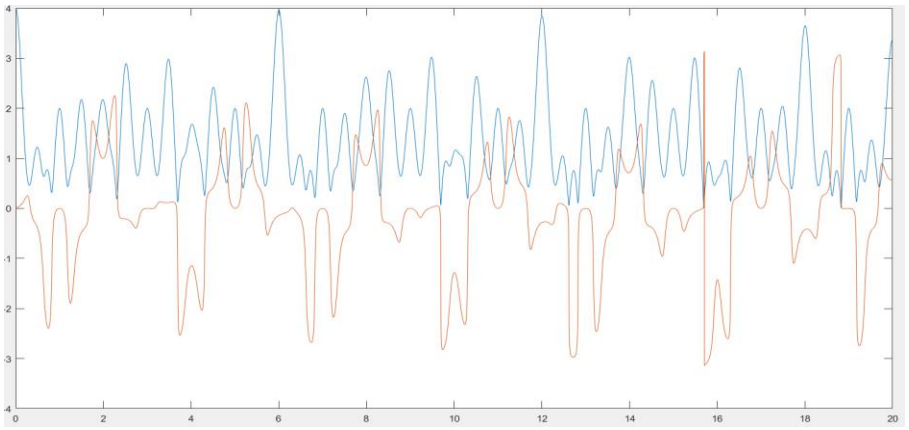
گزارش تمرین کامپیوتری اول درس سیگنال

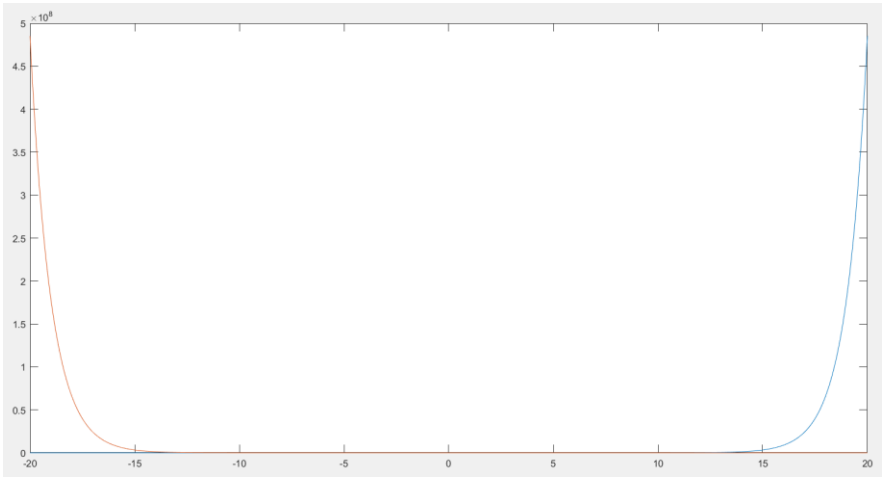
سیدعلی سادات اخوانی – ۸۱۰۱۹۳۴۲۵

بخش اول

۱- می‌خواهیم سیگنال $x_a(t) = 1 + cos(4\pi t) + e_{jt}(cos(0\pi t) + cos(\pi t))$ را در بازه‌ی ۲۰ ثانیه رسم کنیم.
اندازه را با abs(y) و فاز سیگنال را با angle(y) به دست می‌آوریم و رسم می‌کنیم:
نمودار کشیده شده به صورت زیر است:

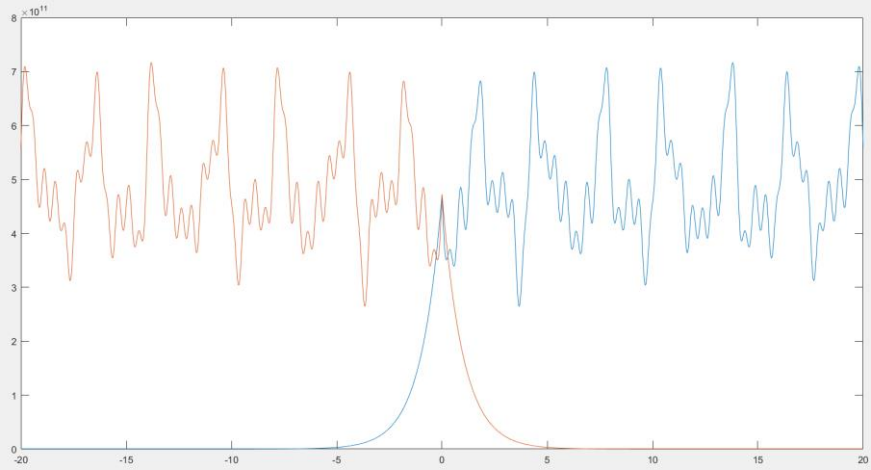


۲ـ سیگنال‌های $h1 = e^{t}$ و $h2 = e^{-t}$ را در بازه‌ی ۲۰ـ تا ۲۰ رسم می‌کنیم.



۳و۴ـ سیگنال بخش ۱ را با پاسخ‌های ضربه‌ی h1 و h2 که در سوال ۲ داده شده‌است کانوالو می‌کنیم. خروجی دو سیستم نسبت به محور yها قرینه است. چون پاسخ ضربه‌ی دو سیستم رابطه‌ی h1(t) = h2(-t) دارد پس باید نسبت به y قرینه باشد.

خروجی‌های زیر به دست می‌آید:



بخش دوم

۵ـ با استفاده از fopen فایل مورد نظر را باز می‌کنیم و خروجی آن file descriptor ما خواهد بود.

```
fileID = fopen('accel_data.txt');
```

در فایل موجود در این سوال ۴ ستون مختلف از اعداد داریم، پس با قالب زیر می‌توانیم آن‌ها را در ماتریس ۴ ستونه ذخیره کنیم:

```
C = textscan(fileID, '%f%f%f%f');
```

حال با استفاده از fclose فایل را می‌بندیم چون دیگر با فایل کاری نداریم و باید بسته شود.

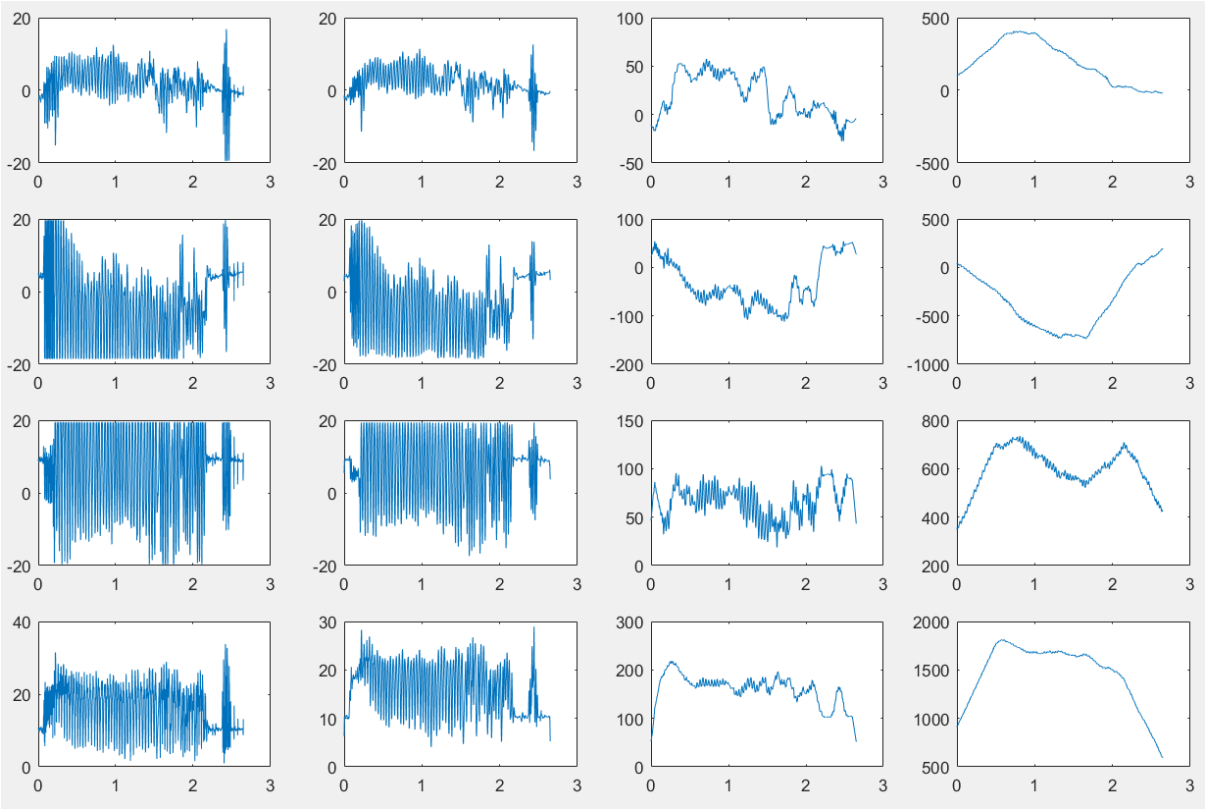
۶ـ چون داده‌های ما خیلی نویزی است از فیلتر میان‌متحرک استفاده می‌کنیم. فیلتر میان‌متحرک یا Moving Average یک فیلتر پایین‌گذر با پاسخ ضربه‌ی متناهی (Finite Impulse Response) است که برای کاهش نویز سیگنال‌های نمونه‌برداری‌شده و یا smoothing آن‌ها استفاده می‌شود.

روش کار این فیلتر به این صورت است که چند داده را گرفته و میانگین آن‌ها را به عنوان خروجی نمایش می‌دهد. برای همین با این روش نویز کاهش پیدا می‌کند.

۷ـ پاسخ سیستم به ورودی های x، y و z و w (در سوال w خواسته نشده بود اما این‌جا آورده شده) در سه ma_length خواسته‌شده را به دست می‌آوریم. برای این کار باید کانولوشن ورودی‌ها را با پاسخ ضربه سیستم

```
h= 0.1*ones(1,ma_length)
```

به دست آوریم. خروجی‌ها به صورت زیر است:



۸ـ سیگنال پس از گذر از فیلتر از ورودی‌ها میانگین می‌گیرد و خروجی به هم نزدیک می‌شود یا به اصطلاح ورودی smooth می‌شود. هرچه ma_length افزایش یابد نویزها کمتر می‌شود. اما اگر بیش از حد طول فیلتر یا همان ma_length را افزایش دهیم ممکن است overfit شود و خروجی نامطلوب باشد.

بخش سوم

برای به دست آوردن tt که بردار زمان مرتبط با tone است، از ۱ تا TD/nd با step برابر FS/1 حرکت می‌کنیم.

می‌دانیم که TD مدت زمانی است که یک Note کامل طول می‌کشد و نسبت Note موجود به Note کامل است. پس باید برای پیدا کردن مدت زمان هر نوت باید TD را به nd تقسیم کنیم.

نکته‌ی دیگر این است که برای برداشتن نمونه با فرکانس FS باید step برابر FS/1 داشته باشیم.

حال اگر بخواهیم تعداد نمونه‌ها را به دست آوریم باید از تناسب استفاده کنیم. تعداد نمونه‌ها برابر است با: FS * TD/n.
طول بردار tt برابر این مقدار خواهد بود.

همچنین برای ساختن مدل یک Note و هارمونیک‌های آن باید فرمولی را که برای هر نوت داریم به‌ازای ۱ تا ۱۰ حساب کنیم.

پس از آن با جمع کردن این ۱۰ هارمونیک به نتیجه‌ای می‌رسیم که شبیه Noteهای موسیقی است.

در آخر کار باید عددی را که پیدا می‌کنیم در تابعی نمایی و میرا ضرب کنیم تا میرایی Noteهای موسیقی را شبیه‌سازی کنیم.

فرکانسهایی که در فایل notes.m برای Noteهای موسیقی داریم از یک فرکانس پایه تشکیل شده است و Noteهای بعدی به‌طور مداوم در 12/1 ضرب شده‌اند)

فرکانسها به شکل مرتب و با ضریب ثابتی به ازای نوتهای مختلف تغییر میکند.

در پیاده‌سازی کد این سیستم، از حلقه‌ی for استفاده شده است.

برای عمل جایگذاری، کدی که زده شده، به ازای طول بردار هر Note مقدار new_sum را تغییر می‌دهد و هارمونیک‌های جمع شده و میراشده با استفاده از ضرب در تابع نمایی میرا را در فاصله‌ی بین temp_sum که در واقع sum_new مرحله‌ی قبل بوده، جایگذاری می‌کند.

پس از آن به سراغ تعداد اعداد لازم برای نوت بعدی می‌رود و با احتساب طول مورد نیاز آن و xtmp جدید و ضرب آن در تابع میرا آن را جایگذاری میکند و برای همه‌ی Noteها این کار را تکرار می‌کند. در نهایت برداری از مجموع همه‌ی Noteها داریم و آن را در یک فایل صوتی می‌ریزیم با استفاده از دستور audiowrite.