به نام خدا

دانشگاه صنعتی خواجه نصیرالدین

دانشکده برق

تشخیص و شناسایی خطا

امتحان پایان‌ترم

شیما سادات ناصری

۴۰۱۱۲۸۱۴

دکتر مهدی علیاری شوره دلی

۲۰ خرداد ۱۴۰۲

فهرست مطالب

عنوان   شماره صفحه

بخش۱: سوال هماهنگ شده

سوال اول

در مسئله بهینه‌سازی در این فرایند از نرم بی‌نهایت استفاده می‌شود که در آن نیاز به معکوس G_d است. حال اگر G_d صفر موهومی داشته باشد، بعد از اعمال معکوس سازی، قطب موهومی ایجاد می‌شود و چون در تعریف نرم بی‌نهایت، حد بالای بزرگترین مقدار ویژه $G(j\omega)$ در نظر گرفته می‌شود $(\forall\omega \in \mathbb{R})$ به ازای ω برابر با فرکانس صفر موهومی مقدار نرم، بی‌نهایت شده و نرم برای آن وجود ندارد. علاوه براین، زمانی که سیستم stricly proper باشد، معکوس آن در RH_∞ وجود ندارد (چرا که برای مثال در حالت siso درجه صورت از مخرج کمتر است و اگر معکوس شود مرتبه صورت بیشتر می‌شود. مشابه همین اتفاق به صورتی دیگر در mimo رخ می‌دهد) و برای محاسبه نرم نیز به مشکل خواهیم خورد. با این حال در کتاب Chen & Patton[1] روشی برای محاسبه در حالت دوم پیشنهاد شده که به صورت زیر می‌باشد.

در تئوری شماره 8.5 آمده است که اگر $\tilde{N}_d(s)$ شامل صفر روی محور موهومی باشد(حالت دوم) و تعمیم یافته روش inner-outer factorization آن به صورت زیر باشد:

$$\tilde{N}_d(s) = G_{do}(s)G_{dz}(s)G_{di}(s)$$

که در آن فقط عبارت $G_{dz}(s)$ شامل تمامی صفرهایی است که روی محور موهومی قرار دارند؛ درنتیجه حل بهینه آن به صورت زیر خواهد بود.

$$Q_{opt}(s) = Q_0(s)G_{do}^{+}(s)\ ,J_{opt} = \left|\left|G_{do}^{+}(s)G_{dz}(s)\right|\right|$$

در این معادله پارامتر Q_0 براساس معادلات زیر بدست می‌آیند:

$$\begin{cases} G_{dz}^{T}(-j\omega)Q_0^{T}(-j\omega)Q_0(j\omega)G_{dz}(j\omega) \le I\ ,\forall\omega \\ G_{dz}^{T}(-j\omega_0)Q_0^{T}(-j\omega_0)Q_0(j\omega_0)G_{dz}(j\omega_0) = I \end{cases}$$

سوال دوم

سیستم زیر در نظر گرفته می‌شود:

$$y = G_{yu}u + G_{yd}d + G_{yf}f$$

که مانده سیستم به صورت زیر بدست می‌آید:

$$r = R(M_u y - N_u u)$$

[1] Robust Model-Based Fault Diagnosis for Dynamic Systems Kluwer (1999)

اولین شرط گفته شده در صورت سوال در صورتی که برقرار نباشد، داریم:

$$rank([G_{yf}\ G_{yd}]) = rank(G_{yd})$$

درنتیجه برای هر $R\widehat{M}_u$ یک ماتریس تبدیل وجود دارد که:

$$R\widehat{M}_u G_{yf} = R\widehat{M}_u G_{yd} T(s)$$

واضح است که $R\widehat{M}_u G_{yd} = 0$ که این موضوع نتیجه می‌دهد:

$$R\widehat{M}_u G_{yf} = 0$$

اما این موضوع مخالف شرط اصلی و مهم ما (فرض مسئله) که perfect decoupling است، می‌باشد. درنتیجه باید شرط داده شده در صورت مسئله برقرار باشد تا اولین شرط مسئله نیز برقرار گردد. این شرط بیان می‌کند که اسپن زیر فضای G_{yf} با زیر فضای G_{yd} متفاوت است؛ به عبارتی G_{yf} باید حداقل ۱ بردار ویژه مستقل و متفاوت از G_{yd} داشته باشد. بدیهی است که درصورتی که این شرط محقق نشود، دو تابع به هم وابسته هستند و نمی‌توان آن‌ها را از هم تمیز داد.

برای شرط دوم، مقدار m به صورت زیر تعریف شده و داریم:

$$rank(\widehat{M}_u) = m$$
$$rank([G_{yf}\ \ G_{yd}]) \le m$$

حال با توجه به اولین شرط، می‌توان نوشت:

$$rank(G_{yd}) < m$$

به عبارتی تعداد ورودی های ناشناخته‌ای که روی خروجی تاثیر دارد باید کوچک‌تر از تعداد سنسور ها باشد.

سوال سوم

برای اثبات نامساوی $\frac{rank(\varphi_o)}{rank(C)} \le s_0 \le rank(\varphi_o) - rank(C) + 1$ در مورد تشخیص خطای همگنی، باید از تعاریف و خواص ماتریس قابل مشاهده، ماتریس فضای حالت و رتبه استفاده کنیم.

ابتدا سمت چپ نامساوی را اثبات می‌کنیم:

$$\frac{rank(\varphi_o)}{rank(C)} \le s_0$$

بر اساس تعریف ماتریس رویت پذیری φ، این ماتریس با افزودن ستون‌های C، CA، CA^2، ...، CA^(n-1) تشکیل می‌شود. بنابراین، رتبه ماتریس φ برابر است با بیشینه تعداد سطرهای مستقل خطی در آن تشکیل می‌شود. از طرفی، رتبه ماتریس C نشان دهنده تعداد سطرهای مستقل خطی در ماتریس C است. بنابراین، می‌توانیم نامساوی را به صورت زیر بازنویسی کنیم:

$$rank(\varphi_o) \leq rank(C) * s_0$$

حال ضرب ماتریس رویت پذیری در ماتریس A را در نظر می‌گیریم:

$$\varphi_o A = [CA, CA^2, CA^3, \dots, CA^n]$$

ضرب $\varphi_o A$ در ماتریس A را دوباره انجام می‌دهیم:

$$\varphi_o A^2 = [CA^2, CA^3, CA^4, \dots, CA^{n+1}]$$

این فرآیند را تا $\varphi_o A^{n-1}$ ادامه می‌دهیم:

$$\varphi_o A^{n-1} = [CA^{n-1}, CA^n, CA^{n+1}, \dots, CA^{2n-2}]$$

توجه کنید که ماتریس $\varphi_o A^{n-1}$ حاوی همان تعداد سطرهای مستقل خطی در ماتریس رویت پذیری است. این امر به این دلیل است که ضرب در ماتریس A رتبه یک ماتریس را تغییر نمی‌دهد. بنابراین، می‌توان نتیجه گرفت که $rank(\varphi_o A^{n-1}) = rank(\varphi)$

حال به ماتریس $[C, CA, CA^2, \dots, CA^{n-1}, CA^n]$ توجه می‌کنیم:

$$[C, CA, CA^2, \dots, CA^{n-1}, CA^n] = [C, CA, CA^2, \dots, CA^{n-1}] * A + [0, 0, 0, \dots, CA^{n-1}]$$

از آنجا که عبارت سمت راست $[0, 0, 0, \dots, CA^{n-1}]$ یک ترکیب خطی از سطرهای $[C, CA, CA^2, \dots, CA^{n-1}, CA^n]$ است، می‌توان نتیجه گرفت که رتبه ماتریس $[C, CA, CA^2, \dots, CA^{n-1}, CA^n]$ برابر است با رتبه ماتریس $[C, CA, CA^2, \dots, CA^{n-1}]$. بنابراین، داریم:

$$rank(\varphi_o A^{n-1}) = rank([C, CA, CA^2, \dots, CA^{n-1}, CA^n]) = rank(\varphi_o)$$

از طرف دیگر، $\varphi_o A^{n-1} = [CA^{n-1}, CA^n, CA^{n+1}, \dots, CA^{2n-2}]$است. بنابراین داریم:

$$rank(\varphi_o) = rank([CA^{n-1}, CA^n, CA^{n+1}, \dots, CA^{2n-2}])$$

حال به ماتریس $C * [A^{n-1}, A^n, A^{n+1}, \dots, A^{2n-2}]$ توجه می‌کنیم:

$$C * [A^{n-1}, A^n, A^{n+1}, \dots, A^{2n-2}] = [CA^{n-1}, CA^n, CA^{n+1}, \dots, CA^{2n-2}]$$

از آنجا که عبارت سمت چپ $[CA^{n-1}, CA^n, CA^{n+1}, \dots, CA^{2n-2}]$ یک ترکیب خطی از ستون‌های $C * [A^{n-1}, A^n, A^{n+1}, \dots, A^{2n-2}]$ است، می‌توان نتیجه گرفت که رتبه ماتریس

$C * [A^{n-1}, A^n, A^{n+1}, ..., A^{2n-2}]$ کمتر یا مساوی رتبه ماتریس $[CA^{n-1}, CA^n, CA^{n+1}, ..., CA^{2n-2}]$ است. بنابراین، داریم:

$$rank(\varphi_o) \leq rank(C * [A^{n-1}, A^n, A^{n+1}, ..., A^{2n-2}])$$

توجه کنید که رتبه ماتریس $C * [A^{n-1}, A^n, A^{n+1}, ..., A^{2n-2}]$ معادل رتبه ماتریس $[C, CA, CA^2, ..., CA^{n-1}, CA^n]$ است. بنابراین، می‌توان نامساوی را به صورت زیر بازنویسی کنیم:

$$rank(\varphi_o) \leq rank([C, CA, CA^2, ..., CA^{n-1}, CA^n])$$
$$\Rightarrow rank(\phi_o) \leq rank(C * [I, A, A^2, ..., A^{n-1}, A^n])$$

از آنجا که این نامساوی برای هر رتبه ای صحیح است، می‌توان نتیجه گرفت که:

$$\frac{rank(\varphi_o)}{rank(C)} \leq rank([I, A, A^2, ..., A^{n-1}, A^n]) = s_0$$

حال سمت راست نامساوی را اثبات می‌کنیم:

$$s_0 \leq rank(\varphi_o) - rank(C) + 1$$

می‌دانیم که $rank(\varphi_o)$ تعداد بیشینه سطرهای مستقل خطی در ماتریس φ_o است و $rank(C)$ تعداد سطرهای مستقل خطی در ماتریس C است. بنابراین، تعداد بیشینه سطرهای وابسته خطی در ماتریس φ برابر است با $rank(\varphi_o) - rank(C)$.

با اضافه کردن یک به این مقدار، ما حالتی را در نظر می‌گیریم که سطرهای ماتریس φ_o تا سطر $rank(\varphi_o) - rank(C)$ به طور خطی مستقل هستند و سطر بعدی آنها را وابسته خطی می‌کند. بنابراین، داریم:

$$s_0 \leq rank(\varphi_o) - rank(C) + 1$$

با این اثبات، نامساوی زیر را اثبات کرده‌ایم:

$$\frac{rank(\varphi_o)}{rank(C)} \leq s_0 \leq rank(\varphi_o) - rank(C) + 1$$

با این توضیحات، نامساوی مورد نظر در مورد تشخیص خطا به روش parity را اثبات کرده‌ایم. در کل هدف ما این بود که مقداری از s عمق را پیدا کنیم که نه به حالت وابسته بودن برسیم و نه آن قدر کم باشد که نتواند کمکی به ما بکند.

بخش۲: سوالات هماهنگ نشده

سوال چهارم

در هر دو روش در با وجود اینکه در روش LDA به صورت سوپروایز پیش می‌رویم اما در روش PCA کاملا بدون توجه به برچسب ها پیش می‌رویم؛ در نهایت برای بهینه سازی، در هر دو از مقادیر ویژه دسته های مورد بررسی استفاده می‌شود.

PCA:هدف پیدا کردن یک نگاشت خطی برای فضای فعلی به فضایی است که هر دسته بیشترین فاصله را داشته باشد.

$$L = a_2^T S a_2 - \lambda(a_2^T a_2 - 1) - \phi a_2^T a_1$$

$$\Downarrow$$

$$\frac{\partial}{\partial a_2} L = Sa_2 - \lambda a_2 - \phi a_1 = 0 \Rightarrow \phi = 0$$

$$\Downarrow$$

$$Sa_2 = \lambda a_2 \quad \text{and} \quad \lambda = a_2^T S a_2$$

که L لاگرانژی براساس معادله اصلی است و مشاهده می‌شود که درنهایت به مقادیر و بردارهای ویژه هر دسته مارا می‌رساند.

LDA: هدف پیدا کردن یک projection vector است به طوری که کوواریانس هر دسته را کم و میانگین بین دسته‌ای را زیاد کند.

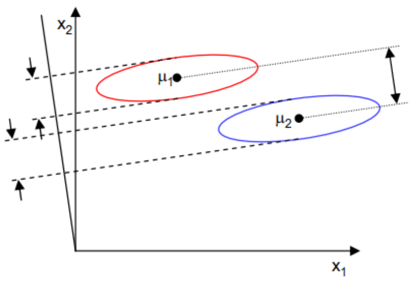
- **The solution proposed by Fisher is to maximize a function that represents the difference between the means, normalized by a measure of the within-class scatter**
  - For each class we define the <u>scatter</u>, an equivalent of the variance, as

$$\tilde{s}_i^2 = \sum_{y \in \omega_i} (y - \tilde{\mu}_i)^2$$

    - where the quantity $\left(\tilde{s}_1^2 + \tilde{s}_2^2\right)$ is called the <u>within-class scatter</u> of the projected examples
  - The Fisher linear discriminant is defined as the linear function $\mathbf{w}^T\mathbf{x}$ that maximizes the criterion function

$$J(w) = \frac{\left|\tilde{\mu}_1 - \tilde{\mu}_2\right|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

  - Therefore, we will be looking for a projection where examples from the same class are projected very close to each other and, at the same time, the projected means are as farther apart as possible



- Similarly, we define the mean vector and scatter matrices for the projected samples as

$$\tilde{\mu}_i = \frac{1}{N_i}\sum_{y \in \omega_i} y \qquad \tilde{S}_W = \sum_{i=1}^{C}\sum_{y \in \omega_i}(y - \tilde{\mu}_i)(y - \tilde{\mu}_i)^T$$

$$\tilde{\mu} = \frac{1}{N}\sum_{\forall y} y \qquad \tilde{S}_B = \sum_{i=1}^{C} N_i(\tilde{\mu}_i - \tilde{\mu})(\tilde{\mu}_i - \tilde{\mu})^T$$

- From our derivation for the two-class problem, we can write

$$\tilde{S}_W = W^T S_W W$$
$$\tilde{S}_B = W^T S_B W$$

- Recall that we are looking for a projection that maximizes the ratio of between-class to within-class scatter. Since the projection is no longer a scalar (it has C-1 dimensions), we then use the determinant of the scatter matrices to obtain a scalar objective function:

$$J(W) = \frac{\left|\tilde{S}_B\right|}{\left|\tilde{S}_W\right|} = \frac{\left|W^T S_B W\right|}{\left|W^T S_W W\right|}$$

- And we will seek the projection matrix W* that maximizes this ratio

- It can be shown that the optimal projection matrix W* is the one whose columns are the eigenvectors corresponding to the largest eigenvalues of the following generalized eigenvalue problem

$$\boxed{W^* = \left[w_1^* \mid w_2^* \mid \cdots \mid w_{C-1}^*\right] = \operatorname{argmax}\left\{\frac{\left|W^T S_B W\right|}{\left|W^T S_W W\right|}\right\} \Rightarrow (S_B - \lambda_i S_W)w_i^* = 0}$$

مشاهده می‌شود که دوباره به مقادیر و بردارهای ویژه هردسته رسیدیم.

سوال پنجم

ج)

$$\dot{x} = Ax + Bu, y = Cx \qquad (3.$$

$$x = y = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix}, u = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}, A = \begin{bmatrix} -0.0085 & 0 & 0.0085 \\ 0 & -0.00195 & 0.0084 \\ 0.0085 & 0.0084 & -0.0169 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.0065 & 0 \\ 0 & 0.0065 \\ 0 & 0 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

$$\dot{x} = (A + \Delta A_F)\,x + Bu + E_f f, y = Cx + F_f f$$

$$E_f = \begin{bmatrix} 0 & B \end{bmatrix} \in \mathcal{R}^{3\times5}, F_f = \begin{bmatrix} I_{3\times3} & 0 \end{bmatrix} \in \mathcal{R}^{3\times5}.$$

البته مقدار ΔA_F با فرض سوال صفر می‌باشد.

شرط زیر بررسی می‌گردد:

$$\frac{rank(W_o)}{rank(C)} \leq s_0 \leq rank(W_o) - rank(C) + 1$$

```
Hos = obsv(A,C)
```

untitled3.mlx | Hos = 9×3

	1	2	3
1	1.0000	0	0
2	0	1.0000	0
3	0	0	1.0000
4	-0.0085	0	0.0085
5	0	-0.0019	0.0084
6	0.0085	0.0084	-0.0169
7	0.0001	0.0001	-0.0002
8	0.0001	0.0001	-0.0002
9	-0.0002	-0.0002	0.0004

```
R_Hos = rank(Hos)
```

R_Hos = 3

```
R_C = rank(C)
```

R_C = 3

```
s_min = R_Hos / R_C
```

s_min = 1

```
s_max = R_Hos + R_C-1
```

s_max = 5

درنتیجه مقدار s=1 در نظر گرفته می‌شود.

```
Hos = [C ; C*A]
```

Hos = 6×3

1.0000	0	0
0	1.0000	0
0	0	1.0000
-0.0085	0	0.0085
0	-0.0019	0.0084
0.0085	0.0084	-0.0169

```
Hus = [D zeros(3,2) ; C*B D]
```

Hus = 6×4

0	0	0	0
0	0	0	0
0	0	0	0
0.0065	0	0	0
0	0.0065	0	0
0	0	0	0

```
Hfs = [Ff zeros(3,2) ; C*Ef Ef]
```

Hfs = 6×6

1.0000	0	0	0	0	0
0	1.0000	0	0	0	0
0	0	1.0000	0	0	0
0	0.0065	0	0	0.0065	0
0	0	0.0065	0	0	0.0065
0	0	0	0	0	0

حال شرط مطابق سوال ۲هماهنگ شده راه بررسی می‌کنیم.

```
% PDD check
R1 = rank([Hos Hfs])
```

```
R1 = 6
```

```
R2 = rank([Hos])
```

```
R2 = 3
```

```
if R1 > R2
    display('PDD is ok')
else
    display('PDD is not ok')
end
```

```
PDD is ok
```

همانطور که مشاهده می‌شود شرط برآورده شده است. حال بردارهای پریتی را بدست می‌آوریم:

```
% calculation of parity vector
[a,b,c] = svd(Hos);
NB = a(:,length(a))';
[ps,J] = eig(NB*NB',NB*(Hfs*Hfs')*NB');
Vs = ps*NB
```

```
Vs = 1×6
    -0.4106   -0.4059    0.8165    0.0052    0.0038   48.3169
```

```
norm(Vs*Hos)
```

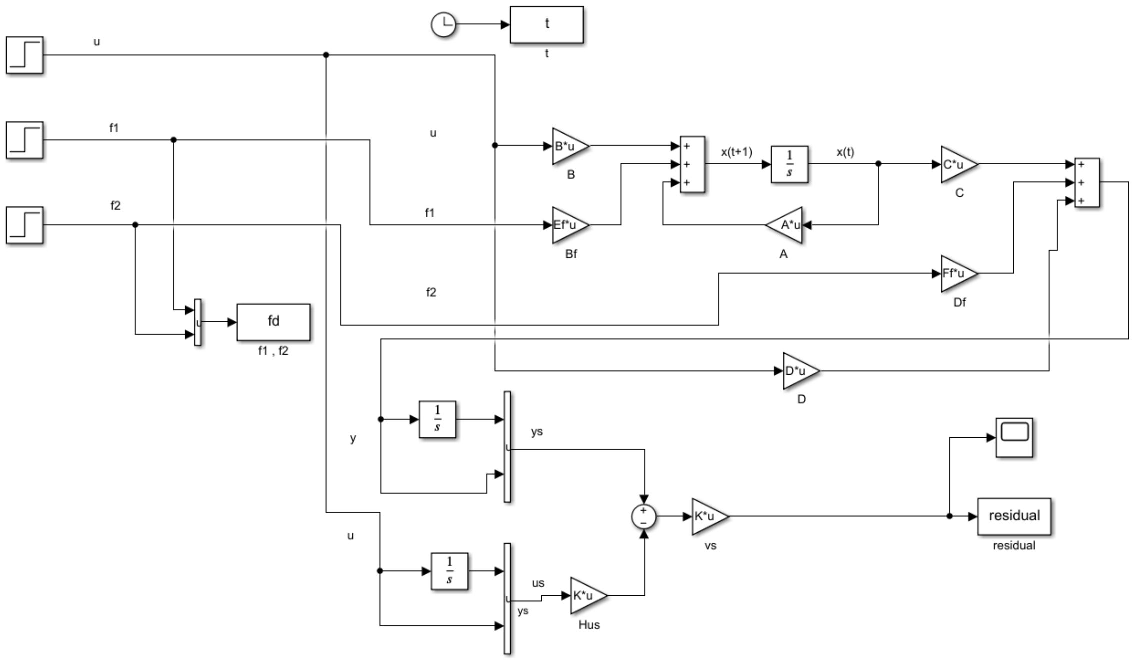
```
ans = 1.3597e-16
```

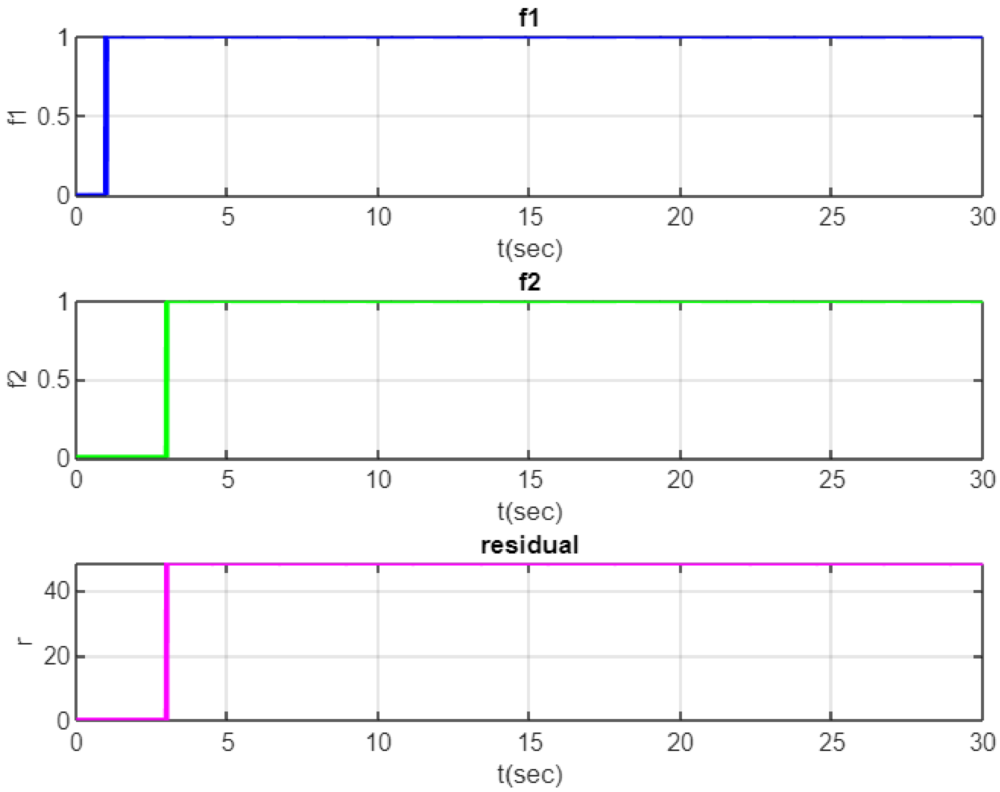
```
norm(Vs*Hus)
```

```
ans = 4.2033e-05
```

```
norm(Vs*Hfs)
```

```
ans = 1
```





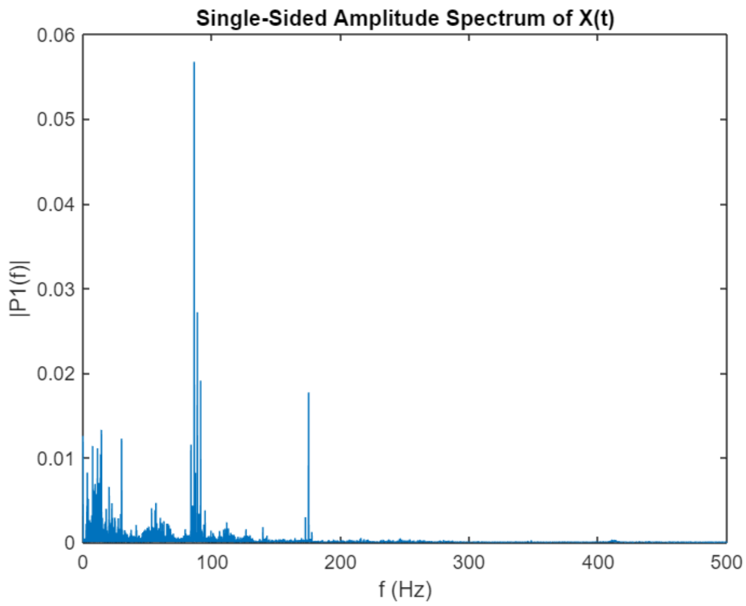
مشاهده می‌شود که سیستم f2 را به موقع تشخیص داده است. هرچند که این حالت صرفا توانسته است f2 را پیدا کند.

سوال ششم

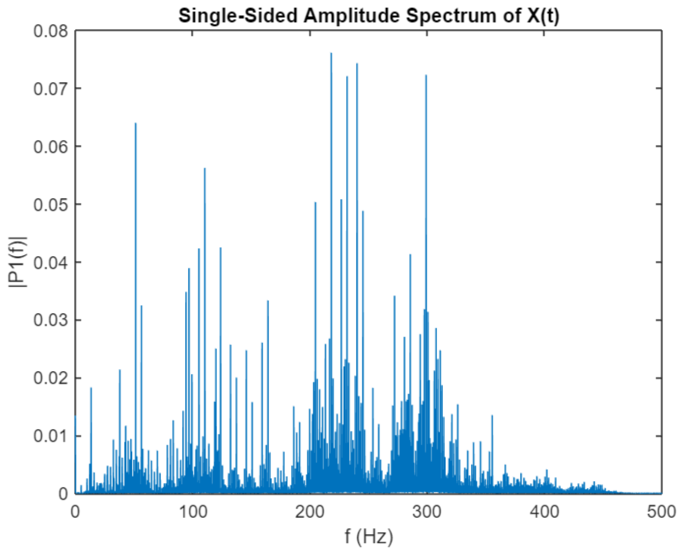
https://colab.research.google.com/drive/1hb6HuhIkENgjKMRUbGKEQvJ0tekpmBD2?usp=sharing

ابتدا در نرم افزار متلب نمودار داده ها بعد از گذر از fft را رسم می‌کنیم.

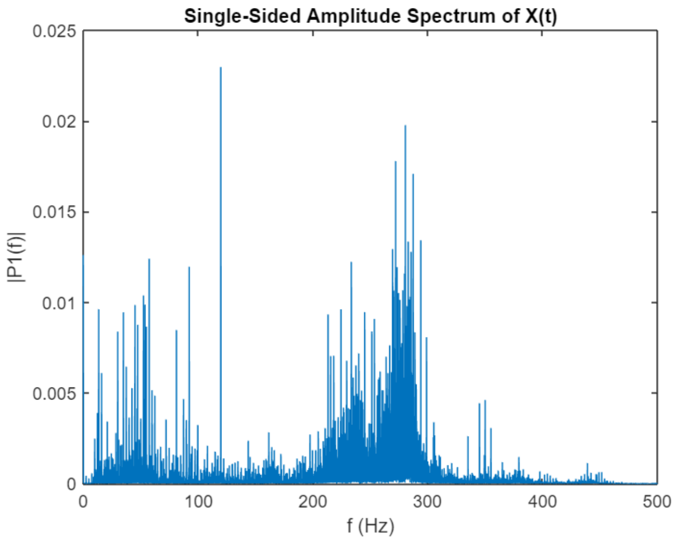
```
normal=[X097_DE_time,X097_FE_time]
n=fft(normal)
L = 243938;
Fs = 1000;
P2 = abs(n/L);
P1 = P2(1:L/2+1);
P1(2:end-1) = 2*P1(2:end-1);
f = Fs*(0:(L/2))/L;
plot(f,P1)
title('Single-Sided Amplitude Spectrum of X(t)')
xlabel('f (Hz)')
ylabel('|P1(f)|')
```



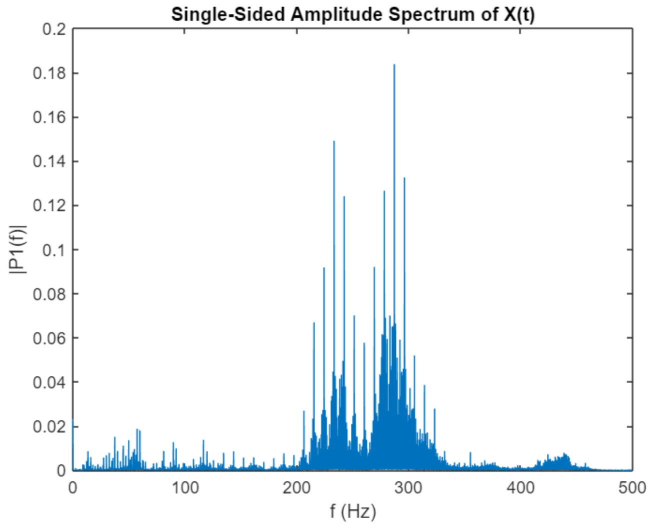
شکل 1 - داده نرمال



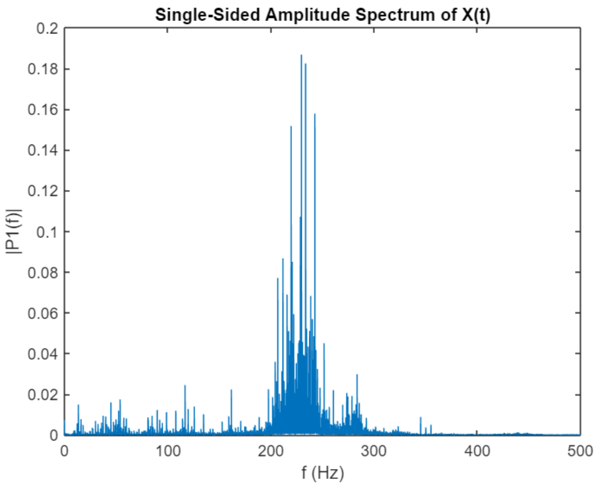
شکل**2** - داده خطای ۱۰۵



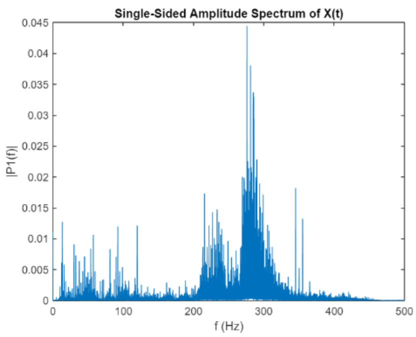
شکل**3** - داده خطای ۱۱۸



شکل4 - داده خطای ۱۳۰



شکل5 - داده خطای ۱۴۴



شکل6 - داده خطای ۱۵۶

همانطور که دیده می‌شود، فرکانس های مشابه زیادی بین داده ها خصوصا داده های خطا وجود دارد. البته هنوز این امکان وجود دارد که بتوان داده نرمال را از خطا تشخیص داد. داده نرمال در فرکانس های

پایین تری نسبت به داده های خطا قرار دارد و در این ناحیه خطا های کمتری وجود دارد. به علاوه اگر بر روی داده های خطا یک نگاشت ساده خطی اعمال شود (برای این مورد LDA می‌تواند گزینه خوبی باشد) به راحتی قابلیت جداسازی پیدا می‌کنند.

ب)

از کدهایی که برای تمرین سری ۳ ارائه شده بود استفاده شد. در این قسمت به دلیل اینکه داده ها هر کدام دارای اندازه های متفاوتی بودند، دوباره فرایند زیر اجرا شد البته با داده های بیشتر.

```
mat = scio.loadmat('data/97.mat')
variables = scio.whosmat('data/97.mat')
# print(variables)

# Choose the appropriate key for your data based on the output of the previous step
data1 = mat['X097_DE_time']
data2 = mat['X097_FE_time']
# Convert the data to a DataFrame
df1 = pd.DataFrame(data1)
df2 = pd.DataFrame(data2)
normal=[]
for i in range(200):
    a=np.concatenate([df1.values[i*400:400+i*400], df2.values[i*400:400+i*400]],axis=1)
    normal.append(a)

normal=np.squeeze(normal)
```

حال ویژگی‌های زیر برای هر داده جداگانه استخراج شد. نکته مهمی که در این جا وجود دارد این است که به دلیل اینکه داده های خروجی fft دارای مقادیر موهومی هستند از آن‌ها یک rms می‌گیریم تا فقط مقدار حقیقی داشته باشد. در انتها نیز تمامی داده ها را با هم concatenate میکنیم.

```
std_normal= normal.std(axis=1)
peak_normal= normal.max(axis=1)
from scipy.stats import skew, kurtosis
skewness_normal= skew(normal, axis=1)
kurtosis_normal= kurtosis(normal, axis=1)
crest_factor_normal = np.max(normal, axis=1) / np.sqrt(np.mean(normal**2, axis=1))
ptp_normal= np.ptp(normal, axis=1)
mean_normal= np.mean(normal, axis=1)
rms_normal = np.sqrt(np.mean(normal**2, axis=1))
abs_mean_normal= np.mean(np.abs(normal), axis=1)
nor_fft_rms=np.sqrt(np.mean((np.fft.fft(normal))**2, axis=1))
feature_normal=np.concatenate([std_normal, peak_normal, skewness_normal, kurtosis_normal, crest_factor_normal, ptp_normal, mean_normal, rms_normal, abs_mean_normal,nor_fft_rms],axis=1)
```

به صورت خیلی ساده برای هر دسته به صورت زیر برچسب تعیین میکنیم.

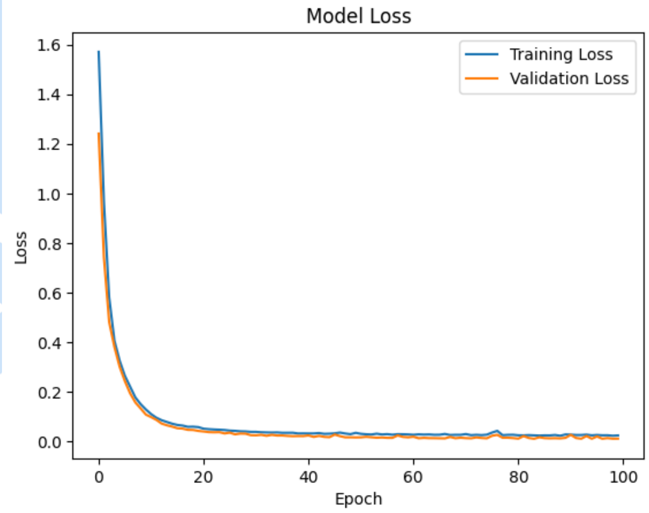
```
X=np.concatenate([feature_normal, feature_f105, feature_f118, feature_f130, feature_f144, feature_f156])
y = np.concatenate((np.zeros(len(feature_normal)), np.ones(len(feature_f105)), 2*np.ones(len(feature_f118)), 3*np.ones(len(feature_f130)),4*np.ones(len(feature_f144)),5*np.ones(len(feature_f156))))
```

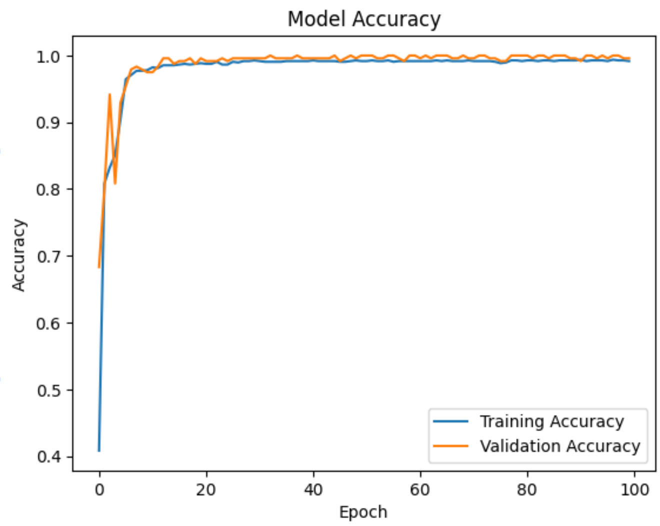
تنها نکته مهم در ادامه این است که باید از loss= sparse_categorical_crossentropy استفاده شود.

ساختار شبکه به صورت زیر می‌باشد.

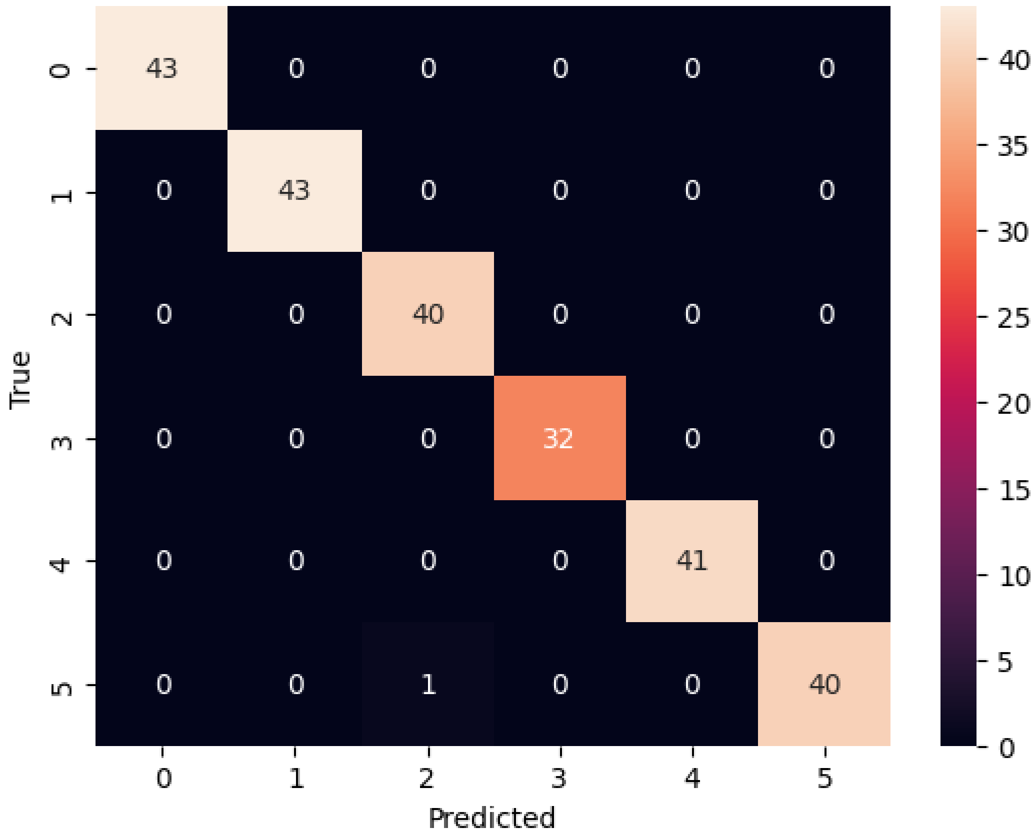
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 64)                1344

 dense_1 (Dense)             (None, 64)                4160

 dense_2 (Dense)             (None, 6)                 390

=================================================================
Total params: 5,894
Trainable params: 5,894
Non-trainable params: 0
_________________________________________________________________
```

خروجی های زیر بدست آمده اند.






```
8/8 [==============================] - 0s 791us/step
Test AUC: 0.9975508130081302
Test Recall: 0.9958333333333333
Test F1-score: 0.9958333333333333
Test Precision: 0.9958333333333333
```



شبکه دارای ۳ لایه(۱ ورودی، ۱ مخفی و ۱ خروجی) می‌باشد. به دلیل اینکه هدف ما دسته بندی است توابع فعال ساز دولایه اول relu و لایه خروجی softmax می‌باشد. نرخ آموزش شبکه ۰.۰۰۱ و تعداد epoch ها ۱۰۰ تا بوده است. همانطور که مشاهده می‌شود شبکه نه overfit شده و نه کم آموزش دیده. ماتریس درهم ریختگی نیز نشان می‌دهد که شبکه توانایی کامل برای جداسازی داده‌ها را دارد.