**به نام خدا**

**پروژه پردازش گفتار**

**سید شایان دانشور**

**9726523**

**استاد درس: دکتر دوست**

**تیر 1400**

سوال 1)

در این پروژه می خواهیم فرکانس گام و فرمنت های یک واکدار را به کمک تبدیل کپستروم بدست اوریم. از فایل a.wav موجود در سایت، در این پروژه استفاده شده است.

مراحل انجام کار:

ابتدا باید فریم بندی انجام داده و پنجره مناسبی روی صوت اعمال کنیم، طول فریم 25 میلی ثانیه و شیفت فریم 10 میلی ثانیه در نظر گرفته شده است. از پنجره همینگ (hamming) برای پنجره بندی (windowing) استفاده کردیم، سپس در هر فریم، مقدار dc را حذف میکنیم تا نویز نداشته باشیم.

FL\_sec = 0.025; % frame length in seconds

FSL\_sec = 0.01; % frame shift length in seconds

lifter\_cutoff = 20; % n in liftering of cepstrum

[data,fs]=audioread('a.wav'); % reading the audio

frame\_length = FL\_sec \* fs;

frame\_shift\_length = FSL\_sec \* fs;

fn=(length(data)- frame\_length)/frame\_shift\_length + 1; % number of frames

window=hamming(frame\_length); % hamming window

pitch\_freqs = zeros(1, fn); % pitch frequencies of frames

formants = zeros(4, fn); % 4 formants of frames

energy\_of\_frames=zeros(1,fn);% energy of frames

for i=1:fn

frame = data((i-1)\* frame\_shift\_length + 1:(i-1)\* frame\_shift\_length + frame\_length); % framing speech

frame = frame.\*window; % windowing frame using hamming window

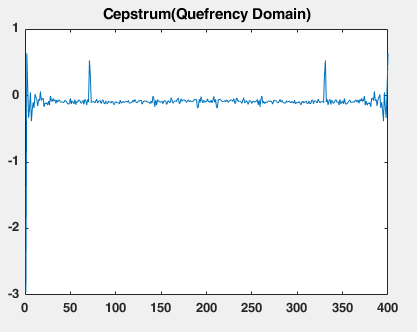
frame = frame - sum(frame)/length(frame); % removing DC component from frame

حال که پیش پردازش مورد نیاز روی فریم انجام شد، روی آن تبدیل کپستروم را پیاده میکنیم، که خود ناشی از سه مرحله ی تبدیل فوریه گسسته، و گرفتن لگاریتم از قدر مطلق آن و در نهایت گرفتن تبدیل فوریه گسسته معکوس از آن می باشد، به طور خلاصه برای هر فریم اینطور توصیف می توان کرد:

quefrency= ifft(log(abs(fft(frame)))); %% cepstrum transform

که همانطور که دیده می شود، روی فریم این تبدیل زده می شود.

تبدیل کپستروم (ضرایب کپسترال) را برای نمونه یک فریم (پر انرژی ترین فریم) را می توانیم ببینیم:



حال برای محاسبه فرکانس گام (Pitch Frequency) باید ابتدا یک لیفتر زمان بالا روی این مقادیر بزنیم و با الگوریتم peak picking ماکسیمم ها را پیدا کنیم، در اینجا در ابتدا برای همه فریم ها این کار را انجام میدهیم، درنهایت فریم های بی انرژی را حذف کرده و از بقیه میانگین میگیریم، دلیل این کار دقت بیشتر در محاسبه فرمنت است.

رابطه محاسبه فرکانس گام به کمک تبدیل کپستروم:

%%%%% calculating pitch frequncy

high\_liftered = high\_time\_lifter(quefrency, lifter\_cutoff); % high time liftering

[pks,locs] = findpeaks(high\_liftered); % peak picking

[~,index] = max(pks);

peak\_index = locs(index); %% finding indexes of peaks

if size(peak\_index)> 0 % sometimes it is empty

pitch\_freqs(i)= fs/peak\_index; %% calculating pitch frequency of frame

end

که همانطور که دیده می شود با ضریب کات آف (همان n در اسلاید های برابر 20) لیفتر زمان بالا زده ایم، سپس به کمک تابع findPeaksموجود در متلب این ماکسیمم های محلی را پیدا کرده و اندیس آن هارا در می آوریم، سپس در صورت وجود چنین ماکسیمم های محلی، با تقسیم کردن فرکانس نمونه برداری گفتار بر مکان (اندیس) اولین ماکسیمم (peak)، فرکانس گام برای این فریم بدست می آید، در زیر تابع لیفتر زمان بالا را مشاهده می کنید: (که صرفا n عنصر آخر را نگه داشته باقی را صفر میکند)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%% Functions %%%%%%%%%%%%%

function output = high\_time\_lifter(frame, n)

high = zeros(1,length(frame));

for i=1:length(frame)

if i >= n

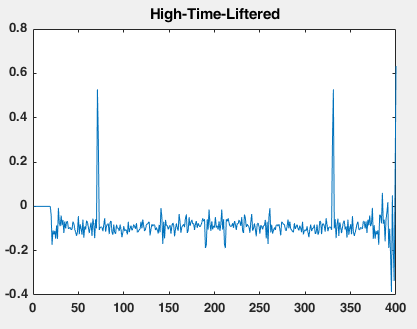
high(i)=frame(i);

end

end

output= high;

end



خب تا به اینجای کار فرکانس گام برای هر فریم بدست آمده، حال میرویم تا فرمنت ها را نیز برای هر فریم بدست آوریم، برای این کار باید روی تبدیل، لیفتر زمان پایین بزنیم ، دوباره روی آن تبدیل فوریه زده و سپس الگوریتم peak picking را اجرا کرده، 4 ماکسیمم محلی که زودتر دیده می شود را حساب کنیم.

low\_liftered = low\_time\_lifter(quefrency, lifter\_cutoff); % low-time liftering

low\_liftered\_dft = abs(log(fft(low\_liftered))); % calculating dft for formants

[~,locs] = findpeaks(low\_liftered\_dft); % peak picking

for formant\_no=1:4

if length(locs)>= formant\_no %% returned locs might be less than 4

formants(formant\_no,i) = locs(formant\_no) \* fs/fn;

end

end

در اینجا از تابع لیفتر زمان پایین استفاده شده و مابقی مشابه فرکانس گام است، با این تفاوت که در الگوریتم peak picking حال باید به دنبال 4 مقدار باشیم و هر کدام موجود بود برای این فریم در نظر بگیریم، در زیر تابع برای لیفتر زمان پایین آمده است:

function output = low\_time\_lifter(frame, n)

low = zeros(1,length(frame));

for i=1:length(frame)

if i < n

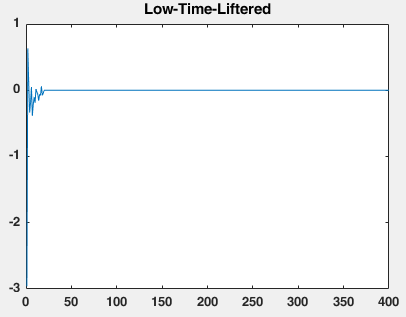
low(i)=frame(i);

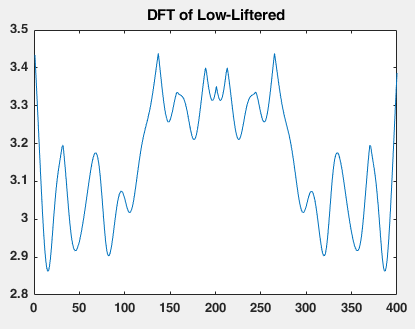
end

end

output= low;

end





حال هم فرکانس فرمت ها و هم فرکانس گام برای هر فریم را بدست آوردیم، اما همه فریم ها کاربردی نیستند و اصلا صوتی غیر از نویز (و یا غیر واکدار احتمالی) در آن ها وجود ندارد، به همین دلیل، انرژی هر فریم را نیز محاسبه میکنیم تا بر اساس آن تصمیم بگیریم. (البته از ZCR نیز می توان استفاده کرد)

% calculating energy of each frame

energy\_of\_frames(i) = sum(frame.\*frame)/ length(frame); % calculating frame's energy

حال باید حداکثر انرژی را مشخص کنیم و فریم هایی که تا حد قابل قبولی به حداکثر انرژی ماکزیمم نزدیک نیستند را در نظر نگیریم:

max\_energy = max(energy\_of\_frames); % finding max energy of a frame

max\_energy\_frame\_index = 0;

for i=1:fn %% finding index of frame with max energy

if max\_energy == energy\_of\_frames(i)

max\_energy\_frame\_index = i;

break

end

end

که در اینجا اندیس فریم با حداکثر انرژی را نیز گرفتیم تا اطلاعات این فریم را بتوانیم رسم کنیم، در ادامه اطلاعات مربوط به محاسبه دوباره تبدیل کپستروم، فیلتر های زمان بالا و پایین و فوریه از فیلتر پایین گذر و نیز رسم آن ها آمده است و استفاده دیگری ندارند:

% recalculating these values for ploting one frame

frame = data(max\_energy\_frame\_index \* frame\_shift\_length + 1:max\_energy\_frame\_index \* frame\_shift\_length + frame\_length);

frame = frame.\*window; % windowing frame using hamming window

frame = frame - sum(frame)/length(frame); % removing DC component from frame

quefrency= ifft(log(abs(fft(frame))));

high\_liftered = high\_time\_lifter(quefrency, lifter\_cutoff);

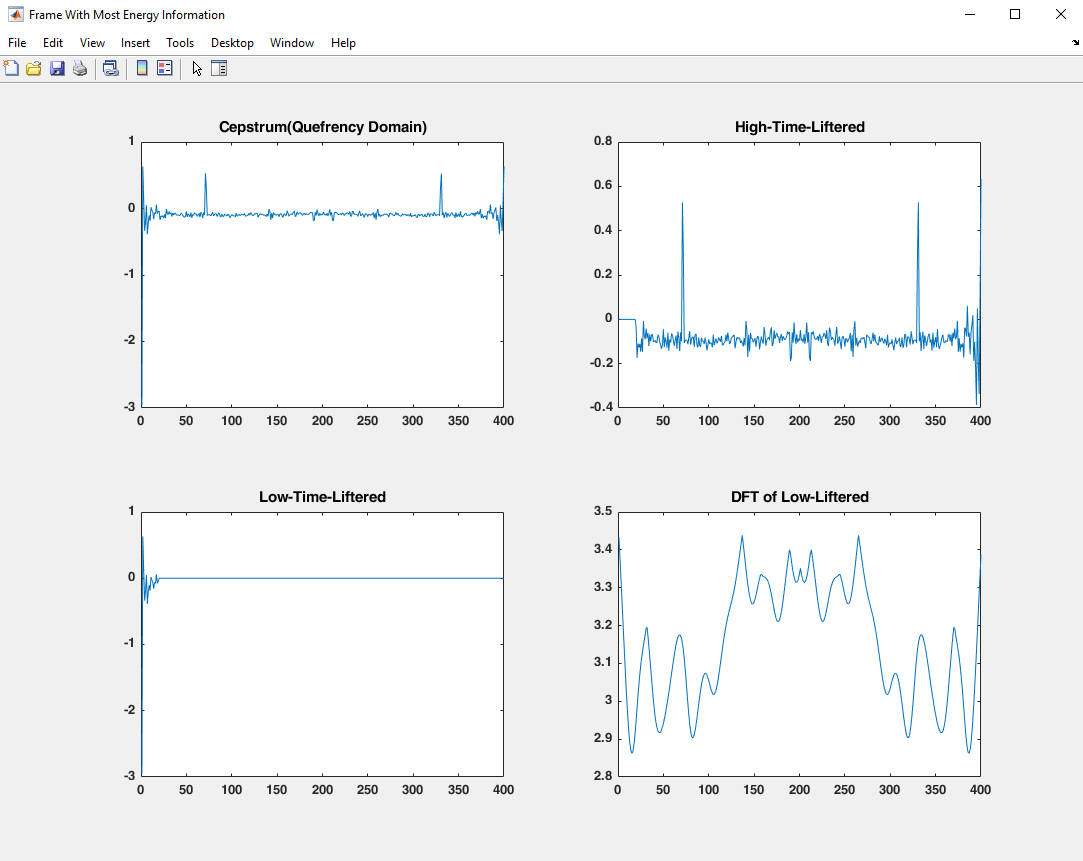
low\_liftered = low\_time\_lifter(quefrency, lifter\_cutoff);

low\_liftered\_dft = abs(log(fft(low\_liftered)));

% ploting cepstrum, high & low liftered , also dft of low time liftered

figure('NumberTitle', 'off', 'Name', 'Frame With Most Energy Information');

subplot(2,2,1);



plot(quefrency);

title('Cepstrum(Quefrency Domain)');

subplot(2,2,2);

plot(high\_liftered);

title('High-Time-Liftered');

subplot(2,2,3);

plot(low\_liftered);

title('Low-Time-Liftered');

subplot(2,2,4);

plot(low\_liftered\_dft);

title('DFT of Low-Liftered');

حال باید فریم های بدون انرژی کافی و همچنین آن هایی که به اندازه کافی برای محاسبه فرکانس گام و فرمنت ها، ماکزیمم نداشتند را حذف کنیم و روی این ها میانگین گرفته و عدد منطقی با خطای کم برای فرکانس گام و فرمنت ها را محاسبه کنیم:

% Going to Calculate Average pitch and formant between frames with

% acceptable energy, so we get a more reliable result.

avg\_pitch\_freq = 0;

avg\_formant1 = 0;

avg\_formant2 = 0;

avg\_formant3 = 0;

avg\_formant4 = 0;

count\_f1 = 0;

count\_f2 = 0;

count\_f3 = 0;

count\_f4 = 0;

count\_pitch = 0;

% neglecting frames with energy lower than half the maximum

for i=1:fn

if energy\_of\_frames(i) > max\_energy/2

if pitch\_freqs(i) > 0

avg\_pitch\_freq = avg\_pitch\_freq + pitch\_freqs(i); %% calculating average of pitch frequencies and ignoring 0's due to size(peak\_index)> 0

count\_pitch = count\_pitch + 1;

end

if formants(1,i) > 0

avg\_formant1 = avg\_formant1 + formants(1,i); %% calculating average of f1 and ignoring 0's due to length(locs)>= formant\_no

count\_f1 = count\_f1 + 1;

end

if formants(2,i) > 0

avg\_formant2 = avg\_formant2 + formants(2,i); %% calculating average of f2 and ignoring 0's due to length(locs)>= formant\_no

count\_f2 = count\_f2 + 1;

end

if formants(3,i) > 0

avg\_formant3 = avg\_formant3 + formants(3,i); %% calculating average of f3 and ignoring 0's due to length(locs)>= formant\_no

count\_f3 = count\_f3 + 1;

end

if formants(4,i) > 0

avg\_formant4 = avg\_formant4 + formants(4,i); %% calculating average of f4 and ignoring 0's due to length(locs)>= formant\_no

count\_f4 = count\_f4 + 1;

end

end

end

avg\_pitch\_freq = avg\_pitch\_freq/count\_pitch; %% final result of pitch frequency

avg\_formant1 = avg\_formant1 / count\_f1; %% final result of formants 1,2,3,4

avg\_formant2 = avg\_formant2 / count\_f2;

avg\_formant3 = avg\_formant3 / count\_f3;

avg\_formant4 = avg\_formant4 / count\_f4;

حال که میانگین فرمنت ها و نیز فرکانس گام را از فریم های با انرژی کافی (فرض کردیم فریم ها با بیش از نصف انرژی ماکیزیمم قابل قبول باشند)، بدست آوردیم، کافی است نتیجه را چاپ کرده و برای داشتن نیم نگاهی از فرکانس گام و فرمنت های فریم های مختلف آن هارا نیز رسم کنیم:

% ploting pitch frequencies

figure('NumberTitle', 'off', 'Name', 'Pitch Frequncies of Frames');

plot(pitch\_freqs);

title('Pitch Frequency');

% ploting formants

figure('NumberTitle', 'off', 'Name', 'Formants of Frames');

hold on

plot(formants(1,:),'r');

plot(formants(2,:),'g');

plot(formants(3,:),'b');

plot(formants(4,:),'c');

title('Formants');

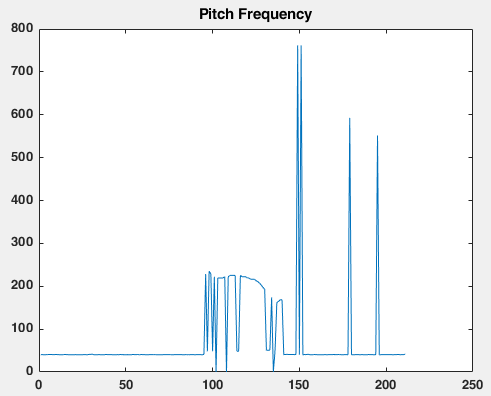
% printing final results (Pitch & Formant)

fprintf("Results: \n Pitch Frequncy %f \n", avg\_pitch\_freq);

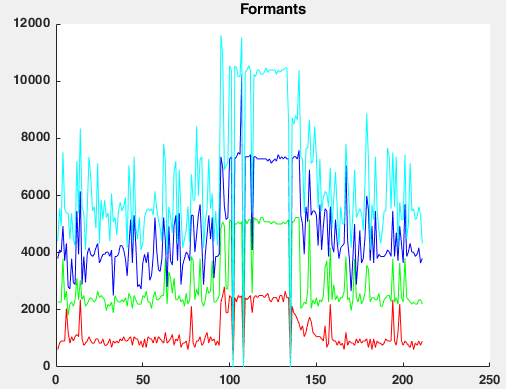
fprintf(" Formants: F1= %f, F2= %f, F3= %f, F4= %f \n", avg\_formant1, avg\_formant2, avg\_formant3, avg\_formant4)

که در زیر نمودار فرکانس های گام و فرمت ها را می بینید:

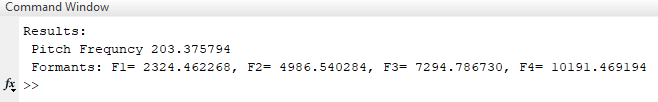
نمودار فرکانس های گام از همه فریم ها:



نمودار فرمنت ها برای فریم های مختلف: (از پایین به بالا فرمنت های 1 و 2 و 3 و 4):



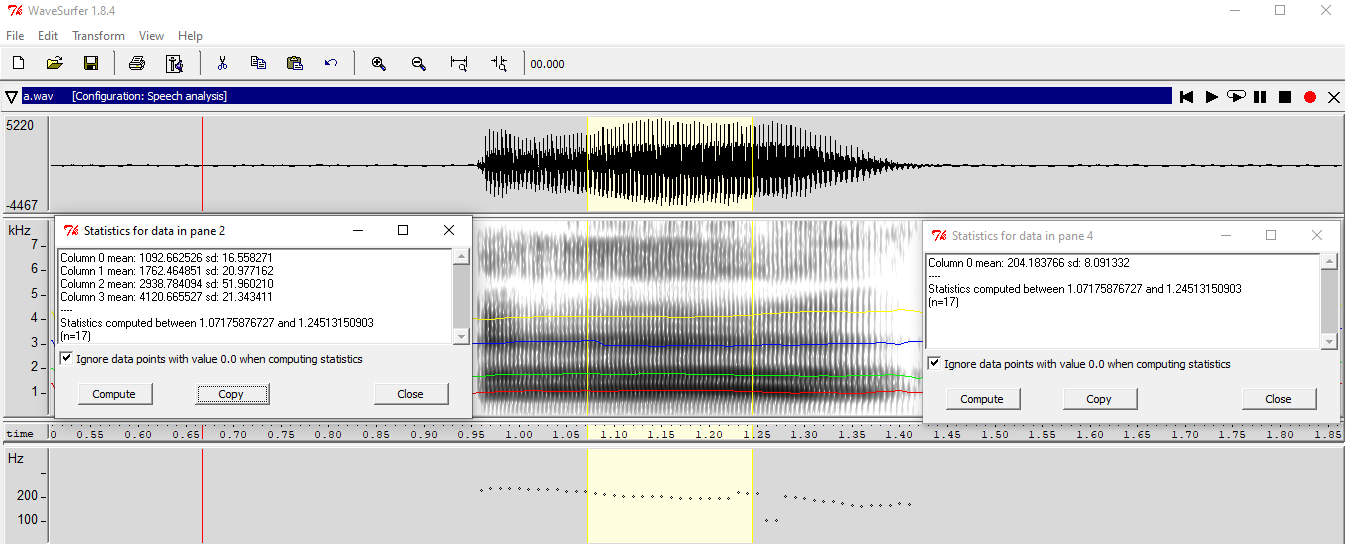
در نهایت نتایج چاپ می شوند:



همانطور که میبینیم مقدار فرکانس گام، برابر 203.375 بدست آمد، وقتی در wave surfer فرکانس گام را در میاوریم، حدود 203 الی 205 است، بنابرین در محاسبه فرکانس گام روش کپستروم کاملا موفق عمل می کند.

اما در محاسبه فرمنت ها چندان موفق عمل نکردیم، فرمنت اول در این روش حدود 2300 بدست آمده، در حالی که در wave surfer حدود 1100 است (حدود 1000 واحد اختلاف)، همچنین فرمنت دوم در اینجا حدود 4900 در آمده در حالیکه در نرم افزار حدود 1800 است (حدود 3000 واحد اختلاف)، این برای فرمنت های سوم وچهارم نیز بدتر می شود و اختلاف برای فرمنت سوم حدود 3100 و برای فرمنت چهارم حدود 5900 واخد اختلاف دارد، بنابرین نمی توان نظر داد که روش کپستروم در این زمینه تا چه حد خوب عمل میکند و به راحتی می توان به دقت بالایی در محاسبه فرمنت رسید. (به لحاظ تئوری به فرمنت ها میرسیم، اما خطای محاسبه و پیاده سازی نسبتا زیاد است)

این مقادیر در wave surfer نیز محاسبه و در زیر آمده است:



سوال دوم)

تبدیل های مختلف برای سهولت یا امکان پذیر کردن تجزیه و تحلیل به کارمی روند، تبدیل Z در بررسی خواص فرکانسی، پایداری و در کل تجزیه و تحلیل سیگنال ها و سیستم های دیجیتالی کاربرد دارد.

با استفاده از تبدیل فوریه می توانیم بگوییم طیف سیگنال یا محتوای فرکانسی آن چگونه است و برای بررسی و مطالعه اطلاعات در حوزه فرکانس مناسب است و ما را از حوزه زمان به حوزه فرکانس می برد، ویژگی های بسیاری را همانطور که در کلاس درس گفته شد، در این حوزه می توان استخراج کرد، اما از طرفی اطلاعاتی را نیز می توان از حوزه زمان بهتر استخراج کرد، که با استفاده از تبدیل فوریه دیگر به این اطلاعات دسترسی نداریم، از طرف دیگر در تبدیل کپستروم این اطلاعات از حوزه زمان وجود دارد و می توانیم از آن ها نیز استفاده کنیم. علاوه بر این ها، تبدیل فوریه کانولوشن که عمل پیچیده ای است را به ضرب تبدیل می کند و تبدیل کپستروم، آن را به جمع تبدیل می کند که بسیار ساده است.

به لحاظ شباهت، تبدیل فوریه پیوسته، حالت خاصی از تبدیل لاپلاس دو طرفه است، تبدیل فوریه گسسته حالت خاصی از تبدیل Z است، بنابرین از تبدیل Z می توان بسیار از خواص موجود در حوزه فرکانس را بدست آورد. تفاوت Z با فوریه گسسته این است که در Z با عدد مختلط سر و کار داریم ولی در فوریه با عدد موهومی.

ولی در عین حال هر دو در حوزه فرکانس اند، تبدیل کپستروم نیز شباهت اش به تبدیل فوریه در این است که در دل خود باید تبدیل فوریه گرفت و کانولوشن را به عمل ساده تری تبدیل می کند، تفاوت آن این است که تبدیل کپستروم در حوزه زمان است ولی فوریه در حوزه فرکانس فعالیت می کند.

پایان