به نام خدا

Blind Source Separation (BSS)

تکلیف شماره

10

محمدرضا آرانی

810100511

دانشگاه تهران

1402/03/21

جدول محتویات

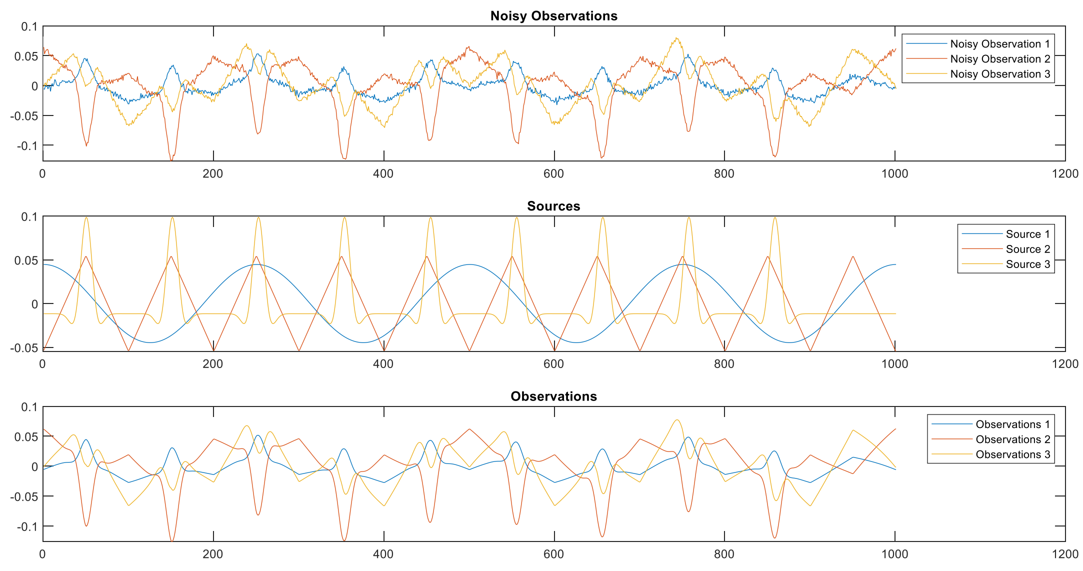
بخش اول:

در این تمرین می خواهیم جداسازی کور منابع را با فرض استقلال منابع حل کنیم.

ماتریس مخلوط کننده A ، ماتریس منابع S و ماتریس Noise در فایل hw10.mat در اختیار شما قرار داده شده است. ابتدا ماتریس مشاهدات X را با رابطه ی X = A S + Noise به دست آورید. هم منابع و هم مشاهدات بدون نویز و هم مشاهدات نویزی را رسم کنید تا ظاهر آنها را ببینید. حال به دید جداسازی کور منابع به مساله نگاه کنید. در واقع فرض می کنیم فقط ماتریس X را داریم و تعداد منابع را هم می دانیم. استراتژی ما این خواهد بود که با ضرب یک ماتریس جدا کننده B در ماتریس Z که شامل داده های سفید شده است، خروجی هایی تولید کنیم که این خروجی ها تا حد ممکن غیر گوسی باشند.

۱- روش اولی که در کلاس مبتنی بر بیشینه سازی Kurt خروجی بیان شد را پیاده سازی کنید. در واقع با رویکرد deflation سطرهای ماتریس B را یک به یک استخراج کنید و با الگوریتم Gradient Projection (GP) بهینه سازی های مرتبط را انجام دهید. توجه داشته باشید ماتریس جدا کننده ی نهایی شما حاصلضرب ماتریس orthonormal نهایی به دست آمده از الگوریتم، در ماتریس سفید کننده است.

متناسب با درخواست اولیه‌ی مسئله، نمودار منابع، مشاهدات و مشاهدات بدون نویز در ابتدا رسم شده است:



شکل 1

در این تکلیف، در واقع *approach* دوم مطرح شده در درس را پیاده‌سازی خواهیم کرد. *Approach* اولی برای مستقل کردن خروجی‌ها از یکدیگر بود که با استفاده از *Kull back Divergence* مطرح شد. در اینجا معیار ما غیرگوسی کردن خروجی‌ها تا حد امکان است. برای اینکار از معیار جدید معرفی شده در درس تحت عنوان *kurtosis* استفاده می‌کنیم:

In probability theory and statistics, **kurtosis** (from Greek: κυρτός, *kyrtos* or *kurtos*, meaning "curved, arching") is a measure of the "tailedness" of the probability distribution of a real-valued random variable. Like skewness, kurtosis describes a particular aspect of a probability distribution. There are different ways to quantify kurtosis for a theoretical distribution, and there are corresponding ways of estimating it using a sample from a population. Different measures of kurtosis may have different interpretations.

The standard measure of a distribution's kurtosis, originating with Karl Pearson,[1] is a scaled version of the fourth moment of the distribution. This number is related to the tails of the distribution, not its peak;[2] hence, the sometimes-seen characterization of kurtosis as "peakedness" is incorrect. For this measure, higher kurtosis corresponds to greater extremity of deviations (or outliers), and not the configuration of data near the mean.

$$\operatorname{Kurt}[X] = \operatorname{E}\left[\left(\frac{X-\mu}{\sigma}\right)^4\right] = \frac{\operatorname{E}\left[(X-\mu)^4\right]}{\left(\operatorname{E}\left[(X-\mu)^2\right]\right)^2} = \frac{\mu_4}{\sigma^4},$$

با این تعریف، سعی می‌کنیم که خروجی‌ها را سنجیده و بهینه کنیم. در واقع این معیار، همان *feature*یی است که میزان گوسی بودن یک توزیع را مشخص می‌کند. هرچه این مقدار به 0 نزدیکتر باشد، توزیع ما به گوسی بودن نزدیکتر است. پس هدف ما می‌تواند *Maximization* تابع هدف زیر باشد:

$$f(b) = |kurt(y)|; y = b^T z;$$

تعریف این تابع هدف به صورت زیر خواهد بود:

$$kurt(y) = E\{y^4\} - 3(E\{y^2\})^2;$$

$$\frac{\partial}{\partial b} f = sign\big(kurt(b^T z)\big)[E\{z * (b^T z)^3\} - 3 * b]$$

و به این ترتیب به دلیل *maximization* قاعده‌ی به روز رسانی به صورت زیر است:

$$b = b + \mu \frac{\partial}{\partial b} f;$$

$$b = \frac{b}{|b|_2}$$

البته توجه شود که قبل از نرمالایز کردن سطر/ستون b باید تصویر آن را از b های قبلی کم کنیم! این کار به کمک رابطه‌ی زیر انجام می‌شود:

```
  for i=1:length(B) % Each Row

         StepSize = sign(kurtosis(B(i,:)*Z))*((Z*(B(i,:)*Z)'.^3)'/length(Z) -3*B(i,:)  ) ;
        %  StepSize = normalize(StepSize,2,"norm");
         B(i,:)        =  B(i,:) + mu*StepSize  ;
         B(i,:)        =  B(i,:)/norm(B(i,:));   % Normalization
         B(i,:)        =  ( eye(size(B)) - B(1:i-1,:)'*B(1:i-1,:)   )* B(i,:)';  %
Orthogonality

    end
```

قسمت-1:

۱-۱- ماتریس جدا کننده ای که در نهایت به دست آوردید را در ماتریس مخلوط کننده ی اصلی ضرب کنید و حاصل را گزارش کنید. ماتریس حاصل باید نزدیک به یک ماتریس permutation باشد به این معنی که در هر سطر و هر ستون فقط یک مقدار غیر صفر داشته باشد.

با استفاده از ساختار کد قبلی و با تغییر منطق به روزرسانی و update کردن Bها، این مسئله به سادگی انجام شد و همگرایی آن بسیار سریعتر از روش‌های قبلی بود.

در نهایت ماتریس مورد نظر که باید شبیه یک ماتریس *Permutation* باشد به صورت زیر خواهد بود:

```
disp("calculated Error Equals to: "+min(Error_Iter_deflate))
calculated Error Equals to: 0.38044
disp(abs(B_hat_best*W*U'*A))
    0.9532    0.2622    0.2230
    0.2366    1.1174    0.4295
    0.1302    0.1319    1.0315
```

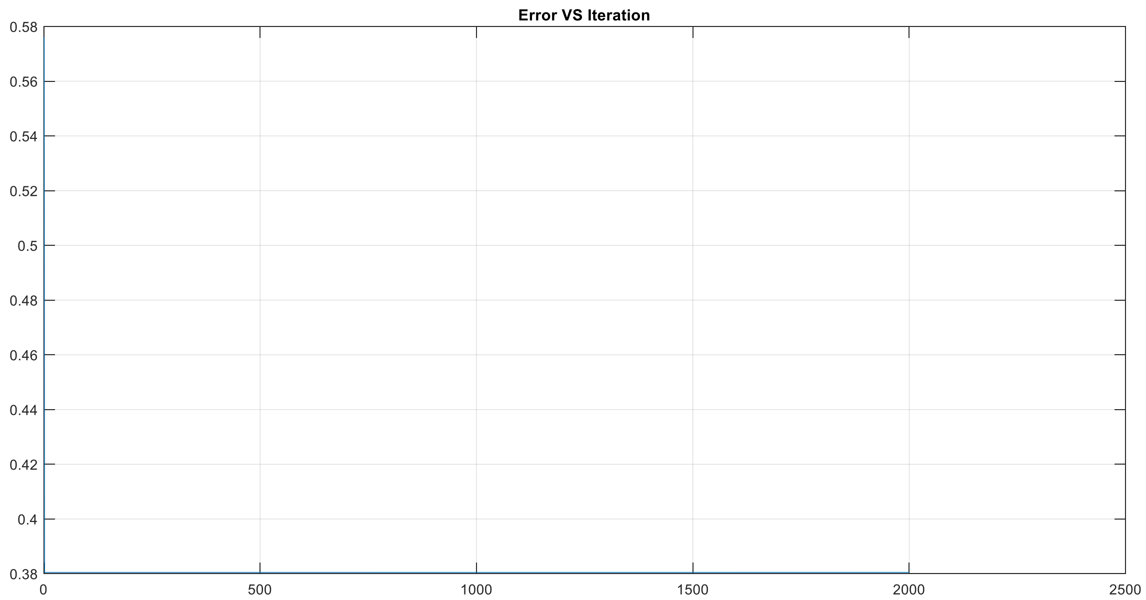
قسمت-2:

۱-۲- ابهام ترتیب و همچنین ابهام scale منابع را برطرف کرده به این معنی که انرژی منابع تخمین زده شده را مساوی انرژی منابع اصلی کنید. منابع تخمین زده شده را روی منابع اصلی رسم کنید. و سپس مقدار خطای زیر را گزارش کنید.

$$E = \frac{\|\hat{S} - S\|_F^2}{\|S\|_F^2}$$

مانند روش‌های قبلی، تمامی حالات محاسبه شده و با رفع ابهام *scale* و *permutation* منابع، خطا محاسبه گردیده است.

نمودار خطا بر حسب تعداد *Iteration* به صورت زیر خواهد بود:

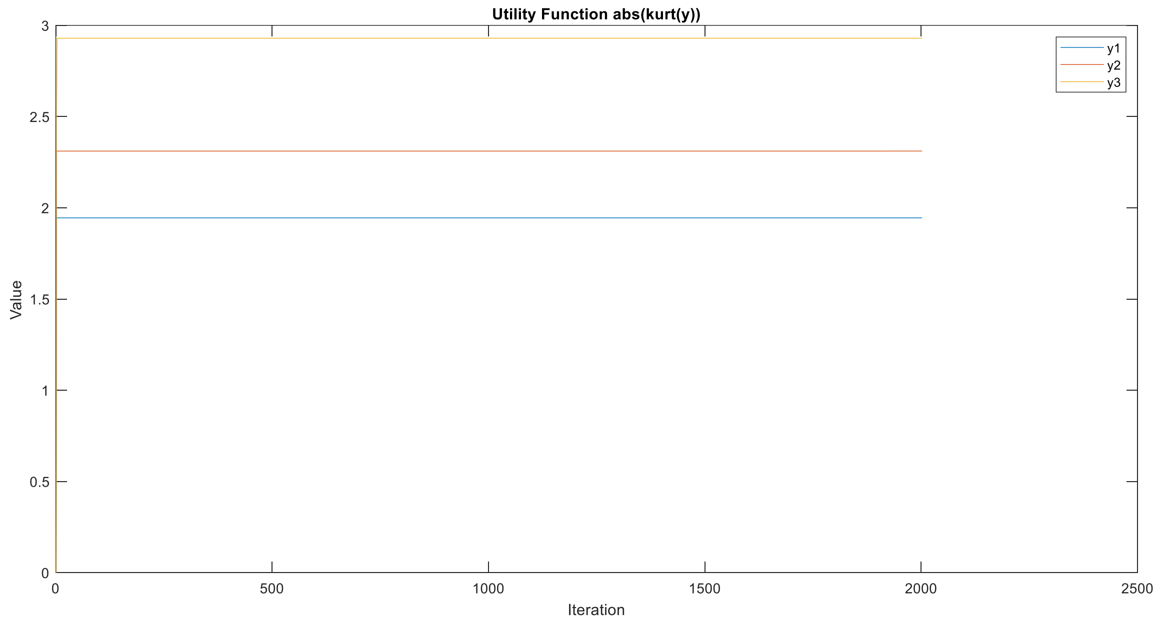


شکل 2

همگرایی تقریبا بعد از 2 یا 3 تکرار رخ داده است!

قسمت-3:

۳-۱- نمودار همگرایی (تابع هدف بر حسب شماره ی iteration) را رسم کنید



شکل 3

$$F(b) = |kurt(y)|$$

مقدار تابع *utility* ما به ازای *Run*های مختلف

بخش دوم:

۲- روش دومی که در کلاس ارائه شد و با رویکرد fixed-point بیشینه سازی Kurt را انجام می داد پیاده سازی کنید (FAST ICA). همه ی نتایج را مشابه قسمت ۱ گزارش کنید.

با توجه به اینکه نقطه‌ی بهینه زمانی اتفاق می‌افتد که مشتق قیدها با مشتق تابع در آن نقطه برابر باشد، پس نقطه‌ی بهینه در این بهینه‌سازی زمانی است که:

* Fixed-Point

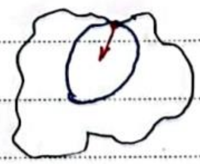
* رویه‌ی دوم برای $\boxed{\text{Maximum}}$ کردن همین "Kurt" انجام میشه

$$\frac{\partial f}{\partial b} = \lambda \frac{\partial g}{\partial b}$$

نقطه‌ی بهینه زمانی اتفاق می‌افتد که مشتق قیدها با

$$g(b) = b^T b - 1 \rightarrow \text{unit norm}$$ بودن

مشتق تابع یکی باشند!



در یک جهت باشند!

$$b = \frac{b}{\|b\|}$$

$$\boxed{b_2^T b_1 = 0}$$

$$b \leftarrow E\{z (b^T z)^3\} - 3b$$

$$b = \frac{b}{\|b\|_2}$$

سرعتش خیلی بهتره از GP

$$|b^{(i+1)T} b^{(i)}| = 1 \rightarrow$$ همگرایی

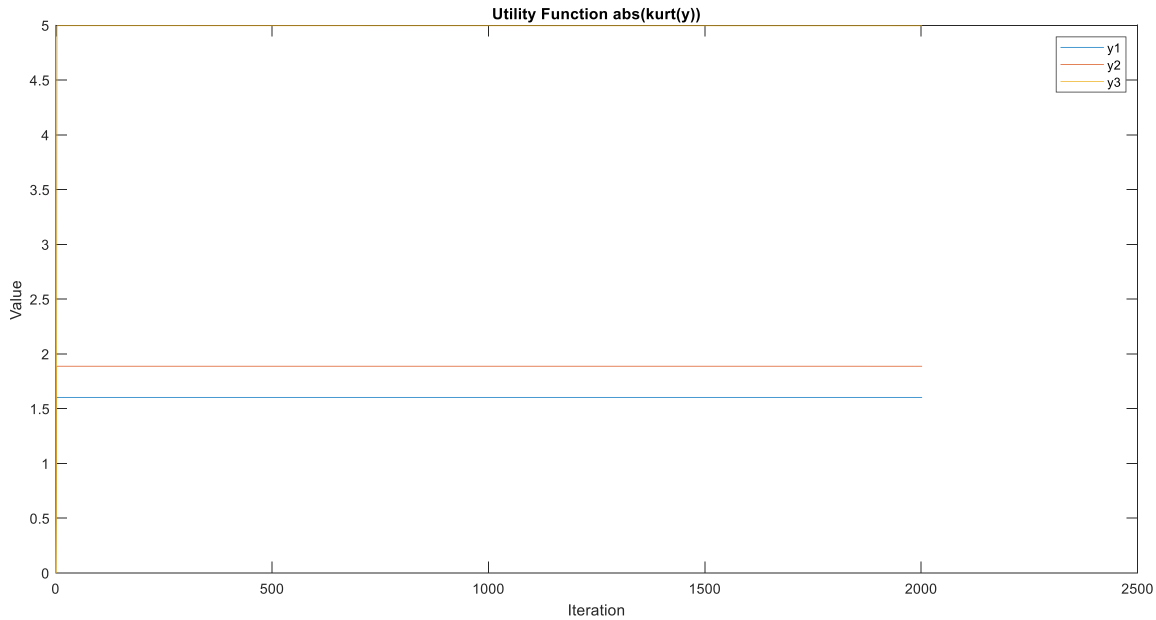
شکل 4

با تغییری جزیی در قاعده‌ی به‌روزرسانی مانند قبل و استفاده از روش عددی نقطه‌ی ثابت، به همگرایی نرم تر و سریعتری خواهیم رسید:

```
disp("calculated Error Equals to: "+min(Error_Iter_deflate))
calculated Error Equals to: 0.30849
disp(abs(B_hat_best*W*U'*A))
    0.9895    0.0259    0.0689
    0.0204    1.0942    0.2385
    0.0440    0.3701    1.1142
```

نمودار تابع *Utility* ما به صورت زیر است:

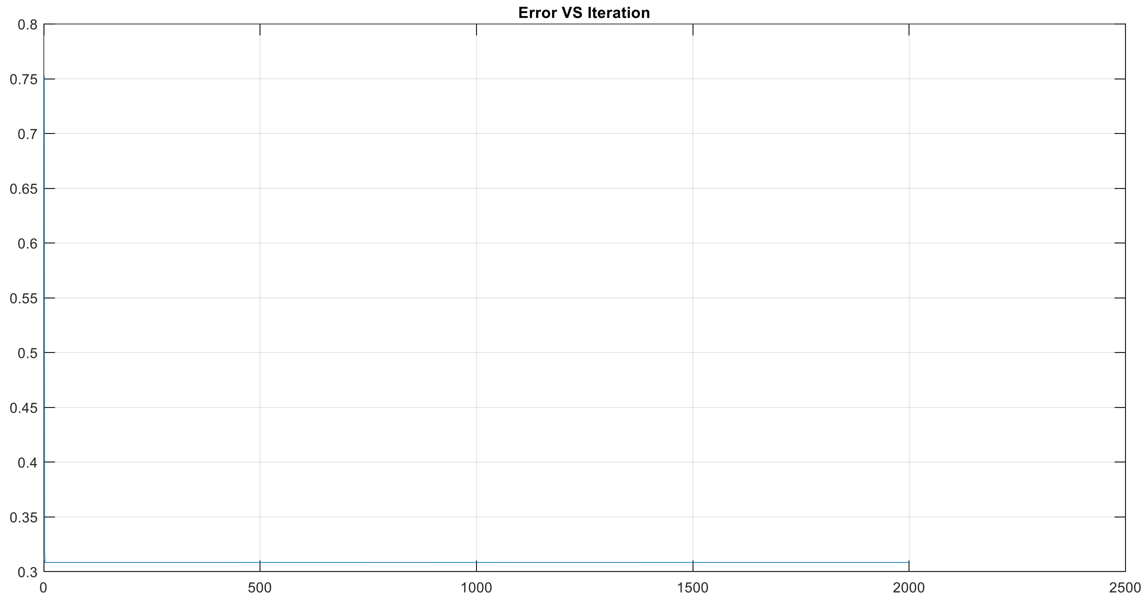
```
figure()
plot(Utility')
title("Utility Function abs(kurt(y))")
xlabel("Iteration")
ylabel("Value")
legend("y1","y2","y3")
```



شکل 5

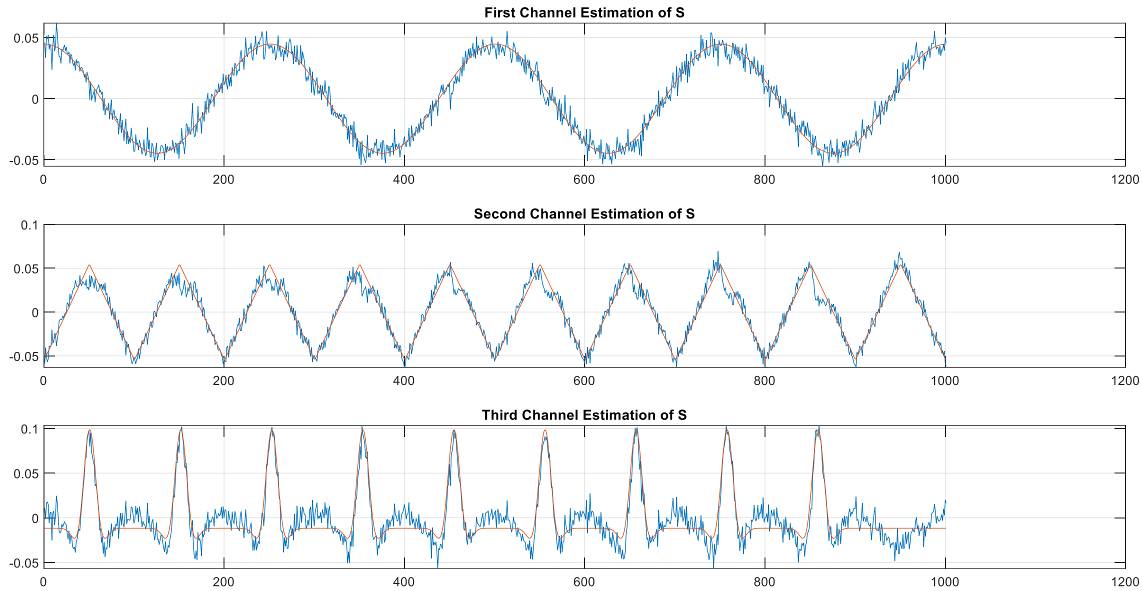
همچنین نمودار خطا بر حسب تعداد تکرار به فرم زیر درخواهد آمد:

```
figure()
plot(Error_Iter_deflate)
grid on
title("Error VS Iteration")
```



شکل 6

سیگنال منابع به صورت بازسازی شده به فرم زیر درخواهند آمد:



شکل 7

بازسازی موفق سیگنال منابع متناسب با هر کانال

1

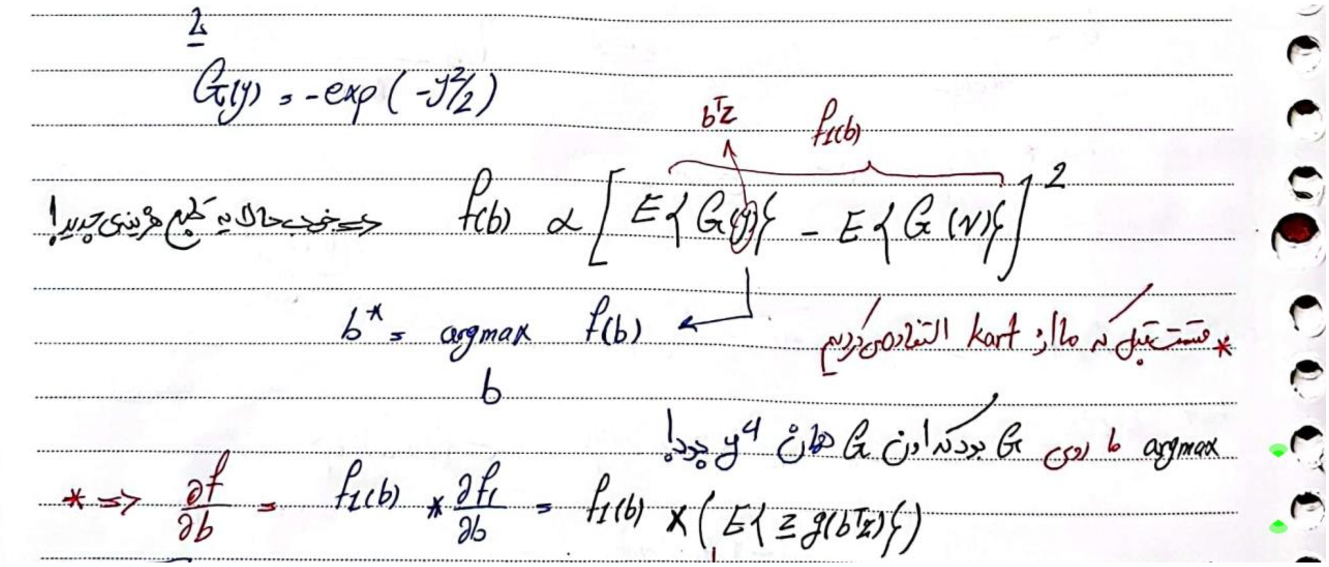
بخش سوم:

۳- روش سومی که در کلاس ارائه شد و به داده ی outlier حساس نبود را با فرض $G(y) = -\exp\left(-\frac{y^2}{2}\right)$ و با فرض استفاده از GP در بهینه سازی ها پیاده سازی کنید. همه ی نتایج را مشابه قسمت ۱ گزارش کنید.

در این روش، به جای *kurt* از تابع دیگری که نسبت به *outlier*ها حساس نباشد استفاده می‌کنیم. در این مثال، تابع داده شده برابر:

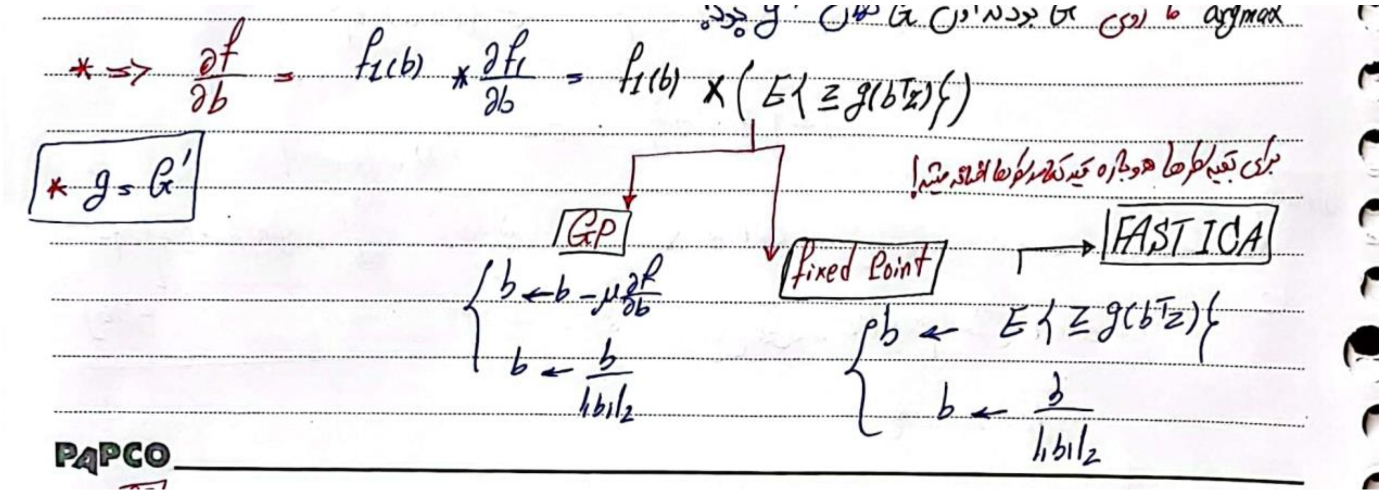
$$G(y) = -e^{-\frac{y^2}{2}}$$

است.



شکل 8

مشتق این تابع هزینه به صورت زیر محاسبه می‌شود:



شکل 9

با استفاده از این تابع،حساسیت ما به *outlier*ها کمتر شده و به درستی مدل را تعیین می‌کنیم.

در مدل فوق، $G(\nu)$ درواقع یک متغیر گوسی است که اختلاف تابع فعلی ما یعنی $E\{G(y)\}$ از آن، می‌شود معیار تابع *Utility*ما.

در این مثال تابع $g(b^T z)$ به صورت زیر خواهد بود:

$$g = \frac{\partial G}{\partial y} = y * e^{-\frac{y^2}{2}}$$

با استفاده از این روش، طی 5 تکرار به شرط توقف رسیدیم.

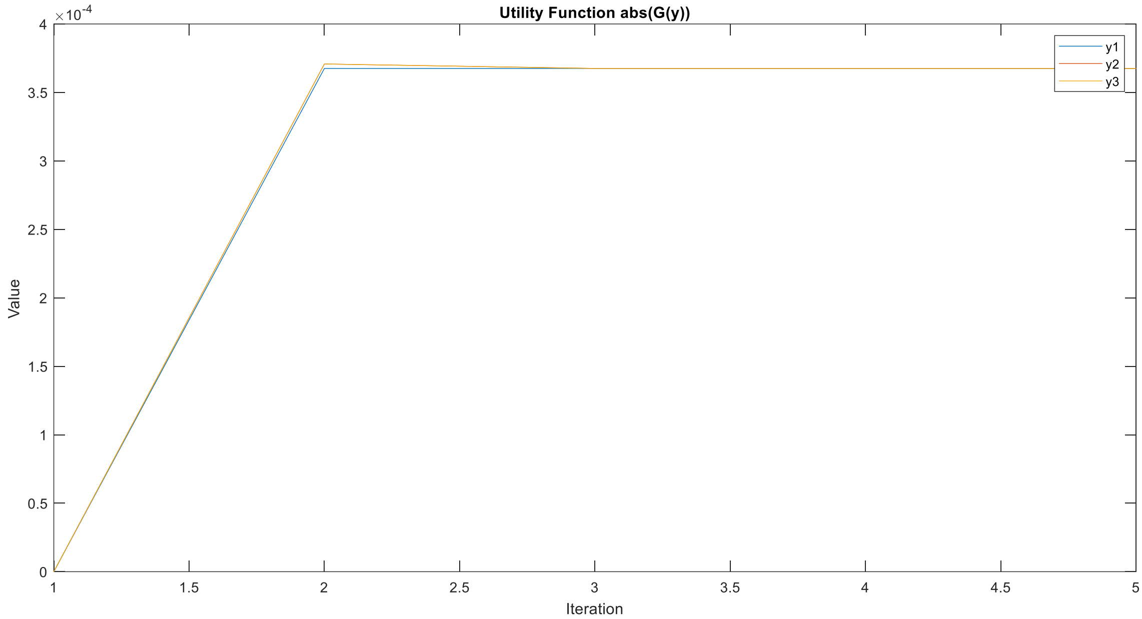
```
        % Update Based on FIxed Point Method
        Part_1_of_Stepsize = F1_Mine(B(i,:)*Z);
        Part_2_of_Stepsize = mean( Z*( (B(i,:)*Z)*exp(-(B(i,:)*Z).^2/2)' )/length(Z) ,2
);
        StepSize      =  Part_1_of_Stepsize*Part_2_of_Stepsize ;
```

همچنین ماتریس نهایی به صورت زیر به همان ماتریس *Permutation* معروف، نزدیک شده است:

```
disp("calculated Error Equals to: "+min(Error_Iter_deflate))
calculated Error Equals to: 0.34033
disp(abs(B_hat_best*W*U'*A))
    0.9714    0.1712    0.1659
    0.1536    1.1226    0.3739
    0.1198    0.2130    1.0657
```

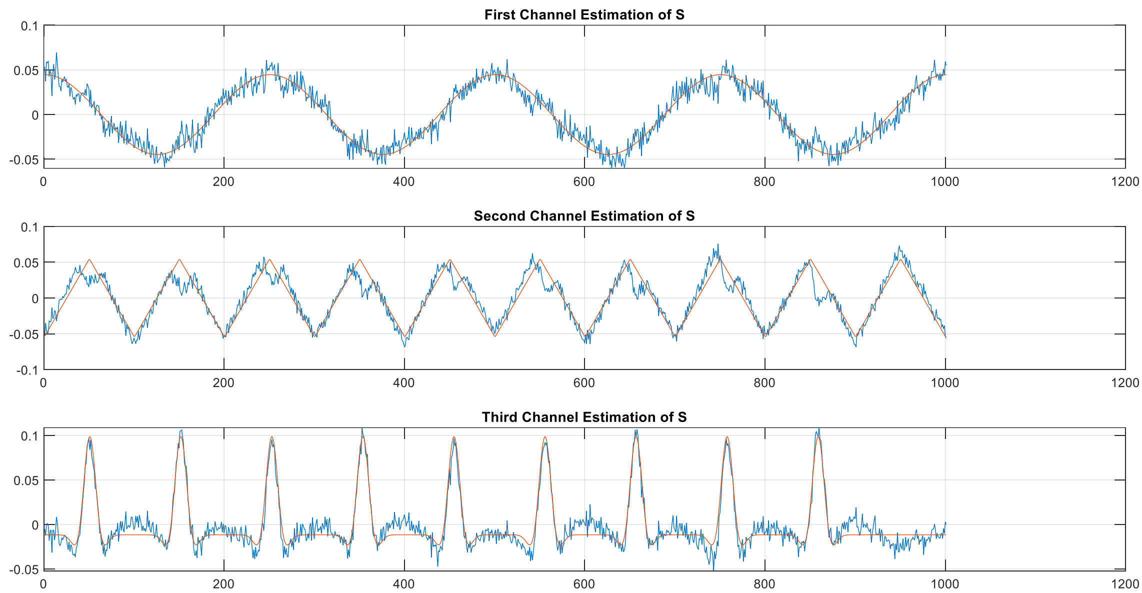
نمودار تابع *Utility* ما بر حسب تکرار به صورت زیر است:

```
[Error_Perm,y_Hat_Chosen,B] = Perm_AMP_Disamb(B,S,Z);
    Error_Iter_deflate(cntr)     = min(Error_Perm);
    y_hat = B*Z;
    New_Utility = [F1_Mine(y_hat(1,:)); F1_Mine(y_hat(2,:)) ; F1_Mine(y_hat(2,:)) ];
    Utility     = [Utility, New_Utility];

…

figure()
plot(Utility')
title("Utility Function abs(G(y))")
xlabel("Iteration")
ylabel("Value")
legend("y1","y2","y3")
```



شکل 10

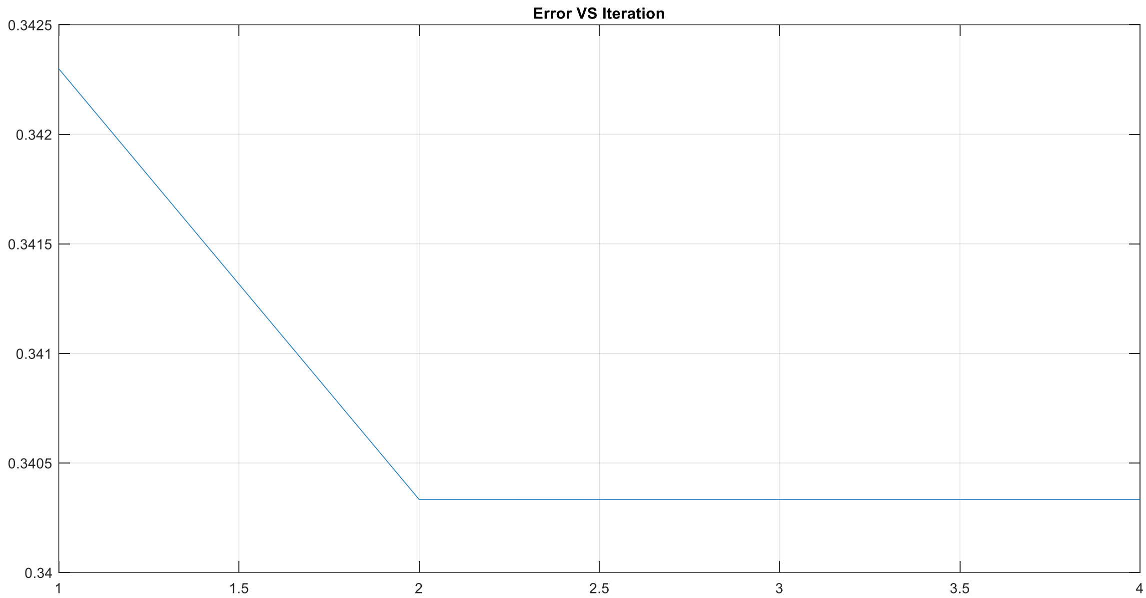
نمودار بازسازی سیگنال منابع نیز آورده شده است:



شکل 11

همگرایی در این شکل نیز مشهود است.

نمودار خطا بر حسب تکرار تا همگرایی نیز به این شکل است:



شکل 12

بخش پهارم:

۴- روش سومی که در کلاس ارائه شد و به داده ی outlier حساس نبود را با فرض $G(y) = -\exp\left(-\frac{y^2}{2}\right)$ و با فرض استفاده از "fixed-point اصلاح شده" (FAST ICA نهایی) در بهینه سازی ها پیاده سازی کنید. همه ی نتایج را مشابه قسمت ۱ گزارش کنید.

با ترکیب این دو روش، یعنی عدم حساسیت به *outlier* ها به همراه استفاده از روش عددی *Fixed Point*، به سرعت همگرا شده و به خروجی زیر می‌رسیم:

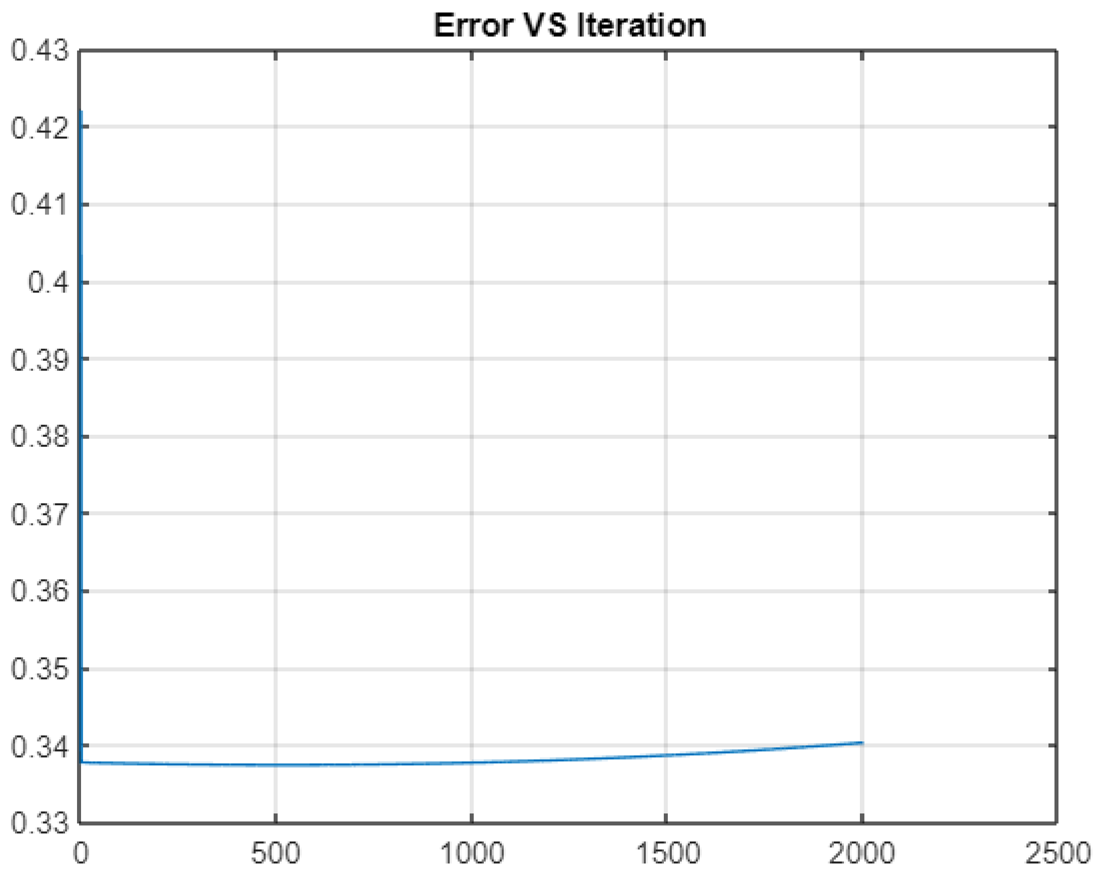
```
disp("calculated Error Equals to: "+min(Error_Iter_deflate))
calculated Error Equals to: 0.33736
disp(abs(B_hat_best*W*U'*A))
    0.9707    0.1861    0.2479
    0.0952    1.1132    0.3289
    0.1742    0.2469    1.0646
```

خروجی به درستی بازیابی شده است و به صورت یک ماتریس *Permutation* ظاهر شده است و تقریبا شرط *Darmoia* رعایت شده است.

تغییرات برای این روش، به فرم زیر بود:

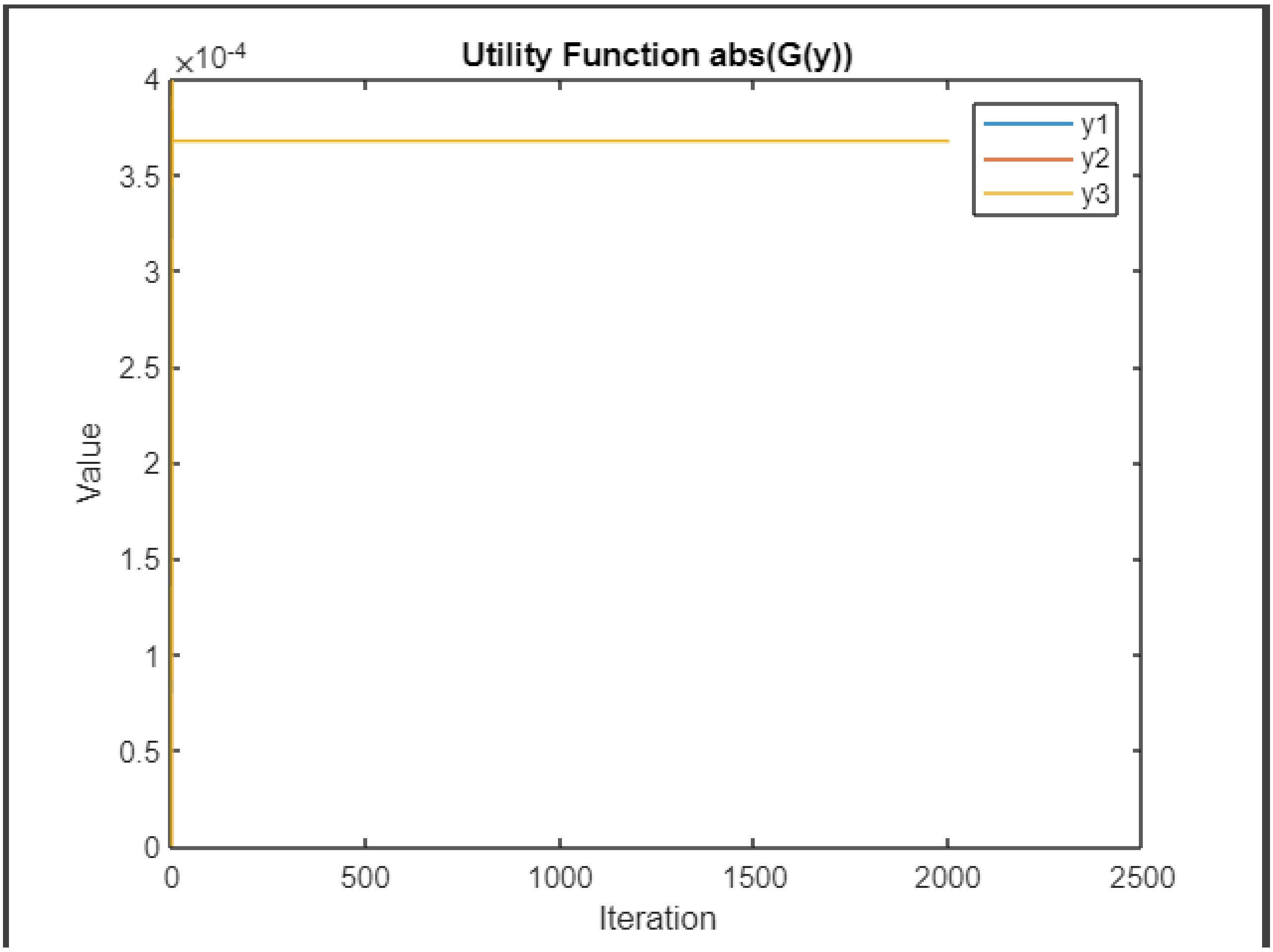
```
Term_1        =  mean( Z*( (B(i,:)*Z)*exp(-(B(i,:)*Z).^2/2)' )/length(Z) ,2 );
Term_2        =  ((B(i,:)*Z).^2*exp(-(B(i,:)*Z).^2/2)' /length(Z))*B(i,:) ;
B(i,:)        =  Term_1 - Term_2';  %B(i,:) + mu*StepSize'  ;
```

نحوه‌ی به روزرسانی همان نحوه‌ی *Fixed Point* است و مقادیری که برای به روزرسانی استفاده می‌شوند، تابع *G(y)* هستند که در اینجا برابر $e^{-\frac{y^2}{2}}$ می‌باشد.



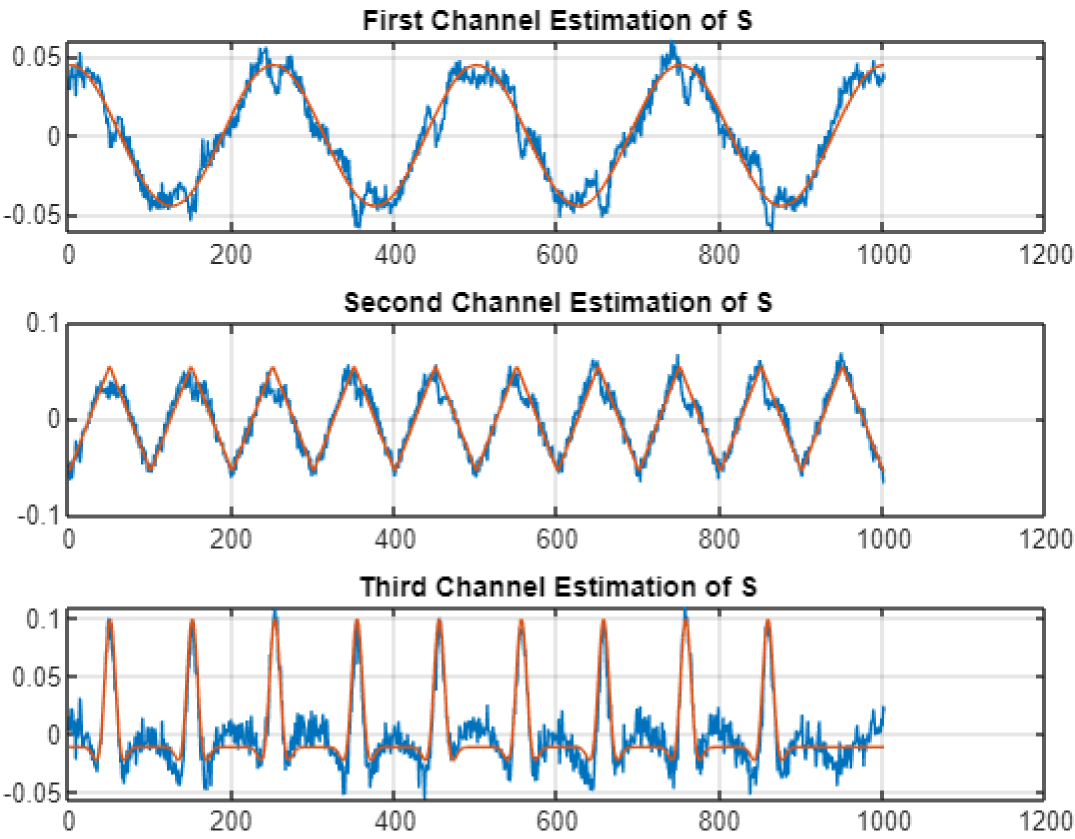
شکل 13

همچنین تابع *Utility*ما به فرم زیر درخواهد آمد:



شکل 14

و در نهایت نمودار بازسازی سیگنال منابع با خطای بازسازی برابر 0.33 در زیر آمده است:


First Channel Estimation of S
0.05
0
-0.05
0
200
400
600
800
1000
1200
Second Channel Estimation of S
0.1
0
-0.1
0
200
400
600
800
1000
1200
Third Channel Estimation of S
0.1
0.05
0
-0.05
0
200
400
600
800
1000
1200

شکل 15

بخش پنجم:

۵- به صورت تجربی و مقایسه ای بیان کنید کدام یک از چهار روش بالا سریعتر همگرا شد و کدام یک کیفیت جداسازی (پارامتر E) بهتری داشت.

با توجه به پارامترهای به دست آمده و به صورت تجربی، به نظر روش *Fixed Point* بسیار سریع تر است در همگرایی تا روش‌های مبتنی بر *GP*.

همچنین باتوجه به اینکه معیار *kurt* نسبت به *outlier*ها حساس است، روش چهارم که ترکیب *Fixed Point* با تابع *G(y)* است می‌تواند بهترین گزینه باشد.

برای مقایسه‌ی *E* به دست آمده داریم:

Method	1	2	3	4
E	0.38	0.3084	0.34	0.33

البته قابل ذکر است که این مقادیر بسیار به هم نزدیک بوده و در این بازه قدرت مقایسه را از ما می‌گیرند.

پایان