

آزمایش 6

پروژه نهایی درس

نام استاد : جناب دکتر مقیمی

نام دانشجو : محمد توزنده جانی

9720783

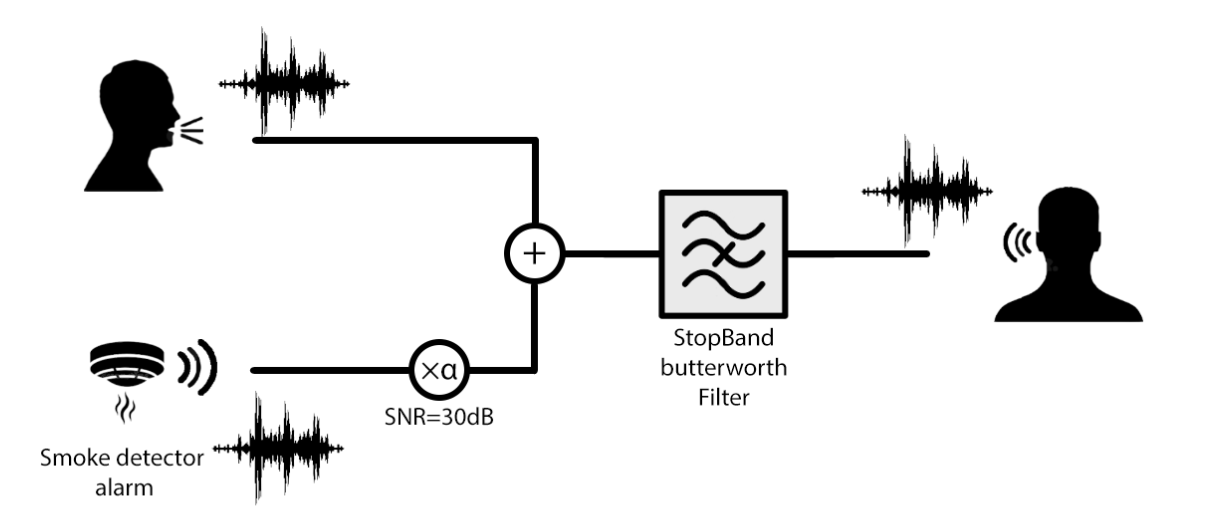
تاریخ ارسال :

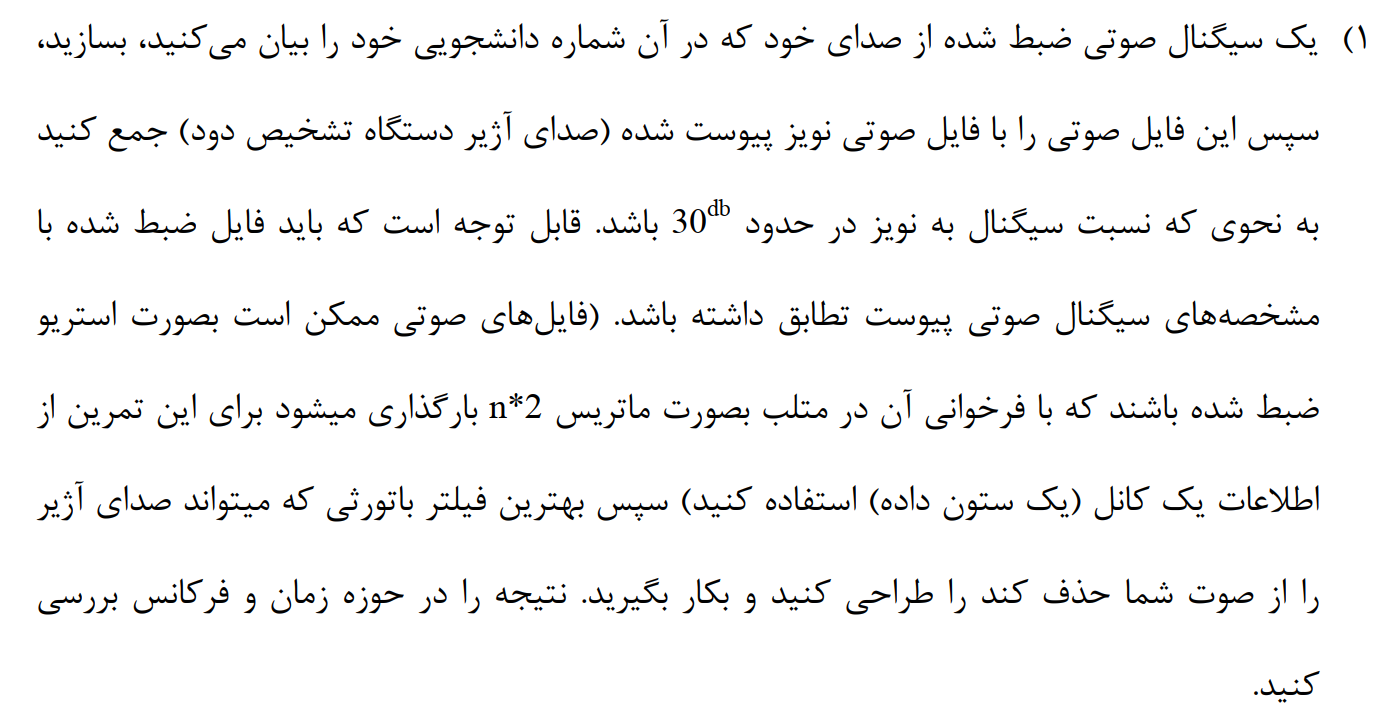
11/03/1401

نیمسال دوم 1400-1401

فهرست مطالب

* ذخیره سازی سیگنال ها ----------------------------------------------------------- 3
* حذف داده های اضافی ------------------------------------------------------------- 3
* یکسان سازی ابعاد ماتریس سیگنال ها ------------------------------------------------ 4
* جمع دو سیگنال داده و نویز --------------------------------------------------------- 4
* بررسی در حوزه ی فرکانس --------------------------------------------------------- 5
* طراحی فیلتر باترورث -------------------------------------------------------------- 7
* بررسی در حوزه ی زمان ------------------------------------------------------------ 8
* لینک های پخش فایل های صوتی ---------------------------------------------------- 11



ذخیره سازی سیگنال ها :

* ابتدا با دستور audioread صوت مربوط به صدای آژیر دستگاه تشخیص دود را به عنوان سیگنال نویز و صوت معرفی خود را به عنوان سیگنال داده اصلی در متلب ذخیره می کنیم:

**clc;**

**clear;**

**close all;**

**%% Create Noise & Data Signal--------------------------------------------**

**[sig,fs]=audioread('1-MyName.mp3'); %signal --> Data Signal**

**[noise,fsn]=audioread('2-Smoke\_Detector.mp3'); %noise --> Smoke\_Detector**

یکسان سازی ابعاد ماتریس سیگنال ها :

* ماتریس های هر سیگنال دارای دو ستون می باشد که ستون دوم مربوط به حالت stereo است.
* جهت حذف داده های اضافی بلااستفاده ، با دستور زیر هر دو سیگنال صوتی را ازstereo به mono تبدیل می کنیم. (ستون دوم هر ماتریس را حذف می کنیم ):

**%% Convert Stereo to Mono -----------------------------------------------**

**sig=sig(:,1);**

**noise=noise(:,1);**

**یکسان سازی ابعاد ماتریس سیگنال ها:**

* در ادامه با دستور زیر ابعاد دو ماتریس را یکسان می کنیم تا مشکلی جهت جمع کردن دو سیگنال با هم نداشته باشیم :

**%% Equalize the dimensions of the matrices -------------------------------**

**sig(483001:length(sig),:)=[];**

**noise(483001:length(noise),:)=[];**

جمع دوسیگنال پیام و نویز :

* می دانیم نسبت سیگنال به نویز برابر 30dB (دسی بل) می باشد ، به عبارت دیگر در جمع دو سیگنال ضریبی مانند در کنار نویز ظاهر می شود :
* *حال رابطه این ضریب را با SNR بدست می آوریم :*
* حال به سادگی ضریب بدست می آید و با توجه به این ضریب سیگنال d که جمع دو سیگنال داده و نویز می باشد نیز بدست می آید.
* کد های مربوط به جمع دو سیگنال با توجه به روابط بالا :

**%% Adding Noise to Signal ---------------------------------------------------**

**% Compute Sout = signal + noise such that SNR = Ps/Pn**

**%s: Input signal**

**%SNR: Desired signal-to-noise ratio**

**%d: Output signal**

**SNR=30; %in dB**

**Es = sum(sig(:).^2);**

**En = sum(noise(:).^2);**

**alpha = sqrt(Es/(SNR\*En));**

**Sout = sig+(alpha\*noise);**

**audiowrite('3-Combined\_signal(SNR=30).wav',Sout,fs); %Save Combined Signal**

* با دستور audiowrite سیگنال جمع شده را با نام دلخواه در پوشه ی متلب و با فرکانس سیگنال داده ذخیره می نماییم .
* با پخش فایل صوتی ایجاد شده ، صدای خود(سیگنال پیام) و صدای آژیر دستگاه(نویز) شنیده می شود.
* حال با طراحی فیلتر باترورث می خواهیم نویز را تا از سیگنال اصلی حذف کنیم.

بررسی در حوزه ی فرکانس:

* با دستور fft تبدیل فوریه سیگنال جمع شده با نویز و سیگنال پیام را بدست می آوریم با مقایسه این دو، محدوده ی طیف فرکانسی نویز مشخص است :

**%% Frequency Analyze -------------------------------------------**

**figure**

**subplot(2,1,1)**

**a0=fft(sig);**

**b0=fftshift(a0);**

**plot(abs(b0),'b','linewidth',1.5);**

**title('Message Fourier transform')**

**xlabel('483000 symbol')**

**grid minor**

**subplot(2,1,2)**

**a1=fft(Sout);**

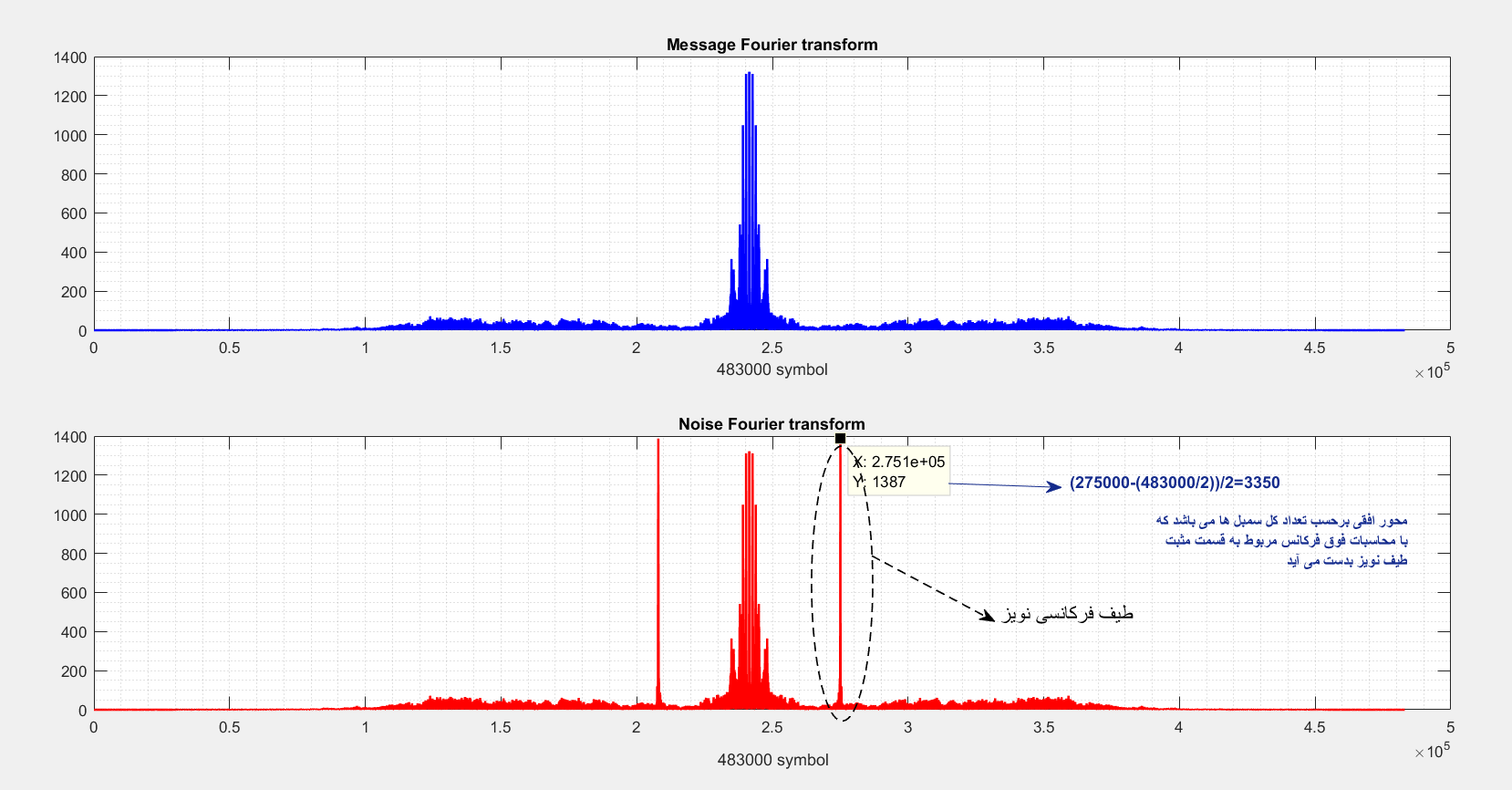
**b1=fftshift(a1);**

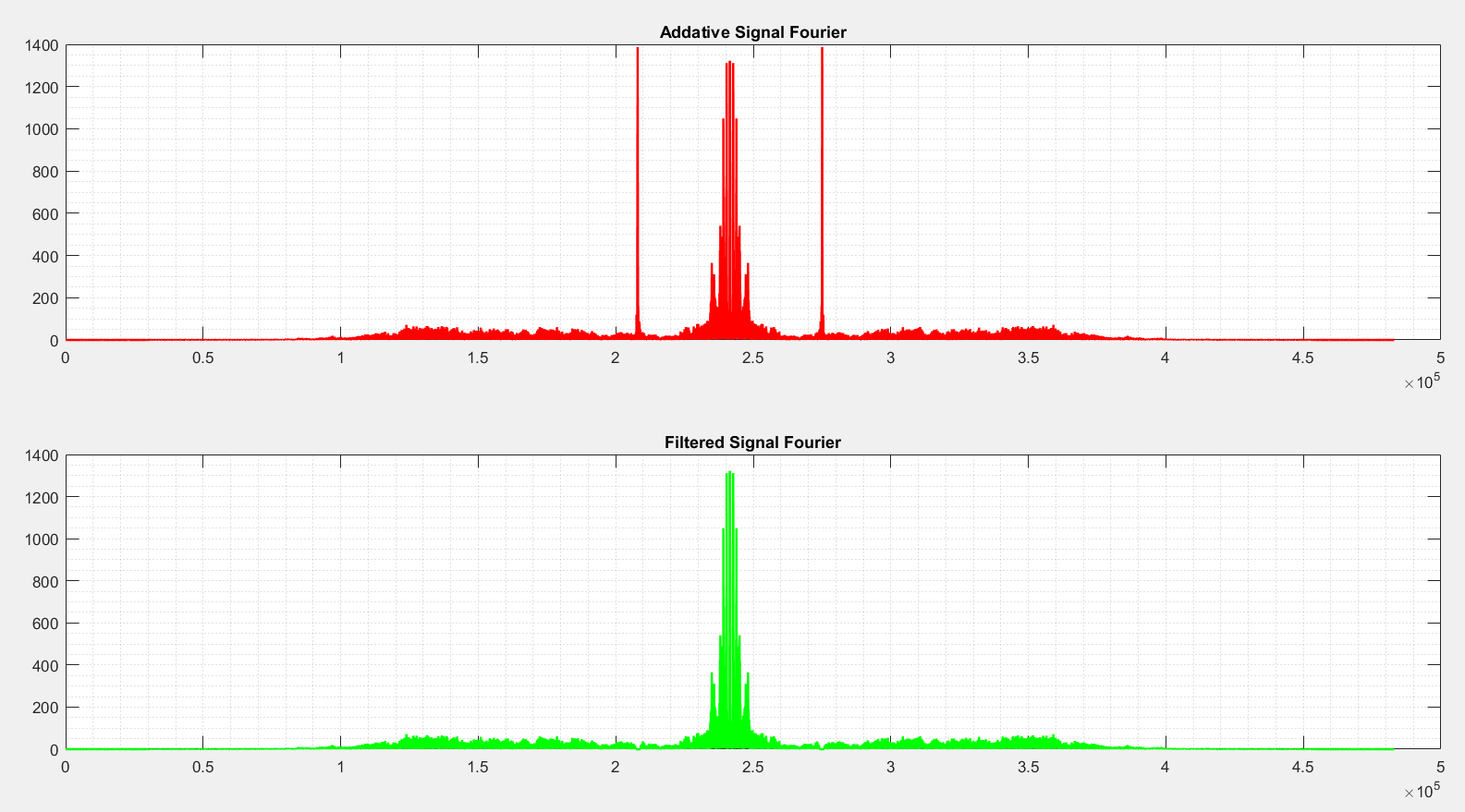
**plot(abs(b1),'r','linewidth',1.5);**

**title('Noise Fourier transform')**

**xlabel('48000symbol')**

**grid minor**

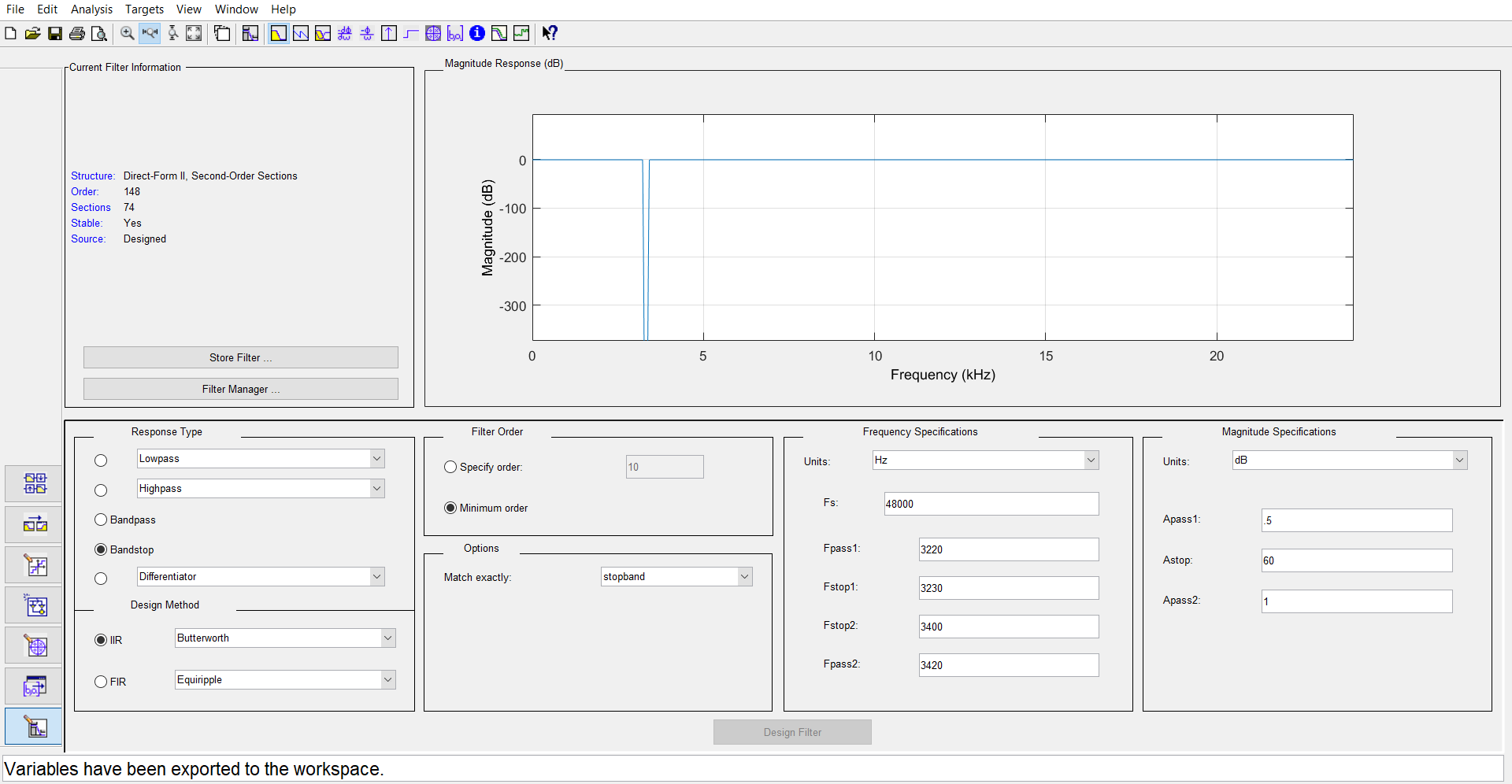
شکل1: طیف فرکانسی سیگنال پیام و سیگنال جمع شده

* حال پس از ساخت سیگنال فیلتر شده تبدیل فوریه آن را در بخش CommandWindow گرفته و با سیگنال جمع شده مقایسه می کنیم : در شکل زیر مشخص است طیف فرکانسی مربوط به نویز تاحد مطلوبی حذف شده است :

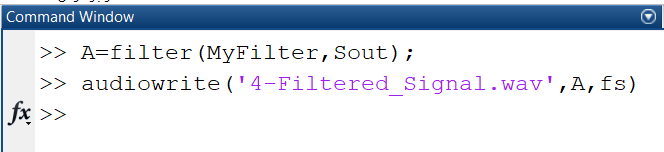
شکل2: طیف فرکانسی سیگنال جمع شده و سیگنال فیلتر شده

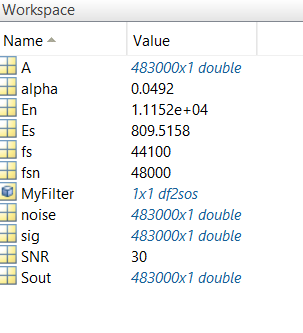
طراحی فیلتر باترورث:

* با توجه به تحلیل در حوزه ی فرکانس در قسمت قبل ، از یک فیلتر باترورث IIR میانگذر (BandStop) برای طراحی استفاده می کنیم.
* یک فیلتر IIR علاوه بر صفر ، قطب نیز دارد.
* اولویت ، طراحی فیلتر هایی با مرتبه پایین تر می باشد زیرا هرچه مرتبه فیلتر پایین تر باشد ، سرعت پردازش اطلاعات توسط فیلتر بالاتر است.
* در پنجره ی FilterDesigner پارامتر های مربوط به فیلتر قرار داده شده است :
* Astop سطح باند توقف بر حسب dB می باشد که برابر 60 قرار می دهیم ، هرچه این مقدار بزرگتر باشد ، درجه فیلتر بالاتر است .
* \* در تحلیل فرکانسی مشاهده کردیم در فرکانس حدود 3325Hz طیف فرکانسی نویز وجود داشت بنابراین در پنجره frequency specifications محدوده ی باند توقف را در محدوده این فرکانس
* به صورت زیر در نظر می گیریم؛ تا سیگنال نویز حذف گردد:



شکل 3: پنجره FilterDesigner و پارامتر های طراحی فیلتر باترورث

* پس از وارد کردن پاراکتر ها از منوی File قسمت Export فیلتر را با نام دلخواه در Workspace ذخیره کرده و با وارد کردن دستور های زیر در CommandWindow سیگنال فیلتر شده خروجی را بدست می آوریم و آن را یه عنوان یک فایل صوتی جدید در پوشه متلب ذخیره می نماییم و نتیجه کار یعنی سیگنال خروجی فیلتر را می شنویم :



شکل 4 : workspace متلب

بررسی در حوزه ی زمان:

* در این بخش ، سیگنال های پیام ، نویز و سیگنال جمع را در حوزه ی زمان رسم و بایکدیگر مقایسه می کنیم:
* شکل موج آبی رنگ سیگنال پیام می باشد.
* شکل موج قرمز رنگ سیگنال نویز(صدای آژیر دستگاه) می باشد.
* شکل موج سیاه رنگ ، حاصل جمع دو سیگنال پیام و نویز است.

**%% Time Analyze----------------------------------------------------**

**figure**

**subplot(3,1,1)**

**plot(sig,'b')**

**title('Message Signal')**

**grid minor**

**subplot(3,1,2)**

**plot(noise,'r')**

**title('Noise')**

**grid minor**

**subplot(3,1,3)**

**plot(Sout,'k')**

**title('Addative Signal')**

**grid minor**

**figure**

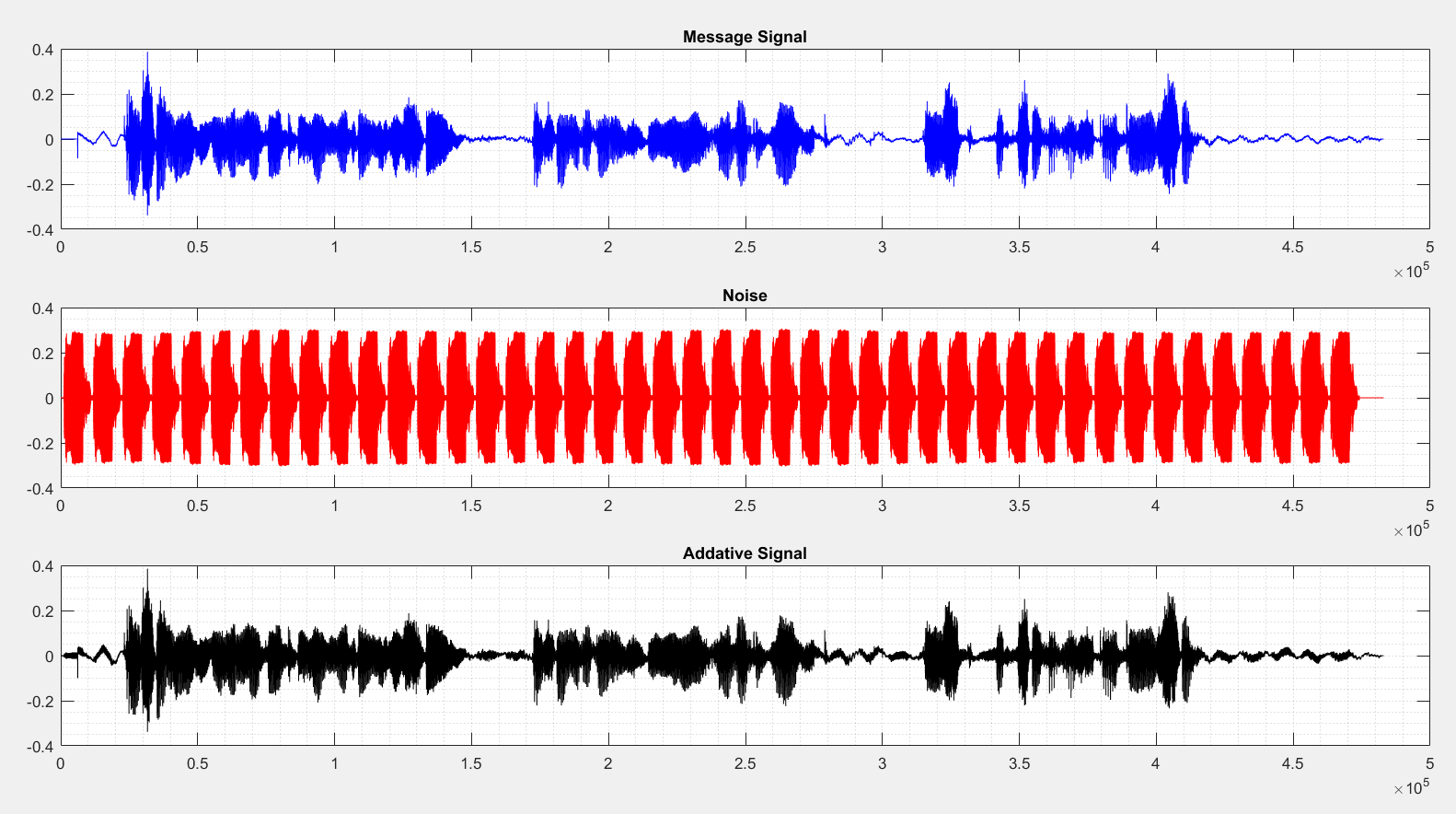
**plot(Sout,'r')**

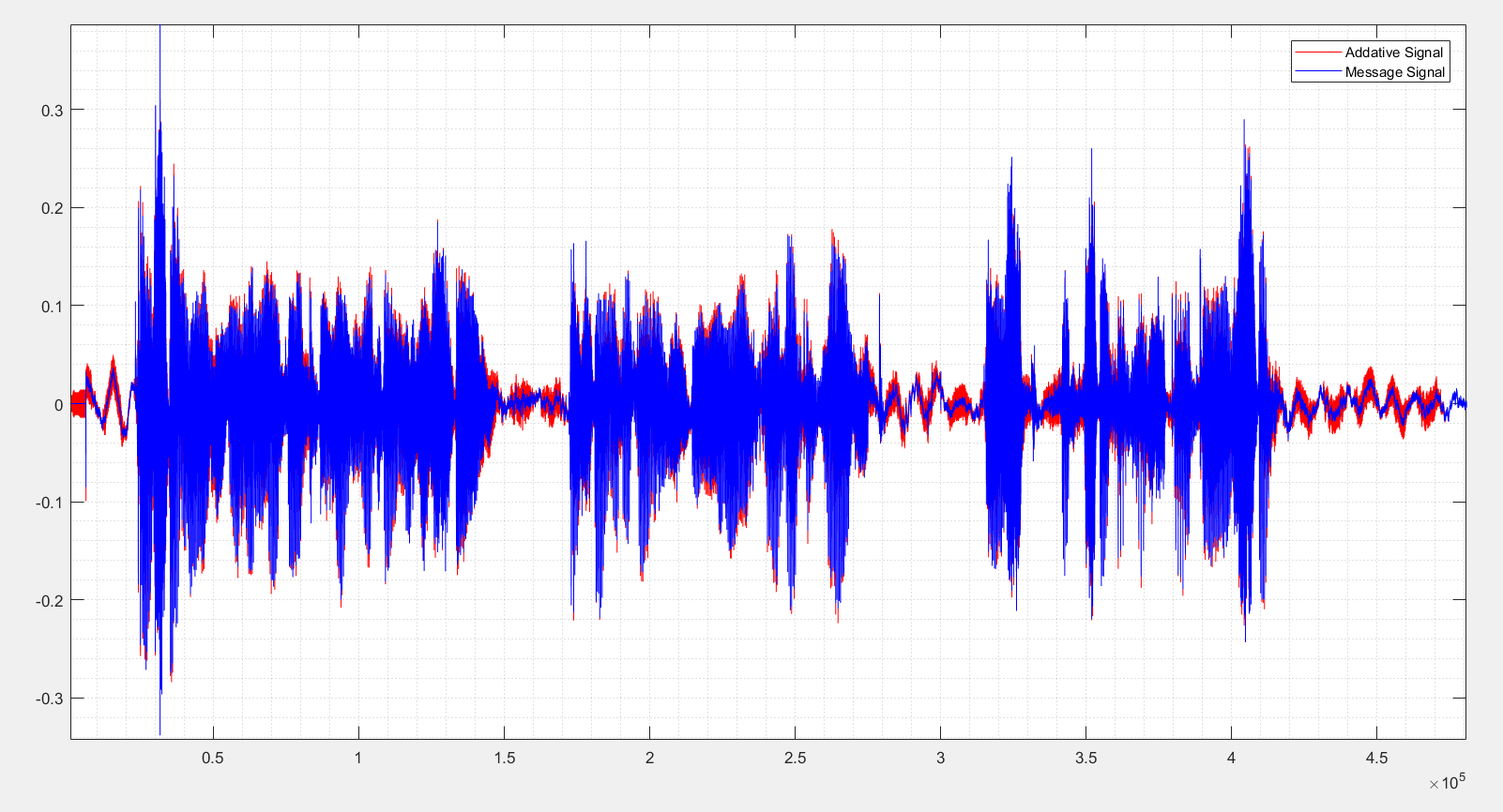
**hold on**

**plot(sig,'b')**

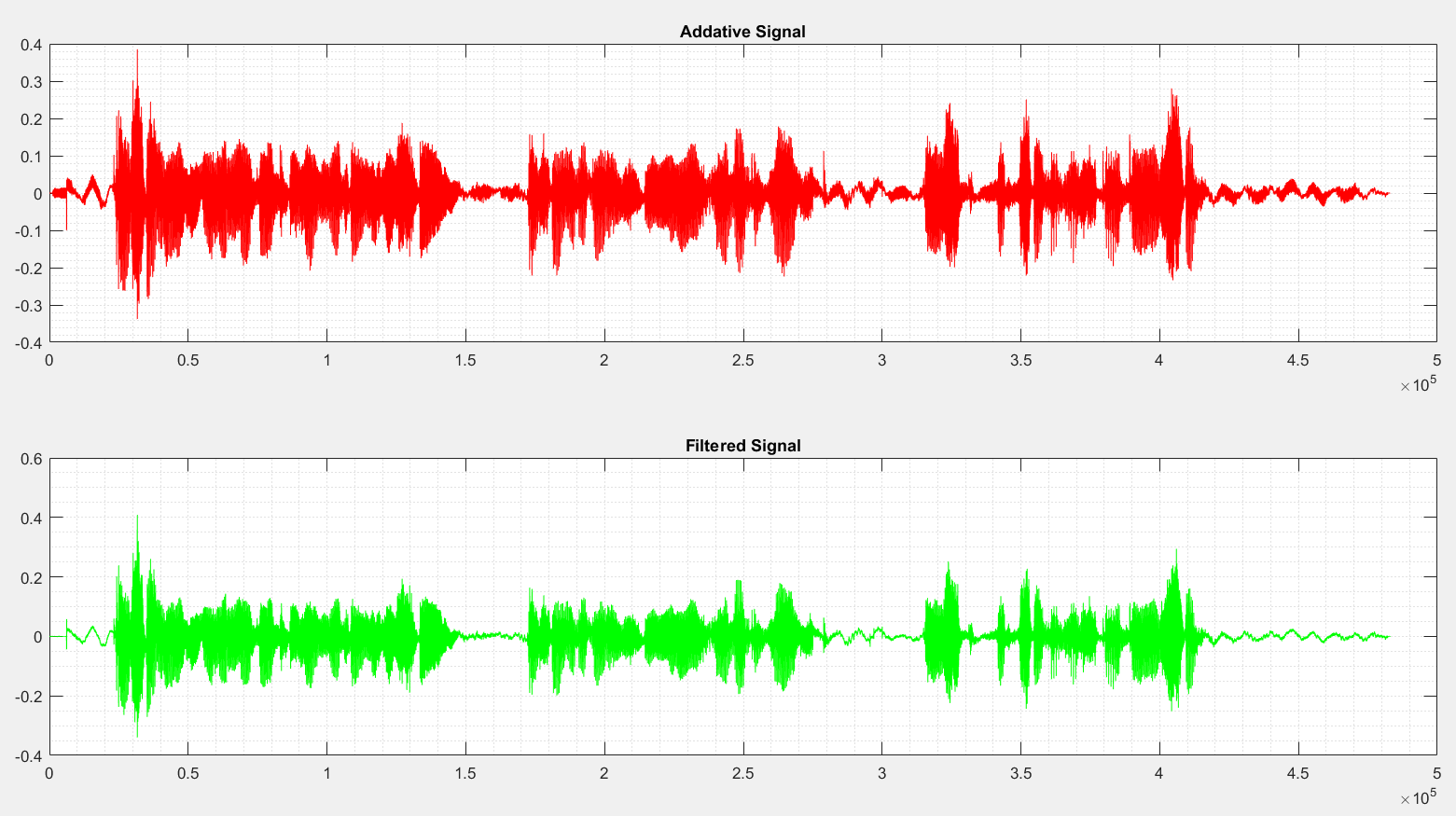
**grid minor**

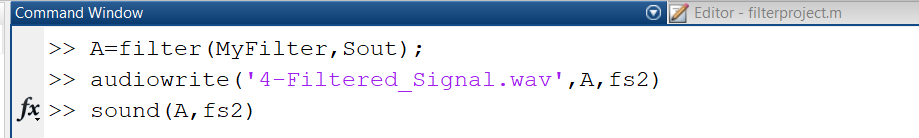
**legend('Addative Signal','Message Signal')**

شکل 5 : سیگنال پیام - سیگنال نویز(صدای آژیر دستگاه) – سیگنال جمع شده در حوزه ی زمان



شکل 5 : نمایش اثر نویز بر سیگنال پیام ( قسمت های قرمز رنگ ، افزایش دامنه سیگنال پیام ناشی از افزوده شدن نویز به آن می باشد).

شکل 5 : سیگنال قرمز ورودی فیلتر (سیگنال جمع) - سیگنال سبز خروجی فیلتر (سیگنال فیلتر شده)

\* جهت ذخیره سیگنال فیلتر شده در WorkSpace و پخش صوت آن در نرم افزار متلب پس از Export کردن فیلتر از پنجره FilterDesigner دستورات زیر در CommandWindow اعمال شود؛

لینک های پخش فایل های صوتی :

برای پخش روی نوشته های زیر کلیک کنید؛

[1- پخش سیگنال پیام](1-MyMessage.mp3)

[2- پخش سیگنال نویز](2-Smoke_Detector.mp3)

[3- پخش سیگنال جمع شده (سیگنال پیام همراه با نویز)](3-Combined_signal(SNR=30).wav)

[4-پخش سیگنال فیلتر شده (خروجی فیلتر باترورث - نتیجه کار)](4-Filtered_Signal.wav)