تکلیف سوم درس پردازش گفتار

دانشکده‌ی مهندسی
کامپیوتر دانشگاه
اصفهان

استاد درس:
دکتر حمیدرضا برادران

دانشجو:
امیررضا صدیقین
۹۹۳۶۱۴۰۲۴

---

۱. ابتدا عبارت "ظاهرش کُنه ضمیرش را لو داد" را با استفاده از wavesurfer با فرکانس نمونه‌برداری ۱۶ کیلوهرتز، بصورت ۱۶ بیتی و مونو در فرمت wav.، با صدای خودتان ضبط کنید.

با استفاده از گزینه‌ی property در wavesurfer تنظیمات خواسته شده را انجام داده‌ایم. سپس عبارت گفته شده ذخیره شده است. این فایل به نام my_voice.wav ذخیره و در کنار کد قرار داده شده است.

پیش از بخش ۲ توابعی برای سهولت کار نوشته شده است که به صورت زیر است:

- تابع read_voice : فایل مربوطه را خوانده و با sample-rate مشخص نمونه برداری شده و در قالب یک آرایه بر می‌گرداند.
- تابع play_voice: در ورودی آرایه‌ای به همراه rate گرفته و آن را پخش می‌کند.
- تابع framing: تابعی که آرایه‌ی سیگنال ورودی را با استفاده از پارامترهای مشخص شده (مثل تکلیف ۲) عملیات فریم‌بندی را انجام می‌دهد.

۲. برای دو فرم واکدار با دو واکه مختلف در فایل ضبط شده، مراحل زیر را انجام دهید. میخواهیم با استفاده از سه الگوریتم مختلف، ضرایب LPC را محاسبه کنیم.

a. ابتدا ضرایب LPC را با الگوریتم معکوس ماتریس اتوکوریلیشن در روش اتوکوریلیشن (ضرایب یول-واکر) بدست آورید. **(راهنمایی: می‌توانید از دستور toeplitz کتابخانه scipy در پایتون، یا دستورات معادل در Matlab استفاده کنید.)**

b. از الگوریتم لوینسون برای محاسبه ضرایب LPC استفاده کنید. **(می‌توانید از دستور levinson در کتابخانه spectrum در پایتون، یا دستورات معادل در Matlab کمک بگیرید.)**

c. از دستورهای built-in برای بدست آورد ضرایب LPC استفاده کنید **(می‌توانید از دستور lpc در کتابخانه spectrum در پایتون، یا دستور معادل در Matlab استفاده کنید.)**

بخش پیدا کردن دو فریم واکه:

برای پیدا کردن فریم‌ها واکه با مقایسه‌ی انرژی آن‌ها یا متناظر با توان ۲ amplitude آن‌ها می‌توان دو فریم واکه را پیدا کرد. برای این کار تابعی تحت عنوان get_sorted_mean_energy_frames_index نوشته شده است که میانگین توان ۲ مقدار سیگنال (متناظر با انرژی) را برای هر فریم حساب کرده و مرتب می‌کند. ۵۰ فریم با کمترین انرژی محاسبه شده

است و از بین آن‌ها دو فریم واکه استخراج شده است که پس از پخش متوجه شده‌ایم که فریم‌های مربوط به /A/ و /a/ هستند.

بخش پیدا کردن دو فریم واکدار:

برای پیدا کردن فریم‌های واکدار باید F0 (pitch frequency) آن‌ها محاسبه شود. فریم‌ها واکدار مقدار F0 غیر صفر دارند و فریم‌های بی‌واک F0 برابر صفر دارند. (البته باید توجه کرد در کنار هم قرار گرفتن فریم‌ها واکدار و بیواک خطایی رخ دهد پس باید مقدار F0 به نسبت بالایی داشته باشد.) تابع get_mean_pithes برای محاسبه‌ی مقدار میانگین pitch نوشته شده است. فریم‌هایی که مقدار خروجی این تابع برایشان بیشتر صفر بوده چاپ شده و دو تا از بین آن‌ها انتخاب شده است که پس از بخش متوجه شده‌ایم فریم‌های مربوط به /r/ و /n/ هستند. در آخر dictionary برای این فریم‌ها ساخته شده است.

پیش از شروع زیر بخش a، دو تابع کمکی نوشته شده است، که به شرح زیر است:

- تابع get_LPC_coeffience_and_show: که dictionary فریم‌ها را دریافت کرده و برای هر کدام با توجه به تابع (الگوریتم) پیدا کردن ضرایب LPC، این ضرایب را پیدا می‌کند و همگی را بر روی یک نمودار نشان می‌دهد.
- تابع get_toeplitz_matrix: این تابع بردار autocorrolated فریم ورودی را حساب می‌کند و سپس ماتریس toeplitz آن را محاسبه می‌کند.

بخش a (پیدا کردن ضرایب LPC با استفاده از روش معکوس ماتریس اتوکورولیشن)

برای پیدا کردن ضرایب LPC با استفاده از روش معکوس ماتریس اتوکورولیشن تابعی تحت عنوان LPCC_with_toeplitz نوشته شده است که در آن ماتریس toeplitz محاسبه شده و سپس با استفاده از فرمول زیر این ضرایب را بدست می‌آورد.

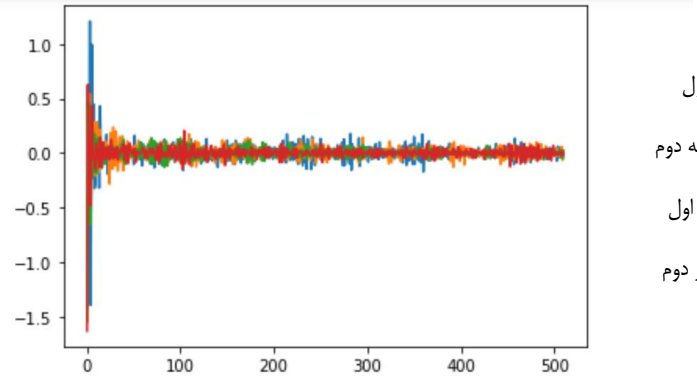
$$\begin{bmatrix} \phi_{ss}(1,1) & \phi_{ss}(1,2) & \dots & \phi_{ss}(1,p) \\ \phi_{ss}(2,1) & \phi_{ss}(2,2) & \dots & \phi_{ss}(2,p) \\ \dots & \dots & \dots & \dots \\ \phi_{ss}(p,1) & \phi_{ss}(p,2) & \dots & \phi_{ss}(p,p) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \dots \\ a_p \end{bmatrix} = \begin{bmatrix} -\phi_{ss}(1,0) \\ -\phi_{ss}(2,0) \\ \dots \\ -\phi_{ss}(p,0) \end{bmatrix}$$

Toeplitz Matrix

ماتریسی که عناصر روی قطرش یکسان هستند

$$\begin{bmatrix} R_{ss}(0) & R_{ss}(1) & \dots & R_{ss}(p-1) \\ R_{ss}(1) & R_{ss}(0) & \dots & R_{ss}(p-2) \\ \dots & \dots & \dots & \dots \\ R_{ss}(p-1) & R_{ss}(p-2) & \dots & R_{ss}(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \dots \\ a_p \end{bmatrix} = \begin{bmatrix} -R_{ss}(1) \\ -R_{ss}(2) \\ \dots \\ -R_{ss}(p) \end{bmatrix}$$

در این روش ماتریس Toeplitz محاسبه شده است. ماتریس سمت چپ و سمت راست محاسبه می‌شود و با استفاده از ضرب ماتریس سمت راست و معکوس ماتریس سمت چپ ضرایب بدست می‌آید.



نمودار آبی: فریم واکه اول

نمودار نارنجی: فریم واکه دوم

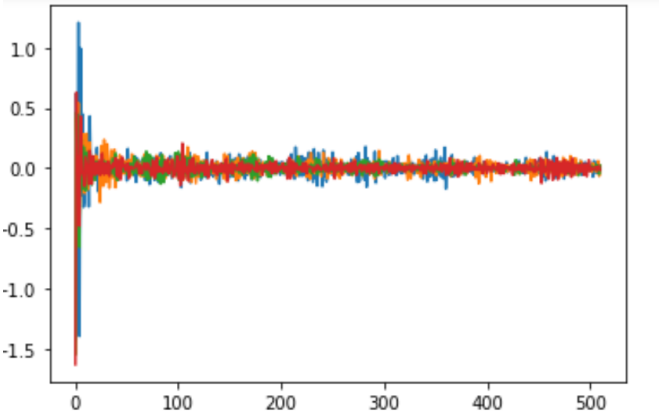
نمودار سبز: فریم واکدار اول

نمودار قرمز: فریم واکدار دوم

نمودار ضرایب LPC برای ۴ فریم به صورت بالا نشان داده شده است. (هر فریم یک رنگ در نمودار)

بخش b (پیدا کردن ضرایب LPC با استفاده از روش لوینسون)

برای پیدا کردن ضرایب LPC با استفاده از روش لوینسون تابعی تحت عنوان LPCC_with_levinson نوشته شده است که در آن ابتدا بردار autocorrelation فریم محاسبه شده و سپس با استفاده از تابع LPCC_with_levinson ضرایب محاسبه می‌شود.



نمودار آبی: فریم واکه اول

نمودار نارنجی: فریم واکه دوم

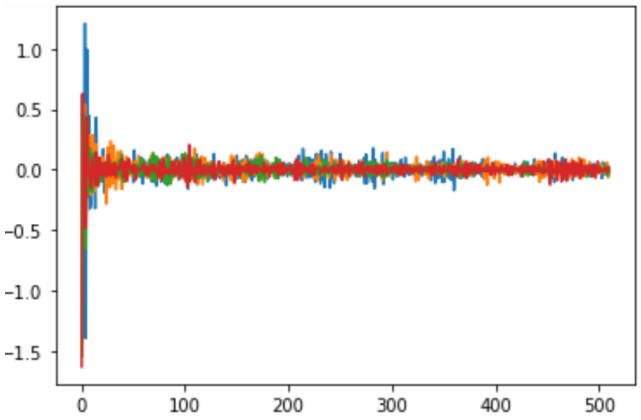
نمودار سبز: فریم واکدار اول

نمودار قرمز: فریم واکدار دوم

نمودار ضرایب LPC برای ۴ فریم به صورت بالا نشان داده شده است. (هر فریم یک رنگ در نمودار)

بخش C (پیدا کردن ضرایب LPC با استفاده از دستورات built-in)

برای پیدا کردن ضرایب LPC با استفاده از روش built-in تابعی تحت عنوان LPCC_with_built_in نوشته شده است که در آن با استفاده از تابع spectrum.lpc این ضرایب پیدا می‌کند.



نمودار آبی: فریم واکه اول

نمودار نارنجی: فریم واکه دوم

نمودار سبز: فریم واکدار اول

نمودار قرمز: فریم واکدار دوم

نمودار ضرایب LPC برای ۴ فریم به صورت بالا نشان داده شده است. (هر فریم یک رنگ در نمودار)

همانطور که مشاهده می‌شود نمودار ضرایب برای هر سه روش به یک صورت است و نتایج یکی است.

۳. مقدار G (گین) در رابطه $H(z) = \frac{G}{A(z)}$ را به کمک lpc بدست آورید.

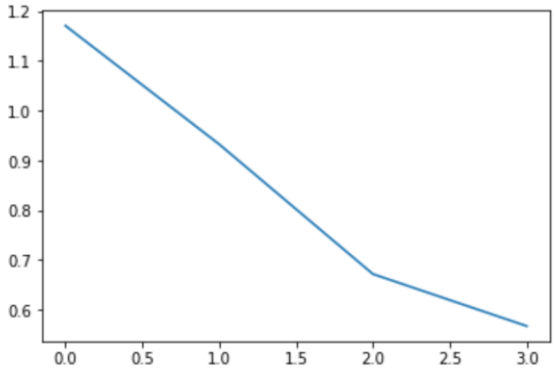
برای پیدا کردن مقادیر G تابعی تحت عنوان get_G نوشته شده است که در آن با استفاده از رابطه‌ی زیر مقدار G محاسبه می‌شود.

$$G^2 = R_{ss}(0) - \sum_{k=1}^{p} a_k R_{ss}(k)$$

مقادیر و نمودار ۴ G بدست آمده به صورت زیر است.

```
vowel_frame1 :  1.1703188939955693
vowel_frame2 :  0.9323481859285366
voiced_frame1 :  0.6717617549806438
voiced_frame2 :  0.5673890825550819

[<matplotlib.lines.Line2D at 0x2191a0834f0>]
```



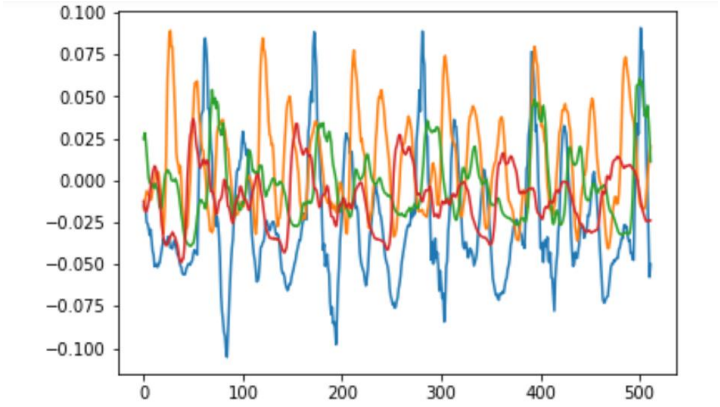
همچنین می‌توانستیم با استفاده از خود تابع spectrum.lpc این مقدار بدست آید.

۴. سیگنال باقی‌مانده یا تحریک را با استفاده از ضرایب LPC بدست آورید.

برای محاسبه‌ی سیگنال باقی‌مانده یا تحریک، تابعی تحت عنوان get_e نوشته شده است که در آن با استفاده از رابطه‌ی زیر این سیگنال را بدست می‌آورد.

$$e[n] = s[n] + \sum_{k=1}^{P} a_k \, s[n-k]$$

نمودار ۴ سیگنال باقی‌مانده برای ۴ فریم ذکر شده به صورت زیر است:



نمودار آبی: فریم واکه اول

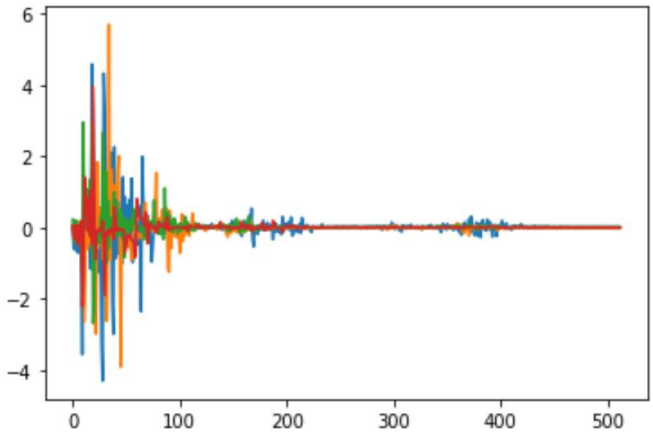
نمودار نارنجی: فریم واکه دوم

نمودار سبز: فریم واکدار اول

نمودار قرمز: فریم واکدار دوم

۵. پاسخ فرکانسی فیلتر مجرای گفتار را ترسیم کنید. **(راهنمایی: می‌توانید از دستور freqz از کتابخانه scipy در پایتون، یا دستورات معادر در Matlab استفاده کنید.)**

برای محاسبه‌ی پاسخ فرکانسی فیلتر مجرای گفتار از تابع freqz استفاده شده است. نمودار H ها به صورت زیر است:



نمودار آبی: فریم واکه اول

نمودار نارنجی: فریم واکه دوم

نمودار سبز: فریم واکدار اول

نمودار قرمز: فریم واکدار دوم

همچنین می‌توان از روش زیر برای محاسبه‌ی Hقدر مطلق و پوش طیف استفاده کرد. (برای بخش ۶ نیز است.)

- محاسبه ضرایب LPC $\{a_k\}$ و گین G برای یک فریم زمان-کوتاه (short-term) سیگنال گفتار
- قرار دادن ضرایب در یک دنباله و اضافه کردن صفر (zero padding) تا حدی که تعداد اعضای دنباله توانی از ۲ شود.

$$a[n] = \{a_0, a_1, a_2, \ldots, a_p, \underbrace{0, 0, \ldots, 0}_{N-(p+1)}\}$$

- گرفتن تبدیل فوریه از دنباله فوق (محاسبه $A[k]$)

$$A[k] = DFT\{a[n]\}$$

- محاسبه دامنه طیف، معکوس کردن آن و ضرب در گین G

$$|H[k]| = G / |A[k]|$$

- اعمال لگاریتم به دامنه طیف بصورت

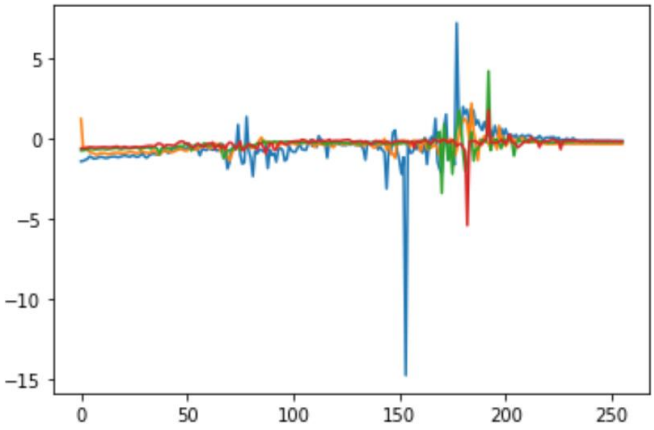
$$20 \log_{10}(\varepsilon + |H[k]|)$$

- یافتن پیک های پوش طیف حاصله به عنوان فرمنت ها

به همین منظور تابعی تحت عنوان get_H نوشته شده است که به همین صورت H را پیدا می‌کند. (برای پوش آن نیز که برای بخش بعد است نیز از همین روش استفاده شده است.)

همچنین می‌توانستیم از تابع scipy.signal.freqz برای همین منظور استفاده کرد که خروجی مشابهی دارد.

نمودار پاسخ فرکانسی برای ۴ فریم به صورت زیر است:



نمودار آبی: فریم واکه اول

نمودار نارنجی: فریم واکه دوم

نمودار سبز: فریم واکدار اول

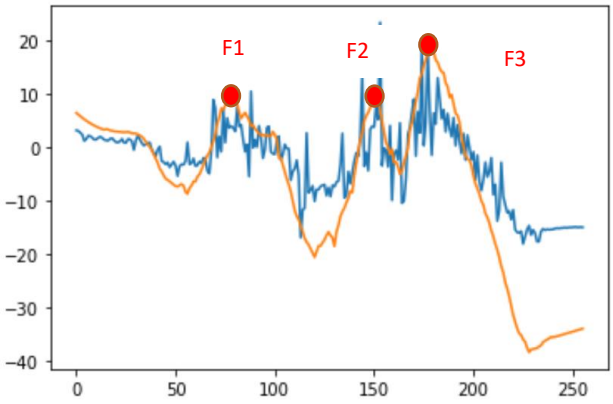
نمودار قرمز: فریم واکدار دوم

۶. پوش طیف را با استفاده از ضرایب LPC ترسیم کنید و با پاسخ فرکانسی قسمت قبل مقایسه کنید. همچنینی بر روی پوش طیف، محل رخداد فرمنت‌ها را نیز مشخص کنید.

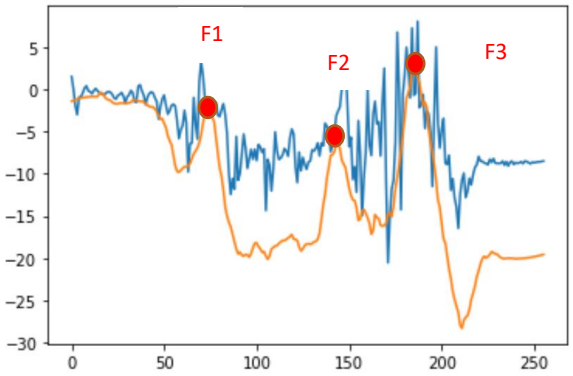
همانطور که در بخش قبل توضیح داده شد. مقدار H برای هر فریم محاسبه شده و نمودار 20.log(epsilone+|H|) کشیده می‌شود. همچنین برای پیدا کردن پوش باید smooth شده‌ی آن را در نظر گرفت برای همین از فیلتر savgol_filter برای smooth کردن استفاده شده است. (برای پیدا کردن فرمنت‌ها فرکانس مهم است و مقدار آن مهم نیست.)

نمودارها به صورت زیر هستند:

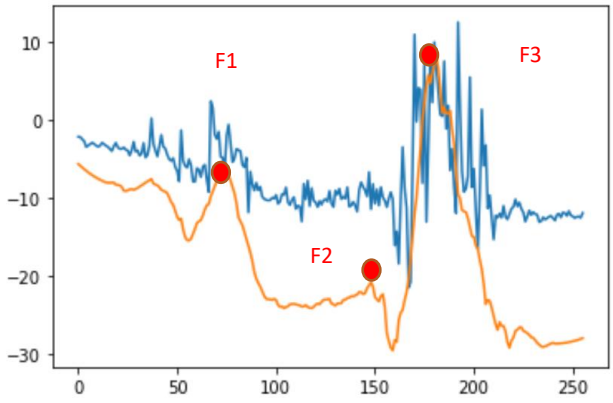
- فریم واکه اول:



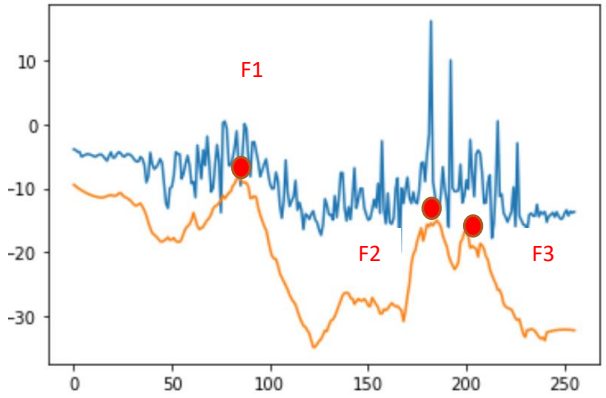
- فریم واکه دوم:



- فریم واکدار اول:



- فریم واکدار دوم:



همانطور که مشاهده می‌شود پوش طیف از روی H بدست می‌آید.

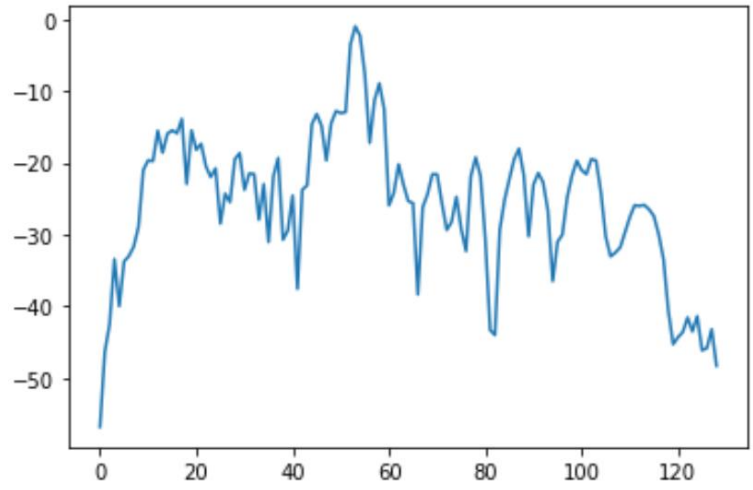
۷. همراه این تمرین، یک فایل گفتاری تلفنی برای شما ارسال شده است. آنرا باز کرده و مراحل پیش‌پردازش را با همان مقادیر گفته شده در تمرین دوم بر روی این فایل انجام دهید.

در این بخش توابع پیش‌پردازش pre_emphasis، framming و windowing مشابه تکلیف قبلی نوشته شده است. همچنین تابع preprocess نوشته شده است که به ترتیب موارد پیش‌پردازش را روی یک فریم با مقادیر ذکر شده انجام می‌دهد.

صوت داده شده خوانده شد و پیش‌پردازش صورت گرفت و فریم‌های پیش‌پردازش شده بدست آمد.

۸. برای یک فریم واکدار، طیف فرکانسی آنرا ترسیم کرده و تفاوت مهم آن با طیف فرکانسی فریم‌های سیگنال گفتار قبلی بیان کنید.

ابتدا به صورت مشابه گفته شده در بخش ۲، یک فریم واکدار پیدا شده است که بعد از پخش متوجه شده‌ایم فریم مربوط به واج /d/ می‌باشد. طیف فرکانسی آن رسم شده است. (با استفاده از فوریه گرفتن و اعمال آن به db)



تفاوت مهم آن با فریم‌های بخش‌های قبل آن تفاوت در بازه‌ی فرکانسی (به دلیل تفاوت در channel و همچنین تفاوت صدای فرد با ما) است. فرکانس فرمنت‌ها متفاوت هستند.

۹. میخواهیم یک الگوریتم VAD ساده پیاده‌سازی کنیم. مراحل زیر را انجام دهید.

a. لگاریتم انرژی کوتاه مدت هر فریم را محاسبه کنید و آنرا ترسیم کنید.

b. یک آستانه ثابت به گونه‌ای انتخاب کنید که با اعمال آن بر منحنی لگاریتم انرژی، بیشتر فریم‌های گفتاری از غیرگفتاری متمایز شود.

c. از یک فیلتر میانه با طول ۵ فریم برای هموارسازی فریم‌های گفتاری آشکارشده از مرحله قبل استفاده کنید. (خروجی این مرحله باید شماره فریم‌های گفتاری باشد)

d. نتیجه الگوریتم خود را با الگوریتم VAD کتابخانه librosa مقایسه کنید.

e. طیف فرکانسی هر فریم را محاسبه کنید.

f. ۲۰ فیلتر مثلثی با دامنه یکسان، توزیع شده به صورت یکنواخت با همپوشانی ۵۰ درصد از هر طرف بر روی محول mel طراحی کنید. فرکانس شروع فیلتر مثلثی اول ۱۵۰ هرتز و فرکانس پایان فیلتر مثلثی آخر ۳۸۰۰ هرتز است. مراحل را به صورت کامل توضیح دهید. (دقت شود که طراحی بانک فیلترها به صورت کامل توسط خودتان انجام شود و از کتابخانه‌های آماده استفاده نکنید.) **(راهنمایی: از فرمول‌های زیر کمک بگیرید)**

$$f_{mel}(f_{hz}) = 2595 \times log_{10}\left(1 + \frac{f_{hz}}{700}\right)$$

$$f_{hz}(f_{mel}) = 700 \times \left(10^{\left(\frac{f_{mel}}{2595}\right)} - 1\right)$$

g. لگاریتم خروجی بانک فیلتر را محاسبه کنید (حاصل باید یک بردار ۲۰ بعدی باشد.)

h. از نتیجه حاصل، DCT بگیرید و تنها ۱۳ ضریب اول را برای هر فریم استخراج کنید.

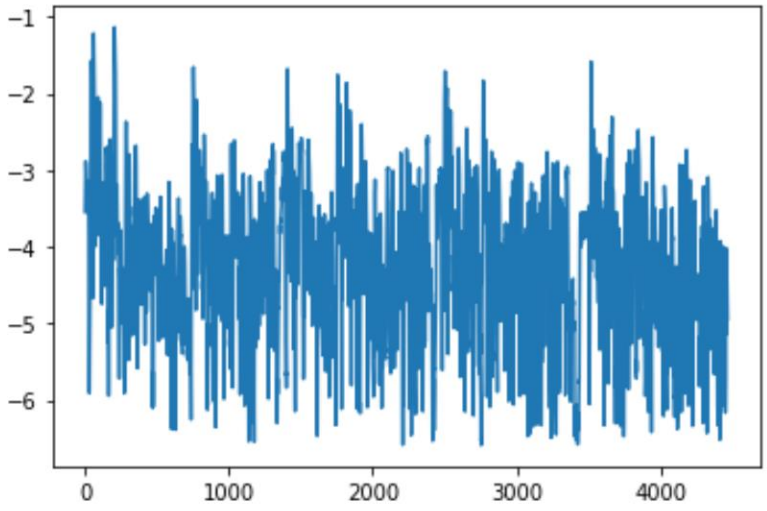
i. به جای ضریب صفرم کپسترال، لگاریتم انرژی کوتاه مدت فریم را جایگزین کنید.

j. بر روی فریم‌های گفتاری نتیجه شده از VAD، نرمالیزه‌سازی CMVN را انجام داده و بردارهای میانگین، انحراف استاندارد و MFCC نرمالیزه شده را بدست آورید.

k. ضرایب مشتق مرتبه اول و دوم و ضرایب MFCC نرمالیزه شده مرحله قبل را با استفاده از رابطه بیان شده در اسلایدهای درس با مقدار $\tau = 2$ محاسبه کنید. (از کد آماده استفاده نکنید.)

بخش a:

برای پیدا کردن لگاریتم انرژی کوتاه مدت، تابعی تحت عنوان log_short_time_energy نوشته شده است و مقدار لگاریتم میانگین انرژی هر فریم را بدست می‌آورد. برای هر فریم این مقدار حساب شده و نمودار آن به صورت زیر است.



بخش b:

مقدار آستانه 5.5- در نظر گرفته شده است و فریم‌هایی که مقدار آن‌ها از 5.5- کمتر بوده است به عنوان فریم‌های سکوت شناسایی شده است. اندیس فریم‌ها نمایش داده شده است.

بخش d:

با استفاده از تابع تحت عنوان نام VAD اندیس فریم‌های سکوت بدست آمده است.

بخش e:

با استفاده از تابع کشیدن طیف، فرکانس طیف فریم‌ها کشیده شده است.