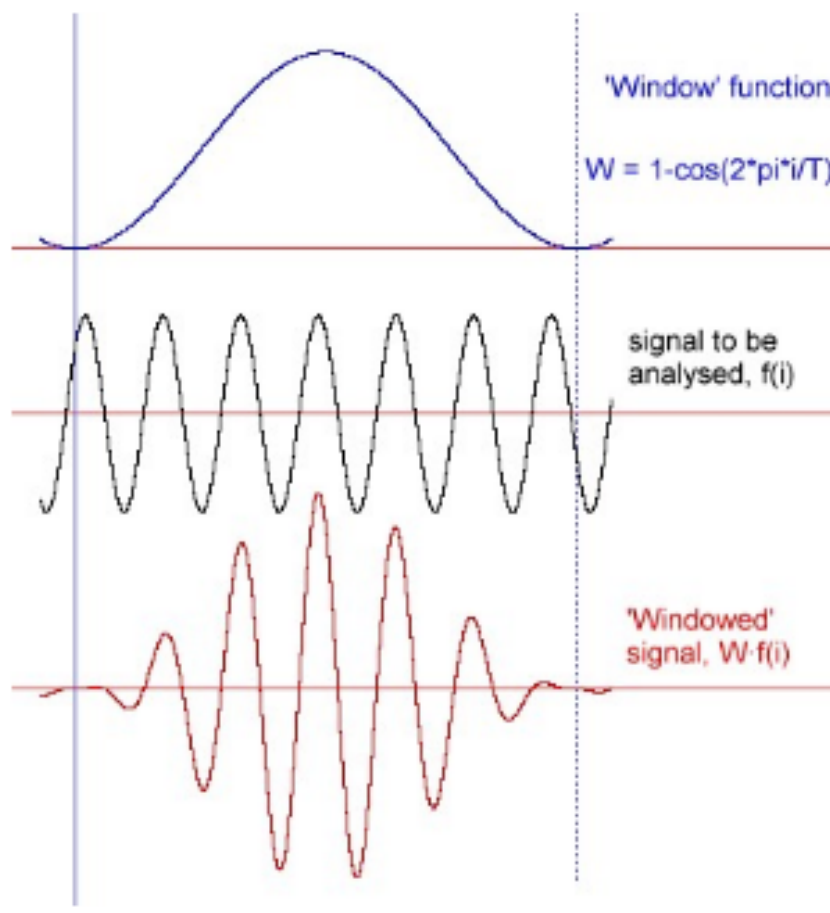


Figura 5 – Efeitos da função de janelamento em um sinal no espectro de frequência

A utilização de uma função de janelamento permite uma definição do período de observação do sinal, redução dos efeitos do vazamento espectral e a separação do sinal de pequena amplitude com frequências muito próximas. A aplicação de uma função de janelamento no tempo, consiste na multiplicação da função original pela função, o que equivale a uma convolução no domínio da frequência. Existem diversas funções de janelamento, as quais possuem diferentes características e aplicações dependendo principalmente dos parâmetros desejados do sinal original.

- Retangular
- Hanning
- Hamming
- Blackman
- Kaiser-Bessel

## Janela Hanning

Dentre as funções de janelamento existentes, a função Hanning é a mais comumente utilizada na produção musical. O formato desta janela é similar ao de meio ciclo de uma onda cossenoidal. Suas características de baixo vazamento espectral e formato de onda bem similar ao formato cossenóide,. Torna-se recomendável utilizar portanto a janela para análises de sinais com transientes maiores que de duração da própria janela.

$$w[n] = 0.5 - 0.5 * \cos\left(\frac{2\pi * n}{N}\right), n = 0, 1, 2, \dots, N - 1 \quad (1.1)$$

$$w[n] = 0.5 - 0.5\cos(2\pi n/N), n = 0, 1, 2, \dots, N-1$$

Figura 6 – Função da janela Hanning

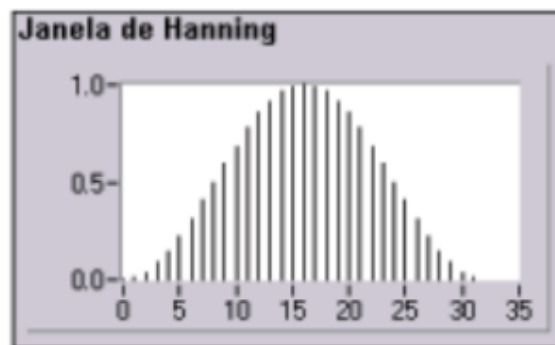


Figura 7 – Função da janela Hanning



## 2 Resultados

### 2.1 Dados de entrada

#### 2.1.1 Dados Gerais

A altura escolhida para as torres foi de 0 e 9 para manter a linha de visada direta em paralelo com o chão e assim facilitando os cálculos e diminuindo os erros, os outros dados estão mostrados na tabela abaixo.

Distância Total (m)	0,96713
$\lambda$	$0,001086957 \times 10^{-6}$
Altura das torres	0/9

#### 2.1.2 Dados dos Obstáculos

X	325
Y	7
$d_1$	0,475
$d_2$	0,49213

### 2.2 Memorial de Cálculo

#### 2.2.1 Frequência

Não foi necessário calcular a frequência, o valor utilizado foi de 920MHz e sua escolha está justificada em [1](#).

#### 2.2.2 Raio da Parábola

Com os valores de  $X$  e  $Y$  do obstáculo o valor encontrado para o raio da parábola foi de:

$$r_{parabola} = 1,886160714 \quad (2.1)$$

#### 2.2.3 Atenuação do obstáculo

Com os valores de  $d_1$  e  $d_2$  o  $\alpha$  encontrado teve valor de :

$$\alpha = 0,6703932617 \quad (2.2)$$

O parâmetro  $H_c$  foi encontrado por meio dos dados fornecidos, como mostrado na figura a seguir:

O parâmetro  $r_f$  foi calculado por meio da equação de *Fresnel*, com o valor dos parâmetros a seguir, foi encontrado no gráfico o valor da atenuação do obstáculo.

$r_f$	8,866
$H_c$	5
$\frac{H_c}{r_f}$	0,563
$L_{obstaculo}$	22,5dB

#### 2.2.4 Atenuação no Espaço livre

Com os valores de  $d_1$ ,  $d_2$  e  $f$  conhecidos o valor encontrado para  $L$  foi:

$$L = 91,43dB \quad (2.3)$$

#### 2.2.5 Atenuação total

A atenuação total consiste nas soma das atenuação encontradas, logo:

$$L_{tot} = L + L_{obstaculo} = 91,43 + 22,5 = 113,9dB \quad (2.4)$$

### 2.3 Escolha do transmissor e da antena

A escolha dos módulos e da antena se deu baseado na frequência escolhida para os cálculos e na versatilidade de cada dispositivo.

O modelo das antenas foi o *Yagi(AirMax Antenna 900Mhz)*, devido a sua faixa de trabalho e alta potência.

O modelo do transmissor foi o *Ubiquiti Networks(Rocket M9)* pot ser recomendado para trabalhar em conjunto com o modelo de antena *Yagi*.

### 2.4 Receptor

Escolha do receptor é encontrado à partir das potências das antenas, do módulo transmissor e das perdas durante a transmissão. Sua potência foi calculada da seguinte forma:

$$P_{receptor} = P_{antena_{tx}} + P_{antena_{rx}} + P_{transmissor} - L_{tot} \quad (2.5)$$

A mesma antena utilizada para transmissão é utilizada para recepção, os dados da sua potência estão disponíveis nos seu *datasheet* onde  $P_{antena_{tx}} = P_{antena_{rx}} = 19dB$ . A

potência do transmissor também está disponível no datasheet, onde  $P_{transmissor} = 28dBm$ . O valor encontrado para potência do receptor foi de  $P_{receptor} = -47.9dBi$

Com o valor de  $-47.9dBi$ , o módulo *Ubiquiti Networks(Rocket M9)* também poderá ser utilizado para recepção, tornando o sistema mais simplificado já que ambos receptores e transmissores estarão utilizando antenas recomendadas no datasheet.



## Variáveis do projeto

Todos os valores utilizados e calculados estão registrados na figura a seguir:

Localização A			Escolha da antena	
Latitude	12° 42' 20.14" S			
Longitude	38° 11' 14" O		Transmissor	
Elevação (m)	40		Modelo	Yagi airMAX
Localização B			potencia dBi	19
Latitude	12° 42' 28.13" S			
Longitude	38° 10' 43.03" O		Receptor	
Elevação (m)	31		Modelo	Yagi airMAX
Distancia D (m)	0,96713		potencia dBi	19
Torre A (m)	0			
Torre B (m)	9		Modulo	
Frequencia	920			
			Transmissor	
X obstaculo (m)	325		Modelo	m900 rocket
Y obstaculo (m)	7		potencia dBm	28
R Obstáculo	1,886160714			
D1 (km)	0,475		RECEPTOR	
D2 (km)	0,49213		Modelo	m9 rocket
alfa	0,6703932617		potencia dBm	-47,93545365
Rf	8,866203803			
Altura visada direta (m)	40			
Altura obstaculo (m)	45			
Hc (m)	5			
h/rf	0,5639392135			
Atuaçãoção Obstaculo	22,5			
Atenuação espaço livre	91,43545365			
Atenuação total (db)	113,9354536			

Figura 8 – Variáveis do projeto

## Referências

- [1] TUDE, Eduardo. Enlace rádio digital ponto a ponto. 2004.

ANEXO

## Função GravaAudio

```
1 function audio = gravaAudio(fs,duration)
2 %Cria um objeto do tipo audio recorder com uma frequencia de
   amostragem fs, 16 bit, canal mono
3 recorder = audiorecorder(fs,16,1);
4 record(recorder);
5 pause(duration);
6
7 audio = getaudiodata(recorder);
8 end
```

## Função istft

```
1 %%Downloaded from http://www.ee.columbia.edu/~dpwe/e4810/matlab/
   s24/istft.m
2
3 function x = istft(d, ftsize, w, h)
4 % X = istft(D, F, W, H)                                Inverse short-time
   Fourier transform.
5 %     Performs overlap-add resynthesis from the short-time
   Fourier transform
6 %     data in D. Each column of D is taken as the result of
   an F-point
7 %     fft; each successive frame was offset by H points (
   default
8 %     W/2, or F/2 if W==0). Data is hann-windowed at W pts, or
9 %     W = 0 gives a rectangular window (default);
10 %     W as a vector uses that as window.
11 %     This version scales the output so the loop gain is 1.0
   for
12 %     either hann-win an-syn with 25% overlap, or hann-win on
13 %     analysis and rect-win (W=0) on synthesis with 50%
   overlap.
14 % dpwe 1994may24. Uses built-in 'ifft' etc.
15 % $Header: /home/empire6/dpwe/public_html/resources/matlab/pvoc/
   RCS/istft.m,v 1.5 2010/08/12 20:39:42 dpwe Exp $
16
17 if nargin < 2; ftsize = 2*(size(d,1)-1); end
```

---

```

18 if nargin < 3; w = 0; end
19 if nargin < 4; h = 0; end % will become winlen/2 later
20
21 s = size(d);
22 if s(1) ~= (ftsize/2)+1
23     error('number of rows should be fftsize/2+1')
24 end
25 cols = s(2);
26
27 if length(w) == 1
28     if w == 0
29         % special case: rectangular window
30         win = ones(1,ftsize);
31     else
32         if rem(w, 2) == 0 % force window to be odd-len
33             w = w + 1;
34         end
35         halflen = (w-1)/2;
36         halff = ftsize/2;
37         halfwin = 0.5 * ( 1 + cos( pi * (0:halflen)/halflen));
38         win = zeros(1, ftsize);
39         acthalflen = min(halff, halflen);
40         win((halff+1):(halff+acthalflen)) = halfwin(1:acthalflen);
41         win((halff+1):-1:(halff-acthalflen+2)) = halfwin(1:
            acthalflen);
42         % 2009-01-06: Make stft-istft loop be identity for 25% hop
43         win = 2/3*win;
44     end
45 else
46     win = w;
47 end
48
49 w = length(win);
50 % now can set default hop
51 if h == 0
52     h = floor(w/2);
53 end
54
55 xlen = ftsize + (cols-1)*h;

```

```

56 x = zeros(1,xlen);
57
58 for b = 0:h:(h*(cols-1))
59     ft = d(:,1+b/h)';
60     ft = [ft, conj(ft([(ftsize/2):-1:2]))];
61     px = real(fft(ft));
62     x((b+1):(b+ftsize)) = x((b+1):(b+ftsize))+px.*win;
63 end;
64 end

```

### Função mostraEspectrograma

```

1 function Y = mostraEspectrograma(S,F,T)
2 %%Mostra o spectrograma do sinal S, recebendo o vetor de
   frequencia F e o vetor de tempo T
3
4 X = abs(S);
5 imagesc(T,F,S);
6 colorbar;
7 axis xy
8 xlabel('Time (s)')
9 ylabel('Freq (Hz)')
10
11 end

```

### Função tabelaDoMaior

```

1 function pitches = tabelaDoMaior()
2 %%Retorna todas as frequencias da escala de Do maior
3
4 pitches = [16.352 18.354 20.602 21.827 24.500 27.500 30.868
   32.703 36.708 41.203 43.654 48.999 55.000 61.735 65.406
   73.416 82.407 87.307 97.999 110.000 123.470 130.800 146.830
   164.810 174.610 196.000 220.000 246.940 261.630 293.660
   329.630 349.230 392.000 440.000 493.880 523.250 587.330
   659.260 698.460 783.990 880.000 987.770 1046.500 1174.700
   1318.500 1396.900 1568.000 1760.000 1975.500 2093.000
   2349.300 2637.000 2793.800 3136.000 3520.000 3951.100
   4186.000 4698.600 5274.000 5587.700 6271.900 7040.000

```

```
7902.100 8372.000 9397.300 10548.100 11175.300 12543.900
14080.000 15804.300 16744.000 18794.500 21096.200 22350.600];
```

5

6 **end**

### Função compareToPitches

```
1 %%A funcao recebe um tom e a tabela contadndo a escola de tons,
   retornando o tom mais proximo da escala
2 function exactpitch = compareToPitches(pitch, pitchtable)
3
4 %%Organiza a tabela em ordem crescente
5 sortedpitchtable = sort(pitchtable);
6
7 %%Cria vetor de zeros
8 midpitchtable = zeros(1,length(sortedpitchtable)-1);
9
10 %%Variavel que indica se o tom mais proximo foi encontrado
11 found = 0;
12
13 %%Gera uma tabela de tons na media dos tons da tabela de tons
   para tornar a comparacao
14 %%mais facil.
15 %%log2(0.5) e utilizado 0.5 afim de contabilizar para percepcao
   do som
16 for i = 1:(length(sortedpitchtable)-1)
17     midpitchtable(i) = (sortedpitchtable(i+1) - sortedpitchtable
        (i))*log2(0.5) + sortedpitchtable(i);
18 end
19
20 %%Caso a frequencia seja menor que o menor tom da tabela, se
   retorna o menor tom
21 if (pitch <= midpitchtable(1))
22     exactpitch = sortedpitchtable(1);
23     found = 1;
24 end
25
26 %%Se o tom exato nao for encontrado ainda considera o maior tom
   da tabela
```

```

27 if (found==0)
28     if (pitch > midpitchtable(length(midpitchtable)))
29         exactpitch = sortedpitchtable(length(sortedpitchtable));
30         found = 1;
31     end
32 end
33
34 %%Se o tom exato nao foi encontrado ainda, compare o a todos os
    valores
35 %da midpitchtable e a saida corresponde ao tom na tabela de tom
36 i = 1;
37 while(found==0)
38     if(pitch > midpitchtable(i) && pitch <= midpitchtable(i+1))
39         exactpitch = sortedpitchtable(i+1);
40         found=1;
41     end
42     i = i+1;
43 end
44
45 end

```

### Função pitchCorrector

```

1 %%A funcao recebe um tom e a tabela contadndo a escola de tons,
    retornando o tom mais proximo da escala
2 function exactpitch = compareToPitches(pitch, pitchtable)
3
4 %%Organiza a tabela em ordem crescente
5 sortedpitchtable = sort(pitchtable);
6
7 %%Cria vetor de zeros
8 midpitchtable = zeros(1,length(sortedpitchtable)-1);
9
10 %%Variavel que indica se o tom mais proximo foi encontrado
11 found = 0;
12
13 %%Gera uma tabela de tons na media dos tons da tabela de tons
    para tornar a comparacao
14 %%mais facil.

```



```

15 %%log2(0.5) e utilizado 0.5 afim de contabilizar para percepcao
    do som
16 for i = 1:(length(sortedpitchtable)-1)
17     midpitchtable(i) = (sortedpitchtable(i+1) - sortedpitchtable
        (i))*log2(0.5) + sortedpitchtable(i);
18 end
19
20 %%Caso a frequencia seja menor que o menor tom da tabela, se
    retorna o menor tom
21 if (pitch <= midpitchtable(1))
22     exactpitch = sortedpitchtable(1);
23     found = 1;
24 end
25
26 %%Se o tom exato nao for encontrado ainda considera o maior tom
    da tabela
27 if (found==0)
28     if (pitch > midpitchtable(length(midpitchtable)))
29         exactpitch = sortedpitchtable(length(sortedpitchtable));
30         found = 1;
31     end
32 end
33
34 %%Se o tom exato nao foi encontrado ainda, compare o a todos os
    valores
35 %da midpitchtable e a saida corresponde ao tom na tabela de tom
36 i = 1;
37 while(found==0)
38     if(pitch > midpitchtable(i) && pitch <= midpitchtable(i+1))
39         exactpitch = sortedpitchtable(i+1);
40         found=1;
41     end
42     i = i+1;
43 end
44
45 end

```

## Código para Testes

```
1 fs = 44100; %% Frequencia de Amostragem
2 winsize = 4000; %% Tamanho da janela
3 NFFT = 4096; %% Numero de amostras da stft
4 noverlap = 2000; %%Numero de amostras que nao sofrerao overlap
5 audio_duration = 3; %%Duracao do audio
6 window = hanning(winsize); %%Janela a ser utilizada na stft
7 signal = gravaAudio(fs ,audio_duration);
8
9 [signal_transformed F T P] = spectrogram(signal , window ,
    noverlap , NFFT, fs);
10
11 sound_corrected = pitchCorrector(signal_transformed , F, P,
    tabelaDoMaior());
12 sound_corrected = conj(sound_corrected); % Faz o conjugado
13 isignal = istft(sound_corrected ,NFFT,winsize ,noverlap); % Faz a
    transformada inversa do sinal
14
15 mostraEspectrograma(isignal ,F,T); % Mostra o espectrograma
16
17 sound(isignal , fs)%toca o som
```