

به نام خدا



دانشگاه صنعتی شریف

دانشکده مهندسی برق

پروژه درس سیگنال‌ها و سیستم‌ها

علی خدنگی

۹۸۱۰۱۴۹۳

علی نوریان

۹۸۱۰۲۵۲۷

استاد مربوطه :

دکتر کربلایی آقاجان

نیمسال ۱۳۹۹

فهرست مطالب

۳	۱ نمونه برداری
۳	۱.۱ تابع HalfBandFFT
۵	۲.۱ تعیین حد پایین فرکانس نمونه برداری
۷	۲ آشنایی با سیگنال‌های EEG
۷	۱.۲ مولفه‌ی P300
۷	۲.۲ باندهای فرکانسی و کاربرد آن‌ها
۹	۳.۲ باند فرکانسی و فرکانس نمونه برداری سیگنال‌های EEG
۱۱	۴.۲ محاسبه فرکانس نمونه برداری
۱۲	۵.۲ پیدا کردن فرکانس قطع سیگنال
۱۳	۶.۲ کاهش فرکانس نمونه برداری
۱۴	۷.۲ epoching
۱۵	۳ خوشه بندی بر مبنای همبستگی
۱۵	۱.۳ روابط همبستگی
۱۶	۲.۳ اعمال الگوریتم خوشه بندی بر روی دیتای ۶۳ کاناله
۱۸	۳.۳ اعمال الگوریتم خوشه بندی روی دیتای ۸ کاناله
۱۹	۴ طراحی فیلتر
۱۹	۱.۴ بررسی فیلتر با فاز خطی
۱۹	۲.۴ اثبات رابطه تاخیر گروه
۲۰	۳.۴ تست عملکرد تابع پیاده سازی شده
۲۱	۴.۴ اعمال تابع تاخیر گروه بر روی فیلتر های استفاده شده در طول پروژه
۲۲	۵ شناسایی کلمات
۲۲	۱.۵ الگوریتم
۲۴	۲.۵ تلاش‌ها
۲۴	۱.۲.۵ ویژگی شماره ۱
۲۴	۲.۲.۵ ویژگی شماره ۲
۲۴	۳.۲.۵ ویژگی شماره ۳
۲۴	۴.۲.۵ شبکه عصبی!
۲۴	۵.۲.۵ ویژگی شماره ۲ - نتیجه مطلوب!
۲۴	۶.۲.۵ شبکه عصبی نتیجه بخش:

۱ نمونه برداری

۱.۱ تابع HalfBandFFT

ابتدا می‌خواهیم تابع HalfBandFFT را بررسی کنیم. کد این تابع به صورت زیر است :

```
function HalfBandFFT(X, Fs, b, my_title)
    if nargin < 3
        b = 1;
    end

    Y = fft(X);

    a = 1;
    L = length(X);
    P2 = abs(Y/L);
    P1 = P2(1:floor(L*a));
    P1(2:end-1) = 2*P1(2:end-1);

    f = Fs*(0:floor(L*a)-1)/L;
    w = 2 * pi * f / Fs;

    if b == 1
        plot(w,P1,'linewidth',1.5);

        if nargin == 4
            title(sprintf('%s',my_title),'Interpreter','latex');
        end

        xlabel('$w$ (rad/s)','Interpreter','latex');
        ylabel('$|H(f)|$', 'Interpreter','latex');
        xlim([0 2*pi/a]); grid minor;
        set(gca,'XTick',0:pi/4:2*pi/a);
        set(gca,'XTickLabel',{'0','\pi/4','\pi/2','3\pi/4','\pi',...
            '5\pi/4','3\pi/2','7\pi/4','2\pi'});

    else
        plot(f(1:floor(L/2)),P1(1:floor(L/2)),'linewidth',1.5);

        if nargin == 4
            title(sprintf('%s',my_title),'Interpreter','latex');
        end

        xlabel('$f$ (Hz)','Interpreter','latex');
        ylabel('$|H(f)|$', 'Interpreter','latex');
        xlim([0,f(floor(L/2))]); grid minor;
    end
end
```

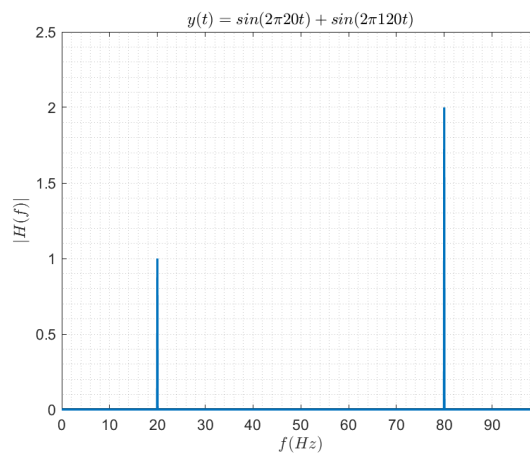
از تابع فوق می‌توان با ۲، ۳ یا ۴ ورودی استفاده کرد. پیش فرض تابع رسم تبدیل فوریه سیگنال در بازه $[0, 2\pi]$ می‌باشد. ورودی b اگر صفر باشد تبدیل فوریه در فرکانس درست و اگر یک باشد در بازه $[0, 2\pi]$ رسم می‌شود. در صفحه بعد چند نمونه از خروجی این تابع رسم شده است.

سیگنال زیر را در نظر میگیریم :

$$y[n] = \sin(2\pi 20n) + 2\sin(2\pi 120n)$$

فرکانس نمونه برداری را برابر ۲۰۰ هرتز در نظر میگیریم در نتیجه تبدیل فوریه این سیگنال به صورت زیر است :

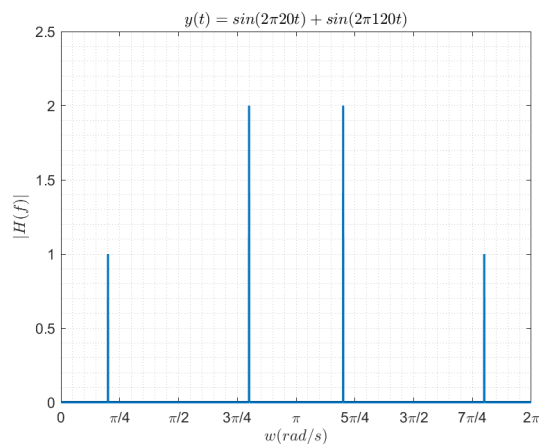
```
Fs = 200;  
t = 0:1/Fs:10;  
y = sin(2*pi*20*t) + 2*sin(2*pi*120*t);  
  
Title = '$y[n] = \sin(2\pi 20n) + \sin(2\pi 120n)$';  
figure;  
HalfBandFFT(y, Fs, 0, Title);
```



شکل ۱

اکنون بار دیگر تبدیل فوریه این سیگنال را در بازه $[0, 2\pi]$ رسم می کنیم :

```
figure;  
HalfBandFFT(y, Fs, 1, Title);
```

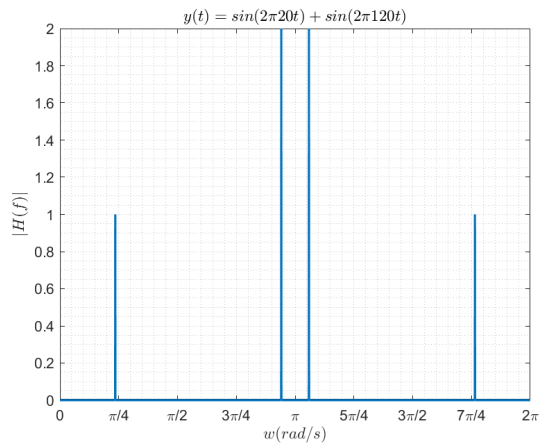


شکل ۲

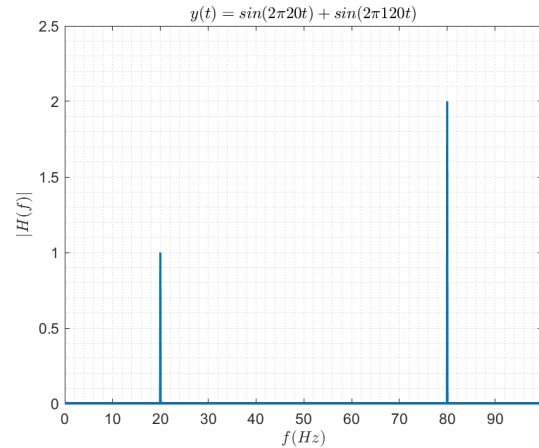
همانطور که مشاهده می شود به در هر دو حالت به درستی رسم شده است. در قسمت بعد پدیده تداخل فرکانسی که به دلیل عدم انتخاب مناسب فرکانس نمونه برداری می باشد را بررسی خواهیم کرد.

۲.۱ تعیین حد پایین فرکانس نمونه برداری

همان سیگنال قسمت قبل را در نظر میگیریم. ماکسیمم فرکانس موجود در سیگنال برابر ۸۰ هرتز است. می توان مشاهده کرد که با کاهش فرکانس نمونه برداری تا دو برابر ۸۰ هرتز، نمایش این فرکانس در بازه $[0, 2\pi]$ به دلیل تقارن نسب به π از دو طرف به مقدار π نزدیک می شود :



شکل ۴: $F_s = 170 \text{ Hz}$

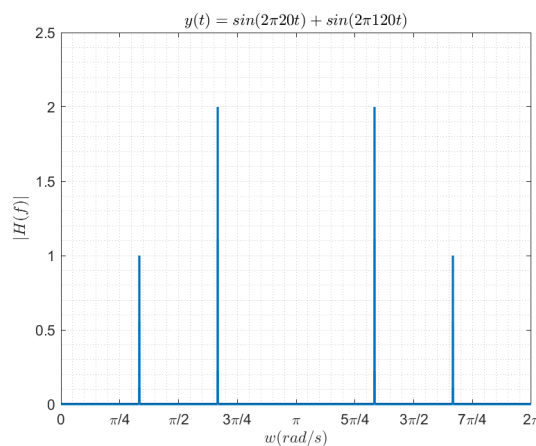


شکل ۳: $F_s = 200 \text{ Hz}$

با رسیدن فرکانس نمونه برداری به ۱۶۰ هرتز این دو بر هم و بر روی π منطبق می شوند و با کاهش بیشتر فرکانس نمونه برداری از محدوده خود عبور کرده و فرکانس بالایی که ۸۰ هرتز می باشد به یک فرکانس پایین تر مپ می شود زیرا با داشتن یک فرکانس نمونه برداری کمتر از ۱۶۰ هرتز قادر به نگه داری محتوای فرکانسی بیشتر از ۸۰ هرتز نخواهیم بود. این امر به پدیده تداخل فرکانسی نایکوئیست معروف است که شرط زیر را برای حداقل فرکانس نمونه برداری به همراه دارد :

$$\omega_s > 2 \times \omega_{Max}$$

شکل زیر پدیده aliasing را نشان می دهد :



شکل ۵: $F_s = 120 \text{ Hz}$

نمونه دیگری از پایین پدیده در صفحه بعد آمده است.

اکنون پدیده تداخل فرکانسی را برای نمونه بیشتر در سیگنال زیر تحقیق می‌کنیم :

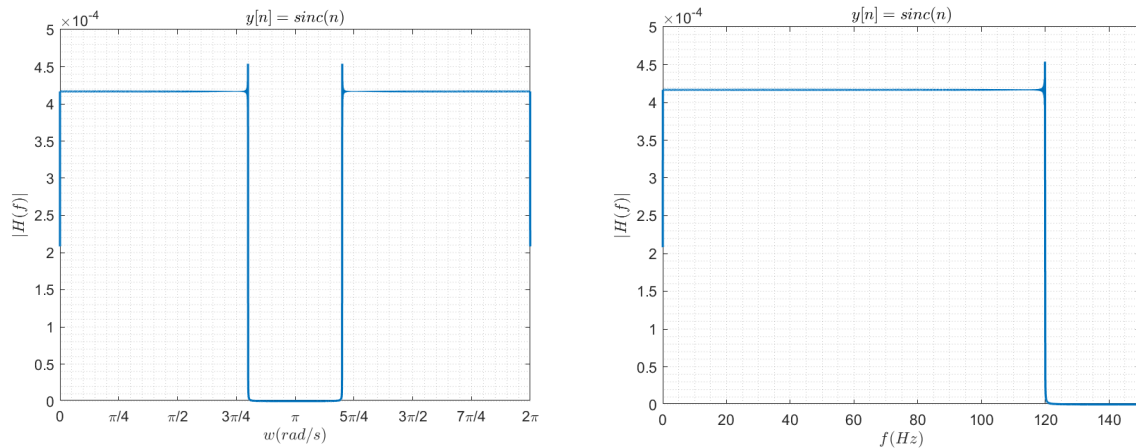
$$y[n] = \text{sinc}[n] = \frac{\sin[2\pi 120n]}{2\pi 120n}$$

می‌دانیم تبدیل فوریه تابع sinc یک پالس مستطیلی می‌باشد. تبدیل فوریه سیگنال فوق را به ازای یک فرکانس نمونه برداری بالا بیش از ماکسیمم فرکانس سیگنال ($F_s = 300\text{Hz}$) رسم می‌کنیم :

```

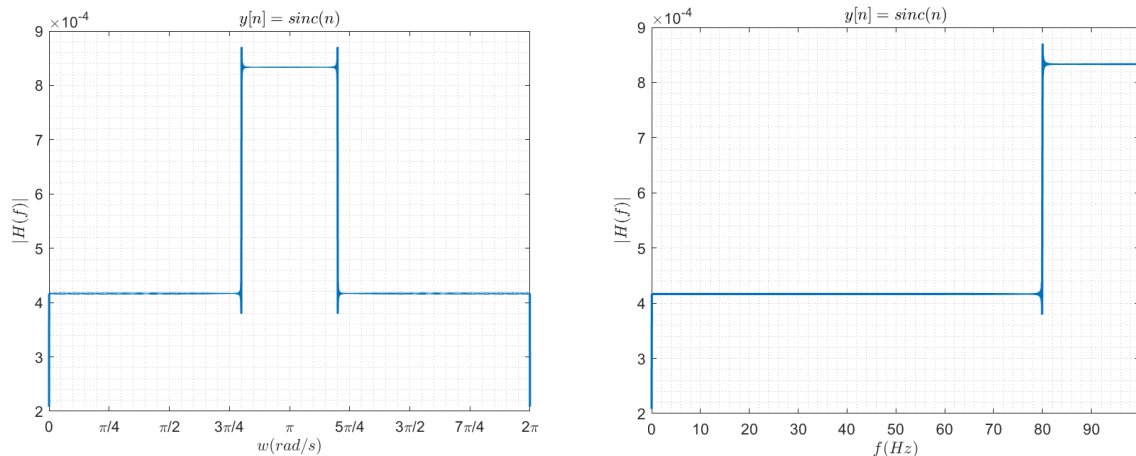
Fs = 300;
t = -10:1/Fs:10-1/Fs;
f0 = 120;
y = sin(2*pi*f0.*t)./(2*pi*f0.*t);
Title = '$y[n] = \text{sinc}(n)$';
figure; HalfBandFFT(y, Fs, 0, Title);
figure; HalfBandFFT(y, Fs, 1, Title);

```



شکل ۶: $F_s = 200\text{Hz}$

همانگونه که انتظار داریم و در شکل فوق نیز مشخص است تبدیل فوریه این سیگنال به صورت یک پالس مستطیلی می‌باشد که بیشترین محتوای فرکانسی آن در $f = 120\text{Hz}$ است. می‌دانیم با کاهش فرکانس نمونه برداری به کمتر از 240 هرتز دو پالس مستطیلی که در هر تناوب تکرار می‌شوند در هم ادغام شده و تداخل فرکانسی رخ می‌دهد. این امر از خروجی کد زیر نیز مشخص است :



شکل ۷: $F_s = 120\text{Hz}$

۲ آشنایی با سیگنال‌های EEG

۱.۲ مولفه‌ی P300

The P300 (P3) wave is an event-related potential (ERP) component elicited in the process of decision making. It is considered to be an endogenous potential, as its occurrence links not to the physical attributes of a stimulus, but to a person's reaction to it. More specifically, the P300 is thought to reflect processes involved in stimulus evaluation or categorization.

It is usually elicited using the oddball paradigm, in which low-probability target items are mixed with high-probability non-target (or "standard") items. When recorded by electroencephalography (EEG), it surfaces as a positive deflection in voltage with a latency (delay between stimulus and response) of roughly 250 to 500 ms.

The signal is typically measured most strongly by the electrodes covering the parietal lobe. The presence, magnitude, topography and timing of this signal are often used as metrics of cognitive function in decision-making processes. While the neural substrates of this ERP component still remain hazy, the reproducibility and ubiquity of this signal makes it a common choice for psychological tests in both the clinic and laboratory.[?]

۲.۲ باندهای فرکانسی و کاربرد آن‌ها

The textbook definition of a frequency band is an interval in the frequency domain, delimited by a lower frequency and upper frequency. [The International Telecommunication Union](#) has assigned designations to these intervals.[?]

A frequency band is an interval in the frequency domain, delimited by a lower frequency and an upper frequency. The term may refer to a radio band or an interval of some other spectrum. The frequency range of a system is the range over which it is considered to provide satisfactory performance, such as a useful level of signal with acceptable distortion characteristics. A listing of the upper and lower limits of frequency limits for a system is not useful without a criterion for what the range represents. Many systems are characterized by the range of frequencies to which they respond. Musical instruments produce different ranges of notes within the hearing range. The electromagnetic spectrum can be divided into many different ranges such as visible light, infrared or ultraviolet radiation, radio waves, X-rays and so on, and each of these ranges can in turn be divided into smaller ranges. A radio communications signal must occupy a range of frequencies carrying most of its energy, called its bandwidth. A frequency band may represent one communication channel or be subdivided into many.[?]

Beginning with the lowest and ending with the highest, we will enumerate the ITU-designated frequency bands and provide examples of their corresponding applications.

Firstly, the Extremely Low Frequency (ELF) band is ideal for underwater communication. Transmitters in the 22 Hz range of this band are useful in pigging, also known as pipeline transportation. The Super Low Frequency (SLF) band is also suitable for submarine communication.

The waves within Ultra Low Frequency (ULF) band are able to penetrate through dirt and rock. Through-the-earth signal transmission is especially useful in secure communications, making it suitable for military applications. TTE is also used in mining. Similarly, the Very Low Frequency (VLF) band can also penetrate dirt and rock for some distance. Thus, geophysicists use VLF-electromagnetic receivers to measure

conductivity in the near surface of the earth. VLF frequencies benefit from their long range and stable phase characteristics, allowing them to be quite versatile. Like ELF and SLF, VLF can also penetrate seawater to some extent; the military can use VLF to communicate with submarines near the surface of the water. Historically, VLF has been used for navigation beacons.

The High Frequency (HF) band is most useful in shortwave radio applications, as well as aviation air-to-ground communications. Dipole antennas, such as the Yagi, quad, and log-periodic antennas, operate within the higher frequencies of the HF band. Because its wavelengths range from one to ten decametres (10 to 100 meters), the HF band is also known as the decametre band. The Very High Frequency band is suitable for similar applications as the HF band. Additionally, whereas AM radio operates within the LF and MF bands, FM radio operates within the VHF band.

The Ultra High Frequency (UHF) band is perhaps most closely integrated into modern civilian life. In addition to military applications, the UHF band is used in satellite television, mobile phones, Wi-Fi, walkie-talkies, and GPS.

Falling within the microwave band, the Super High Frequency (SHF) band is also optimized for wireless communications. Because the relatively smaller wavelengths of microwaves allow them to be directed in narrow beams, the SHF band is optimal for point-to-point communication using parabolic dishes and horn antennas, for example. Patch antennas typically operate within the SHF band as well. Aside from microwave heating, the SHF band is optimal for satellite links and radar transmitters. The SHF band is also known as the centimetre band because its wavelengths range from one to ten centimetres.

Lastly, the Extremely High Frequency (EHF) band is the highest band on our list. It is also known as the millimetre band, because its wavelengths measure between one to ten millimetres. Because its radio waves are able to be absorbed by the gases in the atmosphere, they only have a short range and can only be used for terrestrial communication over about a kilometer. While certain frequency ranges near the bottom of the band are currently used in 5G cellphone networks, the EHF band is most commonly used in astronomy and remote sensing.[?]

Frequency Band Name	Acronym	Frequency Range	Wavelength (Meters)
Extremely Low Frequency	ELF	3 to 30 Hz	10,000 to 100,000 km
Super Low Frequency	SLF	30 to 300 Hz	1,000 to 10,000 km
Ultra Low Frequency	ULF	300 to 3000 Hz	100 to 1,000 km
Very Low Frequency	VLF	3 to 30 kHz	10 to 100 km
Low Frequency	LF	30 to 300 kHz	1 to 10 km
Medium Frequency	MF	300 to 3000 kHz	100 to 1,000 m
High Frequency	HF	3 to 30 MHz	10 to 100 m
Very High Frequency	VHF	30 to 300 MHz	1 to 10 m
Ultra High Frequency	UHF	300 to 3000 MHz	10 to 100 cm
Super High Frequency	SHF	3 to 30 GHz	1 to 10 cm
Extremely High Frequency	EHF	30 to 300 GHz	1 to 10 mm

Figure 8

۳.۲ باند فرکانسی و فرکانس نمونه برداری سیگنال های EEG

Most of the cerebral signal observed in the scalp EEG falls in the range of 1–20 Hz (activity below or above this range is likely to be artifactual, under standard clinical recording techniques). Waveforms are subdivided into bandwidths known as alpha, beta, theta, and delta to signify the majority of the EEG used in clinical practice.[?] The table below shows the EEG frequency bands :

Band	Frequency (Hz)
Delta δ	< 4
Theta θ	4 - 8
Alpha α	8 - 16
Beta β	> 16

Table 1: EEG frequency bands[?]

The practice of using only whole numbers in the definitions comes from practical considerations in the days when only whole cycles could be counted on paper records. This leads to gaps in the definitions. The theoretical definitions have always been more carefully defined to include all frequencies. Unfortunately there is no agreement in standard reference works on what these ranges should be – values for the upper end of alpha and lower end of beta include 12, 13, 14 and 15. If the threshold is taken as 14 Hz, then the slowest beta wave has about the same duration as the longest spike (70 ms), which makes this the most useful value.[?]

Band	Frequency (Hz)
Delta δ	0.25 - 4
Theta θ	4 - 8
Alpha α	8 - 12
Sigma σ	12 - 16
Beta β	16 - 40
Gamma* γ	40 - 100

Table 2: Another divisions for EEG frequency bands[?][?]*

Delta Waves is the frequency range up to 4 Hz. It tends to be the highest in amplitude and the slowest waves. It is seen normally in adults in slow-wave sleep. It is also seen normally in babies. It may occur focally with subcortical lesions and in general distribution with diffuse lesions, metabolic encephalopathy hydrocephalus or deep midline lesions. It is usually most prominent frontally in adults (e.g. FIRDA – frontal intermittent rhythmic delta) and posteriorly in children (e.g. OIRDA – occipital intermittent rhythmic delta).[?]

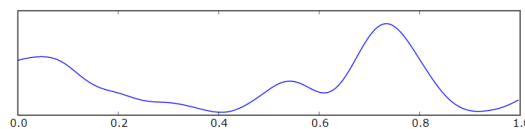


Figure 9: Delta wave

Theta is the frequency range from 4 Hz to 7 Hz. Theta is seen normally in young children. It may be seen in drowsiness or arousal in older children and adults; it can also be seen in meditation.[80] Excess theta for age represents abnormal activity. It can be seen as a focal disturbance in focal subcortical lesions; it can be seen in generalized distribution in diffuse disorder or metabolic encephalopathy or deep midline disorders or some instances of hydrocephalus. On the contrary this range has been associated with reports of relaxed, meditative, and creative states.[?]

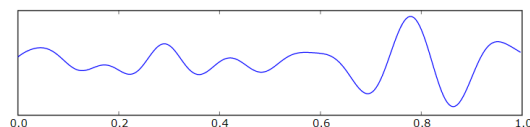


Figure 10: Theta wave

Alpha is the frequency range from 7 Hz to 13 Hz.[81] Hans Berger named the first rhythmic EEG activity he observed the "alpha wave". This was the "posterior basic rhythm" (also called the "posterior dominant rhythm" or the "posterior alpha rhythm"), seen in the posterior regions of the head on both sides, higher in amplitude on the dominant side. It emerges with closing of the eyes and with relaxation, and attenuates with eye opening or mental exertion. The posterior basic rhythm is actually slower than 8 Hz in young children (therefore technically in the theta range).[?]

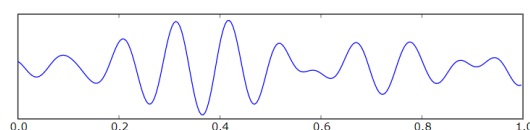


Figure 11: Alpha wave

Beta is the frequency range from 14 Hz to about 30 Hz. It is seen usually on both sides in symmetrical distribution and is most evident frontally. Beta activity is closely linked to motor behavior and is generally attenuated during active movements.[84] Low-amplitude beta with multiple and varying frequencies is often associated with active, busy or anxious thinking and active concentration. Rhythmic beta with a dominant set of frequencies is associated with various pathologies, such as Dup15q syndrome, and drug effects, especially benzodiazepines. It may be absent or reduced in areas of cortical damage. It is the dominant rhythm in patients who are alert or anxious or who have their eyes open.[?]

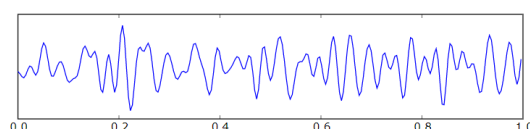


Figure 12: Beta wave

Gamma is the frequency range approximately 30–100 Hz. Gamma rhythms are thought to represent binding of different populations of neurons together into a network for the purpose of carrying out a certain cognitive or motor function.[?]

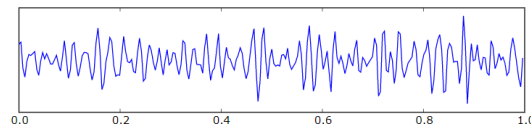


Figure 13: Gamma wave

Therefore, the maximum frequency of the EEG frequency bands is 100 Hz which is the maximum of gamma range. So the sampling frequency must satisfy the following condition :

$$f_s > 2 \times 100 \implies f_s > 200Hz$$

Actually for some Practical reasons the minimum acceptable sampling rate is about 2.5 times greater than the highest frequency of interest but most digital EEG systems will sample at 240 Hz.[?]

۴.۲ محاسبه فرکانس نمونه برداری

با استفاده از سطر اول دیتای مورد نظر که بیانگر زمان می باشد به راحتی می توان به بدست آورد. برای انکار ابتدا داده را لود کرده و با استخراج سطر اول آن، با تقسیم تعداد نمونه ها بر مدت زمان نمونه گیری، فرکانس نموداری بدست می آید :

```
load SubjectData1.mat;

init_signal = train;

init_time = init_signal(1,:);

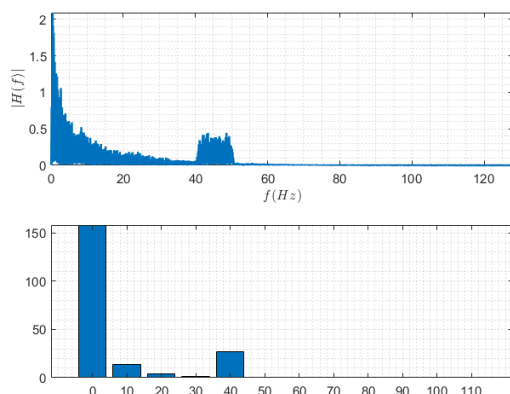
Fs = length(init_time)/(init_time(end)-init_time(1));
```

که این مقدار برابر ۲۵۶ هرتز بدست آمده است.

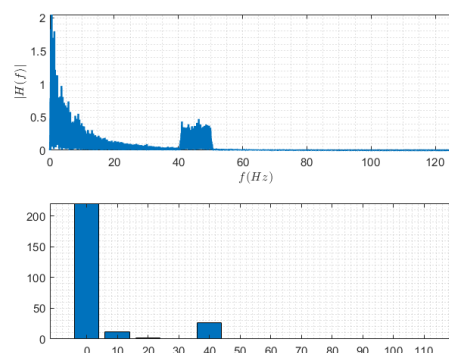
۵.۲ پیدا کردن فرکانس قطع سیگنال

برای فیلتر کردن سیگنال ابتدا نیاز است فرکانس قطع فیلتر را تعیین کنیم. برای اینکار با استفاده از تابعی که در قسمت قبل زده‌ایم طیف فرکانسی کانال‌های مختلف را رسم می‌کنیم. برای نمونه طیف فرکانسی دو کانال در ادامه آمده است. همچنین ملاک دیگری برای مشخص کردن فرکانس قطع فیلتر انرژی سیگنال است. بدین منظور در زیر نمودار طیف فرکانسی هر سیگنال، زیر انرژی آن آن باند فرکانسی از سیگنال نیز مشخص شده است. برای رسم انرژی سیگنال از تابع زیر که تعریف کرده‌ایم استفاده شده است:

```
function plotEnergyBands(input_signal, Fs, window, lastFreq);
```



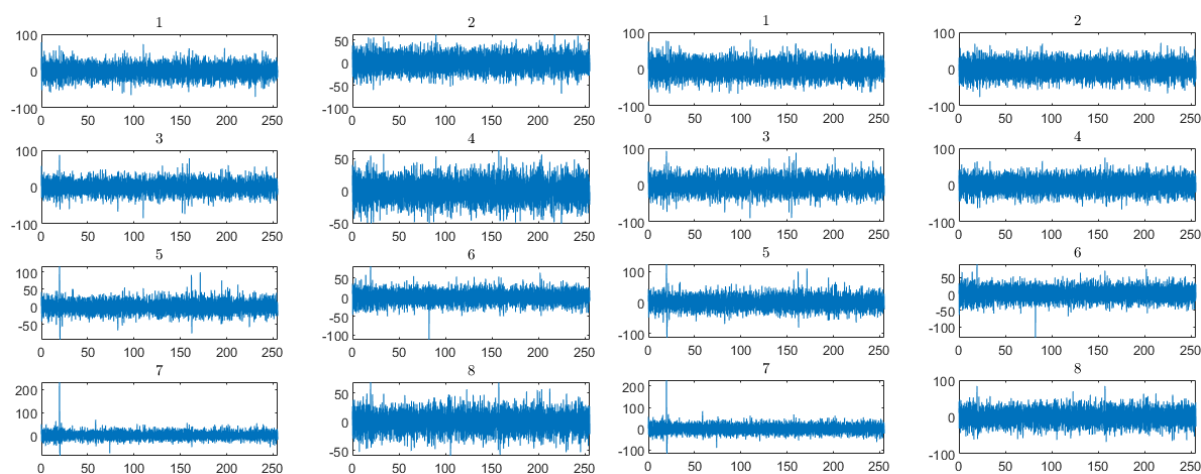
شکل ۱۵



شکل ۱۴

همانطور که در تصاویر فوق مشخص است تجمع انرژی سیگنال در فرکانس‌های پایین می‌باشد. همچنین یک نویز قوی نیز در بازه فرکانسی ۴۰ تا ۵۰ هرتز مشاهده می‌شود. بنابراین یک فرکانس قطع بالا باید ۴۰ هرتز در نظر گرفته شود تا نویز از سیگنال حذف شود. نکته‌ای که قبل از فیلتر کردن باید انجام دهیم حذف مقدار DC سیگنال است زیرا این فرکانس اطلاعاتی را منتقل نمی‌کند. برای این منظور چند کار انجام می‌دهیم. اول میانگین سیگنال را از سیگنال کم می‌کنیم تا مقدار DC از بین برود. همچنین بعد از حذف این مقدار تابع detrend را نیز روی سیگنال اجرا می‌کنیم و در نهایت یک فرکانس قطع پایین حدود نیم تا ۱ هرتز نیز برای فیلتر در نظر می‌گیریم.

بنابراین در مجموع یک فیلتر bandPass از فرکانس ۰.۵ تا ۴۰ هرتز نیاز خواهیم داشت که می‌توان به راحتی از قسمت filterDesigner متلب آن را بسازیم یا با دستور bandpass متلب از این فیلتر استفاده کرد. خروجی کانال‌های دیتای شماره ۱ بعد از اعمال عملیات گفته شده و فیلتر به صورت زیر می‌باشد:

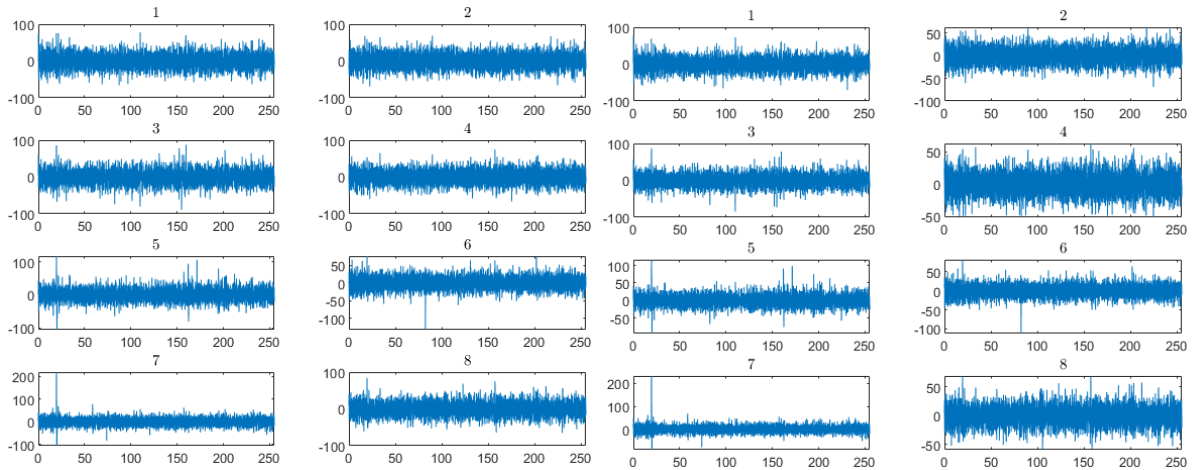


شکل ۱۶: سیگنال‌های اندازه‌گیری شده

شکل ۱۷: بعد از اعمال فیلتر

۶.۲ کاهش فرکانس نمونه برداری

اکنون می‌خواهیم فرکانس نمونه برداری را کاهش دهیم. این مقدار که در قسمت ۴.۲ برابر ۲۵۶ هرتز بدست آمد اکنون بنابر قضیه نایکوویست می‌تواند کاهش یابد. زیرا با اعمال فیلتری که تنها تا فرکانس ۴۰ هرتز را عبور می‌دهد می‌توانیم حداقل فرکانس نمونه‌برداری یعنی ۸۰ هرتز را انتخاب کنیم. در مجموع با توجه به فرکانس ۲۵۶ هرتز اولیه مقدار ۱۲۸ هرتز را برای فرکانس نمونه برداری جدید در نظر می‌گیریم. بنابراین می‌توانیم از هر دو نمونه یک نمونه را برداشته و دیگری را حذف کنیم. تصاویر ۱۶ و ۱۷ نتایج کاهش فرکانس نمونه برداری را نشان می‌دهد. لازم به ذکر است در شبیه سازی برای شفافیت بیشتر در این بخش از فیلتر ۰.۵ تا ۳۰ هرتز استفاده شده است و در نتیجه فرکانس نمونه برداری تا ۶۴ هرتز پایین آمده است. با این حال اثر این عملیات در خروجی کد مشهودتر می باشد.



شکل ۱۸: سیگنال‌های فیلتر شده

شکل ۱۹: بعد از کاهش فرکانس نمونه برداری

کاهش نمونه برداری در تابع زیر انجام می‌شود :

```
function new_signal = reduceSampleRate(signal, Fs, new_Fs)
    if nargin < 3
        fratio = Fs;
    else
        fratio = floor(Fs/new_Fs);
    end
    new_signal = signal(1:fratio:end);
end
```

epoching ۷.۲

اکنون نوبت epoch کردن داده‌ها می‌باشد. این تابع به صورت زیر می‌باشد :

```
function E = epoch(signals,target,Fs)
    E = zeros(size(signals,1),Fs,sum(target));
    before = round(Fs * 0.2);
    after = round(Fs * 0.8);
    indexes = find(target);
    for i=1:size(signals,1)
        for j=1:sum(target)
            index = indexes(j);
            if index-before > 0 && index+after-1 <= length(signals(i,:))
                E(i,:,j) = signals(i,index-before:index+after-1);
            elseif index-before <= 0
                E(i,before-index+1:end,j) = signals(i,1:index+after-1);
            elseif index+after-1 > length(signals(i,:))
                E(i,end-before-length(signals(i,:))+index:end,j) =
                    signals(i,index-before:end);
            end
        end
    end
end
```

این تابع با ورودی گرفتن ماتریس حاوی سیگنال‌های کانال‌ها و همچنین زمان‌های رخ دادن تحریک، ۲۰۰ میلی ثانیه قبل و ۸۰۰ میلی ثانیه بعد از تحریک را جدا کرده و در یک trial ذخیره می‌کند. در پایان با داشتن ۸ کانال و تعداد N تحریک و با توجه به فرکانس نمونه برداری که برابر ۶۴ هرتز می‌باشد (در نتیجه ۱ ثانیه برابر ۶۴ درایه خواهد بود)، در خروجی یک ماتریس $8 \times Fs(64) \times N$ خواهیم داشت.

* از آنجایی که در epoch کردن تعداد درایه‌های هر تریال دارای اهمیت است اگر فیلتر کردن را بعد از epoch کردن انجام دهیم آنگاه طول هر تریال کوچک‌تر شده و یک دسته از اطلاعات مهم از دست رفته‌اند.

* همچنین از آنجایی که در epoch کردن لازم است چند ثانیه قبل و چند ثانیه بعد از تحریک را داشته باشیم، بنابراین لازم است در ابتدا قبل از تحریک اول مدت زمانی سیگنال داشته باشیم و همچنین بعد از تحریک آخر نیز مدت زمانی دیرتر سیگنال را قطع کنیم.

۳ خوشه بندی بر مبنای همبستگی

۱.۳ روابط همبستگی

به شکل زیر عمل می کنیم:

$$\int_{-\infty}^{\infty} (Y(t) - cX(t))^2 dt = \int_{-\infty}^{\infty} Y(t)^2 dt + c^2 \int_{-\infty}^{\infty} X(t)^2 dt - 2c \int_{-\infty}^{\infty} (Y(t)X(t)) dt \geq 0 \quad (۱)$$

$$c = \frac{\int_{-\infty}^{\infty} (X(t)Y(t)) dt}{\int_{-\infty}^{\infty} X(t)^2 dt} \quad (۲)$$

$$\int_{-\infty}^{\infty} Y(t)^2 dt - \frac{(\int_{-\infty}^{\infty} (Y(t)X(t)) dt)^2}{\int_{-\infty}^{\infty} X(t)^2 dt} \geq 0 \quad (۳)$$

$$\frac{(\int_{-\infty}^{\infty} (Y(t)X(t)) dt)^2}{\int_{-\infty}^{\infty} X(t)^2 dt \int_{-\infty}^{\infty} Y(t)^2 dt} \leq 1 \quad (۴)$$

$$-1 \leq \frac{\int_{-\infty}^{\infty} (Y(t)X(t)) dt}{\sqrt{\int_{-\infty}^{\infty} X(t)^2 dt \int_{-\infty}^{\infty} Y(t)^2 dt}} \leq 1 \quad (۵)$$

بر عکس با داشتن رابطه (۵) رابطه زیر نتیجه می شود.

$$\int_{-\infty}^{\infty} (Y(t) - cX(t))^2 dt = 0 \rightarrow a.s \rightarrow Y(t) = cX(t) \quad (۶)$$

هر چه مقدار همبستگی میان دو سیگنال به یک نزدیک تر باشد یعنی رابطه آن ها به خطی نزدیک تر است و این یعنی محتوای فرکانسی دو سیگنال با هم نزدیکی بیشتری دارند.

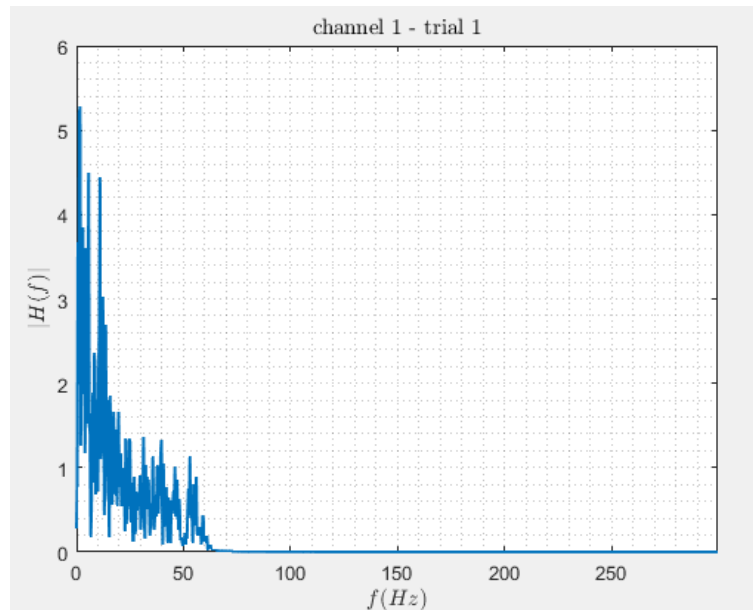
۲.۳ اعمال الگوریتم خوشه بندی بر روی دیتای ۶۳ کاناله

به کمک تابعی که در بخش یک پیاده کردیم طیف فرکانسی trial1 از channel1 را به عنوان نمونه بررسی می کنیم. نتایج به شرح زیر است:

```
load 64channeldata.mat;
channel1_trial1 = data(1, :, 1);

HalfBandFFT(channel1_trial1, 600, 0, 'channel 1 - trial 1');
```

شکل حاصل به صورت زیر است. همان طور که مشخص است نیازی به فیلتر کردن آن نیست و دیتا فیلتر شده است:



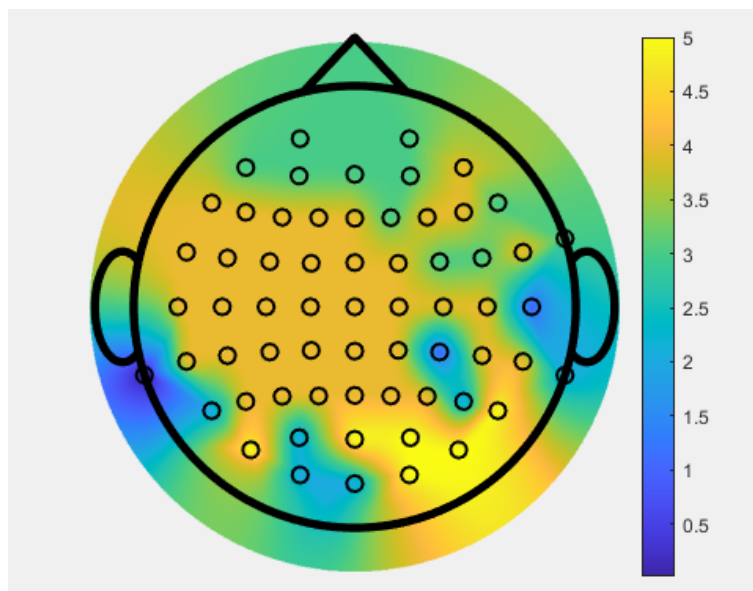
شکل ۲۰: طیف فرکانسی تریال یک از کانال یک

حال با مشاهده بیشینه محتوای فرکانسی می توانیم فرکانس نمونه برداری را تا 120Hz کاهش دهیم. این کار را با تابع reduceSampleRate که پیاده کرده ایم انجام می دهیم. در آخر همه تریال های هر کانال را پشت سر هم قرار می دهیم و ماتریس همبستگی را از روی دیتای بدست آمده محاسبه می کنیم.

```
Arranged = reshape(data, [size(data,1), size(data,2)*size(data,3)]);
reduced = reduceSampleRate(Arranged, 600, 120);
corr = corrcoef(reduced');
distanceMatrix = 1 - corr;
```

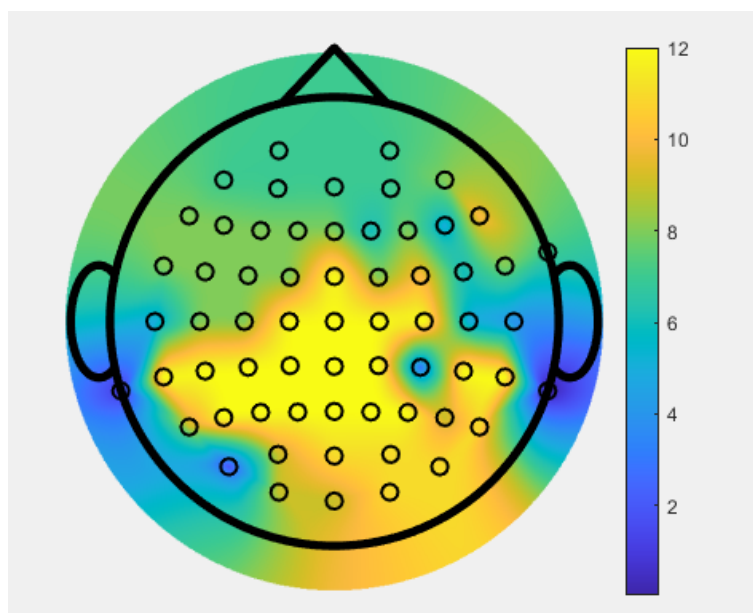
معیار فاصله میان کانال ها را همان طور که در قطعه کد بالا مشخص یک منهی همبستگی می گیریم زیرا هر چه همبستگی به یک نزدیک تر باشد شباهت دو سیگنال بیشتر است. لازم به ذکر است که در این بخش دو الگوریتم WPGMA و UPGMA را پیاده سازی کرده ایم فاصله خوشه ها از خوشه های merge شده نیز برای الگوریتم WPGMA برابر میانگین فاصله از دو خوشه است. اما برای الگوریتم UPGMA این مقدار به صورت میانگین وزن دار محاسبه می شود. (وزن هر خوشه برابر تعداد اعضای آن است)

می‌توانیم با در نظر گرفتن یک مقدار threshold برای مینیمم فاصله میان خوشه‌ها معیار مناسبی برای خوشه‌بندی ارائه دهیم. این مقدار تحت آرگومان DistanceMeasure به تابع خوشه‌بندی داده می‌شود. طبق شهودی که داریم دو الکتروود جلوی سر مربوط به پلک زدن چشم هستند و در گروه مجزایی قرار می‌گیرند همین‌طور الکتروودهای نزدیک گوش باید در یک خوشه‌بندی قرار بگیرند اکثر الکتروودهای پشت سر نیز در یک خوشه و الکتروودهای وسط سر در یک خوشه قرار می‌گیرند. نتیجه خوشه‌بندی با ۶ خوشه برای الگوریتم WPGMA به صورت زیر است:



شکل ۲۱: نتیجه خوشه‌بندی با ۶ خوشه به وسیله الگوریتم WPGMA

خروجی مربوط به الگوریتم UPGMA و ۱۳ خوشه به صورت زیر است:

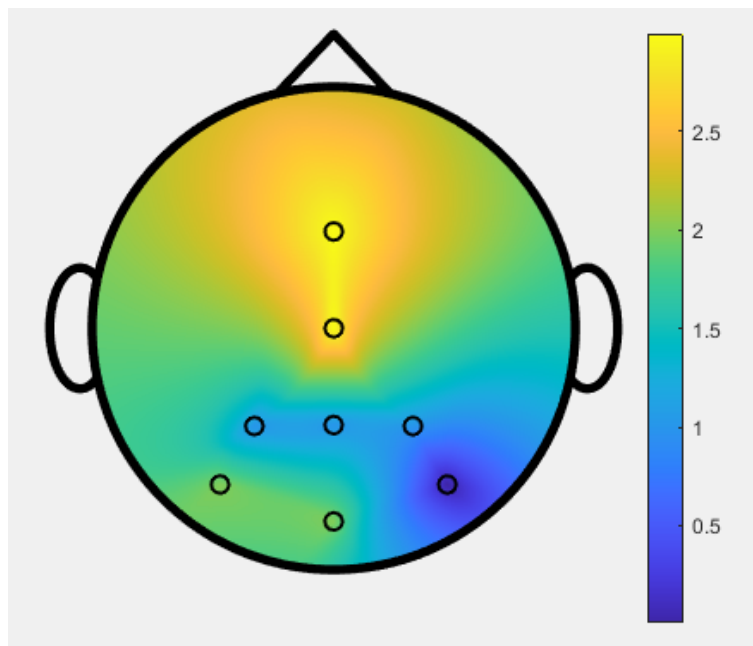


شکل ۲۲: نتیجه خوشه‌بندی با ۱۳ خوشه به وسیله الگوریتم UPGMA

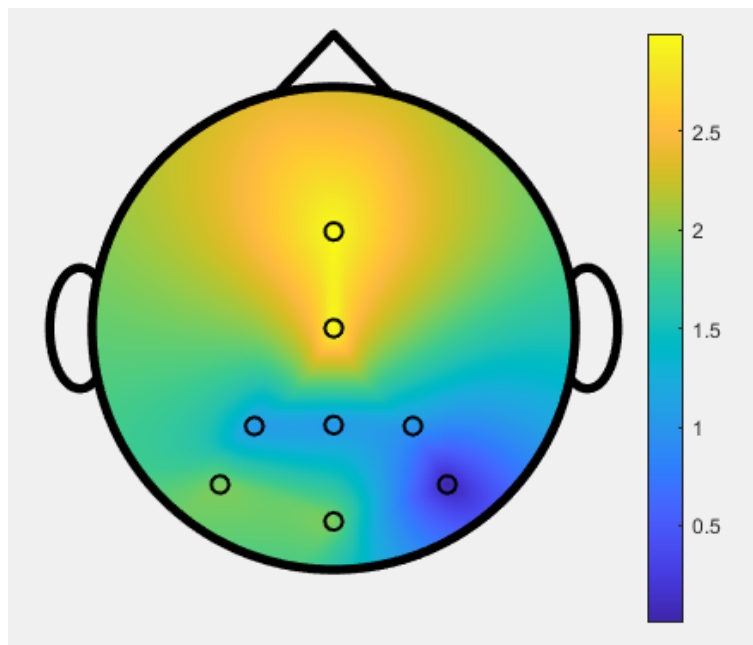
نتایج با شهود فوق برابری می‌کند.

۳.۳ اعمال الگوریتم خوشه بندی روی دیتای ۸ کاناله

همان مراحل بالا را برای دیتای ۸ کاناله انجام می دهیم و مکان الکتروود ها را حدس می زنیم نتایج مشابه نتایج دیتای ۶۴ کاناله به شکل زیر است:



شکل ۲۳: نتیجه خوشه بندی با ۴ خوشه به وسیله الگوریتم WPGMA



شکل ۲۴: نتیجه خوشه بندی با ۴ خوشه به وسیله الگوریتم UPGMA

همان طور که در دو شکل بالا مشخص است اولاً نتایج دو الگوریتم برای این داده یکسان است ثانیاً با نتایج داده ۶۴ کاناله شباهت زیادی دارد و مطابق با شهود بیان شده است.

۴ طراحی فیلتر

۱.۴ بررسی فیلتر با فاز خطی

در این بخش دو فیلتر را مورد بررسی قرار می دهیم، که اندازه آن ها ثابت، اما یکی با فاز غیر خطی و دیگری با فاز خطی. ابتدا فیلتر با فاز غیر خطی را مورد بررسی قرار می دهیم.

$$\Phi(\omega) = \frac{-\pi}{3} \text{sgn}(\omega) \quad (۷)$$

تبدیل فوریه فیلتر به صورت زیر است:

$$H(\omega) = k e^{j\Phi(\omega)} \quad (۸)$$

و ورودی به صورت زیر است:

$$x(t) = \cos(\omega_0 t) + \cos(2\omega_0 t) \quad (۹)$$

در نتیجه تبدیل فوریه سیگنال ورودی به شکل زیر می شود:

$$X(\omega) = \frac{1}{2}(\delta(\omega - \omega_0) + \delta(\omega + \omega_0) + \delta(2\omega - \omega_0) + \delta(2\omega + \omega_0)) \quad (۱۰)$$

با ضرب کردن تبدیل فوریه فیلتر و ورودی و برگرداندن خروجی به حوزه زمان خروجی به صورت زیر می شود:

$$y(t) = k \cos(\omega_0 t - \frac{\pi}{3}) + k \cos(2\omega_0 t - \frac{\pi}{3}) \quad (۱۱)$$

همان طور که از رابطه بالا مشخص است، سیگنال با فرکانس ω_0 به اندازه $\frac{\pi}{3\omega_0}$ و سیگنال با فرکانس $2\omega_0$ به اندازه $\frac{\pi}{6\omega_0}$ شیفت یافته است. هر دو سیگنال ورودی به یک شکل شیفت نخورده اند و در واقع این مسئله باعث اعوجاج می شود. حال به بررسی فیلتر با فاز خطی می پردازیم.

$$\Phi(\omega) = \frac{-\pi}{3} \omega \quad (۱۲)$$

خروجی به صورت زیر می شود:

$$y(t) = k \cos(\omega_0(t - \frac{\pi}{3})) + k \cos(2\omega_0(t - \frac{\pi}{3})) \quad (۱۳)$$

از خروجی بالا در می یابیم که هر کدام از ورودی ها به اندازه $\frac{\pi}{3}$ شیفت یافته اند. و این اثر فاز خطی است. مشخص است که گزاره ذکر شده در توضیحات پروژه در مورد فیلتر اولی که فاز غیر خطی دارد صادق نیست اما در فیلتر دوم که فاز آن خطی است $groupdelay$ برابر $\frac{\pi}{3}$ است و گزاره فوق صادق است.

۲.۴ اثبات رابطه تاخیر گروه

رابطه تبدیل فوریه به شکل زیر است:

$$H(\omega) = A(\omega) e^{j\Phi(\omega)} \quad (۱۴)$$

در نتیجه مشتق تبدیل فوریه به شکل زیر می شود:

$$\frac{d}{d\omega} H(\omega) = \frac{d}{d\omega} A(\omega) e^{j\Phi(\omega)} + j A(\omega) e^{j\Phi(\omega)} \cdot \frac{d}{d\omega} \Phi(\omega) \quad (۱۵)$$

حال از رابطه $groupdelay$ داریم:

$$gd(\omega) = \text{Re}\left\{ \frac{j \frac{d}{d\omega} H(\omega)}{H(\omega)} \right\} \quad (۱۶)$$

در نتیجه:

$$gd(\omega) = -\frac{d}{d\omega} \Phi(\omega) \quad (۱۷)$$

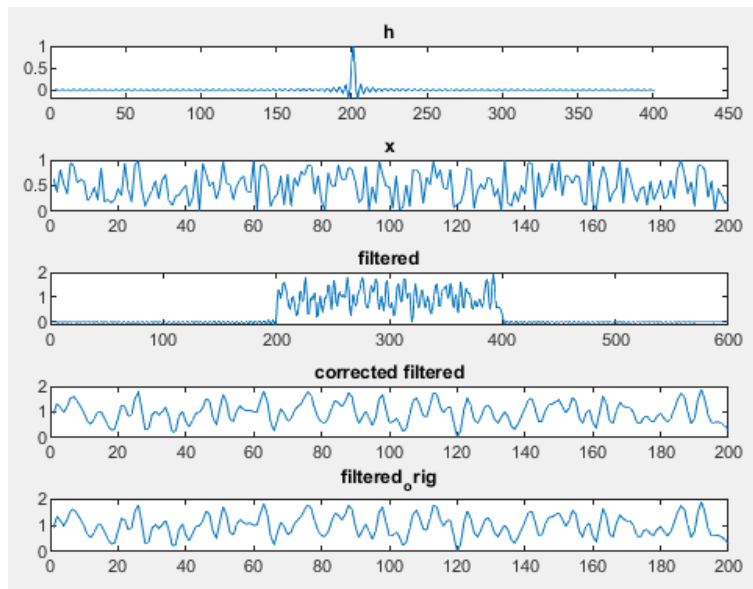
۳.۴ تست عملکرد تابع پیاده سازی شده

در این تست فیلتر پایین گذر را روی ورودی دلخواه اعمال می کنیم و تاخیر گروه را محاسبه می کنیم. همچنین این مقادیر را با توابع آماده متلب نیز محاسبه می کنیم.

```
N = 200;
M = 200;
Fs = 100;
x = rand(1,N);
h = sinc((-M:M) / 2);
H = fft(h, 2*M+1);

gd = groupdelay(h, 2*M+1);
gd_original = grpdelay(h, 1, 2*M+1);
filtered = conv(h,x);
corrected_filtered = zphasefilter(h, x);
filtered_orig = conv(x, h, 'same');
```

شکل نمودار ها به صورت زیر است:



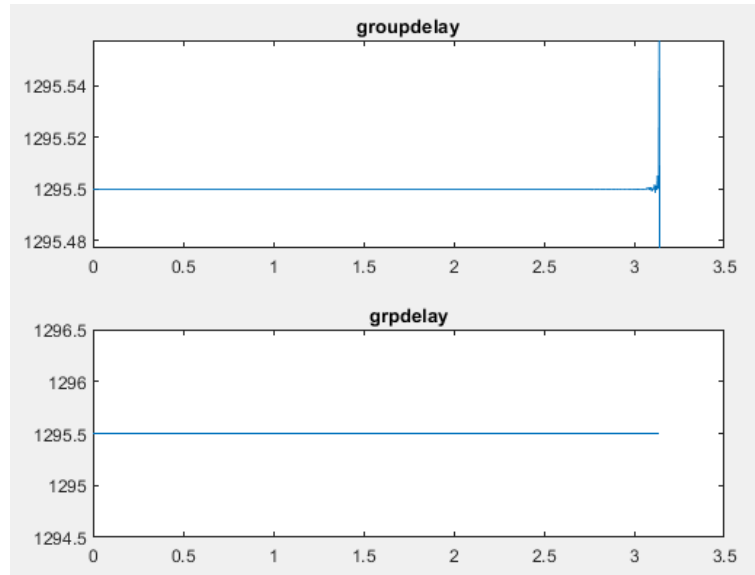
شکل ۲۵: تست یک

نمودار سوم خروجی فیلتر شده پیش از اعمال تاخیر گروه است و نمودار چهارم پس از اعمال تاخیر گروه. نمودار پنجم نیز خروجی فیلتر شده به کمک تابع آماده متلب است (خود به خود تاخیر گروه را حذف می کند). همچنین مقادیر تاخیر گروه به کمک تابع *groupdelay* و *grpdelay* برابر ۲۰۰ بدست آمده است. (برای فیلتر های باز فاز خطی این مقدار ثابت است و تابع فرکانس نیست)

تعریف *DFT* به این صورت است که از تبدیل فوریه تابع نمونه برداری می کند. در نتیجه اگر N بیشتر شود نمونه های بیشتری از تبدیل فوریه فیلتر در بازه صفر تا 2π گرفته می شود و دقت محاسبات افزایش می یابد.

۴.۴ اعمال تابع تاخیر گروه بر روی فیلتر های استفاده شده در طول پروژه

این تابع را روی فیلتر پایین گذر استفاده شده در بخش های قبل اعمال می کنیم. نتیجه به شکل زیر است. دلیل ناپیوستگی های به وجود آمده در خروجی `groupdelay` این است که فیلتر ایده آل نیست و فاز آن کاملاً خطی نیست. ولی در اکثر نقاط این مقدار با مقدار تابع آماده متلب برابری می کند.



شکل ۲۶: تست دو

در مورد سوال فیلتر علی و حقیقی با فاز صفر نیز می توان گفت تابع $k\delta[n]$ دارای ویژگی مورد نظر هست.

۵ شناسایی کلمات

۱.۵ الگوریتم

در این قسمت ۸ داده از ۸ نفر داریم که برخی آزمایش اول را انجام داده‌اند و برخی آزمایش دوم را انجام داده‌اند. مرحله‌ای که در بخش ۲ انجام داده‌ایم را به عنوان مراحل پیش‌پردازش و آماده‌سازی دیتاهای خام اندازه‌گیری شده، انجام می‌دهیم تا دیتای مناسب برای learn را آماده کنیم. این مراحل به صورت زیر هستند :

۱. مرحله اول - فیلتر کردن :

همانطور که در بخش ۲ نیز توضیح داده شده است، قبل از فیلتر کردن مقدار DC سیگنال را حذف می‌کنیم. برای اینکار یکبار میانگین سیگنال را از سیگنال کم کردم و سپس از تابع detrend متلب عبور می‌دهیم. سپس سیگنال را از یک فیلتر میانگذر عبور می‌دهیم تا نویزها را از سیگنال جدا کنیم. این موارد در تابع زیر انجام می‌شود :

```
function new_Subject = bandpassStruct(Subject,fpass);
```

تابع فوق فیلتر میانگذر را بر روی تمام کانال‌های دو ماتریس train و test اعمال می‌کند.

۲. مرحله دوم - کاهش فرکانس نمونه‌برداری :

همانطور که در قسمت ۲ طیف فرکانسی سیگنال‌ها را مشاهده کردیم و از آنجایی که فرکانس قطع فیلتر میان‌گذر قسمت قبل را 0.5 و 40 هرتز انتخاب کرده‌ایم بنابراین می‌توانیم فرکانس نمونه‌برداری را تا ۱۲۸ هرتز پایین بیاوریم. کاهش نمونه‌برداری برای تمام سیگنال‌های دو ماتریس train و test در تابع زیر انجام می‌شود :

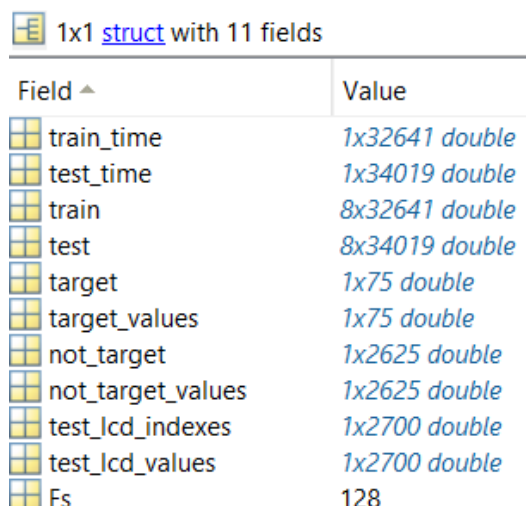
```
function new_Subject = reduceSampleRateStruct(Subject,new_Fs);
```

۳. مرحله سوم - جدا سازی targets و not_targets :

همانطور که در دستور کار آمده است در این قسمت تابع زیر را پیاده‌سازی می‌کنیم :

```
function new_Subject = IndexExtraction(Subject);
```

در تابع فوق تمام اجزای استراکت تفکیک می‌شوند و استراکت شامل ویژگی‌ها به صورت تفکیک شده در خروجی ظاهر می‌شود. خروجی این تابع به صورت زیر می‌باشد :



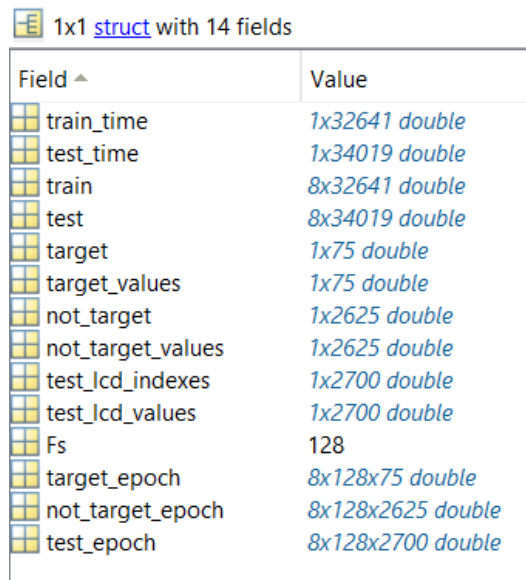
Field ^	Value
train_time	1x32641 double
test_time	1x34019 double
train	8x32641 double
test	8x34019 double
target	1x75 double
target_values	1x75 double
not_target	1x2625 double
not_target_values	1x2625 double
test_lcd_indexes	1x2700 double
test_lcd_values	1x2700 double
Fs	128

شکل ۲۷: خروجی IndexExtraction

۴. مرحله چهارم - epoching :

با توجه به تابع epoch که در بخش ۲ زده شد، اکنون با بهبود آن این فرایند را روی استراکت Subject انجام می‌دهیم و چند epoch به تفکیک برای target، not_target، مجموع target و not_target و در نهایت برای test تشکیل می‌دهیم. این تابع به صورت زیر می‌باشد و خروجی آن در تصویر ۲۸ آمده است :

```
function new_Subject = epochStruct(Subject);
```



Field ^	Value
train_time	1x32641 double
test_time	1x34019 double
train	8x32641 double
test	8x34019 double
target	1x75 double
target_values	1x75 double
not_target	1x2625 double
not_target_values	1x2625 double
test_lcd_indexes	1x2700 double
test_lcd_values	1x2700 double
Fs	128
target_epoch	8x128x75 double
not_target_epoch	8x128x2625 double
test_epoch	8x128x2700 double

شکل ۲۸: خروجی epochStruct

لازم به ذکر است از آنجایی که تمام مراحل فوق باید به ترتیب صورت گیرند و همچنین بهبود کد تمام مراحل ذکر شده در تابع زیر انجام می‌شوند :

```
function new_Subject = preprocessing(subjectName, fpass, Fs_new)
    Subject = load(subjectName);
    Subject = bandpassStruct(Subject, fpass); % bandpass-filtering
    Subject = reduceSampleRateStruct(Subject, Fs_new); % down-sampling
    Subject = IndexExtraction(Subject); % index-extraction
    new_Subject = epochStruct(Subject); % epoching
end
```

۵. مرحله پنجم - ایجاد Train_Features :

اکنون تنها یک مرحله تا قبل از learn باقی می‌ماند آن هم ایجاد فرم مناسب برای توابع learn می‌باشد. برای این قسمت هر درایه را متناسب با یک ویژگی در نظر می‌گیریم. برای این بخش مدل‌های متفاوتی در نظر گرفته شد تا نتیجه نهایی بهبود پیدا کند. این مدل‌ها را در پایان این بخش توضیح خواهیم داد و در این قسمت تنها به توصیف فرم مناسب برای learning می‌پردازیم.

در دستورکار دو تابع برای انجام یادگیری معرفی شده‌است : fitcdiscr و fitcsvm. هر کدام از این دو توابع در حالت کلی دو ورودی دریافت می‌کنند. ورودی اول ماتریسی است که هر سطر آن یک داده برای یادگیری و هر ستون آن بیانگر یک ویژگی می‌باشد. ورودی دوم آن یک بردار که هر درایه آن متناسب با یک سطر ماتریس ورودی اول می‌باشد. در واقع برای هر نمونه ورودی شبکه خروجی آن (که در اینجا target یا not_target بودن است) خروجی را مشخص می‌کند. با اجرای این توابع خروجی یک مدل آموزش دیده می‌باشد که با کمک تابع predict می‌توان خروجی test را مشاهده کرد و کلمه مد نظر فرد را حدس زد.

۲.۵ تلاش‌ها!

مجموعه اندکی از تلاش‌های فراوان برای رسیدن به نتیجه؛
هر ماتریس ورودی برای یادگیری همانطور که گفته شد از یک سری نمونه در سطر و یک سری ویژگی در ستون‌هایش تشکیل شده است. طول هر تریال ۱۲۸ معادل فرکانس نمونه برداری می‌باشد. برای مثال در آزمایش اول ۲۷۰۰ تحریک داریم. بنابراین epoch در ابعاد ۸ در ۱۲۸ در ۲۷۰۰ خواهد بود. در ادامه ویژگی‌ها را برای شفافیت بیشتر برای آزمایش اول تشریح می‌کنیم.

۱.۲.۵ ویژگی شماره ۱

در تلاش اول ۱۲۸ تریال را در بعد ویژگی حفظ کرده و 21600 (8×2700) نمونه را در سطر قرار داده‌ایم.

۲.۲.۵ ویژگی شماره ۲

در تلاش دوم سیگنال هر ۸ کانال که مربوط به هر تحریک هستند را پشت هم در یک بردار چیده‌ایم که در این صورت یک ماتریس ۷۵ در 1024 (8×128) خواهیم داشت.

۳.۲.۵ ویژگی شماره ۳

در تلاش سوم ابتدا ماتریس قسمت قبل را تشکیل داده سپس ۷۵ سطر با که طبق ۵ حرف کلمه ۵ دسته ۱۵ تایی می‌باشد، به ۵ سطر تبدیل می‌کنیم به این صورت که ۱۵ سطری که متناسب با یک نوع تحریک هستند را با یکدیگر میانگین می‌گیریم. بنابراین یک ماتریس ۵ در ۱۰۲۴ تشکیل خواهد شد. سپس ۸ کانال ۱۲۸ تایی که به یک بردار ۱۰۲۴ تایی تبدیل شده‌اند را به حالت اول برمی‌گردانیم و زیر یکدیگر می‌چینیم. بنابراین تعداد ستون‌ها ۱۲۸ و تعداد سطرها ۴۰ (5×8) خواهد شد.

۴.۲.۵ شبکه عصبی!

یک شبکه عصبی سه لایه با یک لایه مخفی برای انحام یادگیری در نظر گرفته شده است. تابع زیر برای این امر پیاده سازی شده است :

```
function [Theta1, Theta2, train_pred, test_pred, Result] = ...  
    NeuralNetworkLearnig(train,y_train,test,test_value,...  
    input_layer_size,hidden_layer_size,num_labels,iter);
```

این تابع با انجام گام به گام یادگیری وزن‌های شبکه عصبی را تشکیل داده و در نهایت خروجی را هم بر روی train و هم روی test گزارش می‌کند. همچنین با توجه به نتایج بدست آمده از test حرف‌های LCD را که target تشخیص داده شده‌اند را برمی‌گرداند. این شبکه از فایل main_NN,m قابل اجرا می‌باشد.
برخلاف تلاش بسیار این تلاش نیز با نتیجه مطلوب همراه نشد!

لازم به ذکر است که تمامی تلاش‌ها با مقادیر مختلف فرکانس و نسبت‌های مختلف target و not_target تست شده‌اند.

خروجی تمام تلاش‌های فوق برای train با موفقیت ۱۰۰ درصد همراه است. دلیل آن هم آموزش شبکه است که با خود train انجام شده است و به شبکه تلاش کرده است تا بر روی داده‌های train, fit شود بنابراین انتظار داریم درصدهای خیلی بالا نزدیک به ۱۰۰ دریافت کنیم.

۵.۲.۵ ویژگی شماره ۲ - نتیجه مطلوب!

این بار با در نظر گرفتن همان ویژگی دوم و با ۵ برابر کردن (۵ بار کپی کردن) target ها نتیجه مطلوب بدست آمد:))

۶.۲.۵ شبکه عصبی نتیجه بخش:

با تغییر وزن بین `target` ها و `not_target` ها به این صورت که داده‌ای `target` را چند بار کپی می‌کنیم، نتیجه مطلوب بدست آمده است.))

* البته نکته مهمی که وجود داشت در ارتباط با سابجکت شماره ۱ بود که بسیاری از تلاش‌ها بر روی آن انجام گرفت و نتیجه نامطلوب بود با اینکه به نظر میرسد داده سابجکت ۱ مناسب نمی‌باشد و همان الگوریتم برای سایر سابجکت‌ها پاسخ مطلوب ایجاد می‌کند.