**UNIVERSIDADE FEDERAL DO PARANÁ**

CURSO DE ENGENHARIA ELÉTRICA

**PROCESSAMENTO DE UM SINAL DE AUDIO PARA DETECÇÃO DE FEEDBACK ACÚSTICO**

Aluno: André Heidemann Iarozinski

Professor: Eduardo Parente Ribeiro, Dr.

CURITIBA

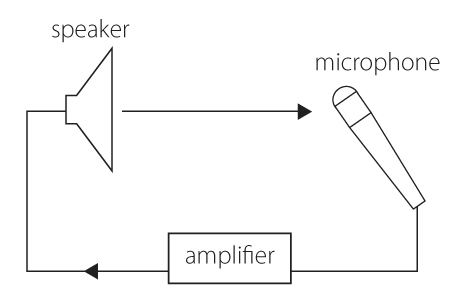
2015

1. **Introdução**

O feedback acústico, mais conhecido como “microfonia” é um problema muito comum em qualquer sistema de áudio. Ocorre devido ao som que é emitido pelo alto falante e é captado pelo microfone, retorna ao microfone criando um ciclo vicioso. Isso provoca uma reação em cadeia onde o som é amplificado sem parar causando o fenômeno conhecido como “feedback acústico”.

Isso ocorre geralmente pelo mau posicionamento dos microfones em um palco, ou pela simples movimentação do cantor. A acústica do ambiente também influencia, pois, devido as reflexões nas superfícies do ambiente, algumas frequências sofrem interferência construtiva fazendo com que o efeito feedback tenha mais chances de ocorrer.

Para prevenir o problema, geralmente o técnico de som responsável, durante a passagem de som realiza testes no local e atenua as frequências que estão sofrendo feedback com um equalizador paramétrico, porém, durante um show, a movimentação no palco é inevitável e a realimentação de certas frequências podem voltar a ocorrer nos microfones.



1. **Objetivo**

O objetivo deste trabalho é desenvolver uma forma eficiente de detecção do efeito feedback em um sinal de áudio. Analisando o espectro de frequências do sinal, será feito um processamento computacional a fim de descobrir qualquer realimentação harmônica indesejável presente no conteúdo espectral.

A princípio o processamento não será feito com um arquivo de áudio .WAV e possivelmente em outra oportunidade será realizado um processamento em tempo real.

1. **Metodologia**

O trabalho será realizado utilizando o software MATLAB, na qual deveram ser criadas as funções necessárias de modo a atingir o objetivo principal.

Utilizando as funções do MATLAB será criado um algoritmo que irá buscar as faixas de frequências que estão sendo realimentadas.

O feedback acontece de forma extremamente rápida, fazendo com que uma frequência atinja o ganho máximo em milésimos de segundo. O algoritmo irá monitorar o espectro e dividi-lo em várias faixas de frequências. A condição para uma frequência ser considerada instável é ter um desvio padrão muito baixo da variação temporal do espectro e estar com amplitude muito acima das faixas de frequências próximas.

**4.Desenvolvimento**

De início foram utilizados 3 arquivos de áudio. Um contendo um trecho instrumental de uma música e outros 2 arquivos de captações de feedback´s. Todos os arquivos foram convertidos para o formato .WAV e modificados para taxa de amostragem de 16kHz.

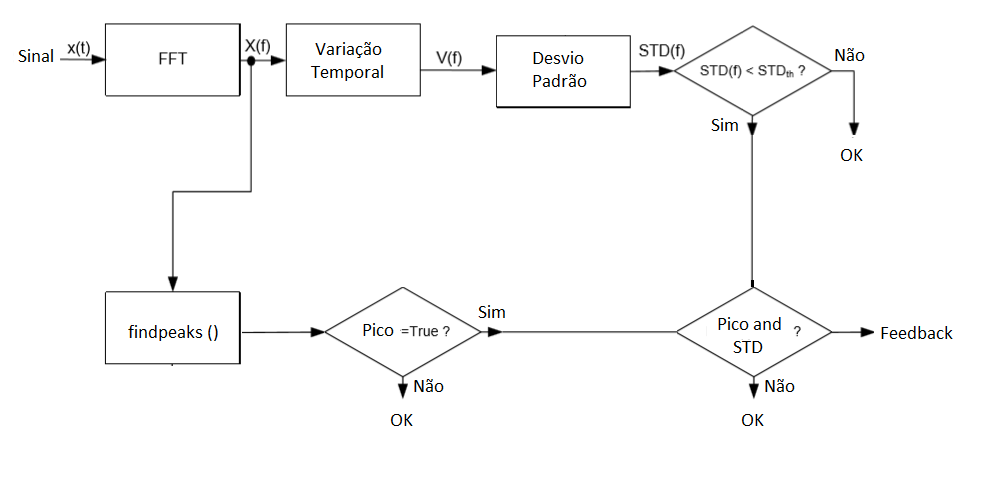
Os arquivos foram nomeados da seguinte forma:

“semfeed.wav” o arquivo com o trecho da música sem alteração

“comfeed1” o arquivo “semfeed” mais a adição de um feedback

“comfeed2” o arquivo “semfeed” mais a adição de um feedback em outra frequência

No MatLab, o algoritmo foi desenvolvido baseado no seguinte fluxograma:



De – Dado um arquivo selecionado, é feita a sua transformada rápida de Fourier. Em seguida é feita a verificação de picos no vetor através da função “findpeaks”. Depois é feita a variação temporal do espectro X(f) com o espectro no instante anterior e calculado o desvio padrão. Se ambas as condições forem satisfeitas para determinada faixa de frequência, então esta, é uma possível faixa de frequência instável.

O processo descrito no fluxograma acima é executado no algoritmo a cada 0.01 segundos até atingir o tempo total de análise (7 segundos). A primeira tentativa de execução do código foi considerado intervalos de tempo de 0.1 segundos mas não se obteve resultados precisos.

%%%%%%%% INICIO DO ALGORITIMO

bs = 100; % "bandsize"(Hz) , tamanho das faixas de freq

nbs=80; %num de bandas

fs=16000; % fornecendo o valor da freq de amost do sinal de audio

g=zeros(1,80);

l=zeros(1,80);

for k=0:700 %%%%%%%%%%%%%%%%%%%% Loop for principal p/ cada uma das 701 janelas de 0.01s do sinal de audio (7.01 seg de audio analisados)

x = audioread('C:\Users\andre iarozinski\Desktop\audios projeto\comfeed1.wav'); %%%% carregando arquivo do sinal de audio

faixas=1:80; %%%%%%%%%%%%% reiniciando os vetores de pico e desv padrao para uma nova comparacao a cada ciclo k

faixas2=1:80;

m=1:80;

b=1:80;

x = x(((fs\*0.01\*k)+1):(fs\*0.01\*(k+1))); %% coletando janelas da entrada/sinal a cada ciclo for"k" para analise,,, x(1:160) , x(161:320), ... x(1441:1600)

S = fft(x); % transformada de janelas de 0.1s da entrada (x) ainda com espectro "duplo" sem unidades(Hz)

L = length(S); % normalizando S

P2 = abs(S/L);

P = P2(1:L/2+1);

P(2:end-1) = 2\*P(2:end-1); % \*\*\* P é o módulo do espectro positivo de S com as amplitudes normalizadas, 8k ptos (0-8kHz)

f = fs\*(0:(L/2))/L; % 16000=fs , agora com frequencia 0 até 8kHz

% a partir do vetor P, crio o vetor b

for i=0:(nbs-1)

b(i+1)= (sum(P(i\*(bs/100)+1:(i+1)\*(bs/100))))/nbs; % criando vetor b, contem a media das amplit p/ cada banda, b(1)=sum(P(1:80))/100,b(2)=sum(P(81:160))/100...

end

%%%%%%%%%%%%%%%%%%%%%%%%%%% Loop de análise de pico

[pks,locs] = findpeaks(b,'Threshold',0.001); %% funcao que retorna valores de pico maiores que o definido e a sua posicao (band de freq)

locs; %%% vetor com a posicao das faixas com pico, mas o vetor nao tem tamanho 50 ainda...

locs=[locs 0];

faixas=1:80; %%% criando vetor de tam 50 para armazenas os valores de "locs"

for c=1:80

if faixas(c)~=locs

faixas(c)=0; %%%%%% zerando faixas de frequencia em que nao foram consideradas pico pela funcao findpeaks

end

end

%%%%%%%%%%%%%%%%%%%% loops p/ analise do desvio padrao STD

if k>1

faixas2=1:80; %% vetor de comparacao que zera os os valores das bandas que tem desv pad maior que 0.1

v=[b;q]; %%% vetor variacao temporal do espectro de uma janela

for f=1:80

ST(f)=std(v(:,f)); %%% cria o vetor do desv padrao entre b e q

end

for f=1:80

if ST(f)>0.00005

faixas2(f)=0; %% vetor de comparacao que zera os os valores das bandas que tem desv pad maior que 0.1

end

end

end

q=b; % armazenando vetor b p/ fazer varicao temp no proximo loop k

%%%%%%%%%%%%%%% se as 2 condicoes forem satisteitas p/ alguma banda, feedback detectado",,, indicar banda, tempo da janela, e plotar fft com pico

if k>1

m=1:80; %%%%% vetor p/ calcular diferenca , satistazer as 2 cond, bandas com pico e com desv pad menor que 0.1

end

for c=1:80

a(c)=faixas(c)-faixas2(c); %%%%% somente as bandas que satisfazem as 2 condicoes nao serao zeradas 2 0 8 0 - 0 1 8 3 = 2 -1 0 -3

if a(c)~=0

m(c)=0;

end

if faixas(c)==0 %%%%%%%%%%% se alguma banda nao foi considerada pico, ela nao pode ser um feedback

m(c)=0;

end

if faixas2(c)==0 %%%%%%%%%%% se alguma banda nao teve desvio pad menor que o definido, ela nao pode ser um feedback

m(c)=0;

end

end

if (sum(m))~=0 %%%%% fazendo a soma do vetor g a cada loop quando se obtem frequencias "suspeitas"

g=(g+m);

end

if (sum(g))>=200 %%%%%%%%% momento em que é detectada as frequencias instáveis

if (sum(g))<=300

tempo=k/100;

tempo %%%%%% mostrando tempo de detecção

end

end

end %%%%%%%%%%%%%%% fim do loop principal

for z=1:80

if g(z)<= 100

g(z) = 0;

end %%%% normalizando amplitudes finais p/1

if g(z)>= 100

g(z) = 1;

end

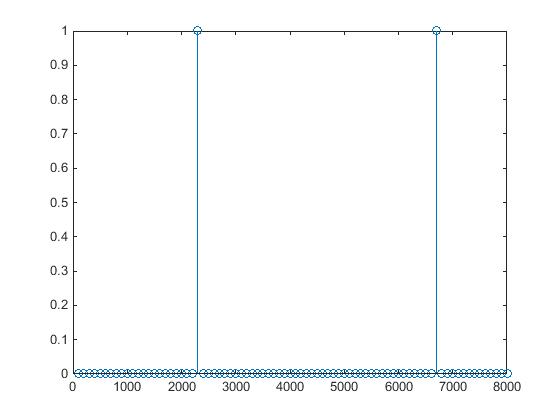
end

d=1:80; %% fator de escala p/ploatr ate 8kHz

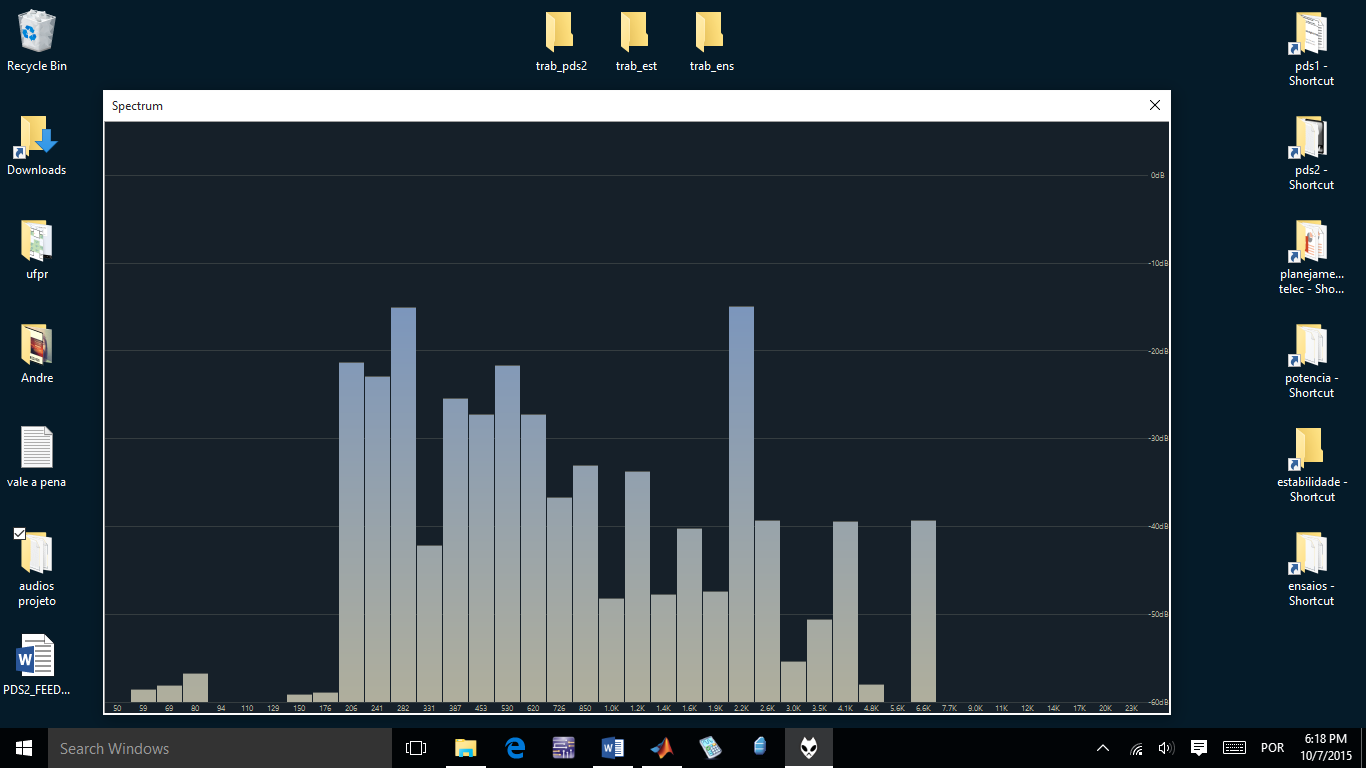
stem(d\*bs,g) %%% plotando bandas de frequencia com fb detectadas

**5.Conclusões**

O algoritmo conseguiu detectar com precisão considerável as frequências problemáticas nos sinais. Para o arquivo “comfeed1” o algoritmo detectou as faixas 2.2k-2.3kHz e 6.6-6.7kHz no instante de tempo 3.57 segundos.



Observando o espectro do sinal entre 3 e 4 segundos onde o eixo das abcissas (frequências) está em escala log. Nota-se claramente a frequência 2.2Khz no meio da imagem. Com amplitude muito próxima da frequência 292Hz (estável). A direita a terceira harmônica do sinal detectada, 6.6kHz.



O algoritmo também não acusou nenhuma frequência no arquivo “semfeed” e conseguiu detectar a frequência instável 1.8kHz no arquivo “comfeed2”.

O tempo de execução do código ficou relativamente alto, na ordem da duração de cada arquivo (10segundos), mas funcionou conforme o esperado.

**6.Referências**

1. Low-Complexity Howling Detection based on Statistical Analysis of Temporal Spectra. Jae-Won Lee1 and Seung Ho Choi2, International Journal of Multimedia and Ubiquitous Engineering Vol.8, No.5 (2013).
2. Evaluation of various Algorithms to detect acoustic Feedback (Toningenieur Projekt.) Sebastian Braun, Betreuung: DI Hannes Pomberger Graz, February 19, 2012

3- Definição do Audio Feedback

<https://en.wikipedia.org/wiki/Audio_feedback>

Acessado em 15/09/2015

4- How Stuff Works – Sobre o fenômeno da microfonia

<http://electronics.howstuffworks.com/gadgets/audio-music/question263.htm>

Acessado em 16/09/2015

5- Site do MatLab sobre PDS aplicado aos sinais de áudio

<http://www.mathworks.com/discovery/audio-signal-processing.html>

Acessado em 17/09/2015