تعریف مساله:

سیستم وینر زیر را با نویز رنگی در نظر بگیرید:

$$y(t)=\sum_{i=1}^{p} a_i \sum_{l=1}^{q} d_l g_l[y(t-i)]+\sum_{j=1}^{n} b_j u(t-j)+\sum_{k=1}^{m} f_k v(t-k)+v(t), \tag{1}$$

که در آن u و y ورودی و خروجی سیستم، v یک نویز سفید با متوسط صفر، و $g_l(^*)$ توابع معلوم غیر خطی هستند. با تعریف:

$$\boldsymbol{a} \coloneqq [a_1,a_2,\ldots,a_p]^{\mathrm{T}} \in \mathbb{R}^p,$$

$$\boldsymbol{d} \coloneqq [d_1,d_2,\ldots,d_q]^{\mathrm{T}} \in \mathbb{R}^q,$$

$$\boldsymbol{b} \coloneqq [b_1,b_2,\ldots,b_n]^{\mathrm{T}} \in \mathbb{R}^n,$$

$$\boldsymbol{f} \coloneqq [f_1,f_2,\ldots,f_m]^{\mathrm{T}} \in \mathbb{R}^m,$$

$$\boldsymbol{\varphi}(t) \coloneqq [u(t-1),u(t-2),\ldots,u(t-n)]^{\mathrm{T}} \in \mathbb{R}^n, \tag{2}$$

$$\boldsymbol{\psi}(t) \coloneqq [v(t-1),v(t-2),\ldots,v(t-m)]^{\mathrm{T}} \in \mathbb{R}^m, \tag{3}$$

$$\boldsymbol{G}(t) \coloneqq \begin{bmatrix} g_1(y(t-1)) & g_2(y(t-1)) & \cdots & g_q(y(t-1)) \\ g_1(y(t-2)) & g_2(y(t-2)) & \cdots & g_q(y(t-2)) \\ \vdots & \vdots & & \vdots \\ g_1(y(t-p)) & g_2(y(t-p)) & \cdots & g_q(y(t-p)) \end{bmatrix} \in \mathbb{R}^{p\times q}. \tag{4}$$

معادله (1) به صورت زیر قابل بازنویسی است:

$$y(t)=\boldsymbol{a}^{\mathrm{T}}\boldsymbol{G}(t)\boldsymbol{d}+\boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{b}+\boldsymbol{\psi}^{\mathrm{T}}(t)\boldsymbol{f}+v(t). \tag{5}$$

فرض کنید:

$$\hat{\theta}_k \coloneqq \begin{bmatrix} \hat{\boldsymbol{a}}_k \\ \hat{\boldsymbol{b}}_k \\ \hat{\boldsymbol{f}}_k \end{bmatrix}$$

$$\hat{\vartheta}_k \coloneqq \hat{\boldsymbol{d}}_k$$

تقریب‌هایی از $\boldsymbol{\theta} \coloneqq \begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \\ \boldsymbol{f} \end{bmatrix}$ و $\boldsymbol{\vartheta} \coloneqq \boldsymbol{d}$ در تکرار k باشند. اگر L طول داده‌ها باشد، با فرض

$$L \gg p+q+n+m$$

تابع هزینه به صورت زیر تعریف می‌شود:

$$J(\boldsymbol{a},\boldsymbol{b},\boldsymbol{f},\boldsymbol{d}) = \sum_{t=1}^{L}[y(t)-\boldsymbol{a}^{\mathrm{T}}\boldsymbol{G}(t)\boldsymbol{d}-\boldsymbol{\varphi}^{\mathrm{T}}(t)\boldsymbol{b}-\boldsymbol{\psi}^{\mathrm{T}}(t)\boldsymbol{f}]^2. \qquad (6)$$

تابع هزینه بالا را به روش کمترین مربعات تکراری بهینه سازی می‌کنیم.

روش کمترین مربعات تکراری

تعاریف زیر را در نظر بگیرید:

$$\boldsymbol{Y}(L) \coloneqq \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(L) \end{bmatrix} \in \mathbb{R}^{L}, \quad \boldsymbol{\Phi}(L) \coloneqq \begin{bmatrix} \boldsymbol{\varphi}^{\mathrm{T}}(1) \\ \boldsymbol{\varphi}^{\mathrm{T}}(2) \\ \vdots \\ \boldsymbol{\varphi}^{\mathrm{T}}(L) \end{bmatrix} \in \mathbb{R}^{L\times n},$$

$$\boldsymbol{\Psi}(L) \coloneqq \begin{bmatrix} \boldsymbol{\psi}^{\mathrm{T}}(1) \\ \boldsymbol{\psi}^{\mathrm{T}}(2) \\ \vdots \\ \boldsymbol{\psi}^{\mathrm{T}}(L) \end{bmatrix} \in \mathbb{R}^{L\times m}, \qquad (9)$$

$$\hat{\boldsymbol{\Psi}}_k(L) \coloneqq \begin{bmatrix} \hat{\boldsymbol{\psi}}_k^{\mathrm{T}}(1) \\ \hat{\boldsymbol{\psi}}_k^{\mathrm{T}}(2) \\ \vdots \\ \hat{\boldsymbol{\psi}}_k^{\mathrm{T}}(L) \end{bmatrix} \in \mathbb{R}^{L\times m}. \qquad (18)$$

$$\hat{\boldsymbol{\psi}}_k(t) \coloneqq [\hat{v}_{k-1}(t-1),\hat{v}_{k-1}(t-2),\ldots,\hat{v}_{k-1}(t-m)]^{\mathrm{T}}. \qquad (16)$$

$$\Upsilon(\hat{\boldsymbol{d}}_{k-1},L) \coloneqq \begin{bmatrix} \hat{\boldsymbol{d}}_{k-1}^{\mathrm{T}}G^{\mathrm{T}}(1) \\ \hat{\boldsymbol{d}}_{k-1}^{\mathrm{T}}G^{\mathrm{T}}(2) \\ \vdots \\ \hat{\boldsymbol{d}}_{k-1}^{\mathrm{T}}G^{\mathrm{T}}(L) \end{bmatrix} \in \mathbb{R}^{L\times p},$$

$$\boldsymbol{\Omega}(\hat{\boldsymbol{a}}_k,L) \coloneqq \begin{bmatrix} \hat{\boldsymbol{a}}_k^{\mathrm{T}}G(1) \\ \hat{\boldsymbol{a}}_k^{\mathrm{T}}G(2) \\ \vdots \\ \hat{\boldsymbol{a}}_k^{\mathrm{T}}G(L) \end{bmatrix} \in \mathbb{R}^{L\times q}. \qquad (10)$$

$$\hat{\boldsymbol{\Xi}}(\hat{\boldsymbol{d}}_{k-1},L) = [\Upsilon(\hat{\boldsymbol{d}}_{k-1},L),\boldsymbol{\Phi}(L),\hat{\boldsymbol{\Psi}}_k(L)], \qquad (21)$$

در معادلات بالا متغیرهای دارای علامت ^ نشان دهنده تقریب متغیر مربوطه توسط الگوریتم است. با دقت در رابطه (16) می‌بینیم که مقادیر نویز v در زمان‌های مختلف در اختیار نیست. بنابراین باید تخمین آن را در الگوریتم به کار ببریم. این کار در هر تکرار با رابطه زیر انجام می‌شود:

$$\hat{v}_k(t) = y(t) - \hat{\boldsymbol{a}}_{k-1}^{\mathrm{T}} G(t)\hat{\boldsymbol{d}}_{k-1} - \boldsymbol{\varphi}^{\mathrm{T}}(t)\hat{\boldsymbol{b}}_{k-1} + \hat{\boldsymbol{\psi}}_k^{\mathrm{T}}(t)\hat{\boldsymbol{f}}_{k-1}. \qquad (17)$$

با استفاده از تعاریف بالا رابطهٔ تخمین پارامترها در تکرار k به صورت زیر می‌باشد:

$$\hat{\boldsymbol{\theta}}_k = [\hat{\boldsymbol{\Xi}}^{\mathrm{T}}(\hat{\boldsymbol{d}}_{k-1},L)\hat{\boldsymbol{\Xi}}(\hat{\boldsymbol{d}}_{k-1},L)]^{-1}\hat{\boldsymbol{\Xi}}^{\mathrm{T}}(\hat{\boldsymbol{d}}_{k-1},L)\boldsymbol{Y}(L), \quad k = 1,2,3,\ldots, \qquad (19)$$

$$\hat{\boldsymbol{\vartheta}}_k = [\boldsymbol{\Omega}^{\mathrm{T}}(\hat{\boldsymbol{a}}_k,L)\boldsymbol{\Omega}(\hat{\boldsymbol{a}}_k,L)]^{-1}\boldsymbol{\Omega}^{\mathrm{T}}(\hat{\boldsymbol{a}}_k,L)[\boldsymbol{Y}(L) - \boldsymbol{\Phi}(L)\hat{\boldsymbol{b}}_k - \hat{\boldsymbol{\Psi}}_k(L)\hat{\boldsymbol{f}}_k], \qquad (20)$$

برای این الگوریتم موارد زیر را در نظر می‌گیریم:

- مقدار اولیه پارامترها یعنی $\hat{\boldsymbol{\theta}}_0$ را غیر صفر و تصادفی و $\hat{\boldsymbol{d}}_0$ به نحوی تعیین می‌شود که $\|\hat{\boldsymbol{d}}_0\| = 1$ و همچنین $\hat{v}_0(t-i)$ به صورت تصادفی تعیین می‌شود.
- بعد از به روز کردن پارامترها قرار می‌دهیم:

$$\hat{\boldsymbol{a}}_k := \hat{\boldsymbol{\theta}}_k(1:p),$$

$$\hat{\boldsymbol{b}}_k := \hat{\boldsymbol{\theta}}_k(p+1:p+n),$$

$$\hat{\boldsymbol{f}}_k := \hat{\boldsymbol{\theta}}_k(p+n+1:p+n+m).$$

- در هر مرحله بعد از به روز رسانی $\hat{\boldsymbol{d}}_k$ آنرا با رابطهٔ زیر نرمالیزه می‌کنیم:

$$\hat{\boldsymbol{d}}_k = \mathrm{sgn}[\hat{\boldsymbol{\vartheta}}_k(1)]\frac{\hat{\boldsymbol{\vartheta}}_k}{\|\hat{\boldsymbol{\vartheta}}_k\|}.$$

- شرط توقف الگوریتم را رسیدن نرم بردار اختلاف $\{\hat{\boldsymbol{a}}_k,\hat{\boldsymbol{b}}_k\hat{\boldsymbol{f}}_k,\hat{\boldsymbol{d}}_k\}$ با $\{\hat{\boldsymbol{a}}_{k-1},\hat{\boldsymbol{b}}_{k-1}\hat{\boldsymbol{f}}_{k-1},\hat{\boldsymbol{d}}_{k-1}\}$ به مقداری کوچک (مثل $\varepsilon > 0$) قرار می‌دهیم.

روش گرادیان نزولی تکراری

این روش مشابه روش قبل است، فقط در آن تغییرات زیر را انجام می‌دهیم:

- بعد از محاسبه $\hat{\boldsymbol{\Xi}}(\hat{\boldsymbol{d}}_{k-1},L)$ در الگوریتم، پارامترهای زیر تعیین می‌شوند:

$$0 < \mu_1(k) \le \frac{2}{\lambda_{\max}[\hat{\boldsymbol{\Xi}}^{\mathrm{T}}(\hat{\boldsymbol{d}}_{k-1},L)\hat{\boldsymbol{\Xi}}(\hat{\boldsymbol{d}}_{k-1},L)]}, \qquad (39)$$

$$0 < \mu_2(k) \le \frac{2}{\lambda_{\max}[\boldsymbol{\Omega}^{\mathrm{T}}(\hat{\boldsymbol{a}}_k,L)\boldsymbol{\Omega}(\hat{\boldsymbol{a}}_k,L)]}. \qquad (40)$$

- بعد از محاسبه نرخ‌های آموزش بالا، پارامترهای مدل توسط روابط زیر به روز می‌گردد:

$$\hat{\boldsymbol{\theta}}_k = \hat{\boldsymbol{\theta}}_{k-1} + \mu_1(k)\hat{\boldsymbol{\Xi}}^{\mathrm{T}}(\hat{\boldsymbol{d}}_{k-1},L)[\boldsymbol{Y}(L) - \hat{\boldsymbol{\Xi}}(\hat{\boldsymbol{d}}_{k-1},L)\hat{\boldsymbol{\theta}}_{k-1}], \qquad (41)$$

$$\hat{\boldsymbol{\vartheta}}_k = \hat{\boldsymbol{\vartheta}}_{k-1} + \mu_2(k)\boldsymbol{\Omega}^{\mathrm{T}}(\hat{\boldsymbol{a}}_k,L)[\boldsymbol{Y}(L) - \boldsymbol{\Omega}(\hat{\boldsymbol{a}}_k,L)\hat{\boldsymbol{d}}_{k-1} - \boldsymbol{\Phi}(L)\hat{\boldsymbol{b}}_k - \hat{\boldsymbol{\Psi}}_k(L)\hat{\boldsymbol{f}}_k], \qquad (42)$$

مثال شبیه سازی‌ها

در شبیه سازی‌های این دو الگوریتم سیستم وینر زیر را در نظر می‌گیریم:

$$y(t) = \sum_{i=1}^{2} a_i[d_1 g(y(t-i)) + d_2 g_2(y(t-i)) + d_3 g_3(y(t-i))] + \sum_{j=1}^{2} b_j u(t-j) + fv(t-1) + v(t),$$

$$g_1(y(t-i)) = y(t-i),$$

$$g_2(y(t-i)) = y^2(t-i),$$

$$g_3(y(t-i)) = y^3(t-i),$$

$$\boldsymbol{\theta} = [a_1, a_2, b_1, b_2, f]^{\mathrm{T}} = [0.25, 0.28, -0.30, 1.00, 0.05]^{\mathrm{T}},$$

$$\boldsymbol{\vartheta} = [d_1, d_2, d_3]^{\mathrm{T}} = [0.80, -0.50, -0.3317]^{\mathrm{T}},$$

$$\Theta = [\boldsymbol{\theta}^{\mathrm{T}}, \boldsymbol{\vartheta}^{\mathrm{T}}]^{\mathrm{T}} = [0.25, 0.28, -0.30, 1.00, 0.05, 0.80, -0.50, -0.3317]^{\mathrm{T}}.$$

همچنین ورودی u(t) به صورت نویز سفید با متوسط صفر و واریانس ۱ که برای پایداری سیستم فیلتر شده است می‌باشد و خروجی آغشته به نویز سفید با واریانس $\sigma^2 = 0.20^2$ و متوسط صفر شده است.

پیاده سازی در Matlab

ابتدا برنامه data.m برای تولید زوج ورودی-خروجی‌های سیستم اصلی را نوشتیم. توضیحات این برنامه به صورت زیر است:

```
N=4998;
```

تعداد نمونه‌های مورد نیاز برای زوج‌های ورودی-خروجی تعیین می‌شود

```
a=[0.25; 0.28];
b=[-0.3; 1];
f=0.05;
d=[0.8; -.5;-.3317];
```

پارامترهای سیستم مورد نظر تعیین می‌شوند.

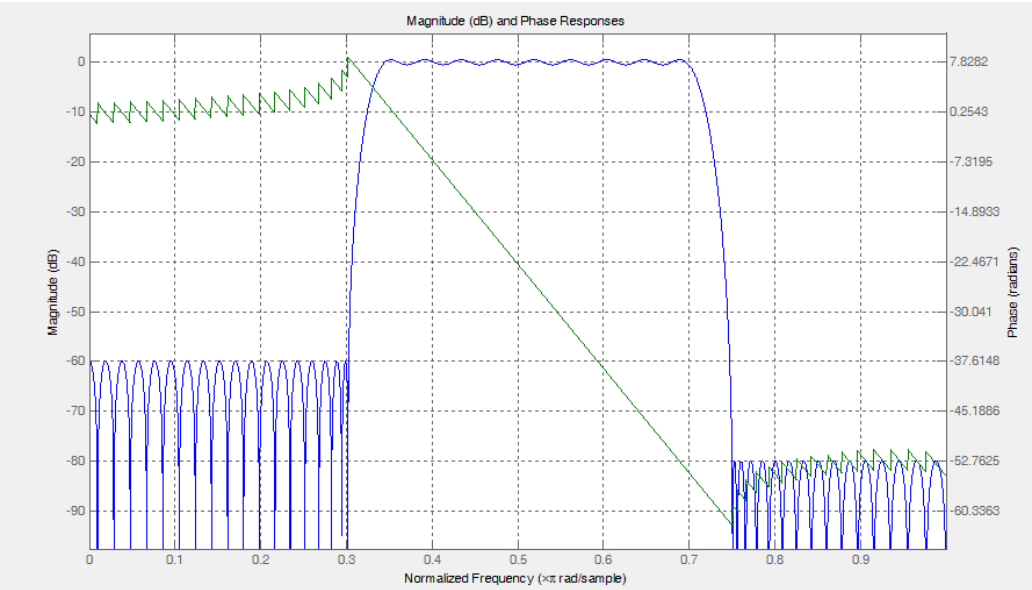
```
u1=random('normal',0,1,1,N);
```

ورودی به صورت نویز سفید با واریانس ۱ و متوسط ۰ تولید می‌شود.

```
Hd=filt;
```

یک فیلتر دیجیتال میان گذر با پاسخ فرکانسی به صورت زیر برای فیلتر کردن نویز تولید می‌شود. تابع filt با استفاده ار جعبه ابزار طراحی فیلتر، فیلتر مورد نظر را می‌سازد.

```
u1=filter(Hd,u1);
```




```
u1=filter(Hd,u1);
```

ورودی نویز را با فیلتر بالا فیلتر می‌کنیم تا پاسخ سیستم پایدار باشد.

```
v1=random('normal',0,.2,1,N);
```

نویز مربوط به خروجی ساخته می‌شود

```
u1=[0 0 u1];
v1=[0 0 v1];
```

شرایط اولیه صفر برای ورودی و نویز در نظر گرفته می‌شود

```
for i=1:N

y1(i+2)=a(1)*(d(1)*y1(i+1)+d(2)*y1(i+1)^2+d(3)*y1(i+1)^3)+...

a(2)*(d(1)*y1(i)+d(2)*y1(i)^2+d(3)*y1(i)^3)+b(1)*u1(i+1)+b(2)*u1
(i)+...
        f*v1(i+1)+v1(i+2);
end
```

در یک حلقه for و به صورت بازگشتی با استفاده از رابطهٔ (1) خروجی سیستم محاسبه می‌شود.

```
save('data.mat','u1','y1')
```

سیگنال‌های ورودی و خروجی در فایل data.mat ذخیره می‌شود.

در ادامه برای پیاده سازی روش کمترین مربعات برنامه main1.m نوشته شد. توضیحات این برنامه به صورت زیر می‌باشد:

```
load data
```

داده‌های تولید شده در برنامه data.m برای استفاده در الگوریتم لود می‌شوند.

```
L=1000;
```

طول داده مورد استفاده در الگوریتم تعیین می‌شود:

```
theta=rand(5,1);
```

مقدار اولیه پارامترها به صورت تصادفی انتخاب می‌شوند

```
theta_h=theta;
```

متغیر theta_h برای ذخیره پارامترها در تکرار قبلی مورد استفاده قرار می‌گیرد

```
d=rand(3,1);d=d/norm(d);
```

مقدار اولیه پارامتر d به صورت تصادفی انتخاب می‌شوند و با تقسیم آن بر نرم d، شرط $\|\hat{\boldsymbol{d}}_0\| = 1$ را برآورد می‌کنیم.

```
v=zeros(L+2,1);
```

برداری با مقادیر صفر برای تخمین مقدار نویز تعریف می‌شود.

```
th=[.25 .28 -.3 1 .05 .8 -.5 -.3317];
```

مقدار اصلی پارامترها برای مقایسه با مقادیر تخمینی، تعریف می‌شوند.

```
it=0;
```

متغیری برای شمارش تعداد تکرارها تعیین می‌شود.

```
while 1
    it=it+1;
```

آغاز حلقه تکرار الگوریتم

```
    kk=200;
    y=y1(kk:kk+L+1)';
    u=u1(kk:kk+L+1)';
```

L+2 تا از نمونه‌های سیگنال‌های ورودی و خروجی جدا می‌شوند تا مورد استفاده قرار بگیرند. ۲ نمونه اضافی برای شرایط اولیه اضافه شده‌اند.

```
    PHI=[];
    GAMA=[];
    OMEG=[];
    PSI1=[];
```

متغیرهای مورد نیاز در الگوریتم در هر تکرار تعریف می‌شوند تا بعدا در الگوریتم مورد استفاده قرار گیرند.

```
    for i=1:L
```

برای هر نمونه از L نمونه عملیاتی را انجام می‌دهیم.

```
        phi=[u(i+1);u(i)];
```

بردار $\boldsymbol{\varphi}(t)$ با رابطه (2) ساخته می‌شود

```
        psi1=v(i+1);
```

بردار $\hat{\boldsymbol{\psi}}_k(t)$ با رابطه (3) ساخته می‌شود

```
        G=[y(i+1) y(i+1)^2 y(i+1)^3;y(i) y(i)^2 y(i)^3];
```

ماتریس G با رابطه (4) ساخته می‌شود.

```
        PHI=[PHI;phi'];
```

ماتریس $\boldsymbol{\Phi}(L)$ با رابطه (9) ساخته می‌شود

```
PSI1=[PSI1;psi1];
```

ماتریس $\hat{\boldsymbol{\Psi}}_k(L)$ با رابطه (9) تشکیل می‌شود.

```
GAMA=[GAMA;d'*G'];
```

ماتریس $\boldsymbol{\Upsilon}(\hat{\boldsymbol{d}}_{k-1},L)$ با رابطهٔ (10) تشکیل می‌شود.

```
    end
```

پایان تکرار for i=1:L.

```
    Y=y(3:end);
```

تشکیل بردار Y با رابطه (9).

```
    E=[GAMA PHI PSI1];
```

تشکیل ماتریس $\hat{\boldsymbol{\Xi}}(\hat{\boldsymbol{d}}_{k-1},L)$ با رابطه (21).

```
    theta=(E'*E)\E'*Y;
```

محاسبه $\hat{\boldsymbol{\theta}}_k$ با رابطه (41).

```
    ak=theta(1:2);
    bk=theta(3:4);
    fk=theta(5);
```

جداسازی a و b و f از بردار $\hat{\boldsymbol{\theta}}_k$.

```
        for i=1:L
            G=[y(i+1) y(i+1)^2 y(i+1)^3;y(i) y(i)^2 y(i)^3];
            OMEG=[OMEG;ak'*G];
        end
```

ساخت ماتریس $\boldsymbol{\Omega}(\hat{\boldsymbol{a}}_k,L)$ از رابطه (10).

```
        d=(OMEG'*OMEG)\OMEG'*(Y-PHI*bk-PSI1*fk);
```

محاسبه $\hat{\boldsymbol{d}}_k$ با رابطه (42).

```
        d=sign(d(1))*d/norm(d);
```

نرمالسازی $\hat{\boldsymbol{d}}_k$.

```
        for i=1:L
            phi=[u(i+1);u(i)];
            psi1=v(i+1);
            G=[y(i+1) y(i+1)^2 y(i+1)^3;y(i) y(i)^2 y(i)^3];
            v(i+2)=y(i+2)-ak'*G*d-phi'*bk+psi1'*fk;
        end
```

محاسبه $\hat{v}_k(t)$ با رابطه (17).

```
    res=[res;it temp norm(temp-th)/norm(th)*100];
```

مقدار پارامترها در هر تکرار به همراه درصد خطای نسبی آن در آن تکرار در متغیر res ذخیره می‌گردد.

```
    if norm(theta_h-theta)<0.0001
        break
    end
```

اگر نرم بردار اختلاف بین پارامترهای هر تکرار با تکرار قبل از آن از مقداری مثبت کوچکتر باشد، تکرارها را متوقف می‌کند، اگر نه به الگوریتم اجازه تکرارهای بیشتری را می‌دهد.

```
    theta_h=theta;
```

پارامترهای هر مرحله برای مقایسه در تکرار بعد در متغیری ذخیره می‌شوند.

```
res
save('W_LSI.mat','res')
```

نتیجه الگوریتم نمایش داده شده و در متغیر W_LSI.mat نیز ذخیره می‌شود.

برای پیاده سازی الگوریتم گرادیان نزولی برنامه main2.m نوشته شده. در این برنامه تمامی متغیرها مشابه برنامه main1.m ساخته می‌شوند و تفاوت آن با برنامه قبل به شرح زیر است:

```
    LR1=1.5/max(eig(E'*E));
```

مقدار پارامتر $\mu_1(k)$ با استفاده از رابطه (39) محاسبه می‌شود.

```
    theta=theta+LR1*E'*(Y-E*theta);
```

پارامتر $\hat{\boldsymbol{\theta}}_k$ با استفاده از رابطه (41) به روز می‌شود

```
LR2=1.5/max(eig(OMEG'*OMEG));
```

مقدار پارامتر $\mu_2(k)$ با استفاده از رابطه (40) محاسبه می‌شود

```
        d=d+LR2*OMEG'*(Y-OMEG*d-PHI*bk-PSI1*fk);
```

پارامتر $\hat{\boldsymbol{d}}_k$ با استفاده از رابطه (42) به روز می‌شود.

```
save('W_GI.mat','res')
```

نتایج اجرای برنامه در فایل W_GI.mat ذخیره می‌شود.

نتایج

با اجرای برنامه‌ها برای L=1000 و L=2000، چهار جدول زیر را در متغیر res خواهیم داشت:

تخمین پارامترها با کمترین مربعات و خطای مربوطه (L=1000)

k	$a1$	$a2$	$b1$	$b2$	f	$d1$	$d2$	$d3$	$\delta(\%)$
1	-0.00659	0.0005	-0.35612	0.842653	-0.04111	0.474574	-0.78084	-0.40629	40.72967
2	0.154543	0.176225	-0.32833	0.881719	0.152848	0.662354	-0.61221	-0.43184	19.71598
3	0.197432	0.247887	-0.31733	0.921543	0.130958	0.712575	-0.57606	-0.40049	12.51208
4	0.207314	0.264988	-0.31346	0.946817	0.10301	0.749594	-0.54252	-0.37918	8.0493
5	0.215027	0.273288	-0.30997	0.964384	0.084515	0.775435	-0.51488	-0.36551	5.093975
6	0.222152	0.279542	-0.30683	0.976933	0.071305	0.79234	-0.49532	-0.35617	3.337294
7	0.227552	0.283808	-0.30435	0.985651	0.062306	0.803188	-0.48218	-0.34985	2.635786
8	0.231306	0.286534	-0.30256	0.991498	0.0564	0.810087	-0.47357	-0.34568	2.598231
9	0.233813	0.288249	-0.30132	0.995322	0.052602	0.814449	-0.46802	-0.34298	2.774719
10	0.235446	0.289324	-0.30051	0.997782	0.050188	0.817193	-0.46448	-0.34126	2.956975

تخمین پارامترها با کمترین مربعات و خطای مربوطه (L=2000)

k	$a1$	$a2$	$b1$	$b2$	f	$d1$	$d2$	$d3$	$\delta(\%)$
1	-0.12378	-0.15659	-0.37986	0.718368	-0.01946	0.224634	-0.85179	-0.47328	63.32758
2	0.16214	0.128481	-0.3344	0.821934	0.171885	0.527587	-0.69167	-0.4932	31.05914
3	0.165024	0.205151	-0.33136	0.870575	0.22273	0.589767	-0.67422	-0.44453	25.7137
4	0.179799	0.221803	-0.33065	0.896862	0.162889	0.654561	-0.6349	-0.41042	18.71005
5	0.19552	0.236471	-0.32443	0.921744	0.130853	0.709119	-0.58773	-0.38952	12.94386
6	0.209639	0.249532	-0.31943	0.942771	0.104242	0.746711	-0.55013	-0.37387	8.538585
7	0.221747	0.259759	-0.31491	0.959246	0.08364	0.772148	-0.52218	-0.3621	5.35917
8	0.231521	0.267292	-0.31098	0.971804	0.068458	0.78937	-0.50193	-0.35349	3.24434
9	0.239039	0.272656	-0.30778	0.981075	0.057619	0.801027	-0.48756	-0.34733	2.169286
10	0.244608	0.276397	-0.30531	0.987744	0.050042	0.80891	-0.47751	-0.343	2.044393

تخمین پارامترها با گرادیان نزولی و خطای مربوطه (L=1000)

k	$a1$	$a2$	$b1$	$b2$	f	$d1$	$d2$	$d3$	$\delta(\%)$
1	0.060423	-0.46704	-0.209	0.879689	0.748747	0.017448	-0.88734	0.460787	105.7037
5	0.161963	0.217067	-0.30752	0.923394	0.162982	0.67578	-0.62516	-0.39051	17.05378
10	0.196779	0.254839	-0.31579	0.94987	0.138575	0.719452	-0.58344	-0.37681	11.50424

20	0.212716	0.273908	-0.30988	0.974279	0.099798	0.773771	-0.5218	-0.35917	5.429514
40	0.230141	0.287089	-0.30278	0.99437	0.062726	0.809767	-0.47521	-0.34418	2.60174
50	0.23383	0.289045	-0.30114	0.997945	0.055127	0.815508	-0.46729	-0.34144	2.823231
80	0.237545	0.290725	-0.29943	1.001313	0.047502	0.820773	-0.45992	-0.33883	3.255796

تخمین پارامترها با گرادیان نزولی و خطای مربوطه (L=2000)

k	$a1$	$a2$	$b1$	$b2$	f	$d1$	$d2$	$d3$	$\delta(\%)$
1	0.229276	-0.25464	0.259441	1.122352	0.895597	0.713679	0.693677	0.097339	114.8429
5	0.299212	0.020519	-0.19999	0.938966	-0.05262	0.989968	-0.01319	-0.14067	42.50849
10	0.34821	0.284299	-0.24011	1.058301	-0.00516	0.917718	-0.28817	-0.2734	19.13994
20	0.301631	0.30444	-0.27697	1.043023	-0.01531	0.876599	-0.37334	-0.30362	12.08278
40	0.271638	0.29123	-0.29252	1.015934	0.010476	0.843966	-0.42786	-0.32351	6.580074
50	0.26602	0.288597	-0.29525	1.010733	0.019144	0.836839	-0.43873	-0.32744	5.415227
80	0.259572	0.285414	-0.29829	1.004635	0.030262	0.828073	-0.45168	-0.33208	4.046975

برای مقایسه دو الگوریتم، در برنامه plots.m روند کاهش خطا در هر دو الگوریتم توسط نتایج ذخیره شده درفایل‌ها رسم می‌شود که با اجرای برنامه شکل زیر قابل مشاهده است: