به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

پردازش سیگنال‌های زمان-گسسته

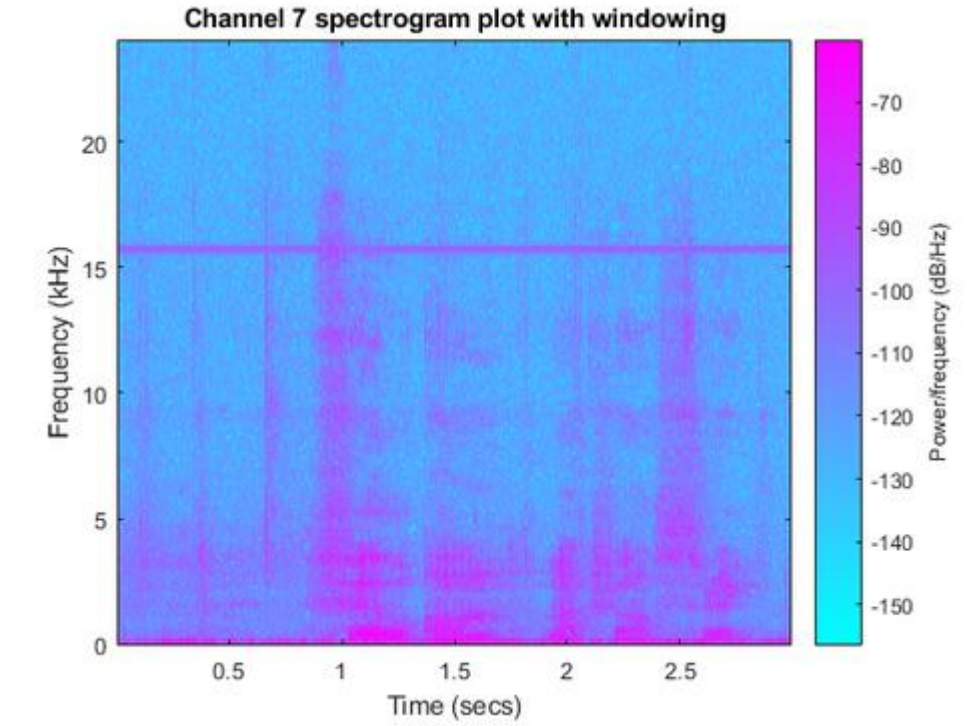
تمرین سری **3**

نام و نام خانوادگی	حمیدرضا علی اکبری خویی
شماره دانشجویی	۸۱۰۱۹۶۵۱۴
تاریخ ارسال گزارش	99/05/06

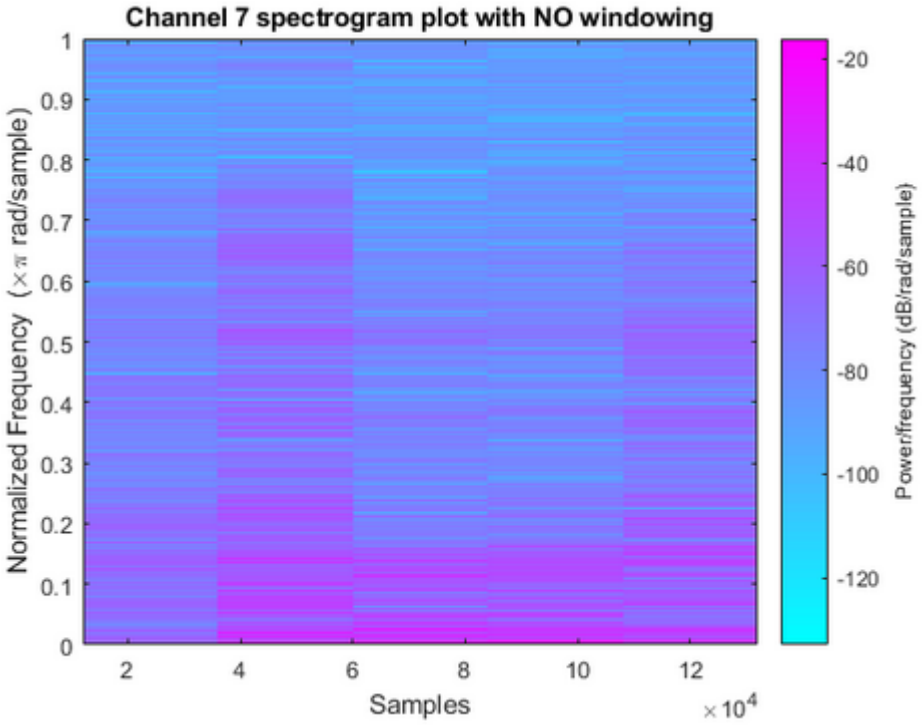
فهرست گزارش سوالات

سوال 1

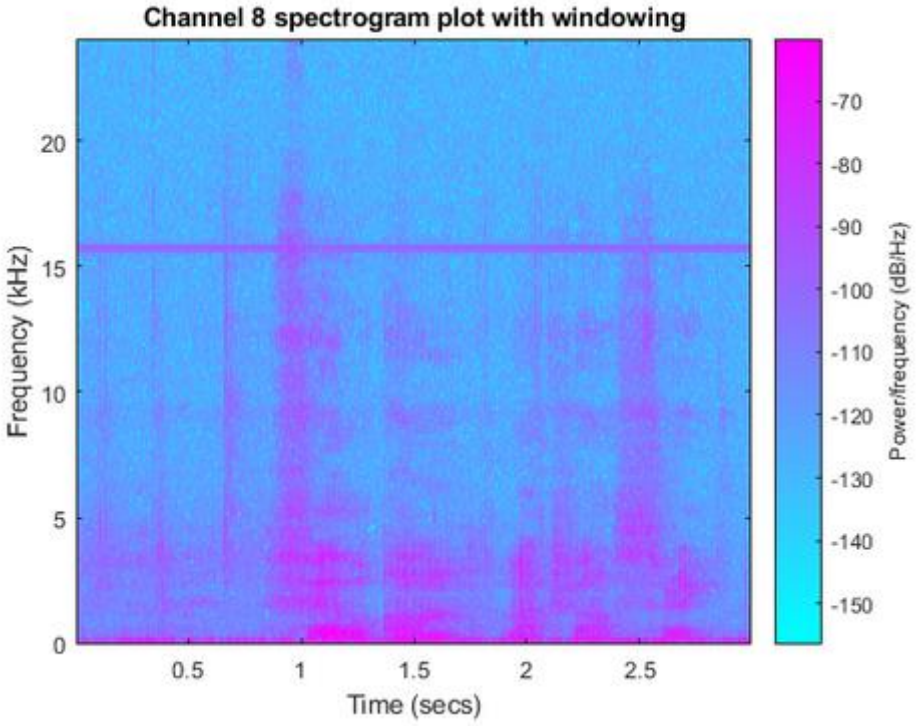
در این قسمت برای رسم اسپکتوگرام ، به دو روش پنجره گذاری و بدون پنجره گذاری استفاده شده است، که در روش پنجره گذاری باید مقدار روی هم رفتگی در نظر گرفته شود که در این روش باعث میشود مقادیر یکم دقیق تر بر حسب هر ثانیه رسم شوند البته اسپکتو گرام اطلاعات خیلی بیشتری نسبت به تبدیل فوریه در اختیار ما میگذارد که باعث میشود بفهمیید در هر بازه زمانی سیگنال دارای چه فرکانسی بوده است که از این روش ها میتوان برای تشخیص صصدا و حتی در مواردی تشخیص نُت های موسیقی استفاده کرد.شکل های اسپکتوگرام با پنجره گذاری و بدون آن به شرح زیر است:



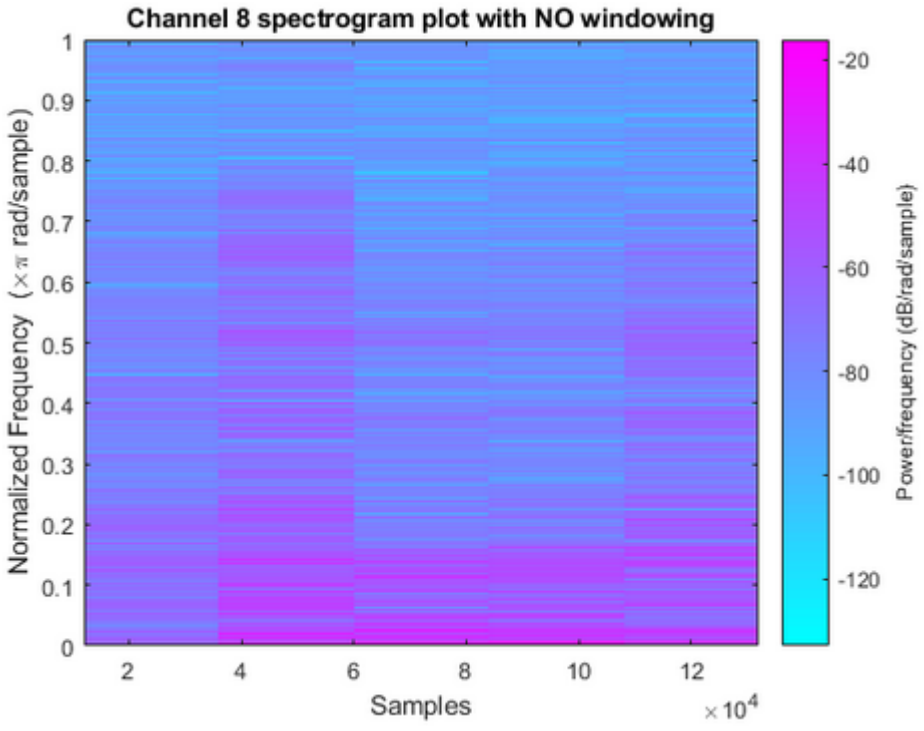
1Figure



2Figure



3Figure



4Figure

البته که اسپکتوگرام برای دو کانال 7 و 8 خیلی شبیه هم میباشد چون که تنها باید اختلاف فاز در آن دو مشاهده شود و به خاطر همین زیاد نمیتوان فرق دو کانال را تشخیص داد ، و این که در روش پنجره گذاری اسپکتوگرام بر حسب زمان_فرکانس کشیده میشود ولی در روش بدون پنجره گذاری اسپکتوگرام به روش نمونه_فرکانس کشیده میشود که دقیقا مثلا قبلی است که چون اگر شماره نمونه را در زمان نمونه برداری ضرب کنیم همان زمان اصلی سیگنال زمان پیوسته را بدست می آوریم!

سوال ۲

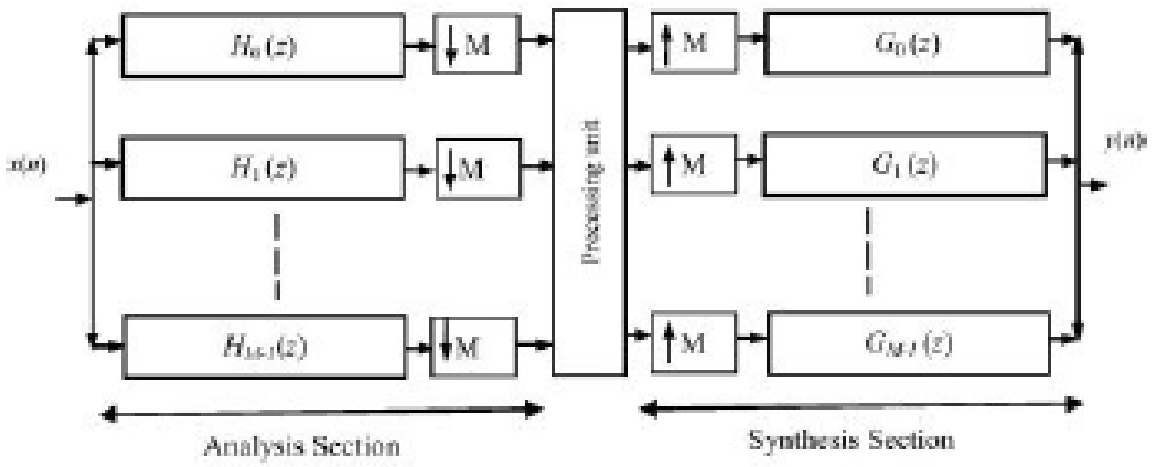


Figure 5

مطابق شکل 5 باید ابتدا حوزه فرکانس را به صورت مساوی به 4 قسمت در هردو کانال 7 و 8 تقسیم کرد و بعد با بلوک های Downsampling میزان حد اقل دیتای مورد پردازش را کم کرد، با این کار میتوان به صورت پردازش موازی که بر رو ی سیگنال انجام میشود میزان زمان برای کل محاسبات لازم را کمتر کرد و بعد از انجام محاسبات میتوان با بلوک های Upsampling کل سیگنال را به صورت خوب بازیابی کرد.البته برای تقسیم به صورت مساوی حوزه فرکانس باید به صورت ایده آل فیلتر هارا طراحی کرد که در این جا با استفاده از FilterDesigner خود متلب فیلتر های FIR با متد Equiripple را طراحی کرد.که میزان Fpass-Fstop را کمترین مقدار گرفت که شبیه خود فیلتر ایده آل عمل کرد!پاسخ فرکانسی فیلتر ها به صورت زیر میباشد:

فیلتر پایین گذر:

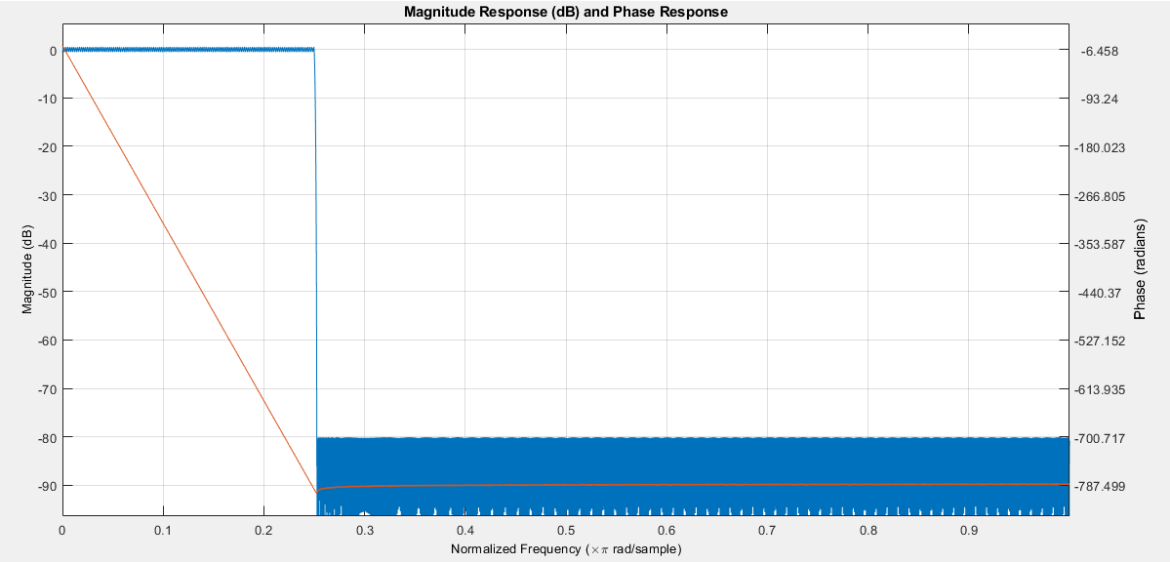


Figure 6

فیلتر های میان گذر:

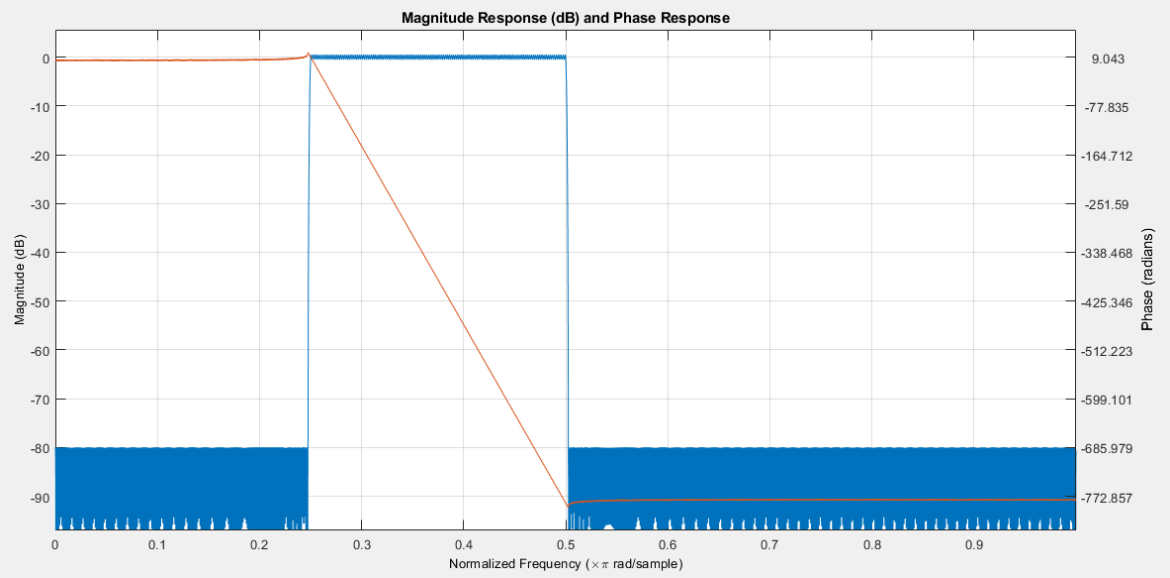


Figure 7

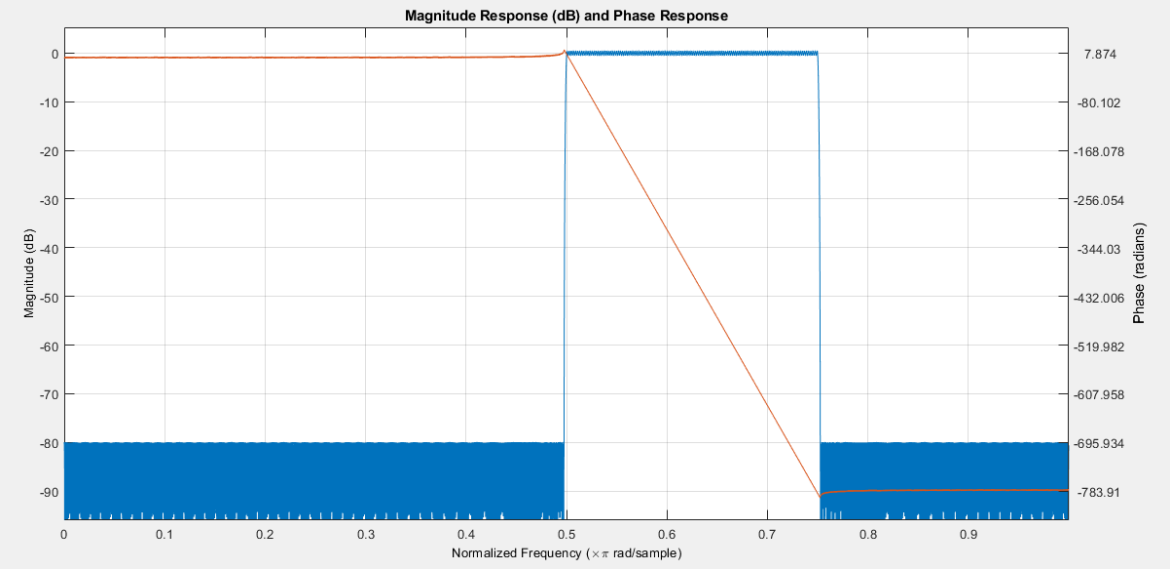
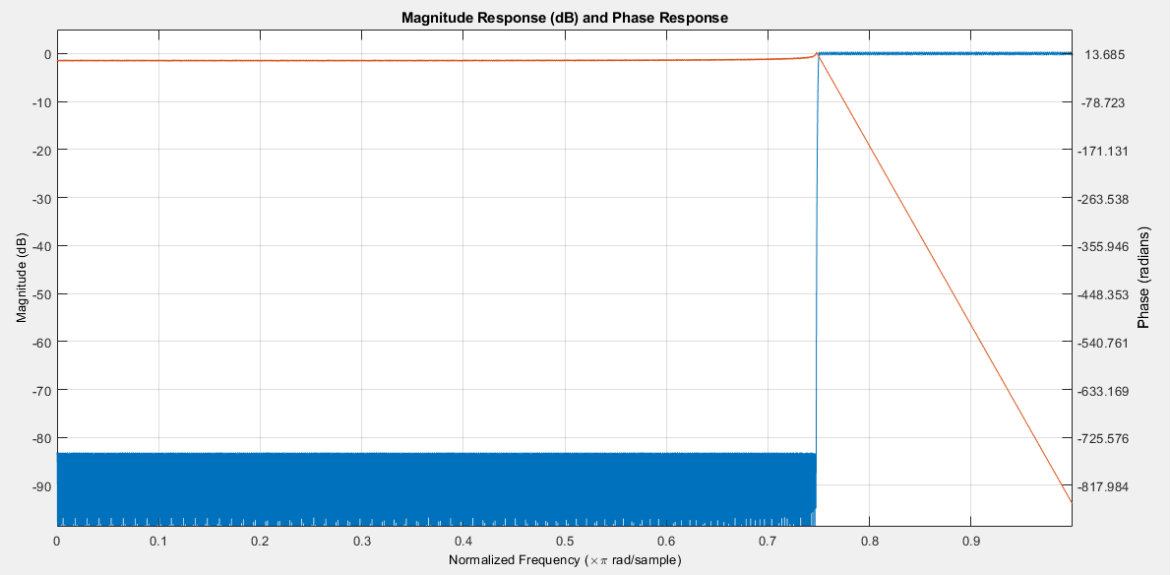


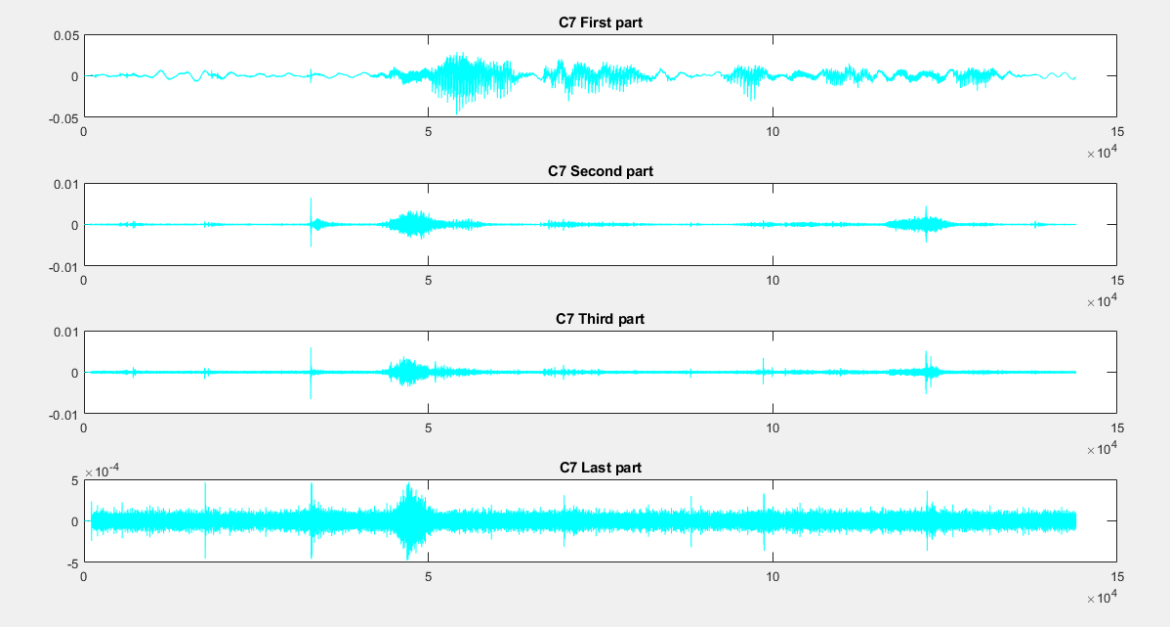
Figure 8

فیلتر بالا گذر:



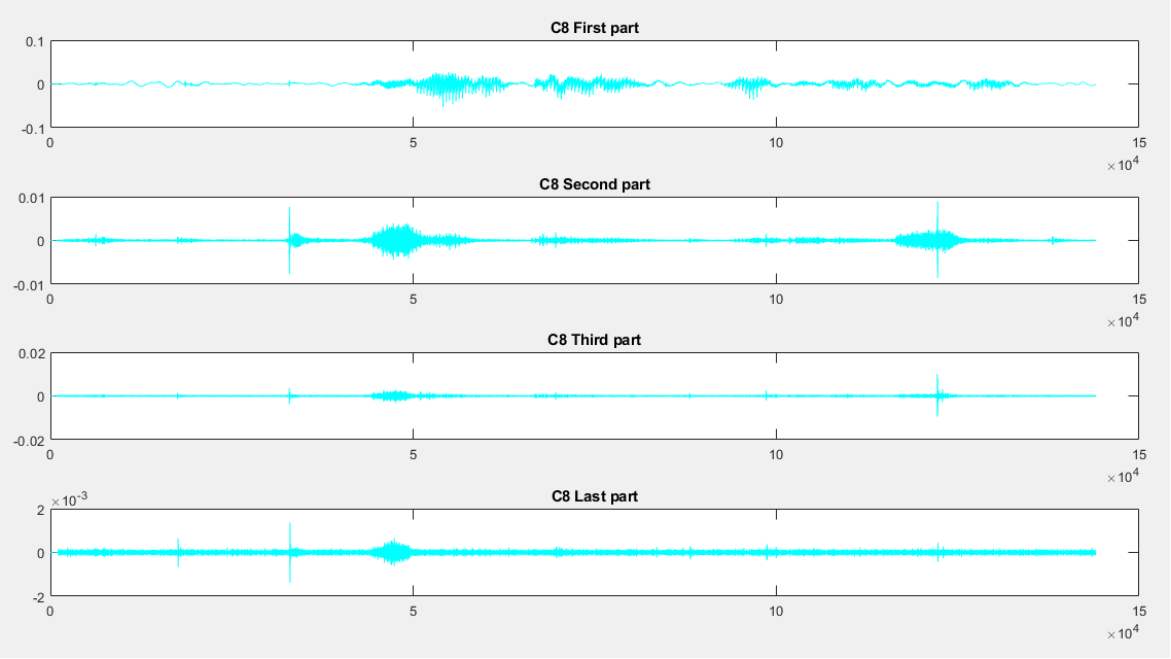
9Figure

خروجی های کانال هفت:



10Figure

خروجی های کانال هشت:



11Figure

مشاهده میشود که با توجه به این که محدوده فرکانسی صدای انسان در محدوده فرکانس های بایین گذر می باشد، بیشتر انرژی سیگنال مربوطه به وجود آمده در ۸ کانال در کانال های اول که مربوط به خروجی های فیلتر بایین گذر میباشدُ بیشتر است!

سوال۳

```
All_data = zeros(144001,8);
All_data(:,1) = data_1;
All_data(:,2) = data_2;
All_data(:,3) = data_3;
All_data(:,4) = data_4;
All_data(:,5) = data_5;
All_data(:,6) = data_6;
All_data(:,7) = data_7;
All_data(:,8) = data_8;

All_Chunked_channels = zeros(256,562,8);
for  data= 1:8
    for chunk= 1:562
        start_index = 256*chunk-255;
        end_index = 256*chunk;
        All_Chunked_channels(:,chunk,data) = All_data(start_index:end_index,data);
    end
end
```

Figure 12

با توجه شکل ۱۲ میتوان دید که چطور ۸ کانال به chunk های مربوطه تقسیم شده اند.البته مقدار کمی دیتا در کل بازه بود که دور ریخته شده اند چون مقدار کل دیتا ها بر ۲۵۶ تقسیم بذیر نبود!

سوال۴

```
correlation_same_subchannels = zeros(562,4);
for sc = 1:4
    for chunk = 1:562
        [~,correlation_same_subchannels(chunk,sc)] = max(abs(xcorr(All_Chunked_channels(:,chunk,sc),All_Chunked_channels(:,chunk,sc+4))));
    end
end
figure
for i = 1:4
 subplot(4,1,i),hist(correlation_same_subchannels(:,i)), title(['subchannel NO.',i,' lagindex distrbution ']),xlim([0  600]);
end
```

13Figure

با توجه به شکل 14 چانک های متناطر هر زیر کانال ه را با هم کورلیشن گرفته ایم تا میزان شباهت به هم را پیدا کنیم، که د حقیقت با توجه به ماتریس تعریف شده آرایه های 0_3 متناظراً با آرایه های 4_7 با هم کورلیشن گرفته میشوند که با توجه به کد بالا این امر بدیهیست!

سوال۵

در این قسم هیستوگرام مربوط به lagindexهارا پیدا میکنیم که البته باید توجه داشت که خروجی کورلیشن متلب دو عدد است که ما با قسمت lagindexکار داریم:

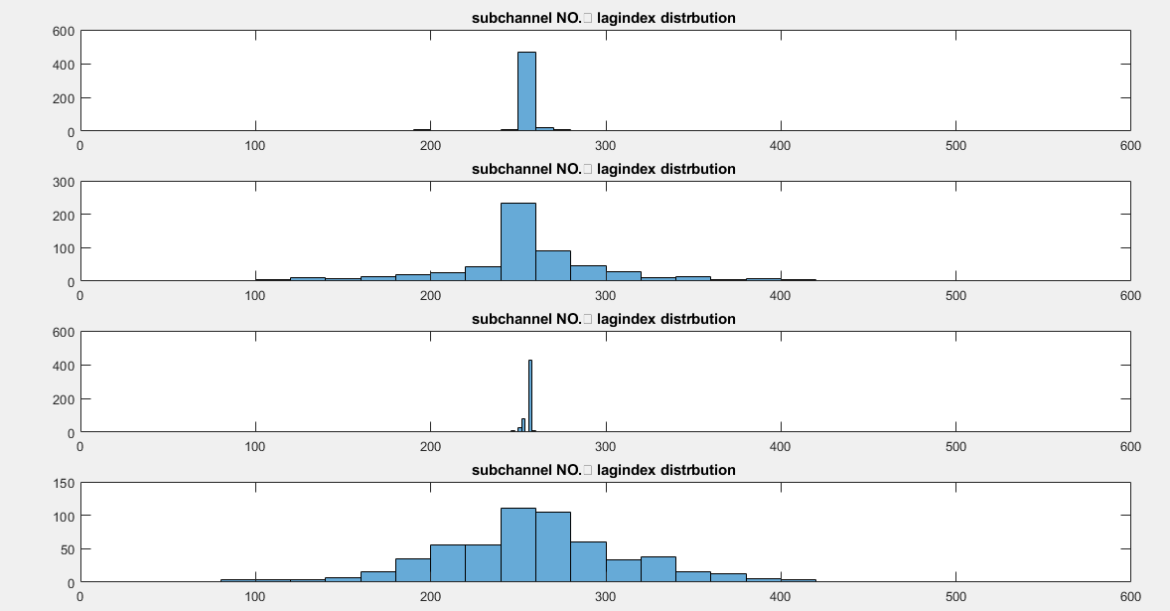


Figure 14

در شکل 14 میتوان دید که عموما مشخصه ها حالت گوسی دارند ، و اینکه برای محاسبه مقدار واریانس باید توجه کرد که واریانس طوری است که با فاصله 3 برابر سیگما از میانگین 99 درصد دیتا ها قرار گرفته اند ، توجه شود محاسبات بر روی کاغذ تقریبی انجام شده است ،پس:

برای اولی:

$$\sqrt{500} = 22.36$$

برای دومی:

$$\sqrt{2500} = 50$$

برای سومی:

مقدار خیلی کوچک میباشد که با توجه به تقریب ها به نظر در حد زیر 10 باشد :

$$\sqrt{20} = 4.5$$

برای چهارمی:

$$\sqrt{2600} = 50.99$$

که برای اطمینان با خود متلب نیز نمودار گوسی را بر هر یک از هیستوگرام ها فیت کرده ام که نتیجه این شد:

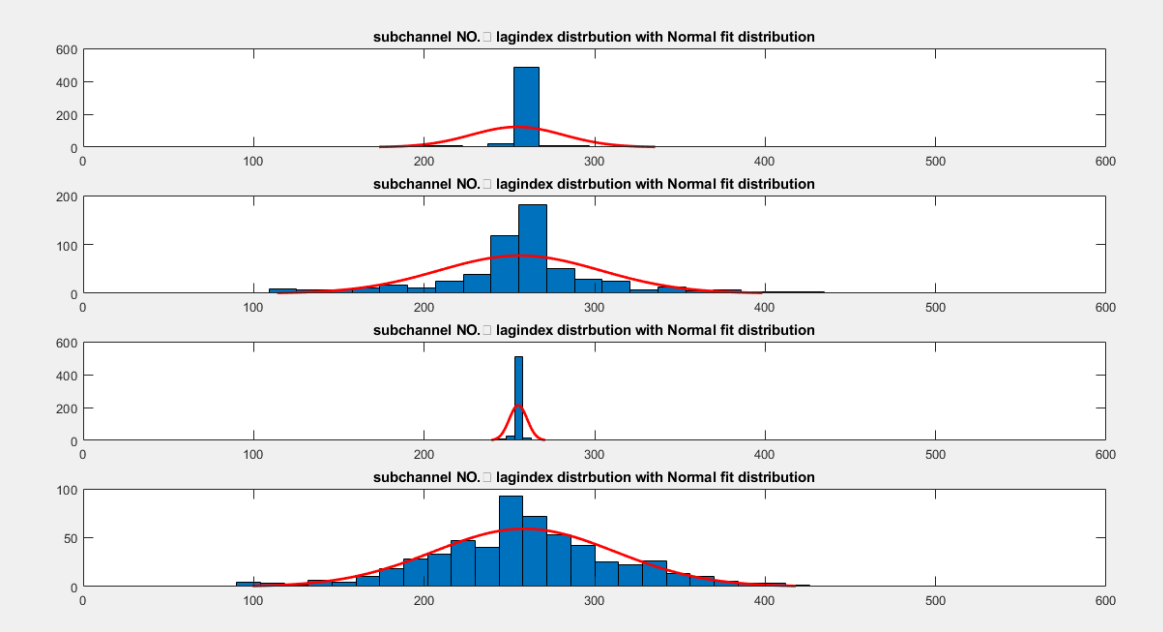


Figure 15

که با استفاده از کد متلب نیز مقادیر سیگما و میانگین را پیدا کرده ام که نتایج به صورت زیر است:

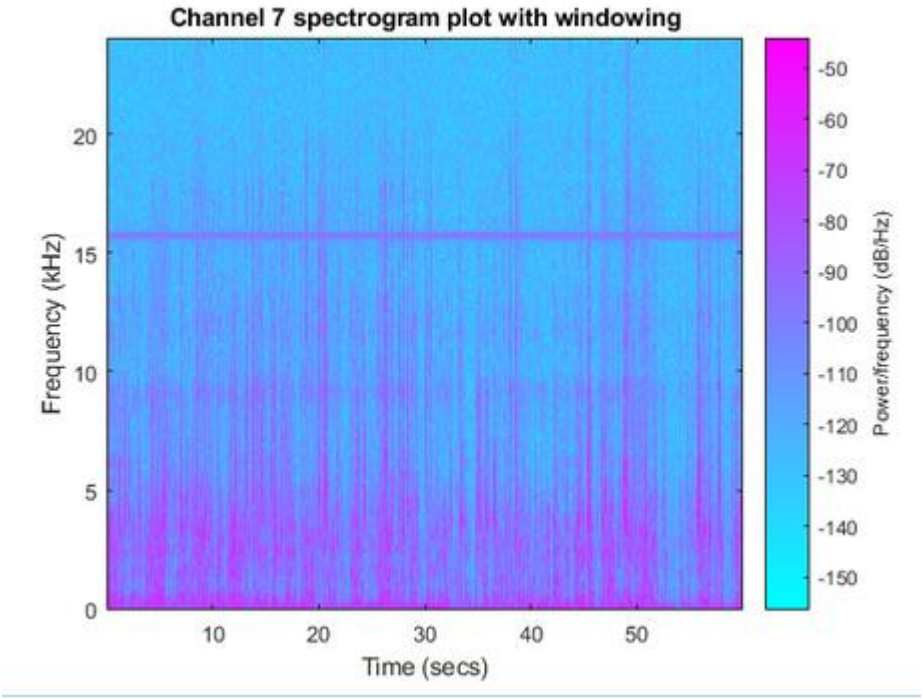
```
first_channel =
  NormalDistribution

  Normal distribution
       mu = 254.532    [252.296, 256.768]
    sigma = 26.9818    [25.4913, 28.6589]
second_channel =
  NormalDistribution

  Normal distribution
       mu = 256.089    [252.16, 260.018]
    sigma = 47.4203    [44.8807, 50.3678]
third_channel =
  NormalDistribution

  Normal distribution
       mu = 255.171    [254.736, 255.606]
    sigma = 5.24678    [4.95693, 5.5729]
last_channel =
  NormalDistribution

  Normal distribution
       mu = 258.669    [254.27, 263.068]
    sigma = 53.0895    [50.1567, 56.3894]
```

16Figure

با توجه به محاسبات انجام شده میتوان دید که نتایج محاسبات نزدیک به مقادیر اصلی می باشد!

سوال۶

با توجه به این که همه کارای سوال های 1-5 را برای وردی های دیگر نیز باید انجام دهیم پس فقط مقادیر محاسبه شده را با خروجی های گرفته شده میاورم که در هر قسمت در صورت نیاز توصیحات مربوط را خواهم داد:



17Figure

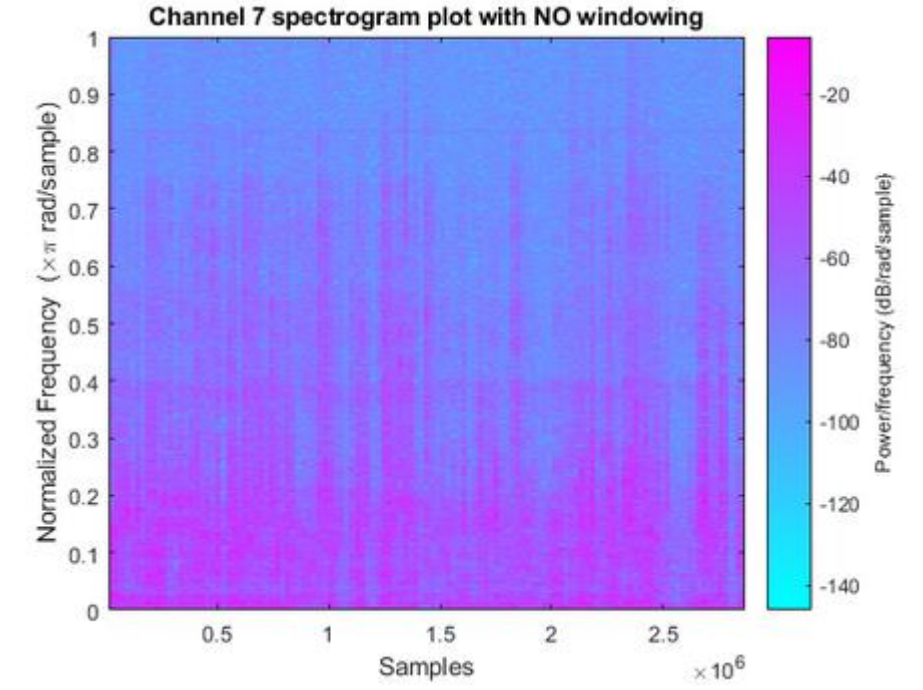
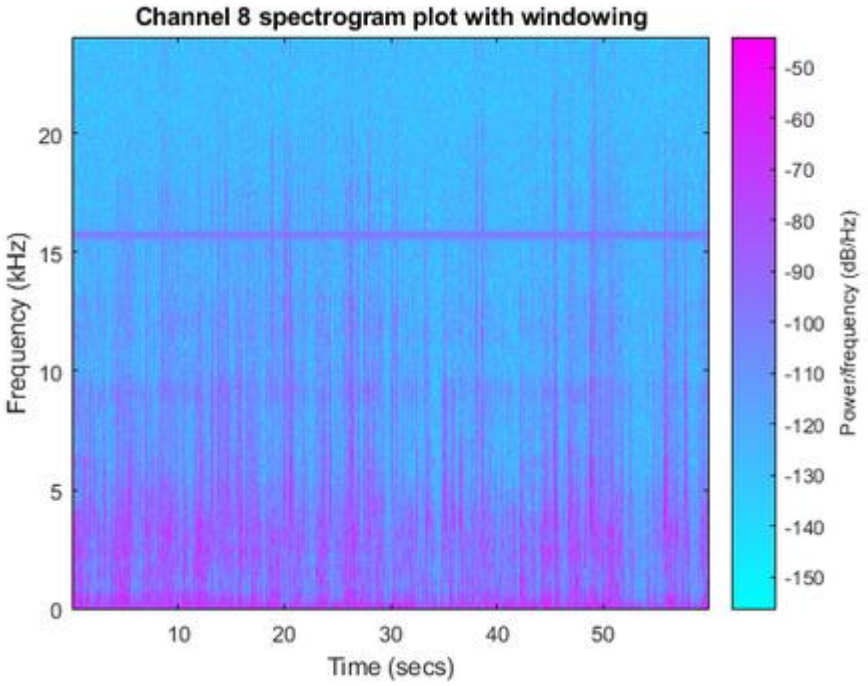
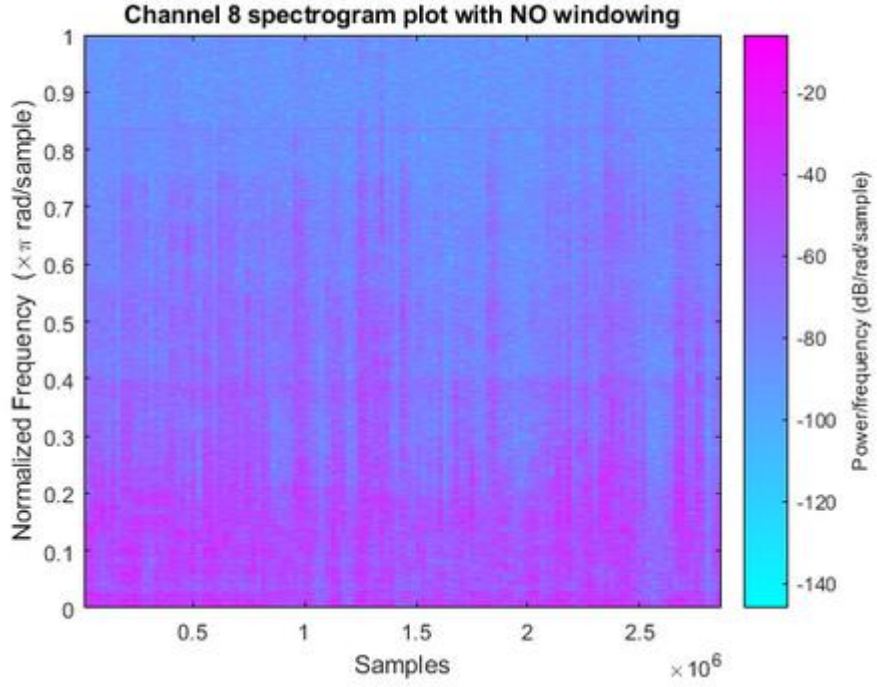


Figure 18



19Figure



20Figure

با توجه به این که مقدار زمانی و طول سیگنال مربوط به ورودی های جدید زیاد است در حد 1 دقیقه پس حال گسسته طور و تیکه تیکه شدن را در قسمت بدون پنجره گذاری برای اسپکتو گرام شاهد نیستیم!

خروجی های بعد از فیلتر مربوط به کانال 7:

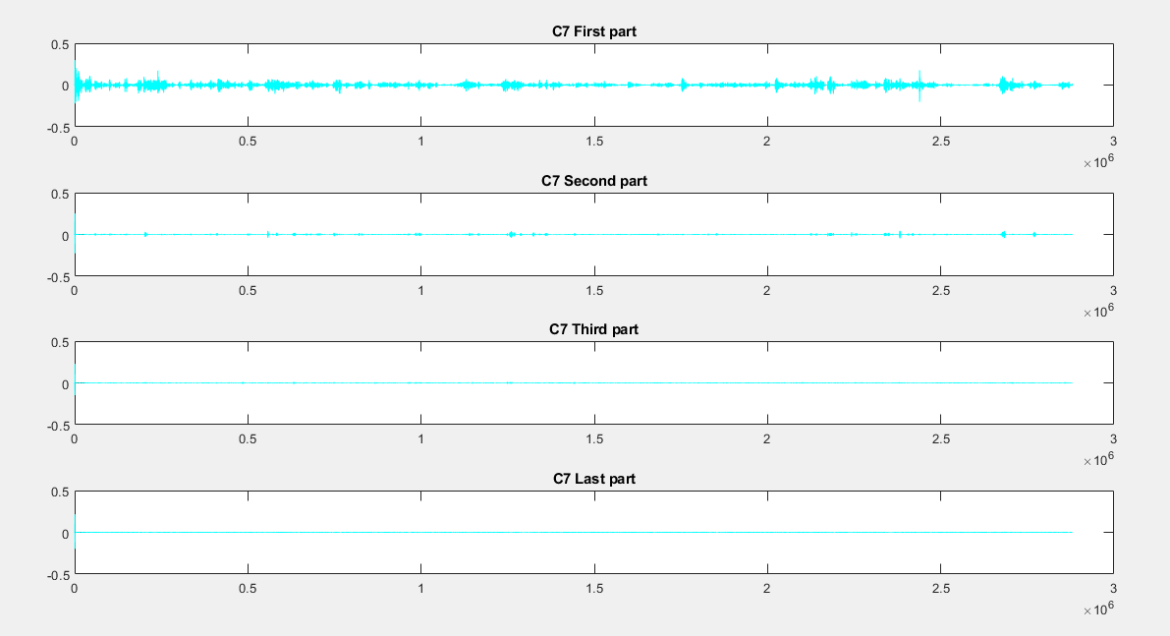


Figure 21

خروجی های بعر از فیلتر مربوط به کانال 8:

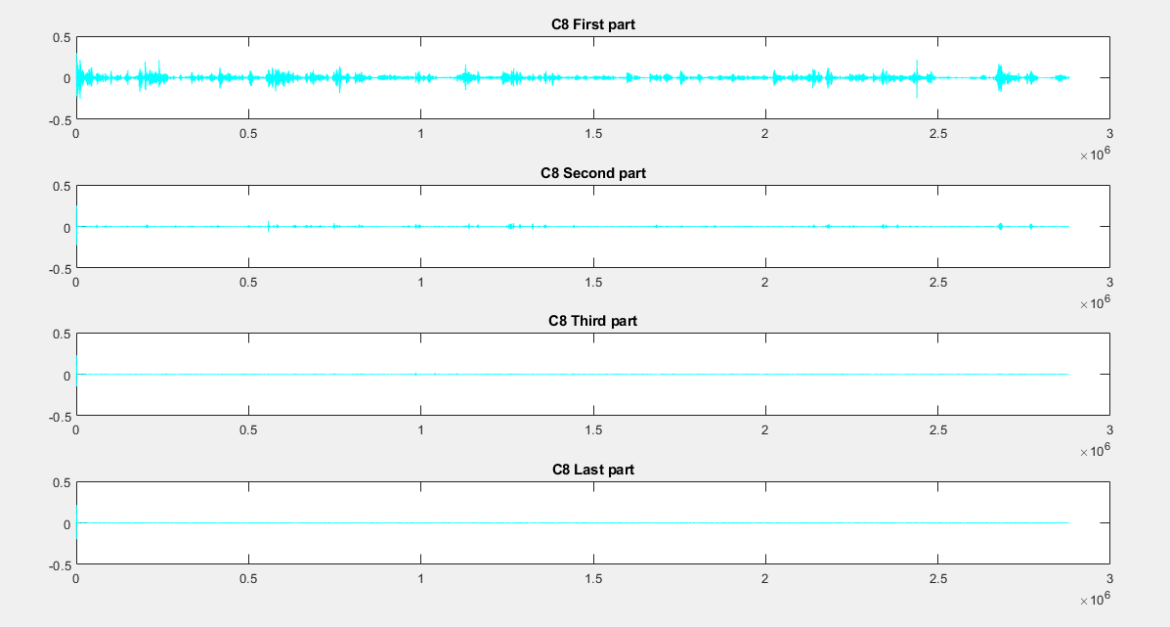
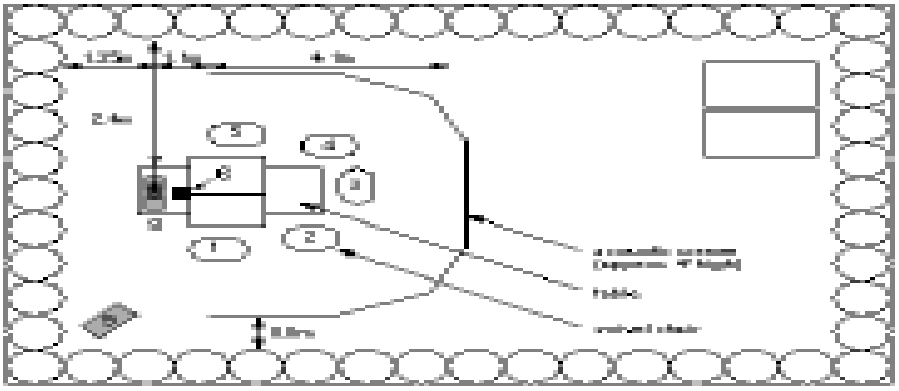


Figure 22

که با توجه به شکل قرار گیری گیرنده های صدا در صورت پروژه و با توجه به این که مقدار فرکانسی صدای انسان کم است انرژی در زیر کانال های 1 زیادتر است:



23Figure

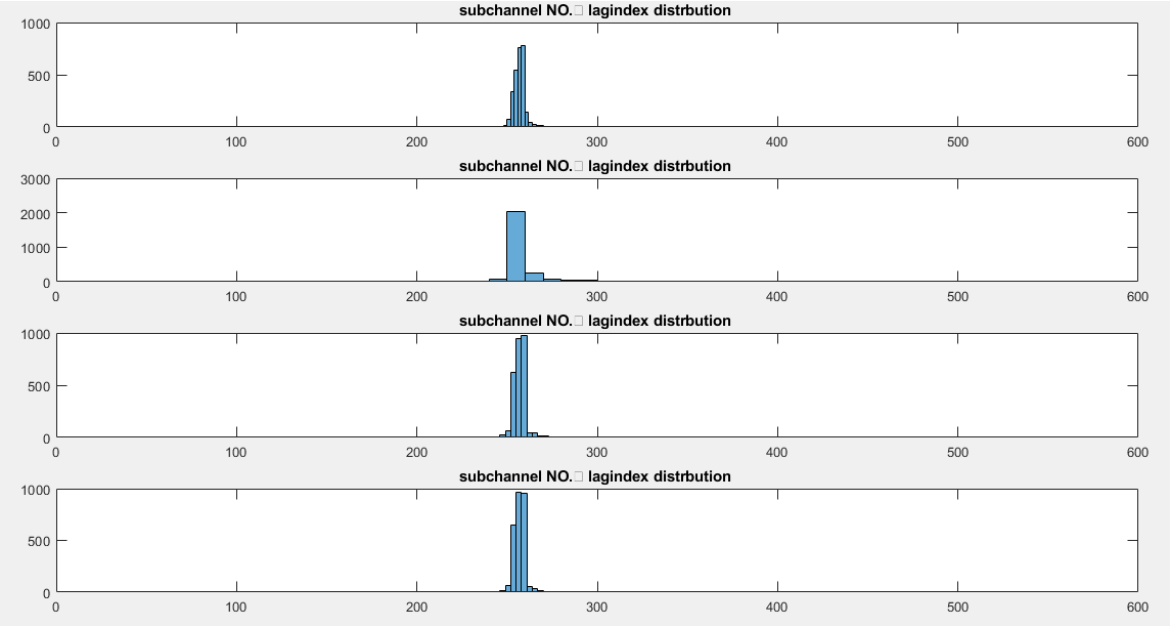
شکل23 قرار گیری گیرنده ها را مشاهده میتوان کرد که دلیل وجود تشابه زیاد در خروجی های مربوط به دوکانال 7 و 8 را توجیح میکند!

```
%All_data = zeros(1220004,8);
All_data(:,1) = decimate(data_1,4);
All_data(:,2) = decimate(data_2,4);
All_data(:,3) = decimate(data_3,4);
All_data(:,4) = decimate(data_4,4);
All_data(:,5) = decimate(data_5,4);
All_data(:,6) = decimate(data_6,4);
All_data(:,7) = decimate(data_7,4);
All_data(:,8) = decimate(data_8,4);

All_Chunked_channels = zeros(256,2812,8);
for  data= 1:8
    for chunk= 1:2812
        start_index = 256*chunk-255;
        end_index = 256*chunk;
        All_Chunked_channels(:,chunk,data) = All_data(start_index:end_index,data);
    end
end
```

24Figure

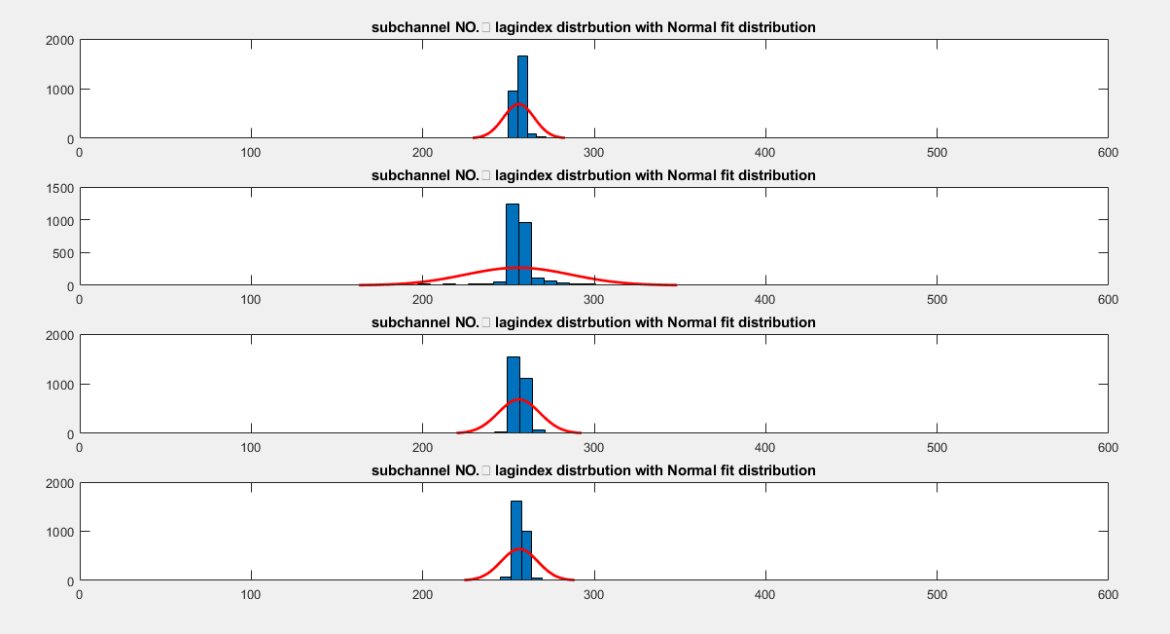
با توجه به شکل 24 میتوان دید که اول کل دیتا هارا که بعد از اعمال فیلتر دریافت کردیم را Downsample میکنیم و باعث میشود با توجه اثبات های قضیه پلی فاز و محاسبات موازی بر روی سیگنال به شدت میزان زمان پردازش لازم را کمتر کند و به علاوه منطق طراحی خود بانک_فیلتر به این گونه میباشد که در شکل 24 میباشد!

هیستوگرام های مربوط به خروجی:



25Figure

خروجی های منحنی نرمال فیت شده بر روی هیستو گرام ها:



26Figure

که برای هر منحنی نرمال فیت شده بر روی هیستوگرام ها مقادیر میانگین و سیگما به صورت شکل 27 بدست می آید:

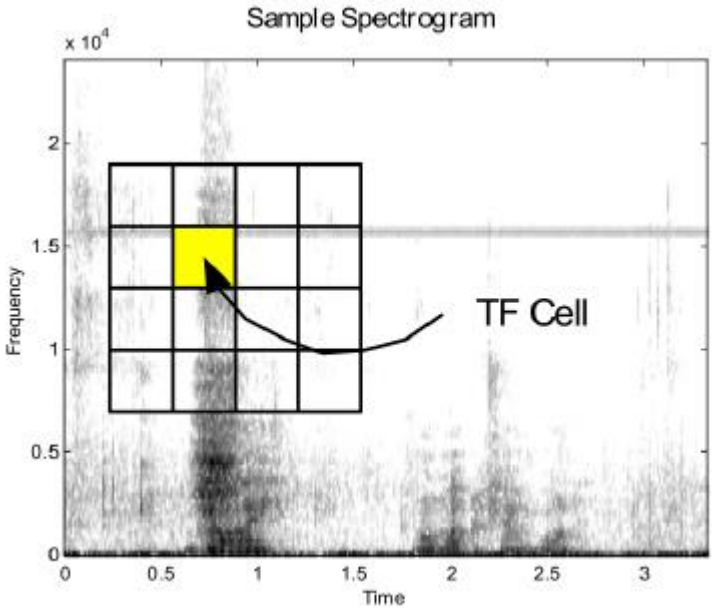
```
first_channel =
  NormalDistribution

  Normal distribution
       mu = 255.991   [255.659, 256.323]
    sigma = 8.97478   [8.74621, 9.21571]
second_channel =
  NormalDistribution

  Normal distribution
       mu = 255.627   [254.483, 256.771]
    sigma = 30.9389   [30.1509, 31.7695]
third_channel =
  NormalDistribution

  Normal distribution
       mu = 256.127   [255.678, 256.575]
    sigma = 12.1331   [11.8241, 12.4588]
last_channel =
  NormalDistribution

  Normal distribution
       mu = 256.365   [255.97, 256.761]
    sigma = 10.6976   [10.4252, 10.9848]
```

Figure 27

باید توجه داشت که مقادیر کورلیشت حساب شده هم در این قسم هم در سوالات قبلی که بر روی هیستوگرام کشیده شده اند میبایست از -256 تا 256 میبودند که در حقیقت شیفت انجام شده است و الان میزان میانگین های حدود 256 میباشد که منطقی میباشد (چرا؟) ، چون که با توجه شکل 23 و با توجه به فایل solo میتوان از اسم فایل فهمید که فرد اصلی که در حال صحبت میباشد در جایگاه سوم قرار دارد که میتوان فهمید با توجه به شکل 23 ، فاصله گوینده 3 از هر دو گیرنده کانال 7 و 8 به یک فاصله میباشد که وقتی که در هر فرکانسی کورلیشن های مربوط به chunk ها که محاسبه میشوند اینگونه است که با توجه به مفهوم کورلیشت بین دو سیگنال ، اگر دو سیگنال میزان اخلاف زمانی یا به صورت دیگر اختلاف فازی نسبت به همدیگر داشته باشند در جواب کورلیشن در همان اختلاف فاز(یا زمانی) میزاد اندازه کورلیشن حداکثر میشود ، با فرض بر این که گوینده سوم که فاصله یکسانی از هر دو کانال 7 و8 (از هردو گیرنده)دارد پس باید با توجه به این که میزان غالب بودن صدای گوینده 3 با توجه به فایلsolo بیشتر است و به عبارتی دیگر دارای انرژی بیشتری است ،پس باید در جواب کورلیشن هایی که گرفته میشود میزان قابل توجهی از جواب های هیستو گرام حول صفر که همان تقریبا میانگین است قرار گیرند چون که گیرنده سوم غالب هست . این توضیح وجود منحنی نرمال برای این گونه توضیع کورلیشن را که میانگین حول صفر دارد را توجیح میکند.

سوال۷

برای محاسبه وزن های مربوط به هر تیکه از اسپکتوگرام داریم:



28Figure

با توجه به شکل 28 ما باید برای هر قسمت از اسپکتوگرام که تقسیم کرده ایم وزن اختصاص بدهیم که در کل باید با توجه به این که حوزه فرکان را 4 قسمت کرده ایم و حوزه زمان را 11250 قسمت پس در کل باید 45000 تا قسمت که اسپکتوگرام را تقسیم کرده ایم وزن اختصاص بدهیم که در نهایت بتوانیم سیگنال خروجی را با توجه به آن باز سازی کنیم!

باید توجه داشت که با توضیحات انتهای سوال 6 این نتیجه گیری درست و بدیهیست که باید مقدار اطلاعات موجود در حول 0 هر یک از کورلیشن های موجود را بگیریم چرا که گوینده اصلی در موقعیت سوم قرار دارد و صدایش بدون اختلاف به شنونده میرسد که در این مرحله با توجه به این میزان سیگما برای محاسبه وزن های مربوط به هرکدام را باید طوری تعیین کرد که اطلاعات موجود در حول صفر بازیابی شوند و بقیه را صفر کند و با توجه به این میزان سیگما را باید در حول 0 در نظر بگیری ولی با توجه به این که باز هم به صورت دقیق دیتای دریافتی با اخلاف زمانی(فازی) به گیرنده های 7 و 8 نمیرسد من میزان سیگمال در حول 0.5 گرفته ام که البته بدیهیست گوینده های دیگر را نیز تحت تاثیر قرار خواهد داد و کاملا به صفر نخواد رساند !

سیگما های گرفته شده برای هر 4 کانال کورلیشن که محاسبه شده است مطابق شکل 29 میباشد:

```
sigma = [0.03;.5;.5;.5];
```

29Figure

که مطابق توضیحات میباشد ، ولی چون در کانال اول میزان انرژِ از دیگر کانال ها خیلی بیشتر است با توجه به توضیحات قبل، میزان سیگمال را بیشتر نزدیک صفر در نظر گرفته ام که همان 0.03 میباشد.

کد متلب مربوط به محاسبه ضرایب و ضرب آنها در هر وزن:

```
W_ch_n = zeros(4,2812);
sigma = [0.03;.5;.5;.5];
mu = [256;256;256;256];
for ch = 1:4
    for n = 1:2812
        temp = (correlation_same_subchannels(n,ch)-mu(ch)).^2;
        W_ch_n(ch,n) = (exp(-temp./(2.*(sigma(ch)).^2)));
    end
end
```

Figure 30

ضرایب خروجی مربوط به هر 4 کانال با استفاده از bubble plot:

31Figure

برای کانال 1 رنگ قرمز

برای کانال دوم رنگ آبی

برای کانال سوم رنگ سبز

برای کانال چهارم رنگ زرد اعمال شده است

که البته با توجه به این که برای کانال اول سیگما 0.03 گرفته شده است در بین ضرایبی که حدود اندازه 0.13 را دارند، ضرایب کانال 1 قرار ندارند و در حقیقت کانال 1 ضرایب را یا یک میکند یا صفر و تقریبا بقیه کانال ها همینطوری هستند، البته با توجه به توزیع میشود فهمید که توزیع ها شبیه هم هستند و آن نقطه هایی که ضرایب حدود 1 دارند غالبا به این برمیگردد که آن کسی که در جایگاه سوم حرف میزند و آن قسمت هاییکه ضرایب صفر دارد و یا حدود0.1 به این برمیگردد که غالبا بقیه گوینده ها حرف میزنند.

سوال8

Question8

```
for i=1:4
    for j=1:2812
    weighted_chunkso(:,j,i)=All_Chunked_channels(:,j,i).*W_ch_n(i,j);
    weighted_chunkso(:,j,i+4)=All_Chunked_channels(:,j,i+4).*W_ch_n(i,j);
    end
end
routput = zeros(2880016,1);
loutput = zeros(2880016,1);
interpolated_weighted_chunkr = zeros(720004,8)
 for j=1:8
   for i=1:2812
    interpolated_weighted_chunkr(256*(i-1)+1:256*i,j)=weighted_chunkso(:,i,j);
   end
 end
 for i=1:8
     interpolated_weighted_chunksr(:,i)=interp(interpolated_weighted_chunkr(:,i),4);
 end
 for i=1:4
     routput(:,1)=routput(:,1)+interpolated_weighted_chunksr(:,i);
 end
  for i=1:4
     loutput(:,1)=loutput(:,1)+interpolated_weighted_chunksr(:,i);
  end
  audiowrite('Rightoutput.wav',routput,Fs);
  audiowrite('Leftoutput.wav',loutput,Fs);
```

Figure 32

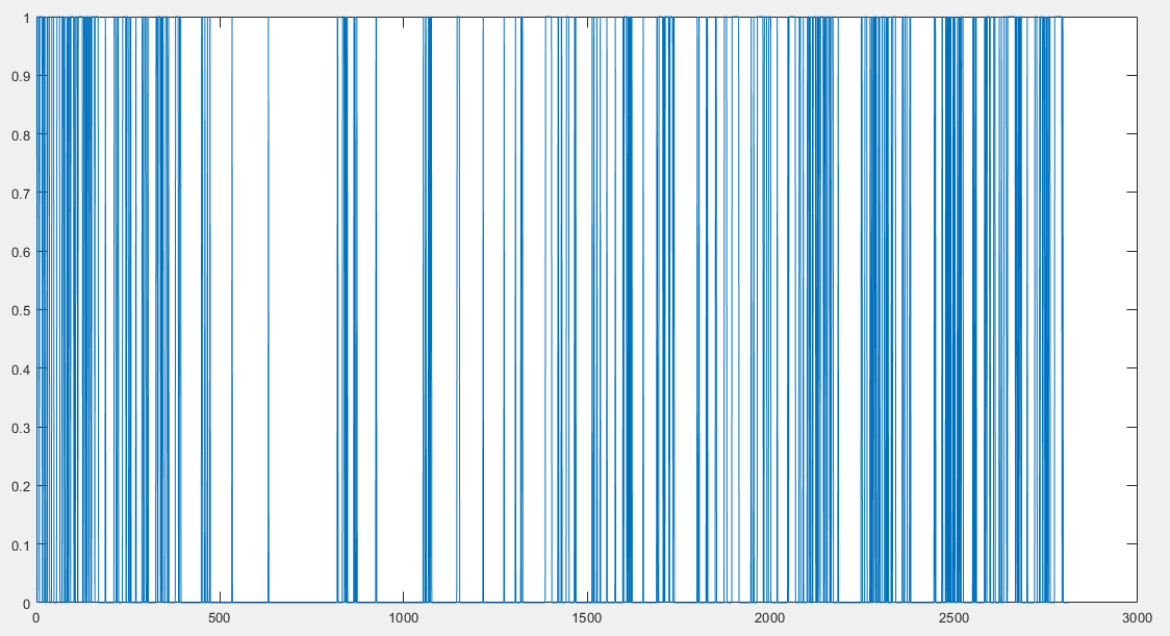
با توجه به شکل 32 خروجی ها را با ضرایب بدست آمده بازیابی میکنیم که با توجه به گسستگی ضرایب یکم صدا ها مخدوش میباشد، ولی به نظر درسته!:) البته در نهایت بنده کانال راست و چپ را ادغام کرده ام ☺ ولی برای کانال راست و چپ به ترتیب با Routput و Loutput ذخیره شده اند.

البته بعد از اعمال ضایب اینترپولیشن(با توجه به شکل5)انجام میدهیم که باید سیگنال اصلی بازیابی کامل بشود و بعد برای کانال چپ و راست 4 زیر کانال را باهم جمع میکنیم!

سوال9

برای نرم تر کردن ضرایب از تابع smooth استفاده میکنیم که دقیقا کار همان فیلتر پایین گذر را انجام میدهد که خروجی حاصل شده به صورت زیر است :

کانال اول:



33Figure

با فیلتر:

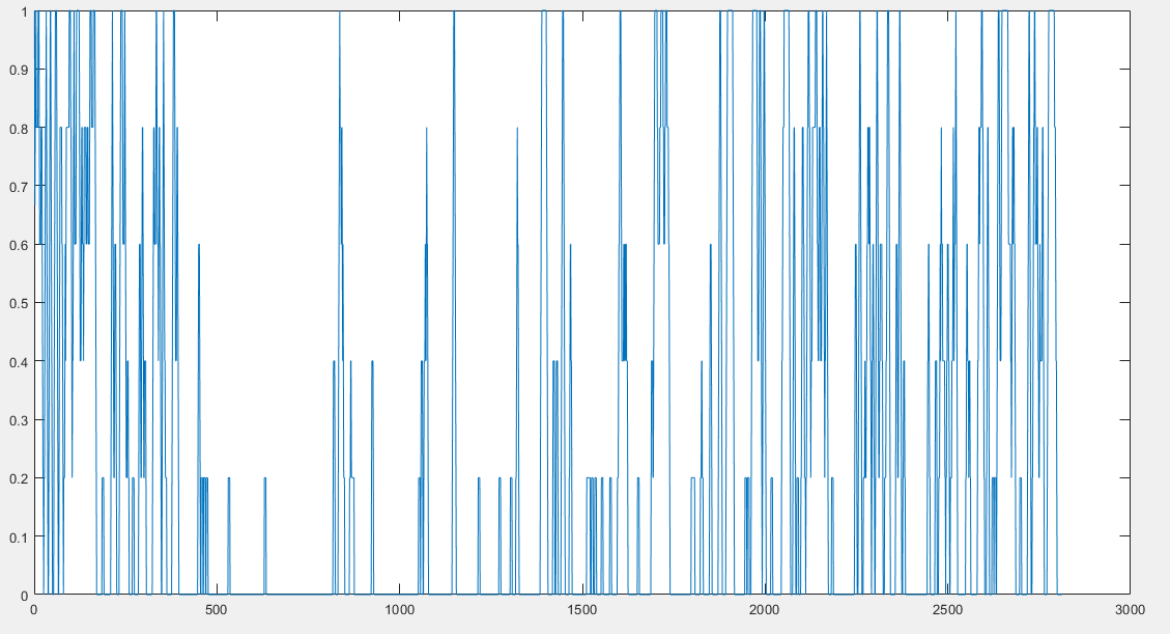


Figure 34

برای کانال دوم:

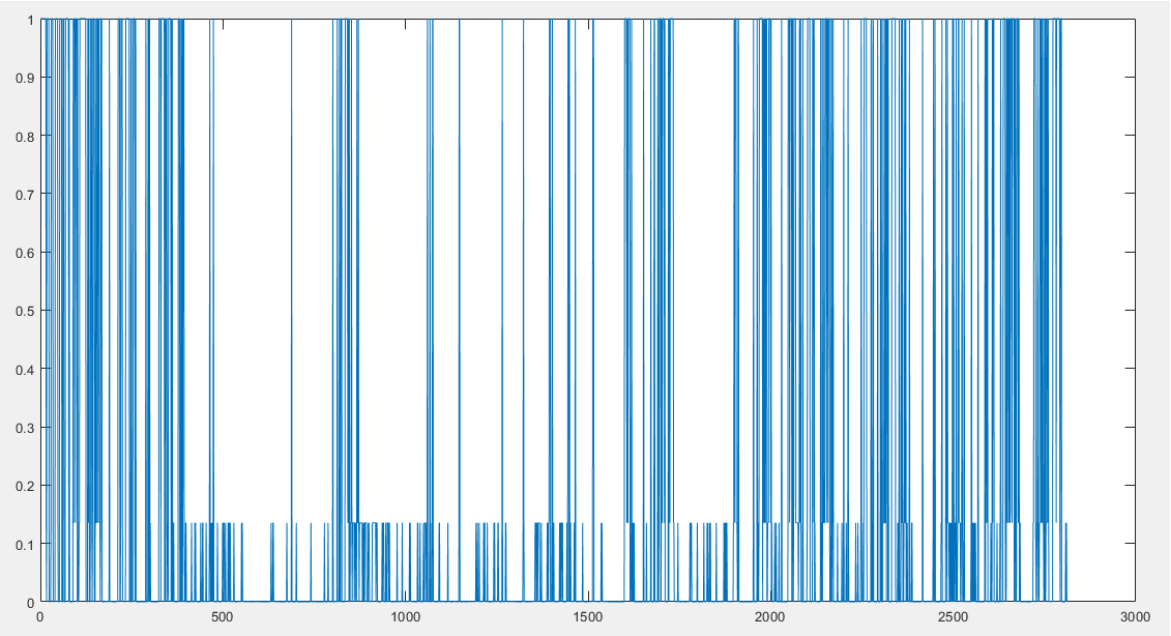
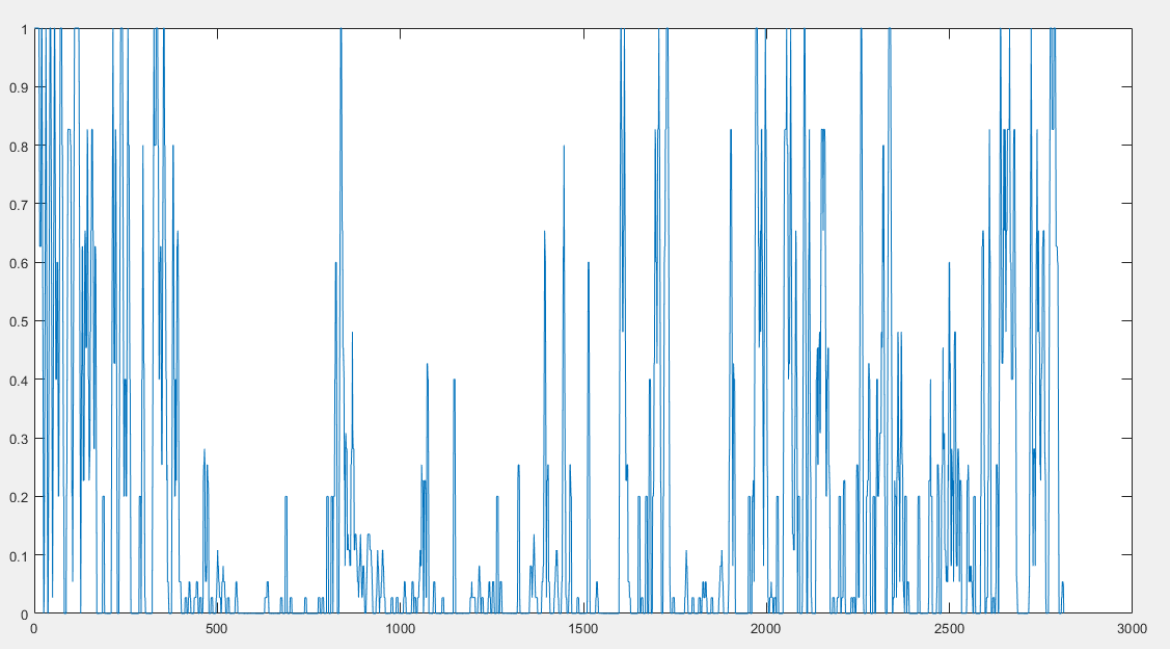


Figure 35

با فیلتر:



36Figure

برای کانال سوم:

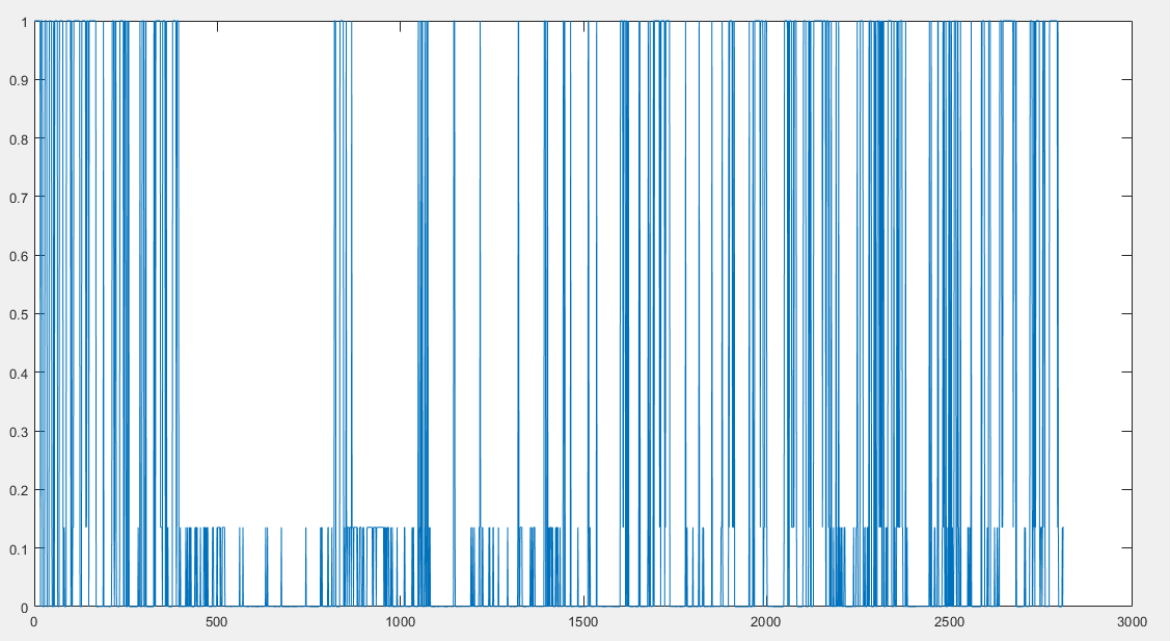
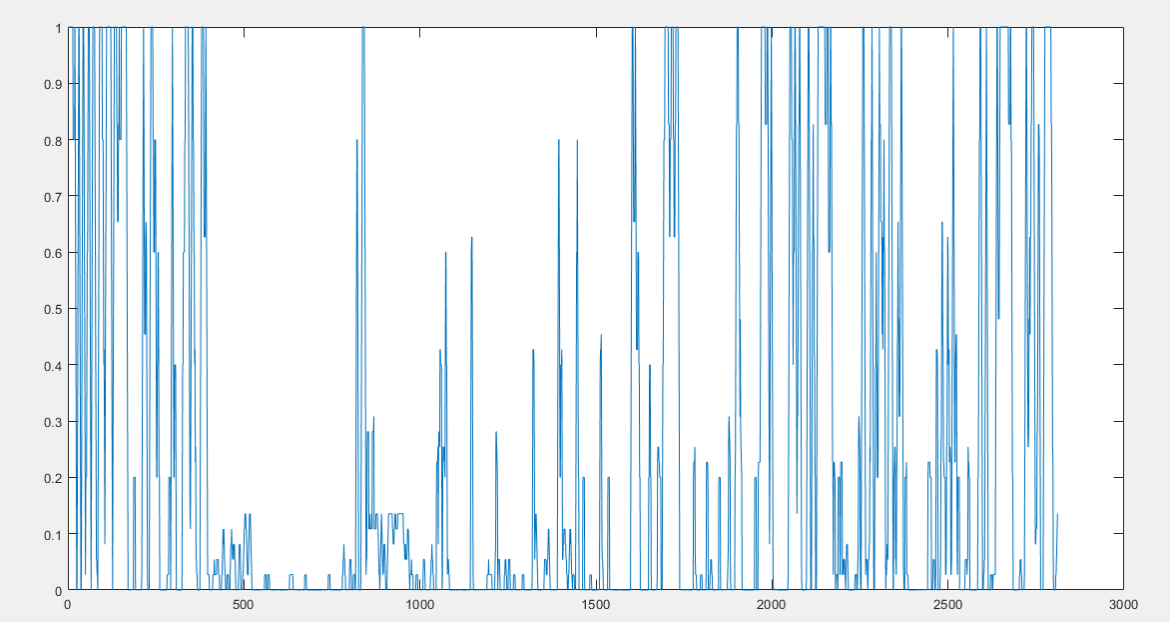


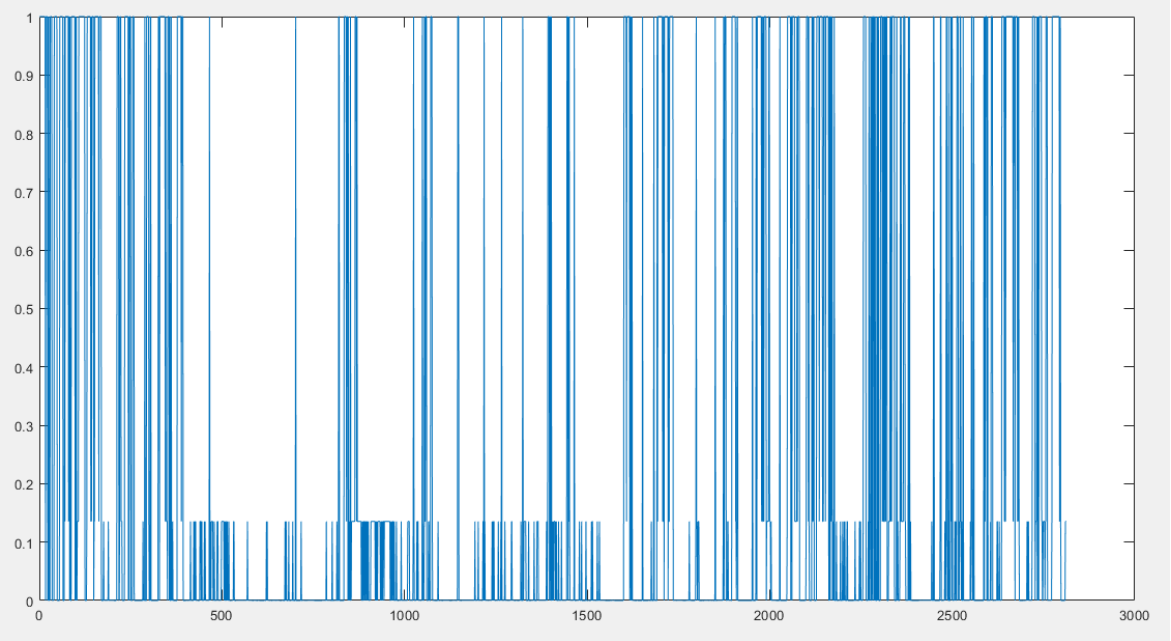
Figure 37

با فیلتر:



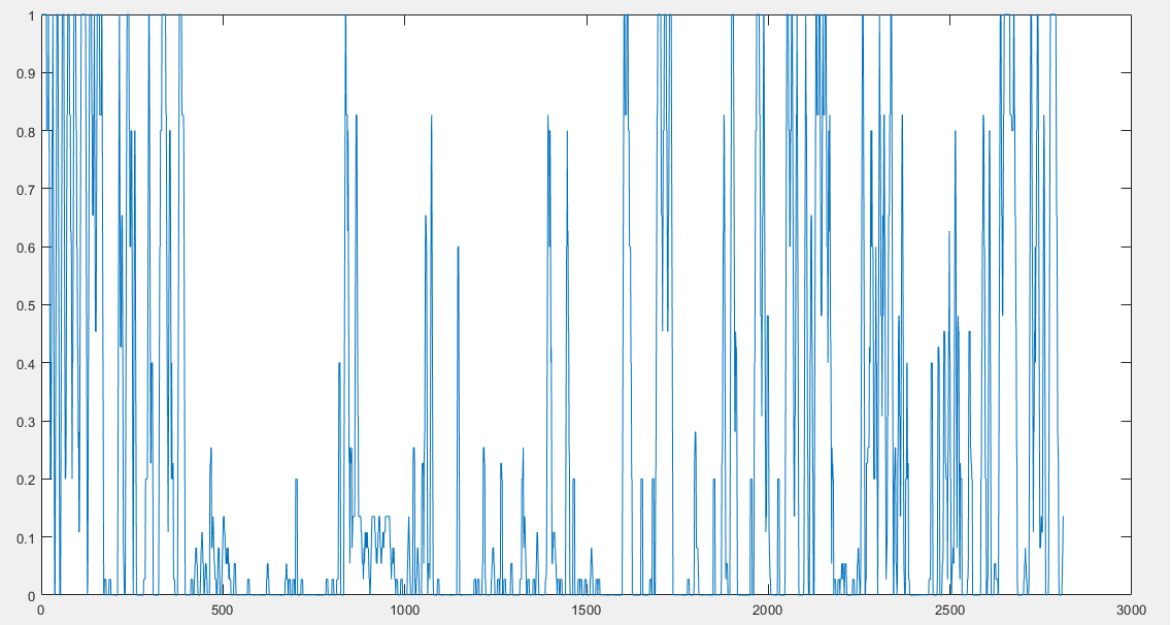
38Figure

برای کانال چهارم:



39Figure

با فیلتر:



40Figure

البته با فیلتر پایین کردن ضرایب باعث میشود که خروجی مقدار گسستگی را از دست میدهد!

سوال10

```
weighted_chunk=zeros(256,11250,8);
 for i=1:4
     for j=1:2812
     weighted_chunk(:,j,i)=All_Chunked_channels(:,j,i).*smooth_wight(i,j);
     weighted_chunk(:,j,i+4)=All_Chunked_channels(:,j,i+4).*smooth_wight(i,j);
     end
 end
 nroutput = zeros(2880016,1)
 lroutput = zeros(2880016,1)
 toutput = zeros(2880016,1)
 interpolated_weighted_chunk = zeros(720004,8)
  for j=1:8
    for i=1:2812
     interpolated_weighted_chunk(256*(i-1)+1:256*i,j)=weighted_chunk(:,i,j);
    end
  end
  for i=1:8
      interpolated_weighted_chunks(:,i)=interp(interpolated_weighted_chunk(:,i),4);
  end
  for i=1:8
      toutput(:,1)=toutput(:,1)+interpolated_weighted_chunks(:,i);
  end
  for i=1:4
      nroutput(:,1)=nroutput(:,1)+interpolated_weighted_chunks(:,i);
  end
  for i=5:8
      lroutput(:,1)=lroutput(:,1)+interpolated_weighted_chunks(:,i);
  end
  so=smooth(toutput);
  x=linspace(0,2880016,10000);
  figure
  plot(so);
audiowrite('Left_smoothed_output.wav',lroutput,Fs);
audiowrite('right_smoothed_output.wav',nroutput,Fs);
audiowrite('Sum_smooth_output.wav',so,Fs);
```

Figure 41

با توجه به شکل 41 اول چانک های وزن دار شده جدید را با وزن های جدید میسازیم و بعد 3 نوع خروجی را در آخر میسازیم که خروجی های کانال راست و چپ به ترتیب با نام های Right_smoothed_outputوLeft_smoothed_outputساخته شده اند و یک خروجیSum-smoothed_output هم داریم که در حقیقت جمع دو خروجی راست و چپ هست به علاوه اینکه دیتای دست آمده را نیز از یک فیلتر پایین گذر رد کرده ایم دوباره تا از گسستگی های موجود در سیگنال خروجی کاسته شود که در نتیجه سیگنال خروجی آخر را بهتر میسازد.

شکل زمانی سیگنال بازیابی شده آخر نیز به صورت زیر است:

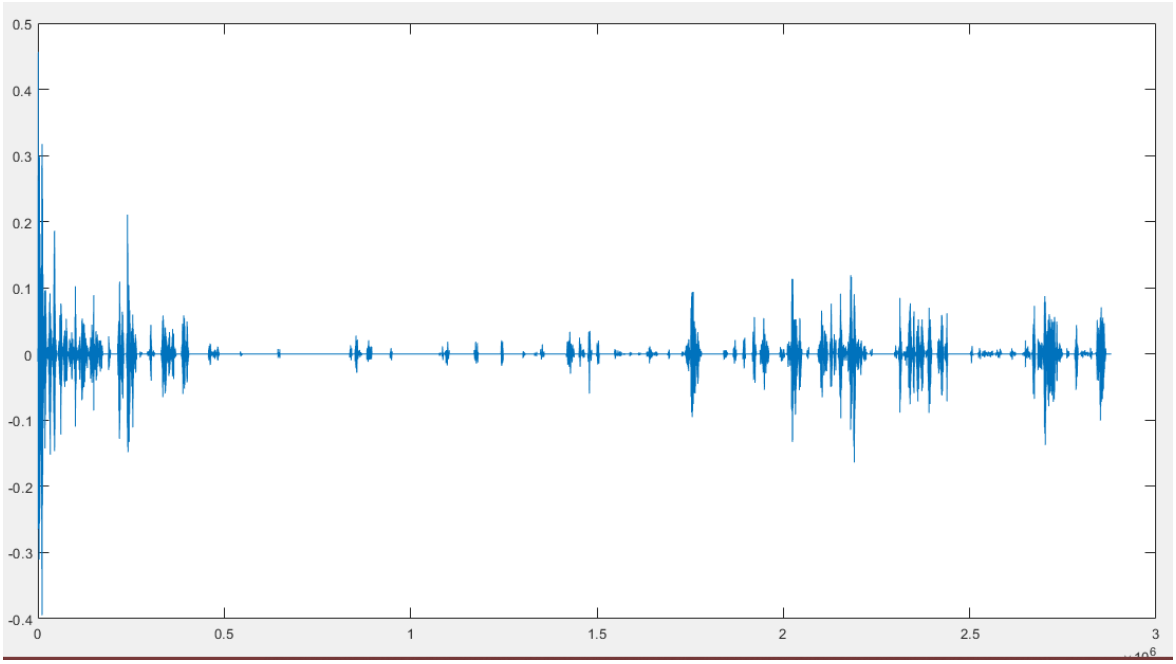


Figure 42

حوزه زمان سیگنال solo به صورت زیر میباشد:

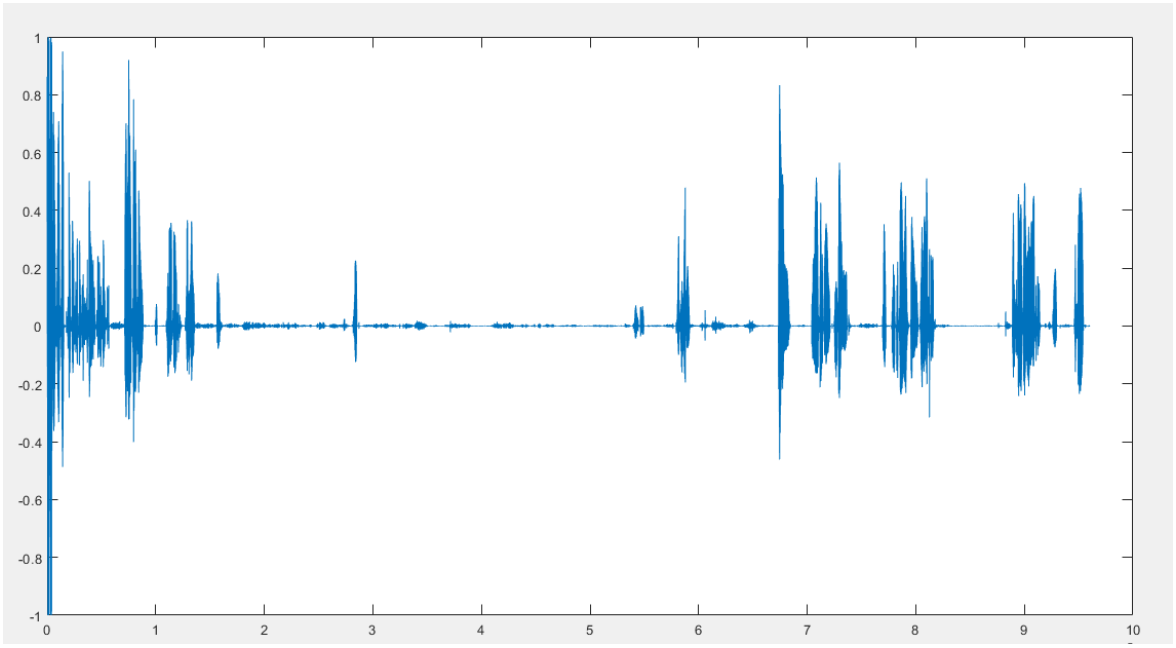


Figure 43

میتوان با بررسی دو خروجی بالا به این نتیجه رسیدک ه بازیابی انجام گرفته تقریبا شبیه و قابل قبول است!

☺