دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)
دانشکده مهندسی برق

سیستم گسسته در زمان

نگارش
ستایش خاصه تراش

استاد
دکتر حیدرعلی طالبی

تیر ۱۴۰۴

چکیده

هدف از این تکلیف، پیاده‌سازی چند مثال در راستای فهم بهتر مفاهیم درس کنترل تطبیقی است. در این تکلیف به تخمین پارامتر های یک سیستم فرضی پرداخته شده و همچنین کارآیی الگوریتم های متفاوت مورد بررسی قرار گرفته است. الگوریتم های استفاده شده شامل Projection، Orthogonal Projection و متود های متفاوت از RLS می باشد. همچنین، همگرایی آن ها و شرط محدودیت پارامتر ها لحاظ شده و اثر تغییر متغیر های موجود در الگوریتم مورد مطالعه قرار گرفته است.

فهرست مطالب

صفحه

فصل اول

مقدمه

مقدمه

تخمین پارامتر یکی از مسائل مهم در زمینه کنترل به شمار می‌رود. هدف اصلی این پروژه، پیاده‌سازی مفاهیم آموخته شده در درس کنترل تطبیقی و به‌کارگیری آن‌ها در یک مسئله است. در بخش اول، به تخمین پارامتر ها با استفاده از الگوریتم های متفاوت تمرکز می‌شود. در بخش بعدی، عملکرد متد های متفاوت الگوریتم RLS مورد ارزیابی قرار می‌گیرد. در بخش آخر نیز تخمین برای پارامتر های متغیر با زمان با الگوریتم های گفته شده انجام می شود.

.

فصل دوم

بررسی الگوریتم های تخمین پارامتر برای سیستم های گسسته در زمان

سوال i: بررسی الگوریتم های تخمین پارامتر برای سیستم های گسسته در زمان

2-1- الگوریتم Projection

در این پروژه، هدف استفاده از الگوریتم Projection برای تخمین پارامترهای یک سیستم خطی گسسته با ورودی تصادفی و ساختار دینامیکی مشخص است. مدل اصلی سیستم به صورت فضای حالت (State-Space) تعریف شده و شامل ماتریس‌های A، B و C است. سیستم سه متغیر حالت دارد و فقط خروجی اول آن استفاده می‌شود.

برای تخمین پارامترها، ابتدا سیستم را به صورت رگرسیون و استاندارد بازنویسی کرده و پارامتر های مجهول و اطلاعات معلوم را به ترتیب با theta و phi بازنویسی میکنیم.

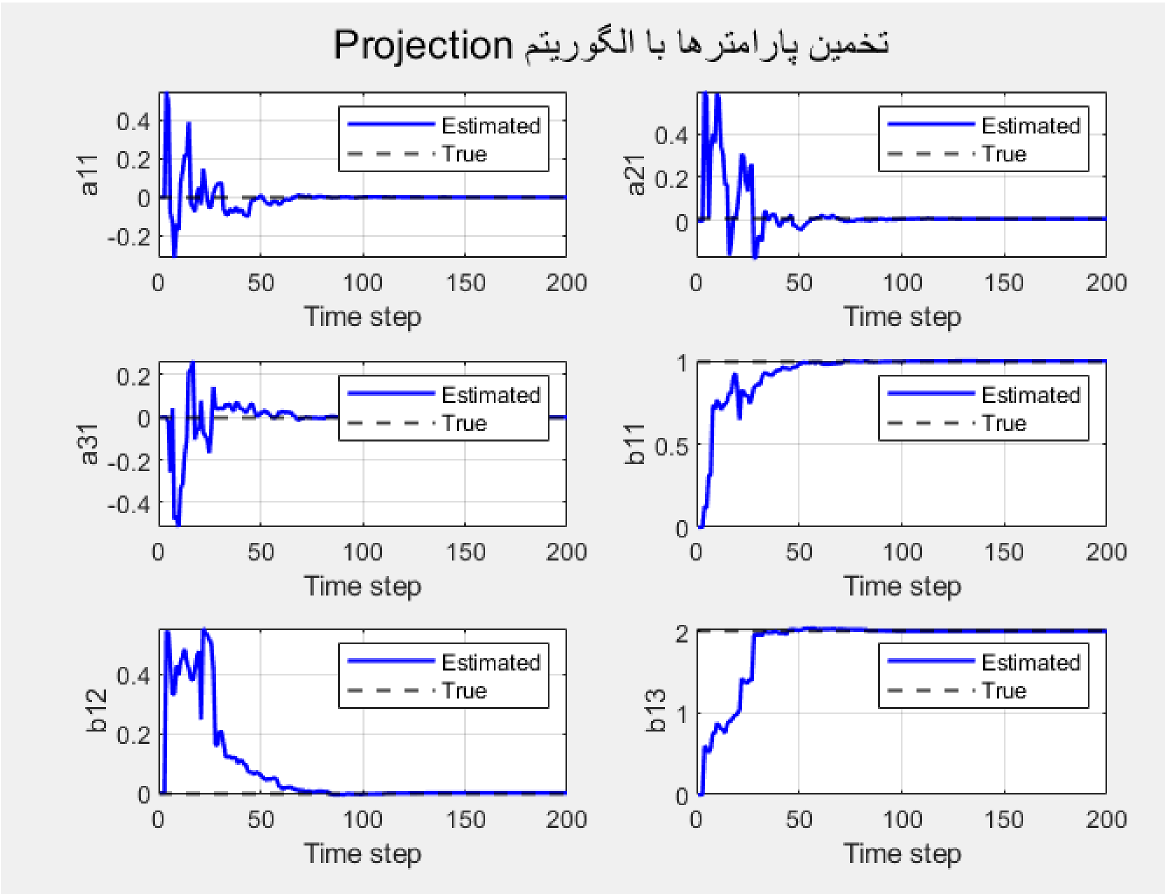
```
theta_hat = zeros(6,1);            % [a11; a21; a31; b11; b12; b13]

phi = [y(t-1); y(t-2); y(t-3); u(t-1); u(t-2); u(t-3)];
```

الگوریتم Projection با هدف جلوگیری از یادگیری نامناسب و انفجار پارامترها عمل می‌کند. در این روش، ابتدا اندازه‌ی بردار رگرسیون یا همان phi اندازه‌گیری می‌شود و اگر بزرگ‌تر از یک کران آستانه (threshold) باشد، فرآیند آپدیت تخمین انجام نمی‌گیرد. این شرط باعث می‌شود که فقط داده‌های "معتبری" که دارای انرژی مناسب هستند، در فرآیند یادگیری مشارکت کنند.

در این کد، مقدار آستانه برابر با 10 در نظر گرفته شده است. همچنین به‌منظور پایداری عددی، در مخرج به‌روزرسانی از یک مقدار کوچک استفاده شده تا از تقسیم بر صفر جلوگیری شود.

```
% ===== الگوریتم Projection =====
theta_hat = zeros(6,1);            % [a11; a21; a31; b11; b12; b13]
theta_history = zeros(6, T);
threshold = 10;                    % کران ϕ برای اعمال Projection

for t = 4:T
    phi = [y(t-1); y(t-2); y(t-3); u(t-1); u(t-2); u(t-3)];

    if norm(phi) <= threshold      % شرط Projection
        y_hat = phi' * theta_hat;
        e = y(t) - y_hat;

        denom = phi' * phi + 1e-6;
        theta_hat = theta_hat + (phi * e) / denom;
    end

    theta_history(:,t) = theta_hat;
end
```

در پایان، تخمین نهایی پارامترها به همراه مقدار واقعی آن‌ها مقایسه شده و برای هر پارامتر، نمودار مقایسه‌ای شامل مقدار تخمینی و مقدار واقعی رسم شده است. نتایج نشان می‌دهند که الگوریتم Projection در صورت تنظیم مناسب آستانه، می‌تواند به خوبی به سمت مقادیر واقعی همگرا شود و رفتار پایداری از خود نشان دهد.



پارامتر های نهایی نیز به صورت زیر نمایش داده می شوند که با پارامتر های اصلی سیستم یکسان می باشند.

```
:Projection تخمین نهایی پارامترها با
theta_hat = [a11; a21; a31; b11; b12; b13]
    0.0000
    0.0100
   -0.0000
    1.0000
   -0.0000
    2.0000
```

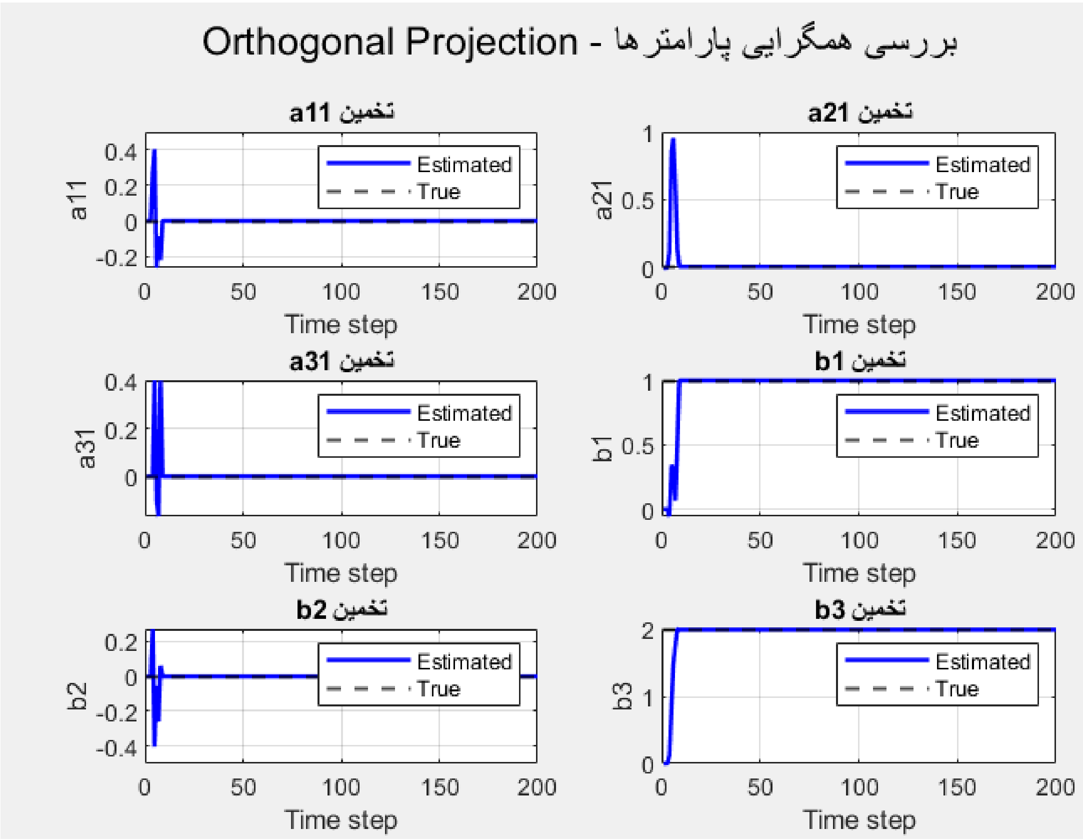
2-2- الگوریتم Orthogonal Projection

در این بخش، الگوریتم **Orthogonal Projection**برای تخمین ضرایب پیاده‌سازی شد. ورودی سیستم یک نویز گوسی با توزیع N(0,1) بود و خروجی تنها شامل متغیر x1 بود که با استفاده از آن تخمین پارامترها صورت گرفت.

الگوریتم تخمین مبتنی بر به‌روزرسانی بازگشتی بردار پارامتر θ با استفاده از نگاشت متعامد است، به‌طوری‌که در هر گام زمانی، اگر انرژی تحریک به اندازه کافی باشد، تخمین به‌روزرسانی شده و ماتریس کوواریانس نیز تصحیح می‌شود.

```
% ===== الگوریتم Orthogonal Projection =====
theta_hat = zeros(6, 1);          % [a11; a21; a31; b1; b2; b3]
theta_history = zeros(6, T);
P = eye(6);                       % مقدار اولیه P

Phi_all = zeros(6, T-3);          % برای بررسی تحریک پایدار

for t = 4:T
    phi = [y(t-1); y(t-2); y(t-3); u(t-1); u(t-2); u(t-3)];
    phit = phi';
    denom = phit * P * phi;

    % ذخیره بردار phi برای بررسی رتبه
    Phi_all(:, t-3) = phi;

    if abs(denom) > 1e-6
        e = y(t) - phit * theta_hat;
        theta_hat = theta_hat + (P * phi) * (e / denom);
        P = P - (P * phi * phit * P) / denom;
    end

    theta_history(:,t) = theta_hat;
end
```

در قدم اول، **همگرایی پارامترها** بررسی شد. برای این منظور، تاریخچه تخمین هر پارامتر در طول زمان رسم شد و مشاهده شد که ضرایب تخمینی به مقدار واقعی خود نزدیک شده و پایدار می‌مانند. این موضوع نشانه موفقیت الگوریتم در یادگیری پارامترهای سیستم است.



در مرحله دوم، **شرط تحریک پایدار (Persistency of Excitation)** بررسی شد. این شرط تضمین می‌کند که بردارهای ورودی $\phi(t)$ در طول زمان اطلاعات کافی برای شناسایی تمام ضرایب را دارند. برای بررسی این شرط، ماتریس $\Phi=[\phi(4),\ldots,\phi(T)]$ تشکیل و رتبه آن محاسبه شد. نتیجه نشان داد که رتبه ماتریس ۶ بوده که به معنای ارضای کامل شرط تحریک پایدار است.

```
رتبه ماتریس Phi_all (برای بررسی تحریک پایدار) = 6
```

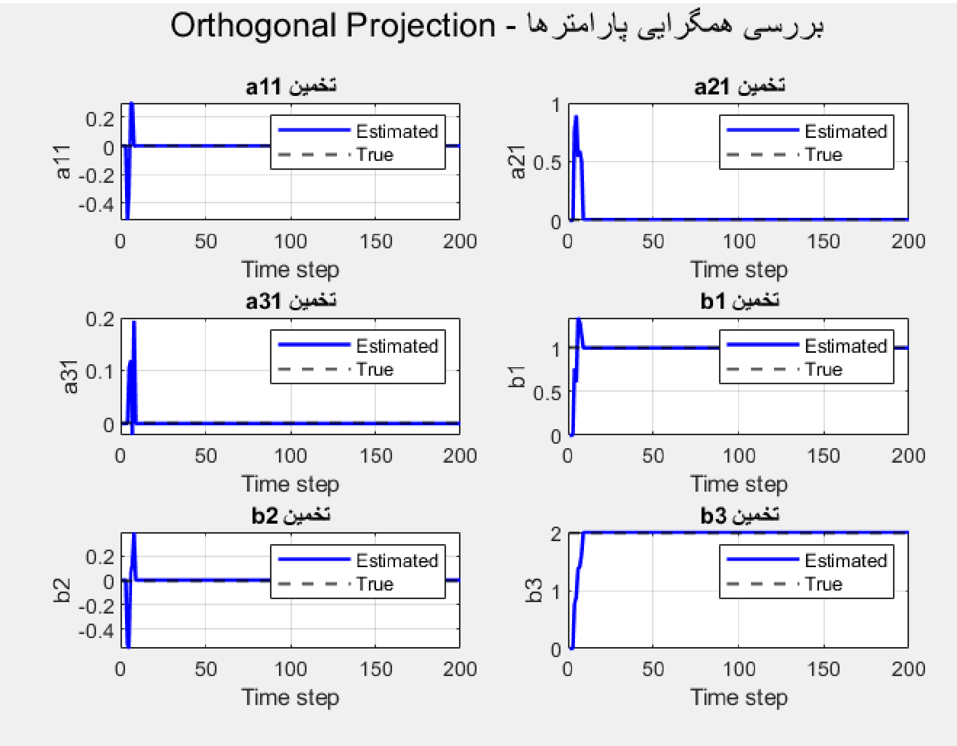
در نهایت، برای بررسی **محدود بودن تخمین‌ها(Boundedness)** ، بیشینه مقدار مطلق تخمین هر پارامتر در طول زمان محاسبه شد. نتایج نشان دادند که تخمین‌ها دارای دامنه محدودی بوده و دچار نوسانات شدید یا واگرایی نشده‌اند، که صحت الگوریتم را از لحاظ پایداری عددی تأیید می‌کند.

a11	a21	a31	b1	b2	b3
0.40152	0.95543	0.40108	1	0.40465	2.0103

در مجموع، الگوریتم Orthogonal Projection در این شبیه‌سازی توانست ضرایب سیستم را با دقت بالا و رفتار پایدار تخمین بزند و شرایط نظری لازم برای همگرایی را نیز به‌خوبی برآورده کرد.

```
:Orthogonal Projection تخمین نهایی پارامترها با
theta_hat = [a11; a21; a31; b1; b2; b3]
    0.0000
    0.0100
   -0.0000
    1.0000
    0.0000
    2.0000
```

همچنین برای بررسی تغییرات متغیر های موجود در کد ماتریس P به 100I تغییر داده شد که همانطور که در تصاویر زیر مشاهده می شود دقت افزایش یافته و نوسان ها کاهش می یابند و با سرعت بیشاری همگرا می شود.



a11	a21	a31	b1	b2	b3
0.5245	0.89396	0.19487	1.345	0.55709	2

2-3- الگوریتم RLS

در این بخش، پیاده‌سازی الگوریتم (RLS) Recursive Least Squares برای تخمین ضرایب سیستم دینامیکی با مدل حالت خطی بوده است. سیستم دارای سه متغیر حالت و یک ورودی تصادفی بوده و تنها خروجی مشاهده‌شده متغیر x1 می‌باشد. مدل سیستم به‌صورت یک معادله حالت در نظر گرفته شده و ضرایب ماتریس‌های A و B در آن مشخص و ثابت هستند.

ابتدا شبیه‌سازی سیستم انجام شد؛ با استفاده از یک ورودی نویزی و مدل خطی $x(t+1)=Ax(t)+Bu(t)$ ، خروجی سیستم $y(t)=Cx(t)$ تولید شد. سپس بردار ویژگی $\phi(t)$ شامل ۳ مقدار گذشته از خروجی و ۳ مقدار گذشته از ورودی تشکیل شد. این بردار برای تخمین ضرایب ۶ تایی شامل سه ضریب ai و سه ضریب bi استفاده شد.

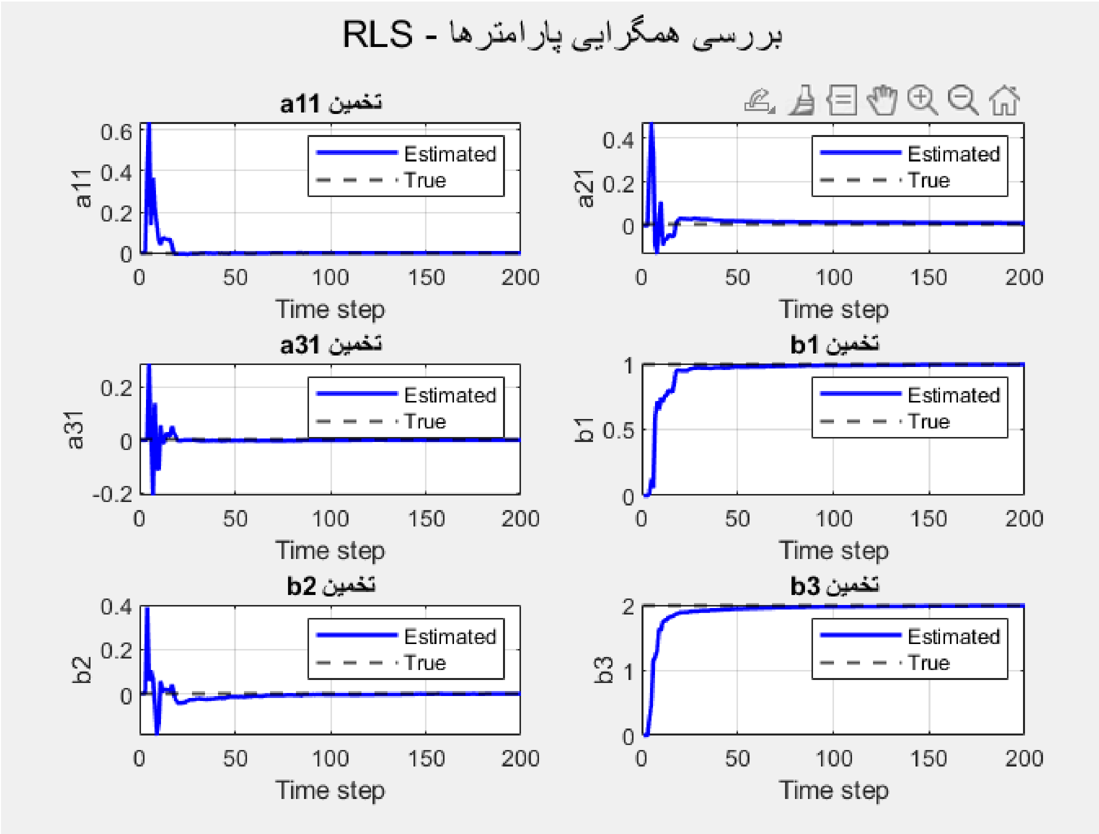
```
theta_hat = zeros(6, 1);        % [a11; a21; a31; b1; b2; b3]
```

الگوریتم RLS با مقداردهی اولیه مناسب برای بردار تخمین و ماتریس کوواریانس آغاز شد. در هر تکرار، ابتدا خطای تخمین $e(t)$ محاسبه شد و سپس بردار تخمین پارامترها theta و ماتریس کوواریانس $P(t)$ بر اساس معادلات به‌روزرسانی شدند. هدف الگوریتم این است که تخمین‌ها با مشاهده داده‌ها در طول زمان به مقدار واقعی پارامترها همگرا شوند.

```
for t = 4:T
    phi = [y(t-1); y(t-2); y(t-3); u(t-1); u(t-2); u(t-3)];
    phit = phi';
    Phi_all(:, t-3) = phi;     % ذخیره برای بررسی تحریک پایدار

    e = y(t) - phit * theta_hat;
    denom = 1 + phit * P * phi;

    theta_hat = theta_hat + (P * phi) * (e / denom);
    P = P - (P * phi * phit * P) / denom;

    theta_history(:,t) = theta_hat;
end
```

برای ارزیابی عملکرد الگوریتم، سه معیار مورد بررسی قرار گرفت:

1. **همگرایی پارامترها** :نمودار پارامترهای تخمینی در مقایسه با مقادیر واقعی نشان داد که الگوریتم RLSقادر است ضرایب واقعی را با دقت خوبی تخمین بزند و به آن‌ها همگرا شود.



2. **بررسـی شـرط تحریک پایدار** :(Persistency of Excitation) برای اینکه تخمین‌ها معتبر باشند، بردارهای ورودی باید مستقل خطی باشند. این موضوع با محاسبه رتبه ماتریس Φ بررسی شد و نتیجه حاکی از برقرار بودن این شرط در این آزمایش بود.

```
رتبه ماتریس Phi_all (برای بررسی تحریک پایدار) = 6
شرط تحریک پایدار برقرار است.
```

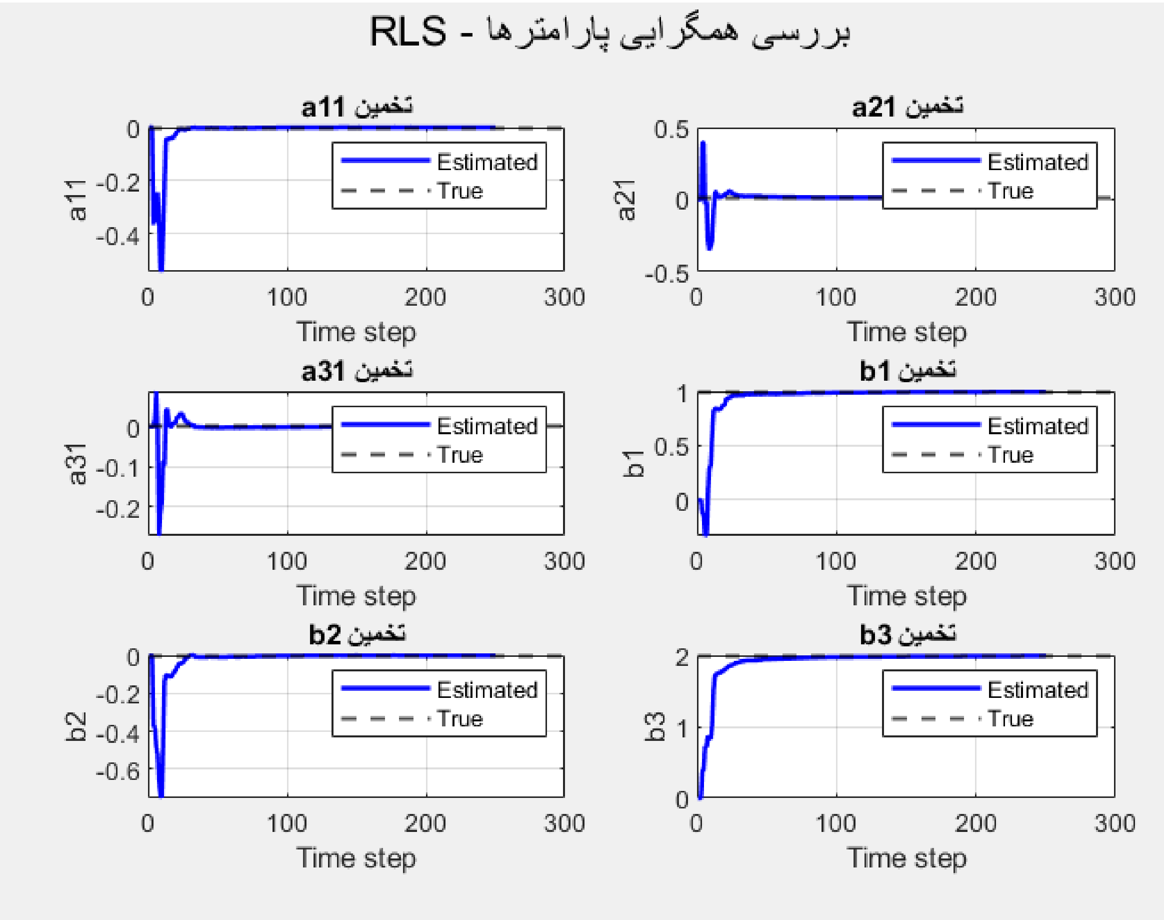
3. **محدود بودن تخمین‌ها** :(Boundedness) برای اطمینان از عدم ناپایداری الگوریتم، بیشینه مقدار مطلق تخمین‌های هر پارامتر در طول شبیه‌سازی محاسبه شد. مقادیر به‌دست آمده در محدوده‌های منطقی قرار داشتند.

```
بیشترین مقدار مطلق هر پارامتر در طول تخمین:
     a11        a21       a31        b1         b2        b3
   _______    _______   ______    _______    ______    ______

   0.63762    0.47382   0.2886    0.99319    0.3889    1.9852
```

نتایج پارامتر ها:

```
تخمین نهایی پارامترها با RLS:
theta_hat = [a11; a21; a31; b1; b2; b3]
   -0.0000
    0.0125
   -0.0002
    0.9963
    0.0004
    1.9892
```

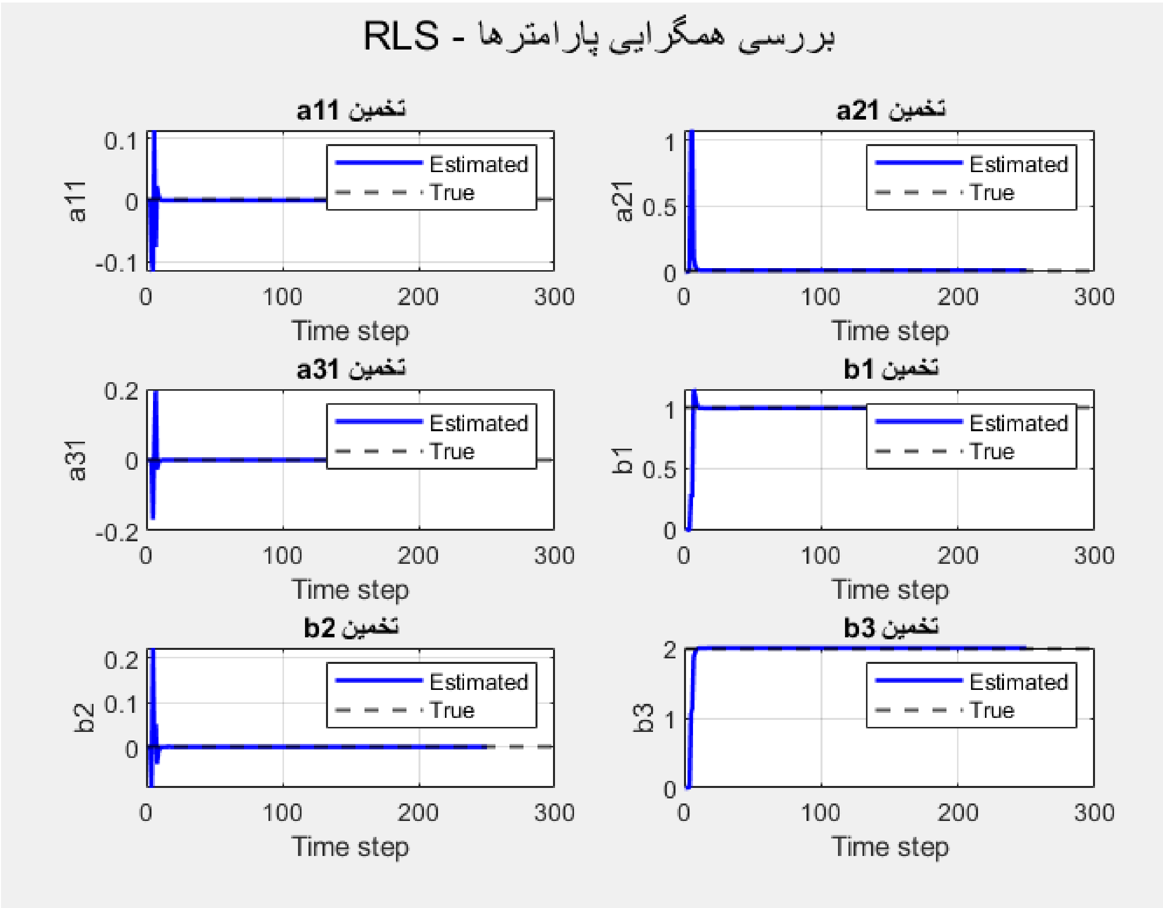
همانطور که مشاهده می شود کد اولیه نوشته شده هنوز سیستم را با دقت کافی نمی تواند تخمین بزند به همین دلیل در گام اول نمونه ها را این بار به جای۲۰۰ به ۲۵۰ تغییر داده و نتایج را بار دیگر مشاهده میکنیم.




```
تخمین نهایی پارامترها با RLS:
theta_hat = [a11; a21; a31; b1; b2; b3]
   -0.0002
    0.0118
   -0.0004
    0.9955
   -0.0002
    1.9899
```

نتایج کمی بهبود می یابند.

در گام دوم این بار ماتریس P را از I به 100I تغییر می دهیم.




```
theta_hat = [a11; a21; a31; b1; b2; b3]
   -0.0000
    0.0100
    0.0000
    1.0000
   -0.0000
    1.9999
```

نتایج به‌دست آمده نشان می‌دهد که الگوریتم RLS با تنظیمات مناسب و تحریک ورودی مناسب، می‌تواند با دقت بالا پارامترهای سیستم را تخمین بزند. این الگوریتم پایه‌ای برای نسخه‌های پیشرفته‌تر مانند RLS با وزن‌دهی نمایی یا تغییر ماتریس کوواریانس محسوب می‌شود.

فصل سوم

بررسی متد های الگوریتم RLS برای تخمین پارامتر های سیستم گسسته

سوال ii: انواع متد های RLS

در این بخش، هدف تخمین ضرایب یک سیستم دینامیکی خطی گسسته با استفاده از سه نسخه‌ی متفاوت از الگوریتم **Recursive Least Squares (RLS)** است. برای این منظور، سیستم با ضرایب مشخص و ورودی تصادفی گاوسی شبیه‌سازی شده و خروجی آن جهت تخمین ضرایب مورد استفاده قرار گرفته است. پارامترهای مورد تخمین شامل ضرایب ماتریس‌های A و B می‌باشند و مجموعاً ۶ پارامتر داریم.

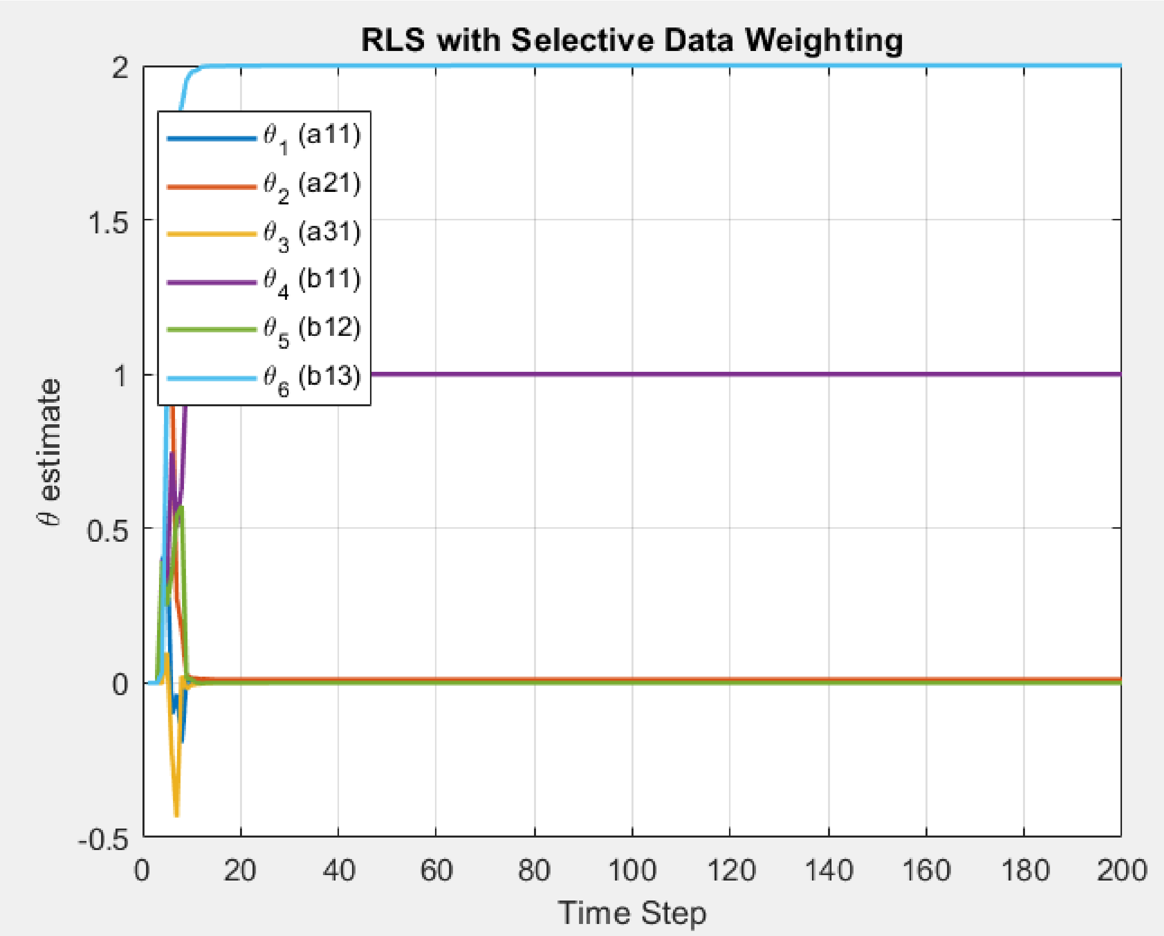
3-1- الگوریتم RLS متد Selective Data Weighting

در این نسخه، از مفهوم وزن‌دهی گزینشی (Selective Weighting) استفاده شده است. در هر گام زمانی، ابتدا اهمیت بردار رگرسیون $\varphi(t-1)$ با استفاده از عبارت $\phi t\ P\ \phi$ سنجیده می‌شود. اگر مقدار آن بزرگ‌تر از آستانه‌ی ε باشد، وزن بالاتری (k1) به داده داده می‌شود وگرنه وزن کمتری (k2) تعلق می‌گیرد. این روش باعث کاهش تأثیر داده‌های با اطلاعات ضعیف و جلوگیری از تخریب تخمین می‌شود. تخمین پارامتر و به‌روزرسانی ماتریس کوواریانس نیز با توجه به این ضریب اصلاح‌شده انجام می‌گیرد.

```
% بردار رگرسیون ϕ(t-1)
phi = [y(t-1); y(t-2); y(t-3); u(t-1); u(t-2); u(t-3)];
phit = phi';

% بررسی اهمیت داده
phi_P_phi = phit * P * phi;
if phi_P_phi >= epsilon
    a = k1;
else
    a = k2;
end

% خطای پیش‌بینی
e = y(t) - phit * theta_hat;

% مخرج بهینه‌شده با وزن
denom = 1 + a * phi_P_phi;

% آپدیت θ
theta_hat = theta_hat + a * (P * phi) * (e / denom);

% آپدیت ماتریس کوواریانس
P = P - a * (P * phi * phit * P) / denom;

% ذخیره تخمین‌ها
theta_history(:,t) = theta_hat;
```

که k1 و k2 به ترتیب ۱ و ۰.۰۰۱ انتخاب شده اند.

نتیجه در نهایت به صورت زیر می باشد که نشان می دهد پارامتر ها به درستی تخمین زده شده اند.



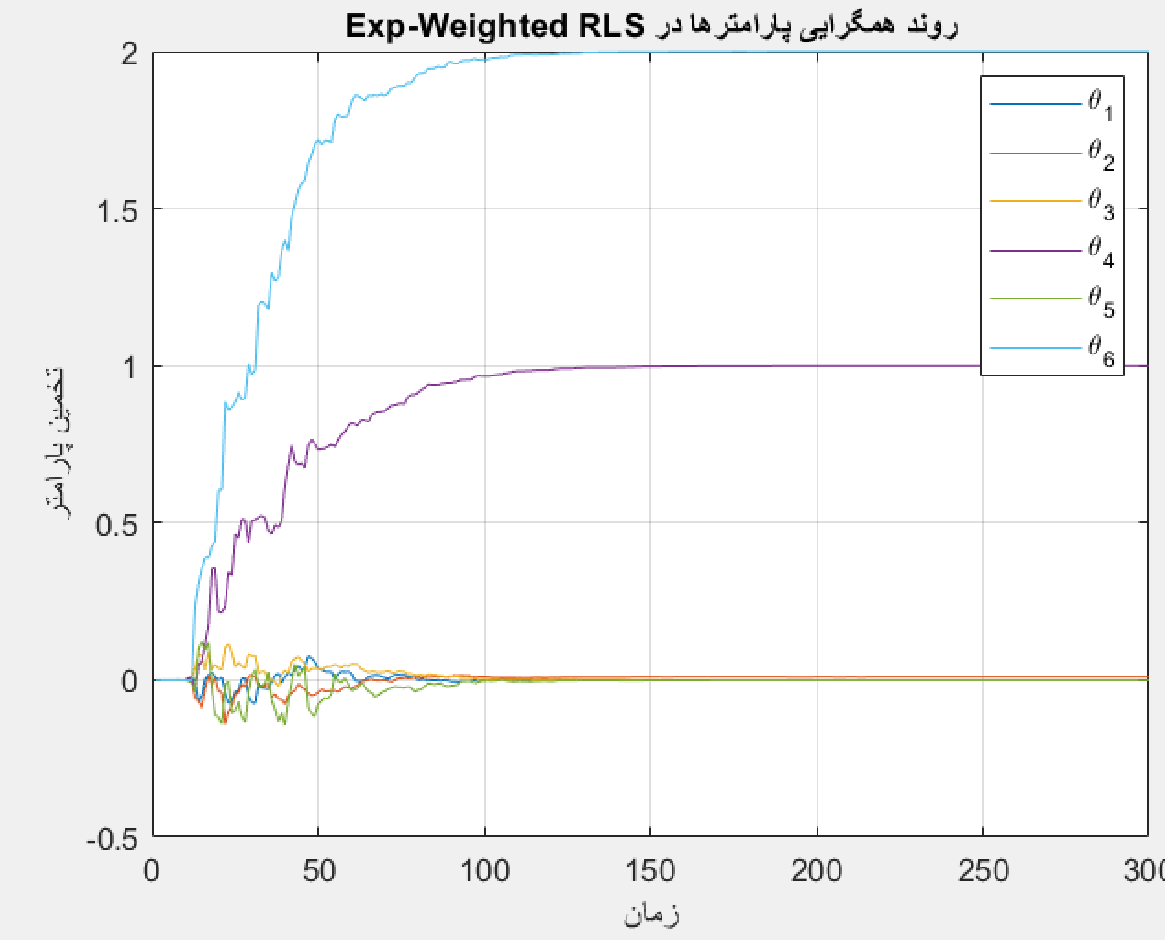

```
:(Modified RLS (Selective Weighting تخمین نهایی پارامترها با
theta_hat = [a11; a21; a31; b11; b12; b13]
    0.0000
    0.0100
   -0.0000
    0.9999
    0.0000
    1.9999
```

3-2- الگوریتم RLS متد Exponentially Weighted

در این نسخه از الگوریتمRLS ، وزن‌دهی داده‌ها بر اساس یک فاکتور فراموشی نمایی $\alpha(t)$ انجام می‌شود که در طول زمان به‌روزرسانی می‌گردد. در این روش، داده‌های اخیر وزن بیشتری دارند، چرا که فرض بر آن است که سیستم ممکن است تغییر کند. $\alpha(t)$ نیز به صورت دینامیکی از ترکیب α_0 و مقدار قبلی خود به‌روزرسانی می‌شود. این روش برای سیستم‌هایی که پارامترهایشان در حال تغییر هستند کارایی بالایی دارد.

```
alpha(1) = 0.7;                     % alpha(0)
alpha0 = 0.99;
eps_safe = 1e-6;

% ===== اجرای الگوریتم =====
for t = 1:T-1
    % phi(t-1)  برای t=1 یعنی phi(0)
    if t < 4
        phi = zeros(6,1); % چون y(t-1), y(t-2), y(t-3) نداریم
    else
        phi = [y(t); y(t-1); y(t-2); u(t); u(t-1); u(t-2)];
    end
    phit = phi';

    % مخرج
    denom = alpha(t) + phit * P(:,:,t) * phi;


    % آپدیت P(t)
    P(:,:,t+1) = (1/alpha(t)) * ...
        (P(:,:,t) - (P(:,:,t) * phi * phit * P(:,:,t)) / denom);

    % خطای پیشبینی
    y_pred = phit * theta_hat(:,t);
    e = y(t+1) - y_pred;

    % آپدیت θ(t+1)
    theta_hat(:,t+1) = theta_hat(:,t) + (P(:,:,t+1) * phi) * (e / denom);

    % آپدیت α(t+1)
    alpha(t+1) = alpha0 * alpha(t) + (1 - alpha0);
end
```

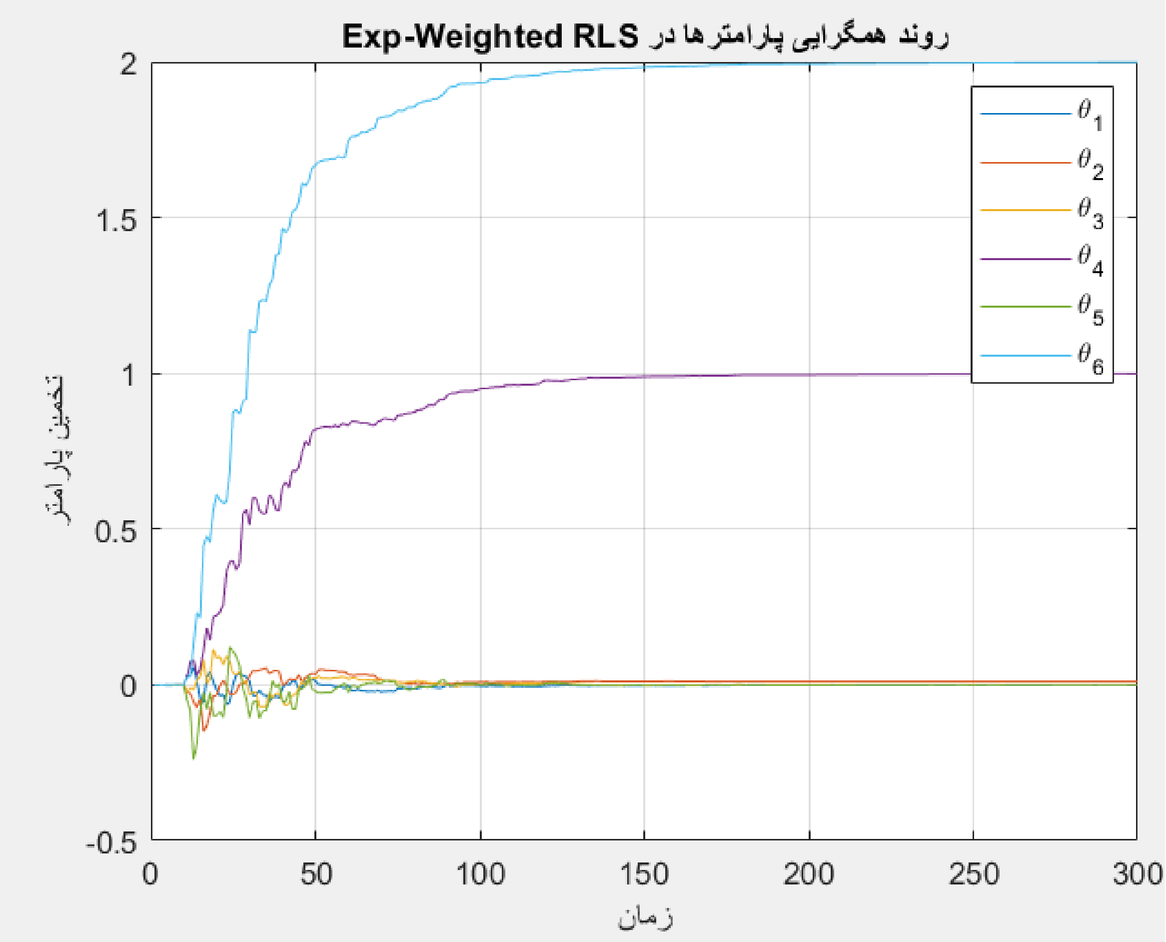
نتایج این الگوریتم به صورت زیر می باشد.




```
: Exp-Weighted RLS تخمین نهایی پارامترها با
theta_hat = [a11; a21; a31; b11; b12; b13]
    0.0000    0.0100    0.0000    1.0000   -0.0000    2.0000
```

همانطور که مشاهده می شود نسبت به متد قبل از سرعت پایین تر برخوردار است اما با دقت بیشتری تخمین میزند.

اگر مقدار اولیه الفا را ۰.۷ این بار به ۰.۹ تغییر دهیم نتیجه به صورت زیر حاصل می شود:




```
تخمین نهایی پارامترها با Exp-Weighted RLS :
theta_hat = [a11; a21; a31; b11; b12; b13]
   -0.0001    0.0101    0.0000    0.9990   -0.0000    1.9984
```

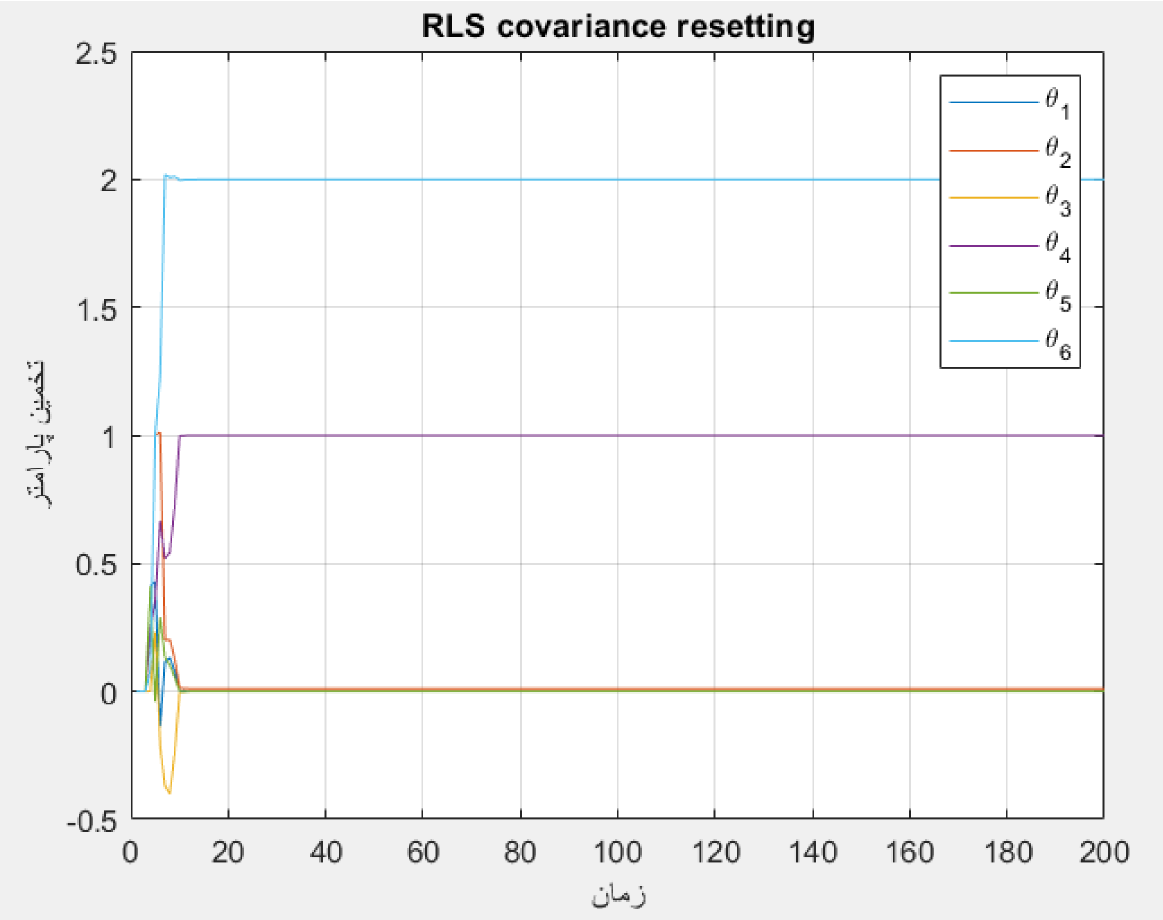
همانطور که مشخص است عملکرد سیستم تصعیف می شود و این وابستگی الگوریتم به این پارامتر را نشان میدهد.

3-3- الگوریتم RLS متد Covariance Resetting

در این نسخه، الگوریتم RLS کلاسیک با یک ویژگی جدید ترکیب شده است: هر چند گام زمانی مشخص (در این کد 30 گام)، ماتریس کوواریانس P به مقدار اولیه بزرگ بازنشانی می‌شود. این کار موجب بازیابی حساسیت الگوریتم به تغییرات در داده‌ها شده و از کند شدن یا انجماد ماتریس P در طول زمان جلوگیری می‌کند. در گام‌های دیگر، به‌روزرسانی معمولی RLS انجام می‌گیرد.

```
% ===== اجرای الگوریتم =====
for t = 2:T
    % ϕ(t-1)
    if t < 4
        phi = zeros(6,1);  % چون داده های کافی نداریم
    else
        phi = [y(t-1); y(t-2); y(t-3); u(t-1); u(t-2); u(t-3)];
    end
    phit = phi';

    % استفاده از P(t-2)
    P_prev2 = P(:,:,t-1);
    theta_prev = theta_hat(:,t-1);

    % مخرج
    denom = 1 + phit * P_prev2 * phi;


    % خطای پیشبینی
    e = y(t) - phit * theta_prev;

    % آپدیت θ
    theta_hat(:,t) = theta_prev + (P_prev2 * phi) * (e / denom);

    % بازتنظیم کوواریانس در بازه های مشخص
    if mod(t, reset_period) == 0
        P(:,:,t) = K0 * eye(6);    % ریست
    else
        P(:,:,t) = P_prev2 - (P_prev2 * phi * phit * P_prev2) / denom;
    end
end
```

نتیجه به صورت زیر حاصل میشود.




```
تخمین نهایی پارامترها با RLS (Covariance Resetting):
theta_hat = [a11; a21; a31; b11; b12; b13]
    0.0000    0.0100   -0.0000    1.0000   -0.0000    2.0000
```

که نسبت به دو الگوریتم قبلی از دقت و سرعت بالاتری برخوردار می باشد.

3-4- مقایسه سه متد

در پایان اجرای هر الگوریتم، روند همگرایی پارامترهای تخمینی در قالب نمودار رسم گردید. نتایج نشان داد که:

- الگوریتم **Selective RLS** در حذف اثر داده‌های کم‌اطلاعات عملکرد مناسبی داشت و همگرایی نسبتاً یکنواختی از خود نشان داد.
- الگوریتم **Exp-Weighted RLS** به دلیل وزن بیشتر داده‌های اخیر، در شرایطی که سیستم دچار تغییرات لحظه‌ای شود، تطبیق‌پذیری بالاتری دارد؛ اما ممکن است در سیستم‌های با پارامتر ثابت، به دقت پایین‌تری برسد.
- الگوریتم **Covariance Resetting RLS** با بازتنظیم دوره‌ای، قادر است مجدداً به اطلاعات جدید حساس شود، ولی اگر دوره‌ی ریست به‌درستی انتخاب نشود، ممکن است نوسان در تخمین‌ها ایجاد کند.

در نهایت، **تمام الگوریتم‌ها قادر به تخمین نسبی درست پارامترهای سیستم بودند** ولی **توازن بین سرعت همگرایی، پایداری تخمین، و حساسیت به داده‌های جدید** بسته به نیاز مسئله تعیین‌کننده انتخاب الگوریتم بهینه است.

فصل چهارم

عملکرد الگوریتم ها در صورت متغیر شدن پارامتر ها

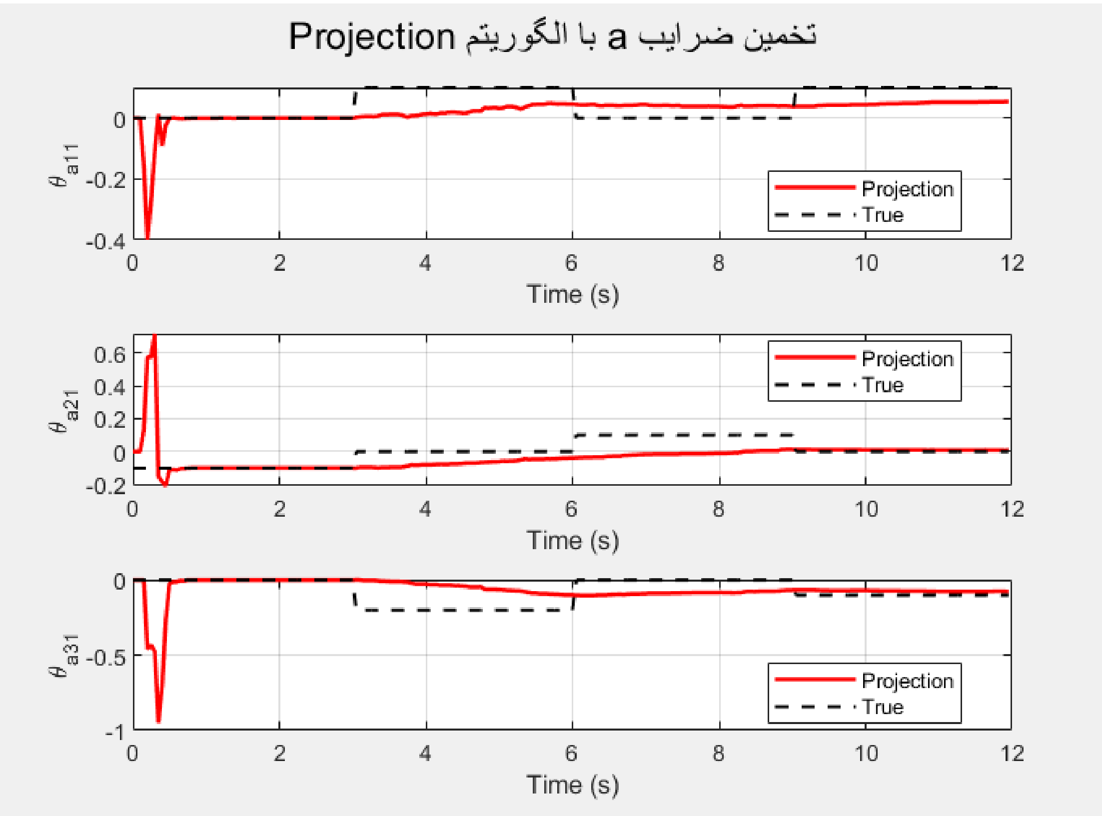
سوال iii: عملکرد الگوریتم ها در صورت متغیر شدن پارامتر ها

در این بخش، هدف تخمین ضرایب یک سیستم خطی گسسته با دینامیک زمان‌متغیر است. ضرایب ماتریس A به صورت تکه‌ای در زمان تغییر می‌کنند و رفتار سیستم بر اساس این ضرایب و یک ورودی تصادفی گاوسی شبیه‌سازی می‌شود. تمرکز اصلی بر تخمین ضرایب a11، a21 و a31 است که در طول زمان مقدارشان تغییر می‌کند.

4-1- Projection

در این الگوریتم، شرطی روی نُرم بردار ورودی φ اعمال می‌شود. تنها در صورتی که انرژی سیگنال φ کمتر از آستانه تعیین‌شده باشد، تخمین به‌روزرسانی می‌شود. این کار باعث حذف تأثیر نمونه‌های با اطلاعات ضعیف می‌شود.

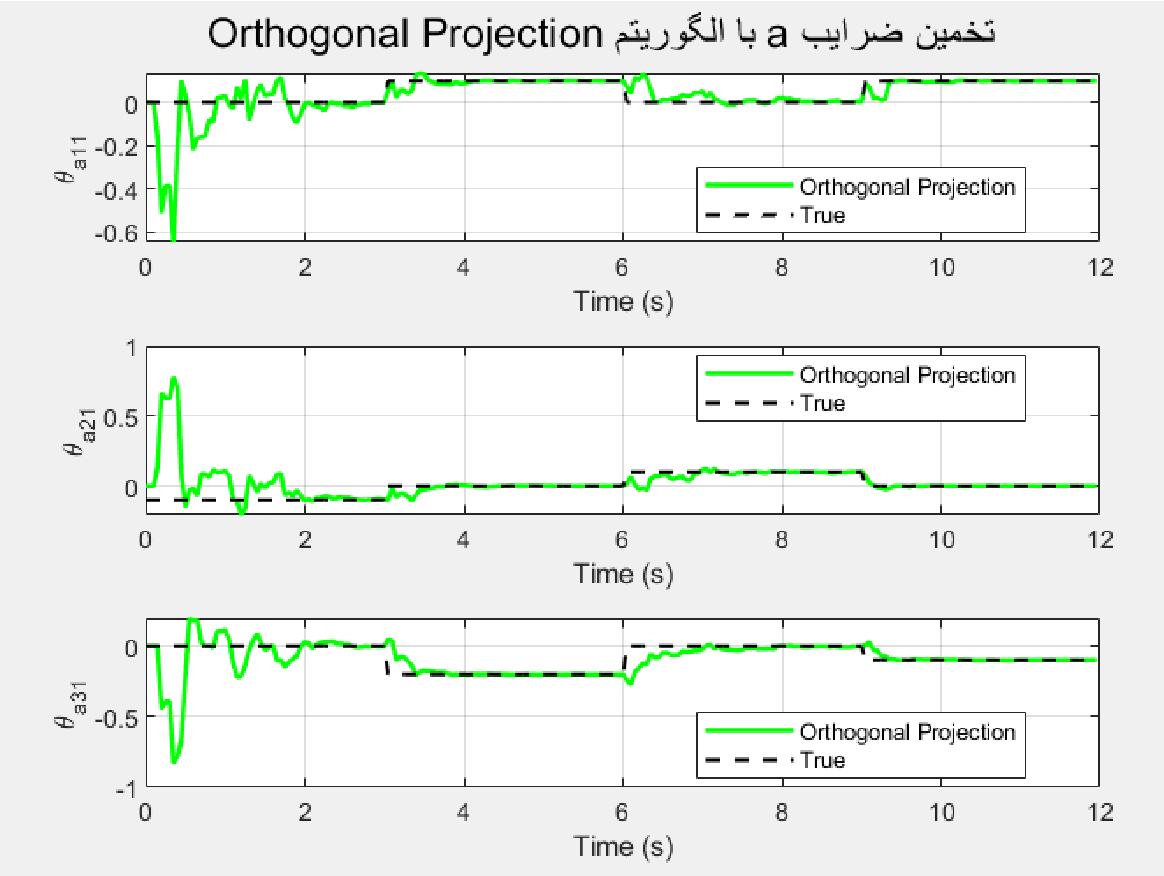
نتیجه این الگوریتم به صورت زیر می باشد که همانطور که مشاهده می شود عملکرد ضعیفی در پارامتر های متغیر با زمان دارد و سرعت پایین آن باعث می شود از تغییرات جا بماند:



2-4- Orthogonal Projection

نسخه ساده‌تر الگوریتم قبلی است که به‌جای استفاده از ماتریس کوواریانس، تنها با استفاده از ضرب داخلیφ ، به‌روزرسانی ضرایب را انجام می‌دهد. این روش دارای پیچیدگی محاسباتی کمتر اما دقت پایین‌تر نسبت به RLS است.

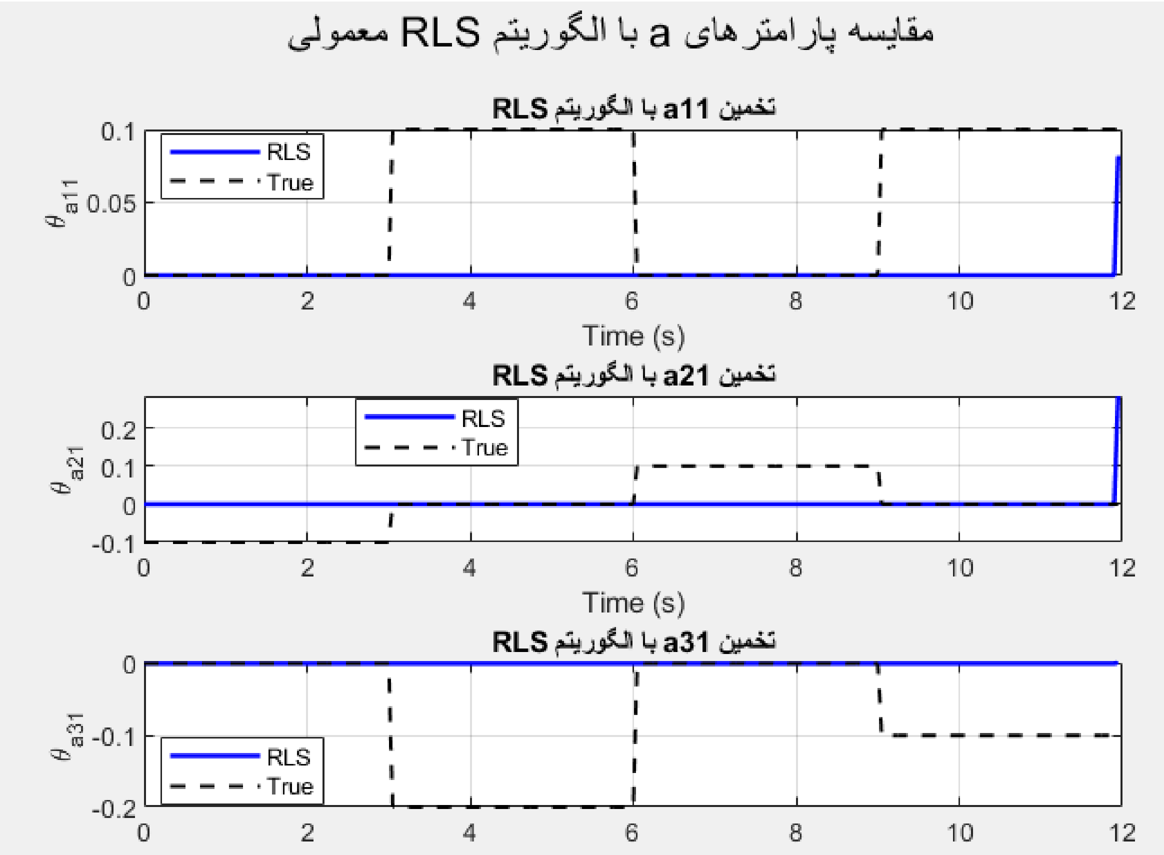
نتیجه این الگوریتم به صورت زیر می باشد که همانطور که مشاهده می شود عملکرد خوبی در پارامتر های متغیر با زمان دارد ولی دارای نوسان های متعدد می باشد:



Standard RLS-4-3

الگوریتم RLS کلاســیک با اســتفاده از معادلات به‌روزرســانی ماتریس کوواریانس P و بردار تخمین θ، عملکرد مناسبی در تعقیب ضرایب دارد، اما در شرایطی که ضرایب ناگهان تغییر کنند، کند عمل می‌کند

نتیجه این الگوریتم به صورت زیر می باشد که همانطور که مشاهده می شود عملکرد بسیار ضعیفی در پارامتر های متغیر با زمان دارد و بسیار کند عمل می کند:

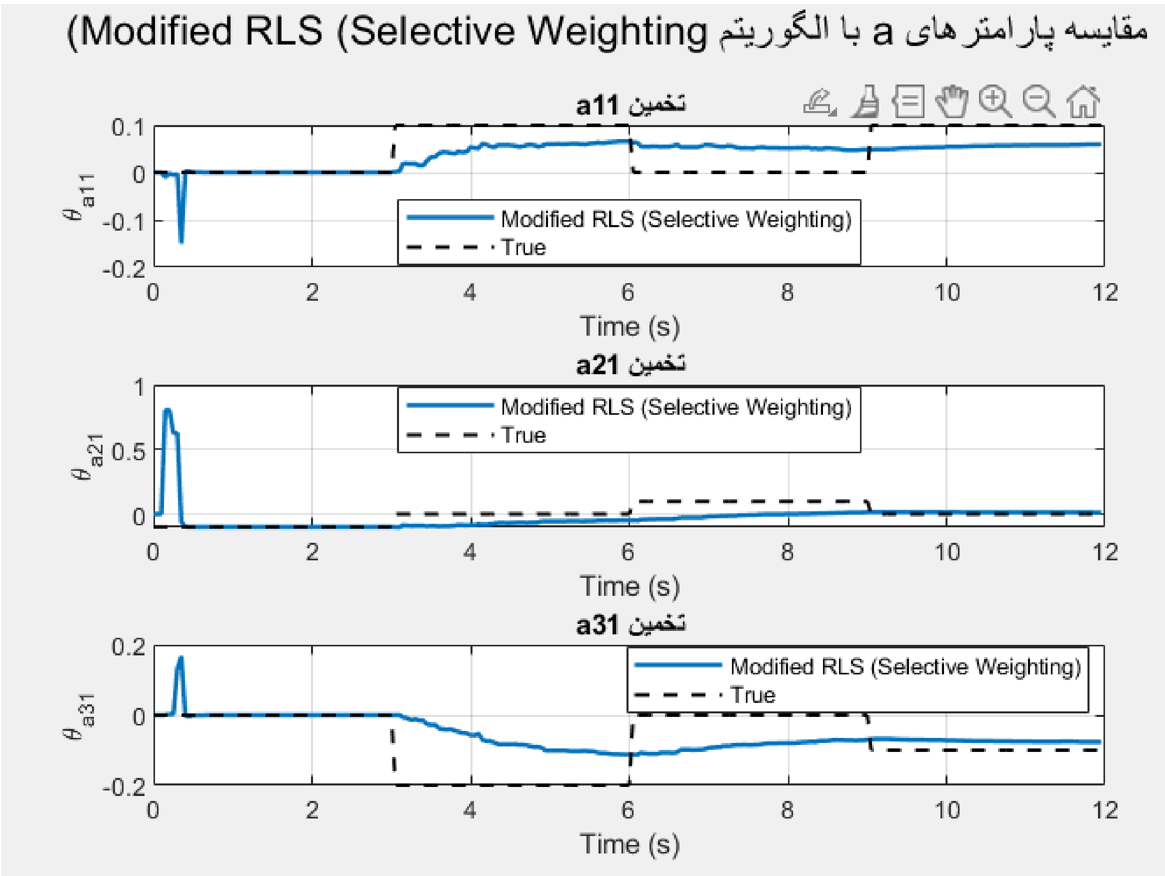


.

4-4- Modified RLS with Selective Weighting

ر این روش، از دو وزن متفاوت k1 و k2 برای بروزرسانی تخمین استفاده شده است. اگر قدرت اطلاعاتی ورودی یعنی ϕTPϕ از آستانه خاصی بیشتر باشد، وزن بزرگ‌تر به آن داده می‌شود. این الگوریتم بین داده‌های مفید و غیرمفید تمایز قائل می‌شود.

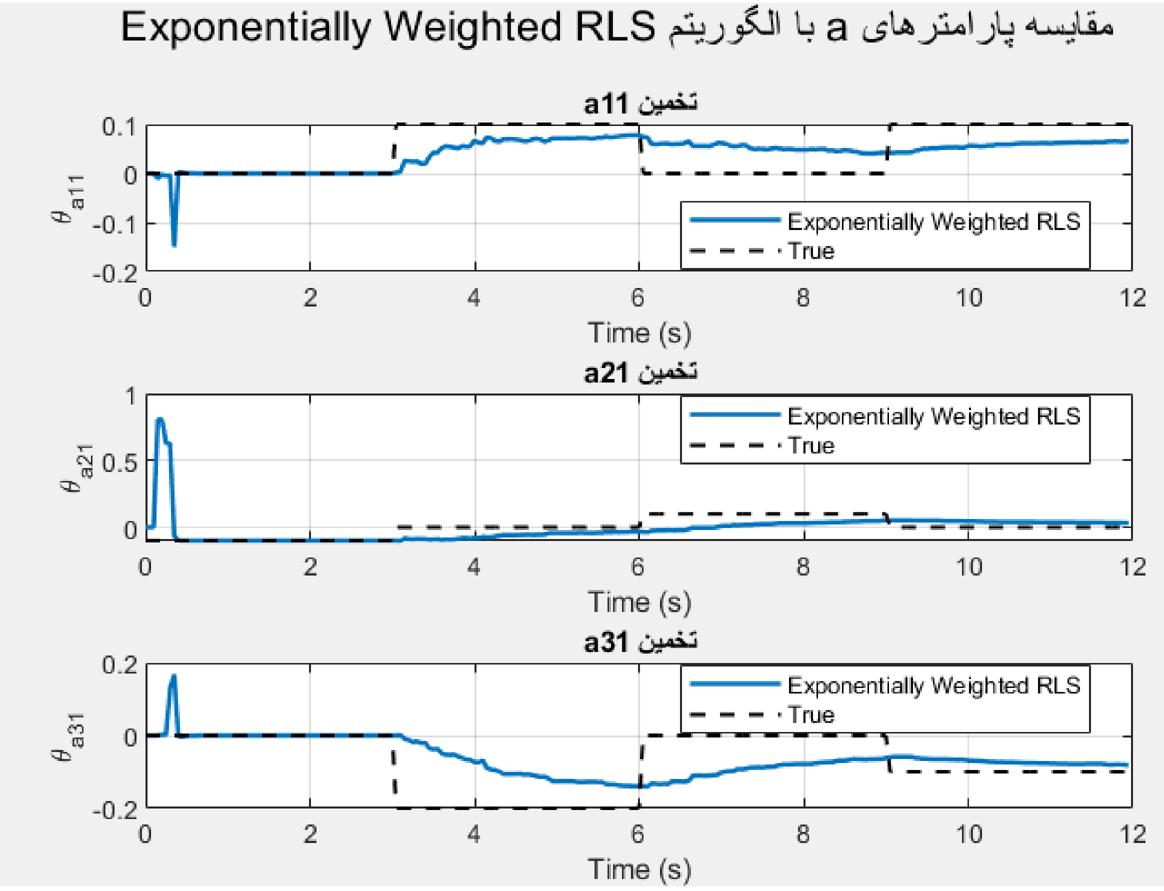
نتیجه این الگوریتم به صورت زیر می باشد که همانطور که مشاهده می شود عملکرد تقریبا ضعیفی در پارامتر های متغیر با زمان دارد و کند بودن آن باعث می شود نتواند به درستی تطبیق پیدا کند:



4-5- Exponentially Weighted RLS

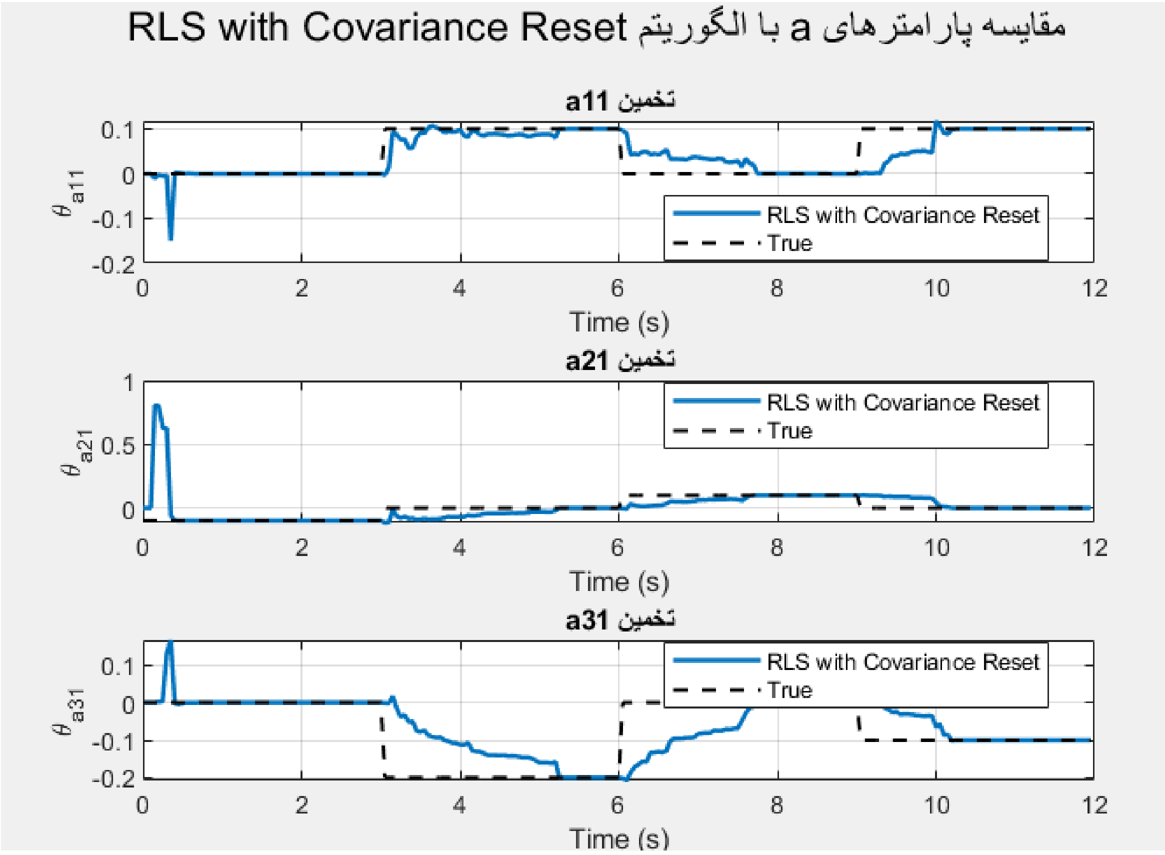
در این الگوریتم، با تعریف ضـــریب فراموشـــی λ ، تأثیر داده‌های قدیمی به‌مرور زمان کاهش می‌یابد. این ویژگی باعث بهبود تطبیق با ضرایب زمان‌متغیر می‌شود، اما اگر تغییرات ناگهانی باشد، پاسخ آهسته‌تری دارد.

نتیجه این الگوریتم به صـورت زیر می باشـد که همانطور که مشـاهده می شـود عملکرد تقریبا ضـعیفی در پارامتر های متغیر با زمان دارد و کند بودن آن باعث می شود نتواند به درستی تطبیق پیدا کند:



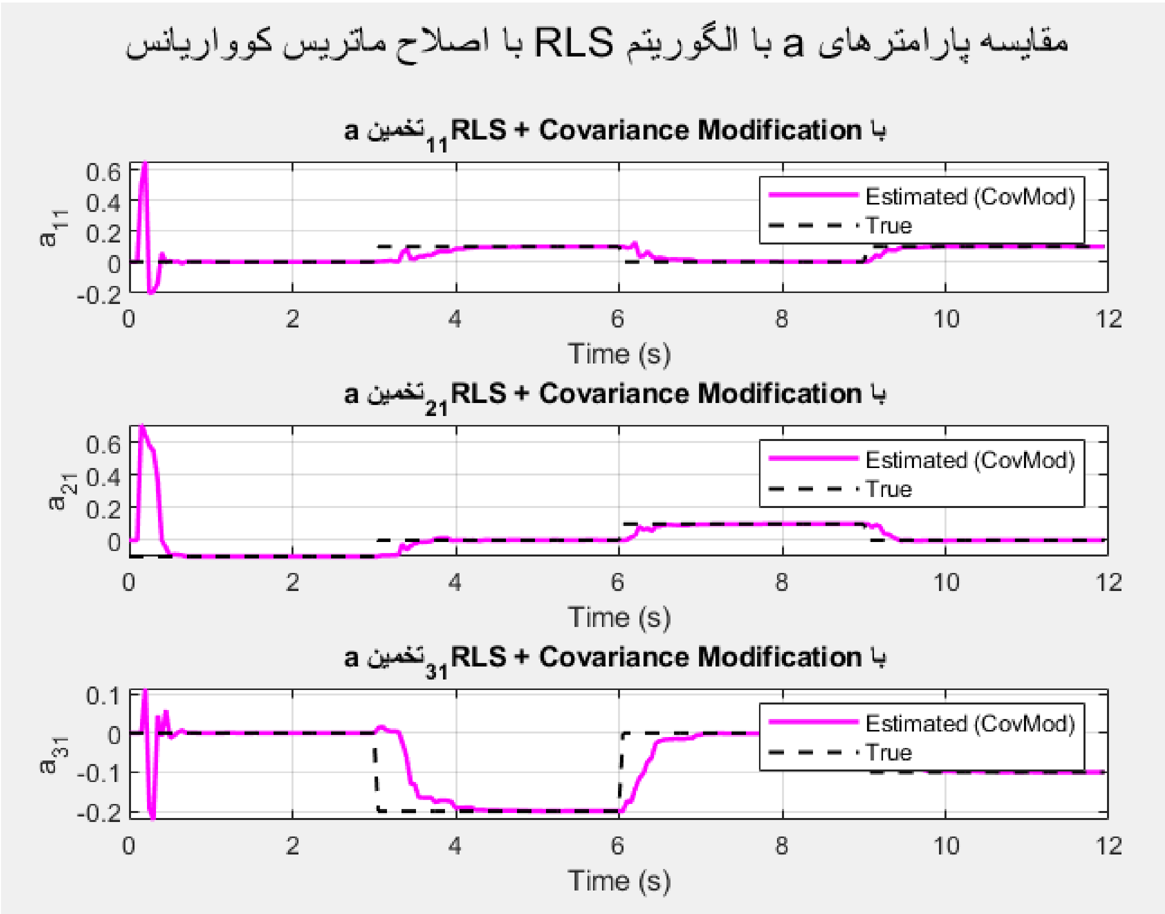
4-6- RLS with Covariance Reset

در این روش، ماتریس کوواریانس P پس از هر تعداد نمونه مشخص (مثلاً هر 50 نمونه)، مجدداً به مقدار اولیه ریست می‌شود. این الگوریتم برای دنبال‌کردن تغییرات ناگهانی مناسب است، اما ممکن است باعث نوسان در تخمین شود.



4-7- RLS with Covariance Modification

در این نسخه پیشرفته، پس از مرحله کلاسیک به‌روزرسانیRLS ، یک جمله اصلاحی Q(t) به ماتریس کوواریانس اضافه می‌شود. این جمله متناسب با مربع خطای تخمینe(t)^2 طراحی شده تا در صورتی که خطا بزرگ باشد، به الگوریتم اجازه دهد سریع‌تر با تغییر ضرایب واقعی همگام شود. مقدار Q محدود شده تا از واگرایی جلوگیری گردد. این روش نوعی تطبیق ساختاری درون الگوریتم RLS ایجاد کرده که همگرایی سریع و پایداری بالا را به‌صورت همزمان فراهم می‌کند.



4-8- نتیجه گیری

الگوریتم **Projection** به دلیل سادگی ساختار خود، در برابر نویز مقاوم است و می‌تواند در بسیاری از شرایط عملکرد قابل قبولی داشته باشد، اما به علت استفاده محدود از داده‌ها، در تخمین ضرایب متغیر دقت بالایی ندارد.

الگوریتم **Orthogonal Projection** نیز محاسبات بسیار سبکی دارد، اما به‌دلیل حذف ساختار ماتریس کوواریانس و استفاده صرف از ضرب داخلی بردار ورودی، به نویز حساس است و در برابر تغییرات ضرایب، سرعت همگرایی مناسبی ندارد.

الگوریتم **RLS کلاسیک** با استفاده از به‌روزرسانی ماتریس کوواریانس، دقت خوبی در تخمین ضرایب ایستا دارد؛ اما در مواجهه با ضرایب زمان‌متغیر یا تغییرات ناگهانی، با تأخیر نسبتاً زیاد واکنش نشان می‌دهد.

در مقابل، الگوریتم **Modified RLS with Selective Weighting** با اعمال وزن‌های مختلف بر اساس قدرت اطلاعاتی ورودی، توانایی تمایز بین داده‌های مفید و داده‌های کم‌ارزش را دارد. این ویژگی باعث می‌شود تنها داده‌هایی که اطلاعات کافی دارند، در تخمین پارامتر اثرگذار باشند؛ با این حال، حساسیت به انتخاب پارامترهای آستانه و وزن‌ها، یکی از نقاط ضعف آن است.

الگوریتم **Exponentially Weighted RLS** از طریق ضریب فراموشی λ داده‌های قدیمی را کم‌اثر می‌کند، بنابراین توانایی خوبی در تطبیق پیوسته با سیستم‌هایی با تغییرات نرم دارد. اما در برابر تغییرات جهشی یا ناگهانی در ضرایب، کند عمل می‌کند و تخمین ممکن است مدت زیادی از مقدار واقعی فاصله داشته باشد.

الگوریتم **RLS با Reset دوره‌ای**، با ریست کردن دوره‌ای ماتریس کوواریانس، باعث می‌شود که سیستم بتواند سریع‌تر با تغییرات ناگهانی ضرایب هماهنگ شود. این ویژگی برای سیستم‌هایی با پرش‌های لحظه‌ای مفید است، ولی در بازه‌هایی که پارامترها ثابت باقی می‌مانند ممکن است باعث بروز نوسانات و ناپایداری در تخمین شود.

در نهایت، الگوریتم **RLS with Covariance Modification**تلفیقی از دقت، پایداری و توانایی بالا در انطباق با شرایط متغیر سیستم ارائه می‌دهد. در این روش، یک جمله $Q(t)$ به صورت دینامیکی به ماتریس کوواریانس اضافه می‌شود که متناسب با میزان خطای تخمین است. در نتیجه، اگر خطا بزرگ باشد، ماتریس کوواریانس اجازه می‌دهد که بردار پارامتر با سرعت بیشتری به مقدار واقعی نزدیک شود.

این الگوریتم حتی در شرایطی که ضرایب ناگهانی تغییر می‌کنند، همچنان پایدار باقی می‌ماند و همگرایی خوبی از خود نشان می‌دهد.

بنابراین، می‌توان نتیجه گرفت که الگوریتم RLS با اصلاح ماتریس کوواریانس(Covarianc) در مقایسه با سایر روش‌ها، بهترین عملکرد کلی را دارد و مناسب‌ترین گزینه برای تخمین پارامتر در سیستم‌های با تغییرات زمان‌متغیر یا غیرایستا محسوب می‌شود.

Abstract