

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 2 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 2( عملیات فریم‌بندی را انجام می‌دهد.



# **بخش پیدا کردن دو فریم واکه:**

برای پیدا کردن فریم‌ها واکه با مقایسه‌ی انرژی آن‌ها یا متناظر با توان 2 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 2 مقدار سیگنال (متناظر با انرژی( را برای هر فریم حساب کرده و مرتب می‌کند. 50 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:**

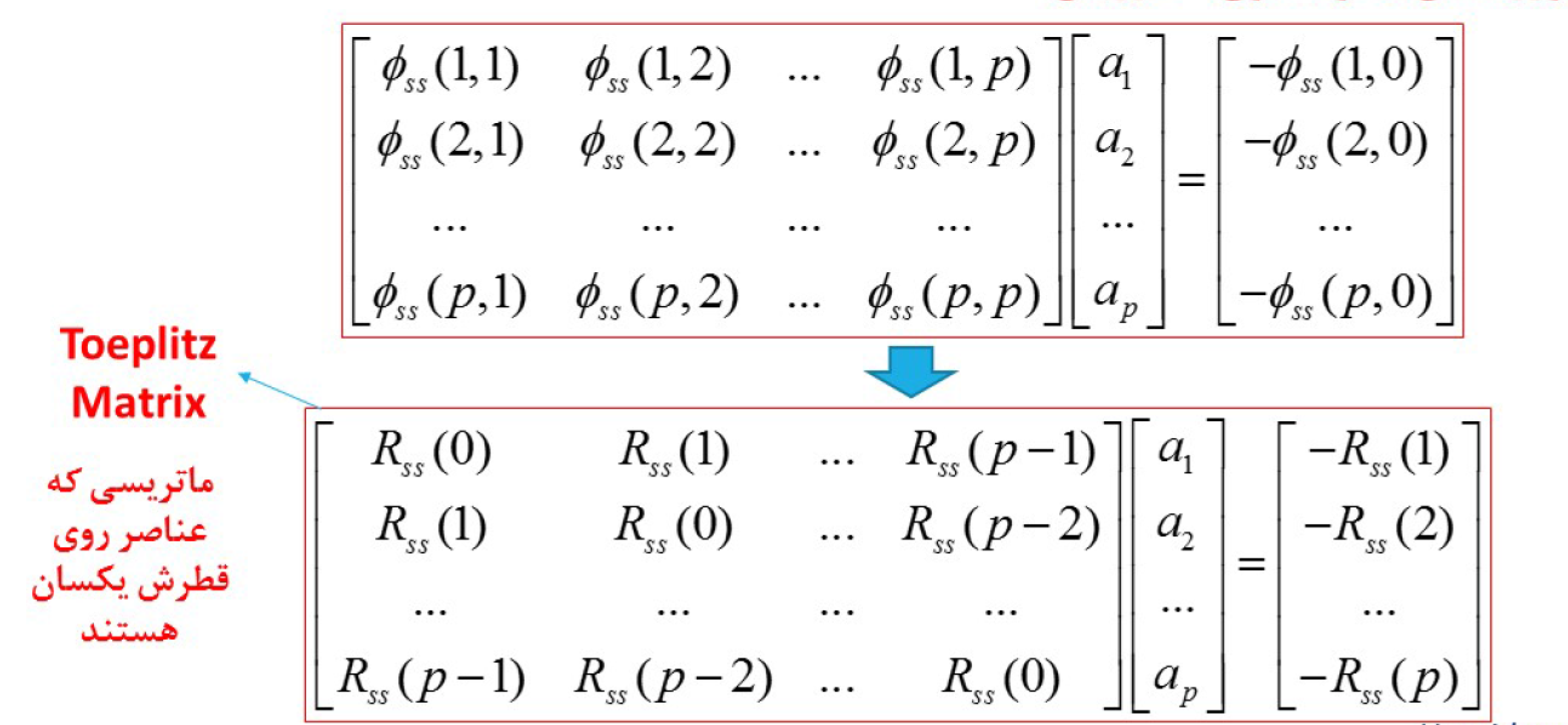
برای پیدا کردن فریم‌های واکدار باید F0 (pitch frequency) آن‌ها محاسبه شود. فریم‌ها واکدار مقدار F0 غیر صفر دارند و فریم‌های بی‌واک F0 برابر صفر دارند. (البته باید توجه کرد در کنار هم قرار گرفتن فریم‌ها واکدار و بیواک خطایی رخ دهد پس باید مقدار F0 به نسبت بالایی داشته باشد.) تابع get\_mean\_pithes برای 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:

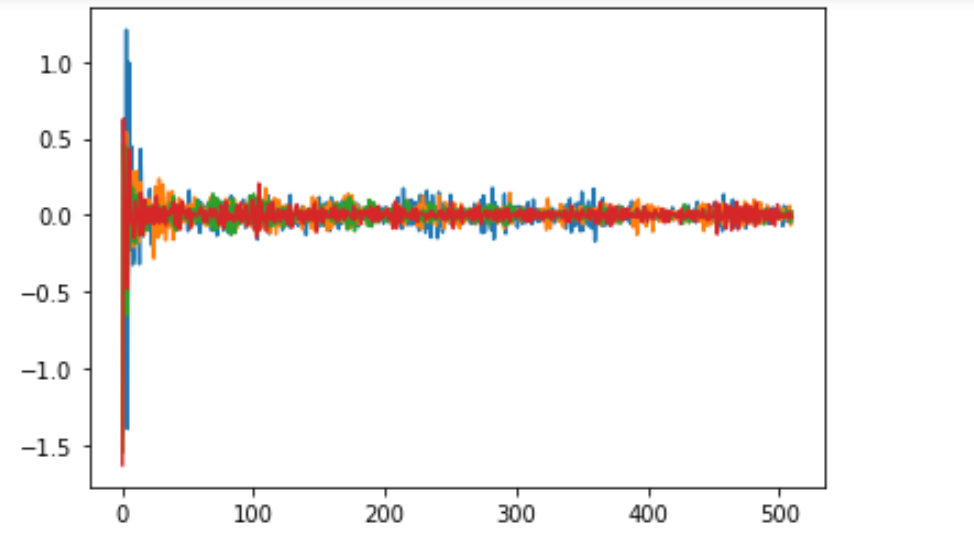
* تابع get\_LPC\_coeffience\_and\_show: که dictionary فریم‌ها را دریافت کرده و برای هر کدام با توجه به تابع (الگوریتم) پیدا کردن ضرایب LPC، این ضرایب را پیدا می‌کند و همگی را بر روی یک نمودار نشان می‌دهد.
* تابع get\_toeplitz\_matrix: این تابع بردار autocorrolated فریم ورودی را حساب می‌کند و سپس ماتریس 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.



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

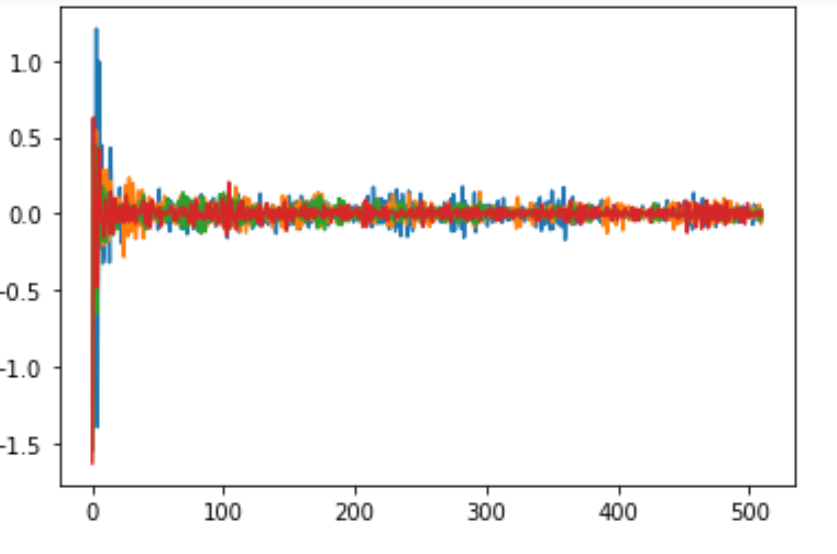
نمودار سبز: فریم واکدار اول

نمودار قرمز: فریم واکدار دوم

نمودار ضرایب LPC برای 4 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

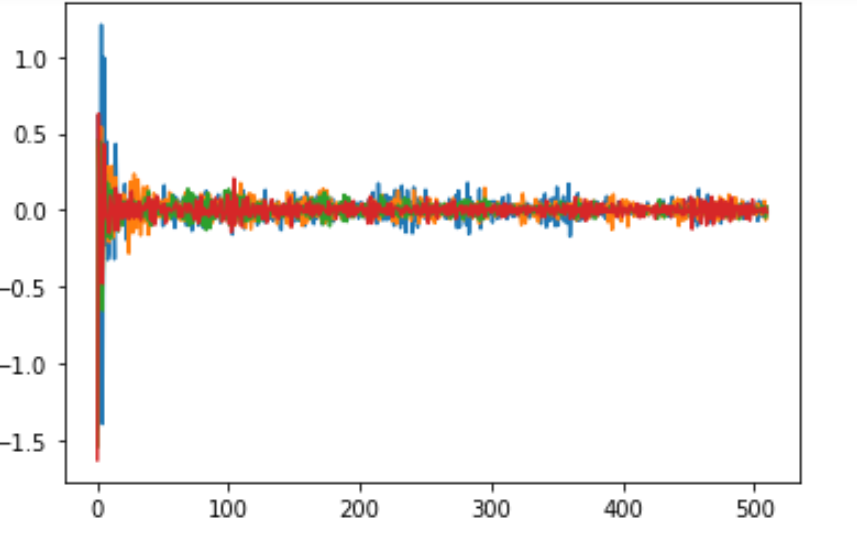
نمودار سبز: فریم واکدار اول

نمودار قرمز: فریم واکدار دوم

نمودار ضرایب LPC برای 4 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

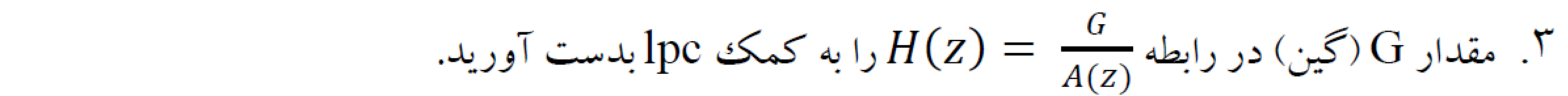
نمودار نارنجی: فریم واکه دوم

نمودار سبز: فریم واکدار اول

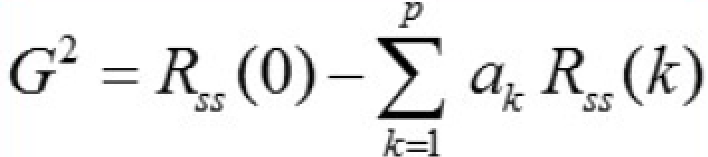
نمودار قرمز: فریم واکدار دوم

نمودار ضرایب LPC برای 4 فریم به صورت بالا نشان داده شده است. (هر فریم یک رنگ در نمودار)

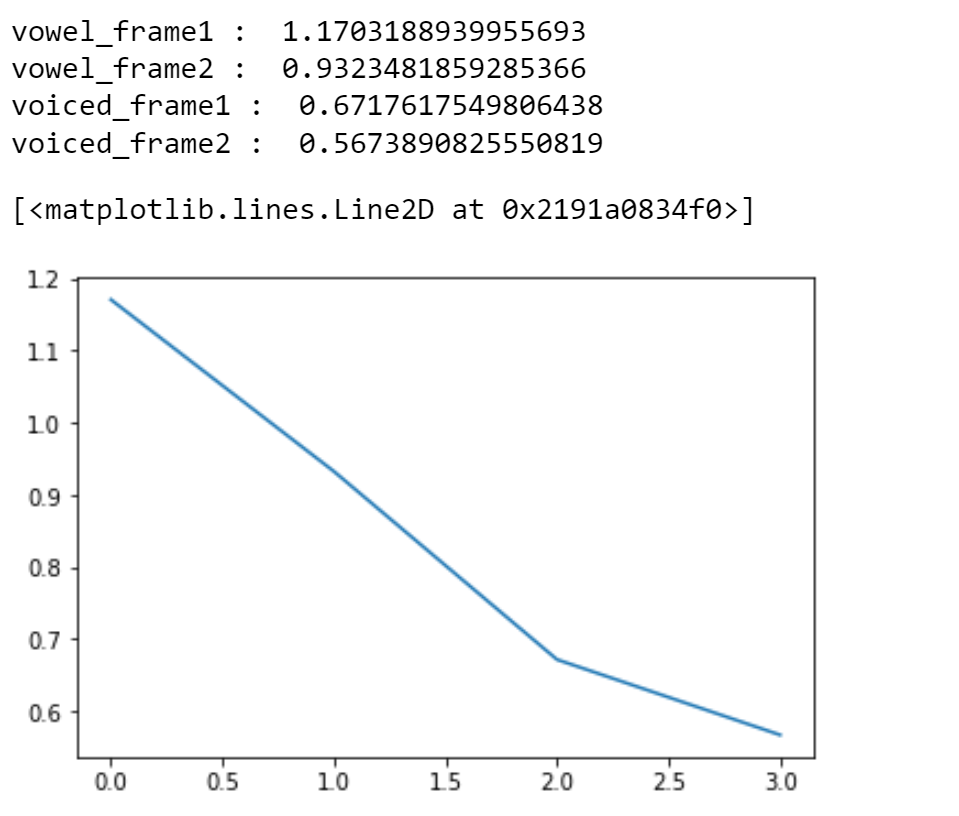
همانطور که مشاهده می‌شود نمودار ضرایب برای هر سه روش به یک صورت است و نتایج یکی است.



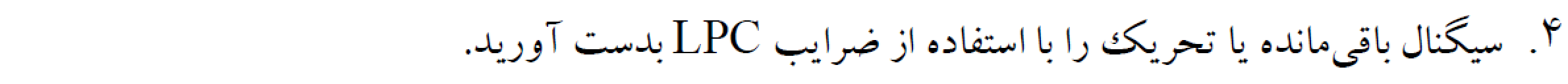
برای پیدا کردن مقادیر G تابعی تحت عنوان get\_G نوشته شده است که در آن با استفاده از رابطه‌ی زیر مقدار G محاسبه می‌شود.



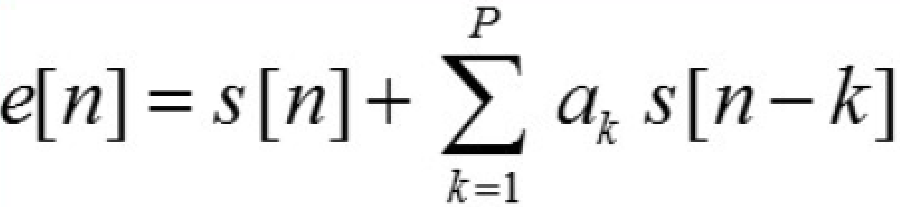
مقادیر و نمودار 4 G بدست آمده به صورت زیر است.



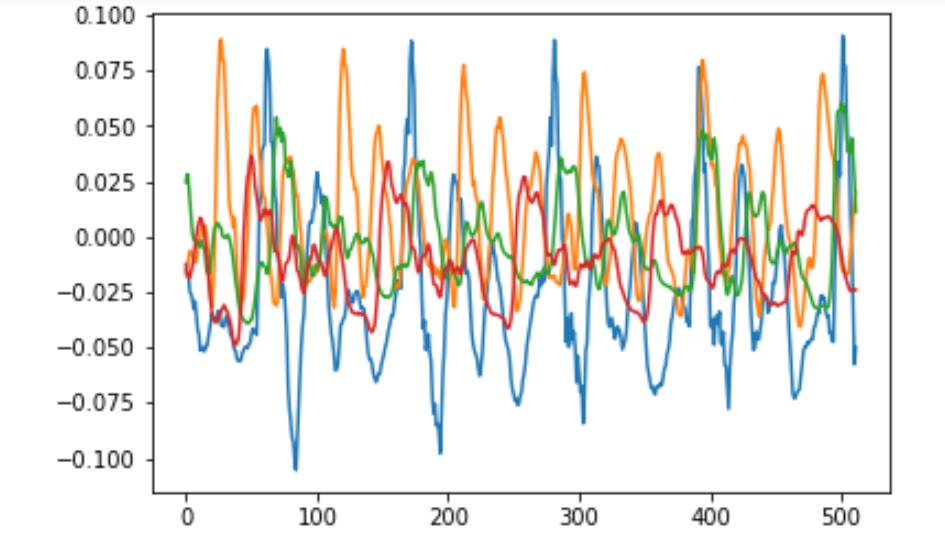
همچنین می‌توانستیم با استفاده از خود تابع spectrum.lpc این مقدار بدست آید.



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.



نمودار 4 سیگنال باقی‌مانده برای 4 فریم ذکر شده به صورت زیر است:

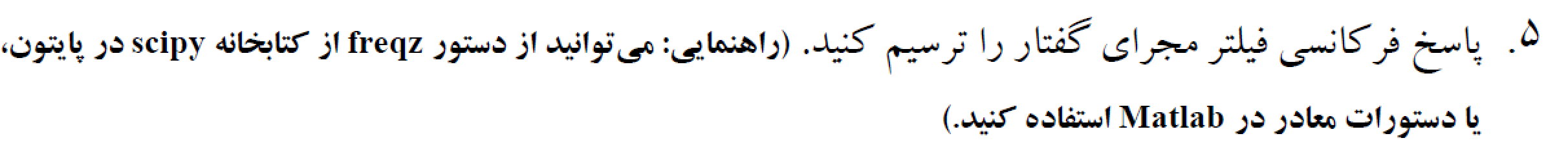


نمودار آبی: فریم واکه اول

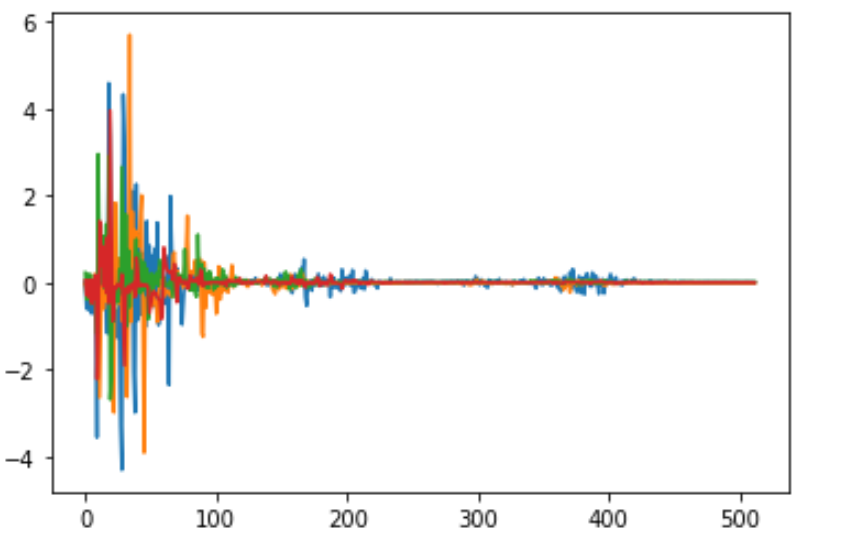
نمودار نارنجی: فریم واکه دوم

نمودار سبز: فریم واکدار اول

نمودار قرمز: فریم واکدار دوم



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



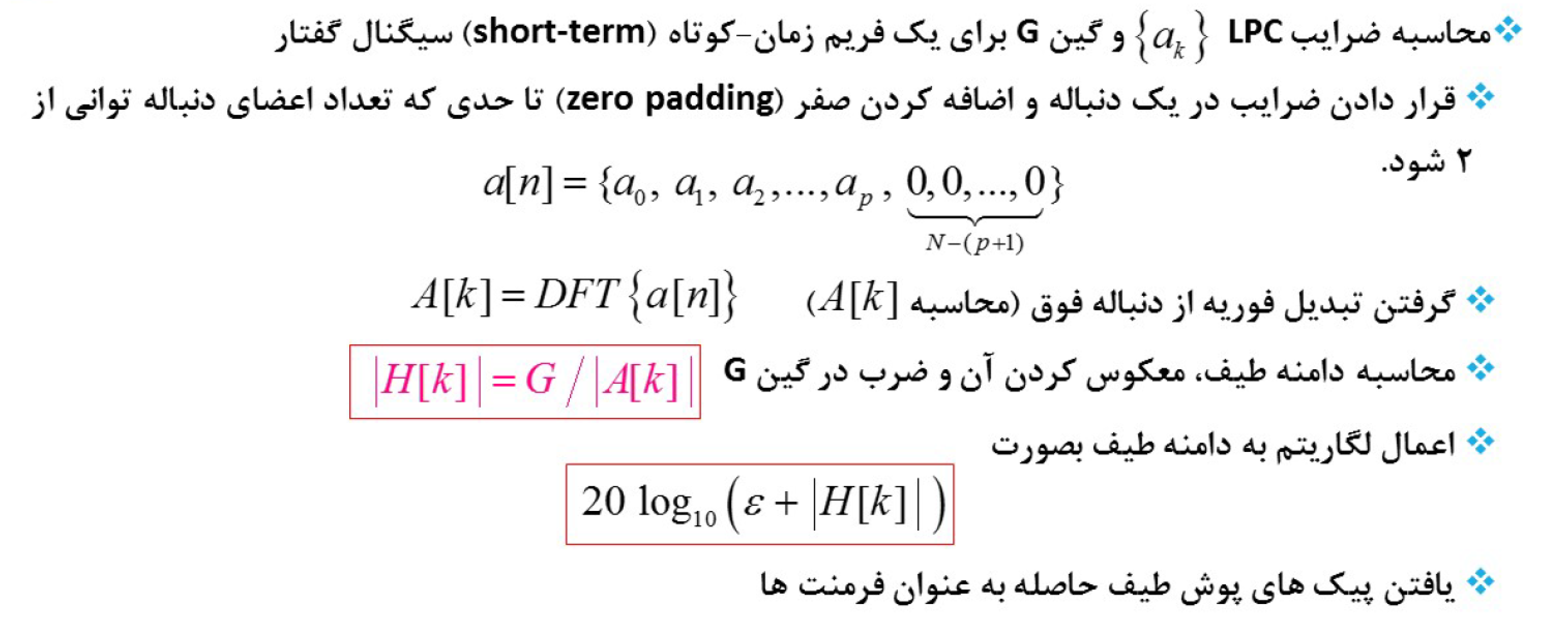
نمودار آبی: فریم واکه اول

نمودار نارنجی: فریم واکه دوم

نمودار سبز: فریم واکدار اول

نمودار قرمز: فریم واکدار دوم

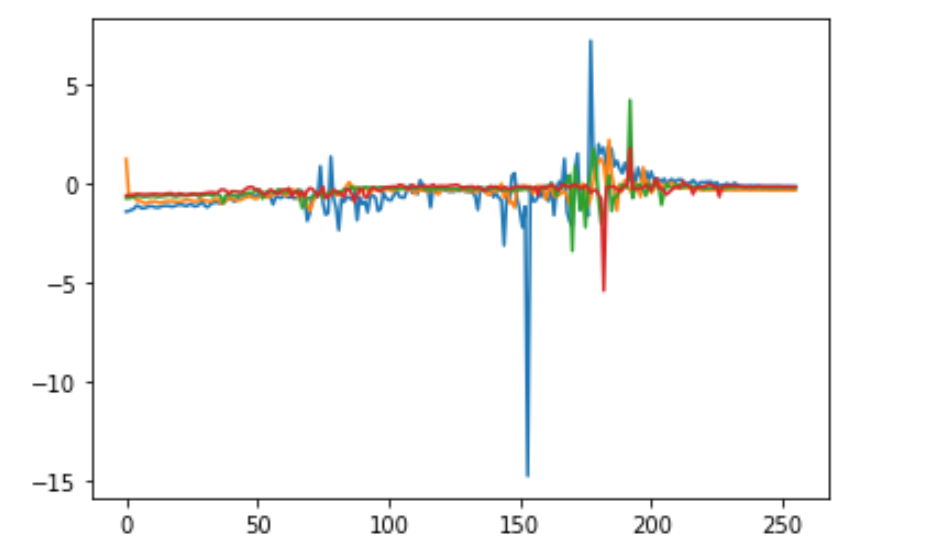
همچنین می‌توان از روش زیر برای محاسبه‌ی Hقدر مطلق و پوش طیف استفاده کرد. (برای بخش 6 نیز است.)



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 4 فریم به صورت زیر است:

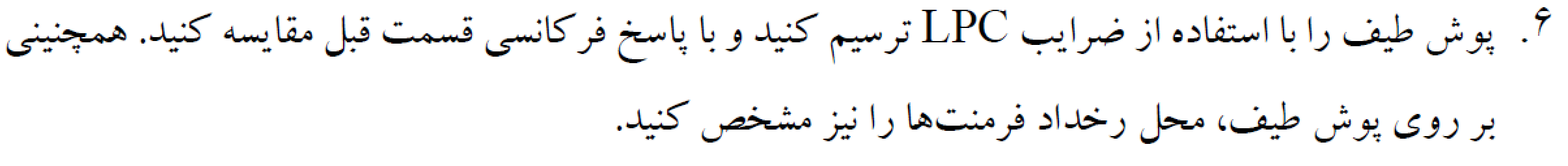


نمودار آبی: فریم واکه اول

نمودار نارنجی: فریم واکه دوم

نمودار سبز: فریم واکدار اول

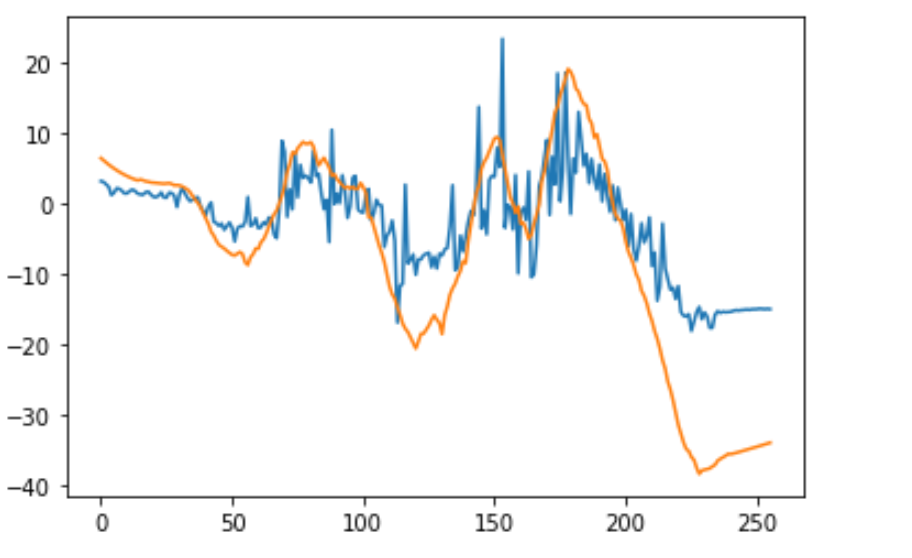
نمودار قرمز: فریم واکدار دوم



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:

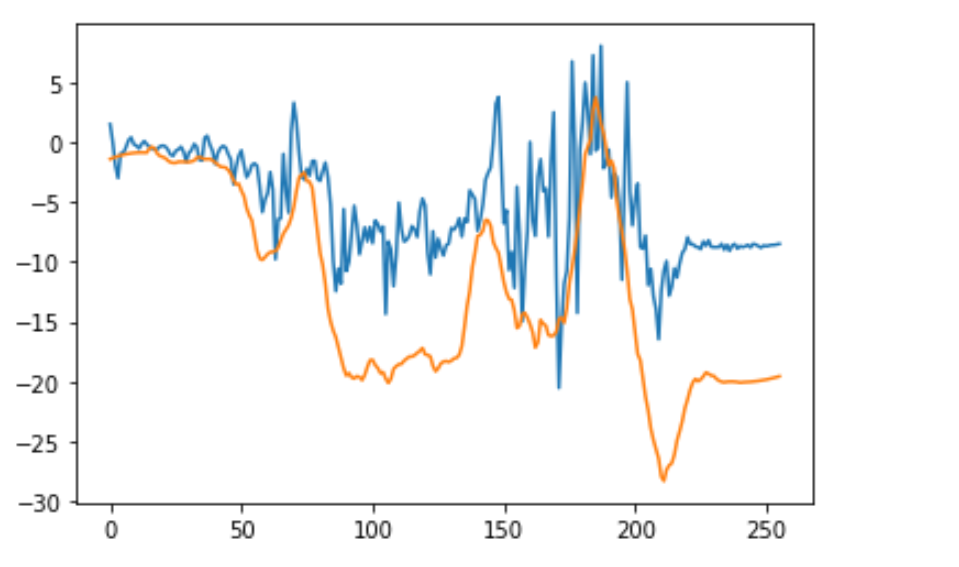


F3

F2

F1

* فریم واکه دوم:

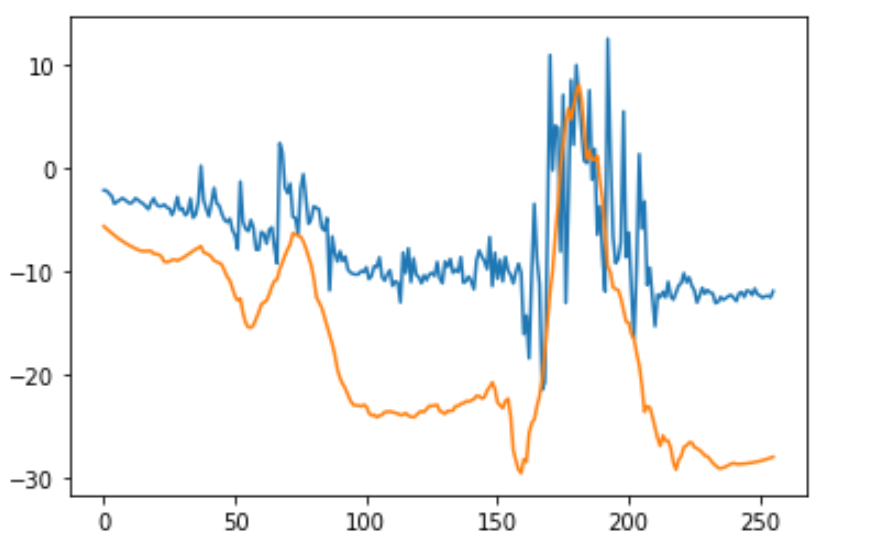


F3

F2

F1

* فریم واکدار اول:

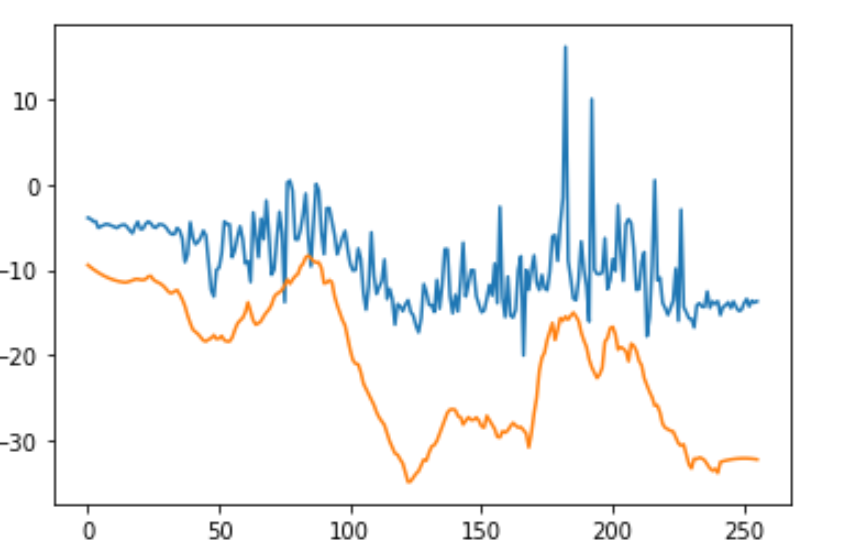


F3

F2

F1

* فریم واکدار دوم:

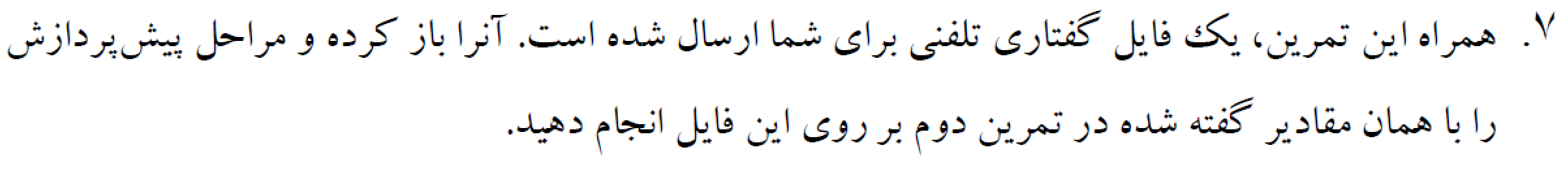


F3

F2

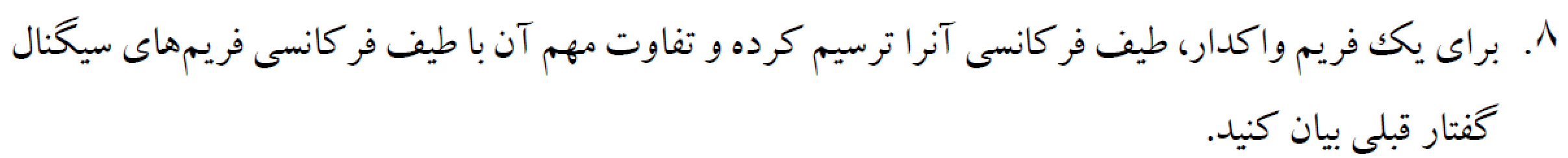
F1

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

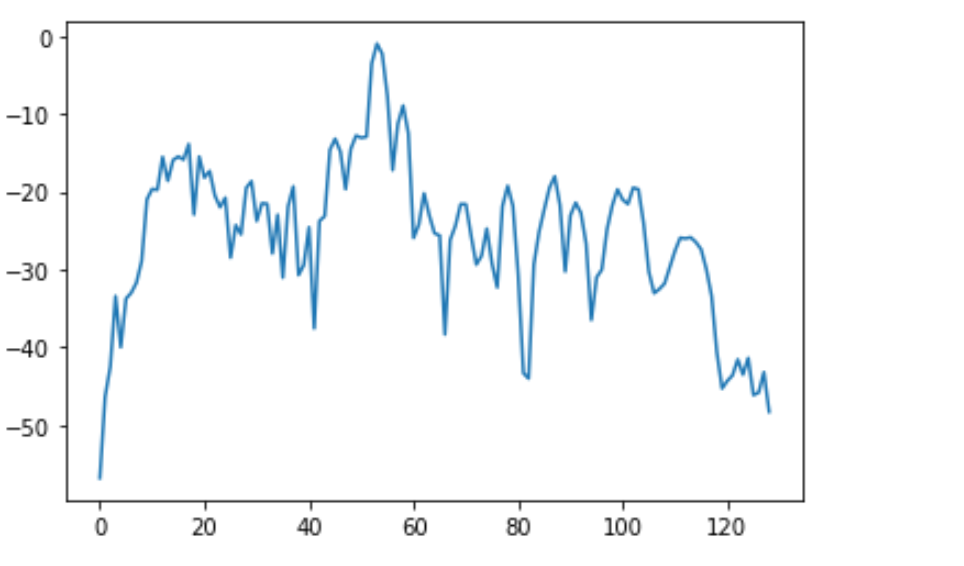


در این بخش توابع پیش‌پردازش pre\_emphasis، framming و windowing مشابه تکلیف قبلی نوشته شده است. همچنین تابع preprocess نوشته شده است که به ترتیب موارد پیش‌پردازش را روی یک فریم با مقادیر ذکر شده انجام می‌دهد.

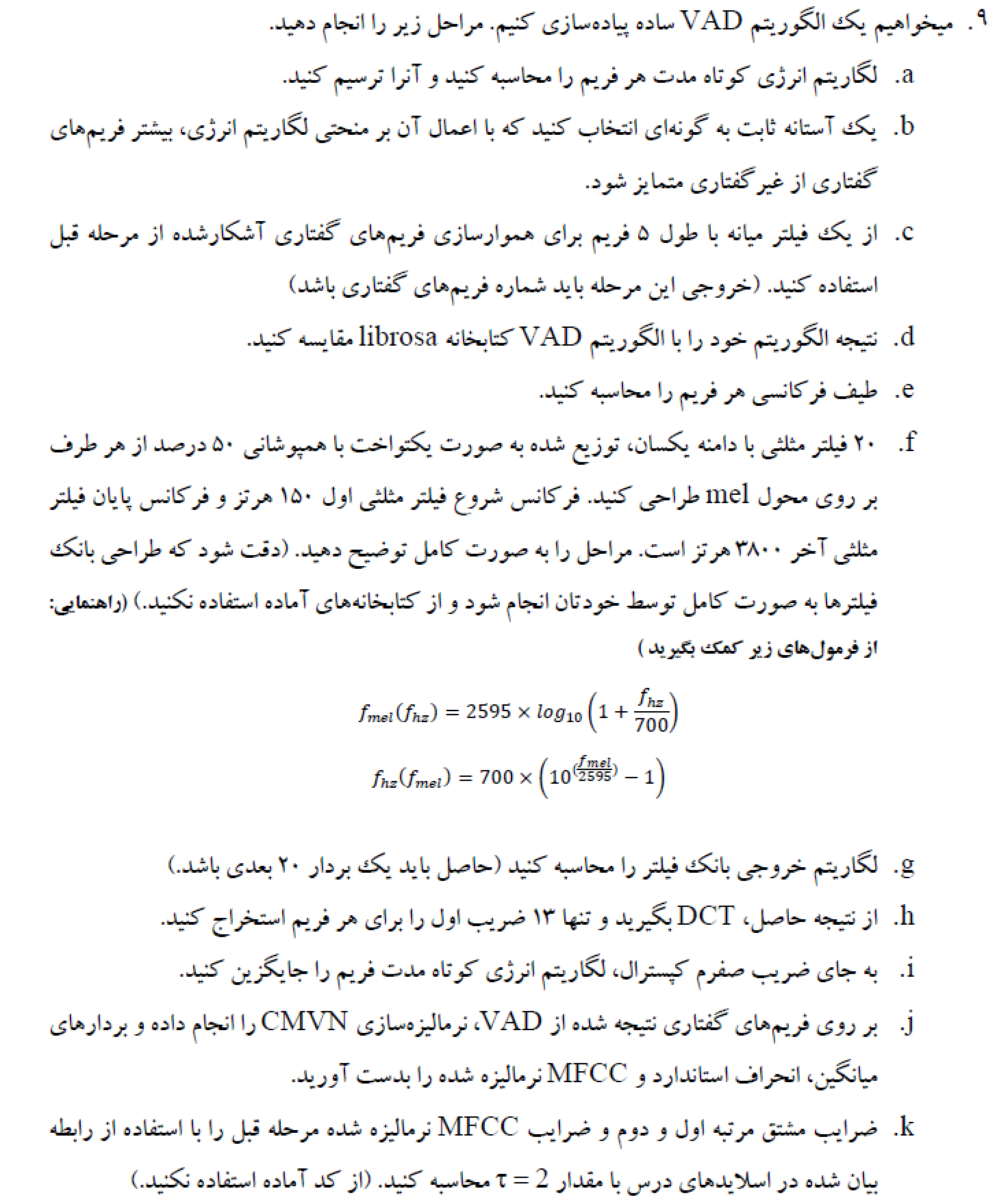
صوت داده شده خوانده شد و پیش‌پردازش صورت گرفت و فریم‌های پیش‌پردازش شده بدست آمد.



ابتدا به صورت مشابه گفته شده در بخش 2، 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)

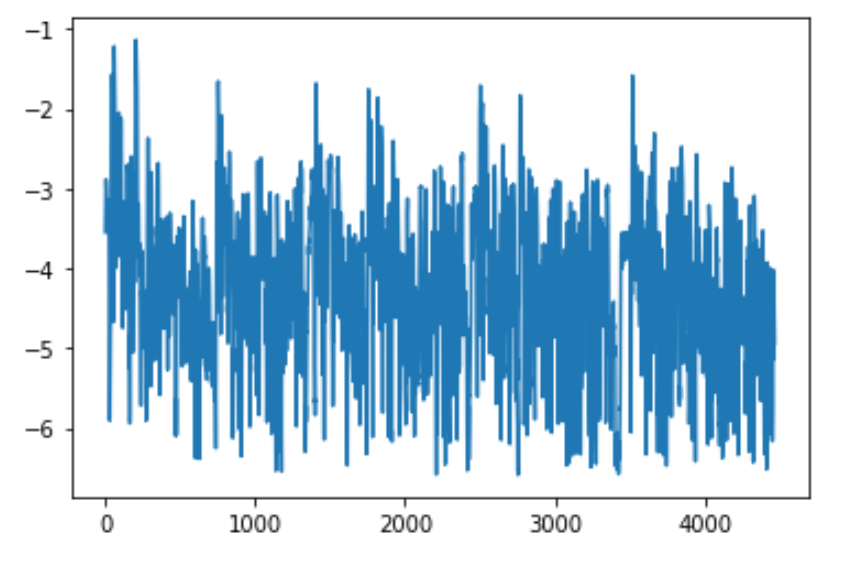


تفاوت مهم آن با فریم‌های بخش‌های قبل آن تفاوت در بازه‌ی فرکانسی ( به دلیل تفاوت در channel و همچنین تفاوت صدای فرد با ما) است. فرکانس فرمنت‌ها متفاوت هستند.



# **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.