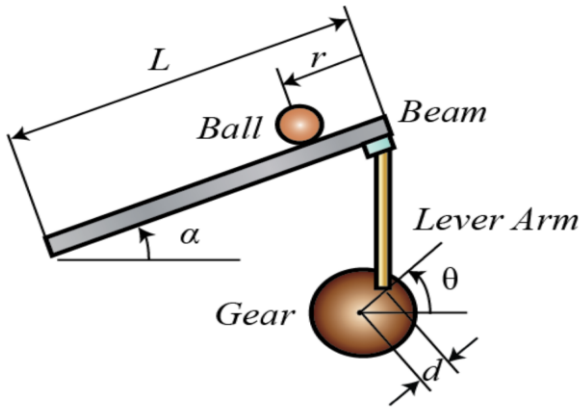
به نام خدا

گزارش پروژه توپ و میله

هستی محمدزاده، علیرضا مداح

- مقدمه :

سیستم کنترلی توپ و میله به صورت کلی مدل های مهم آزمایشگاهی در ارتباط با شاخه سیستم های مهندسی کنترل می باشد. مکانیزم کلی توپ و میله شامل دو میله و یک توپ سخت بر روی آن است. به این صورت که توپ آزادانه در امتداد میله تحت تغییر زاویه میله توانایی حرکت دارد و موقعیت توپ بر اساس سنسور مسافت سنج اندازه گیری می شود.



شماتیک کلی سیستم توپ و میله

- تعیین معادله حاکم بر سیستم و پارامتر های مربوطه :

بر اساس شماتیک کلی سیستم کنترلی میله و توپ، زاویه سروو موتور (θ) زاویه ای بر میله اعمال میکند (α). حال در اثر تغییر زاویه میله می توان انتظار تغییر موقعیت توپ را داشت که فاصله توپ از سر میله را به عنوان خروجی سیستم کنترلی خود (r) تعریف می کنیم. در ادامه به گزارش معادله غیرخطی سیستم می پردازیم :

$$\left(\frac{J}{R^2}+m\right)\ddot{r}+mg\sin\alpha-mr\dot{\alpha}^2=0$$

معادله غیرخطی سیستم حاکم بر سیستم

که ثابت ها و پارامتر های این معادله عبارت است از :

۱- m : جرم توپ = $0.11\ kg$

۲- R : شعاع توپ = $0.015\ m$

۳- d : آفست بازوی اهرم = $0.03\ m$

۴- g : شتاب گرانش زمین = $9.8\ m/s^2$

۵- L : طول میله $= 1.0\ m$

۶- J : اینرسی توپ $= 9.99 \times 10^{-6}\ (kg \cdot m^2)$

۷- r : موقعیت توپ

۸- α : زاویه میله با سطح افق

۹- θ : زاویه سروو موتور

- تعیین رنج ورودی و خروجی :

برای تحلیل رنج منطقی ورودی و خروجی سیستم، ابتدا بر روی رنج ورودی تحقیق می کنیم :

بر اساس شماتیک سیستم کنترلی توپ و میله می توان اینگونه بیان داشت که حداکثر زاویه برای اعمال سروو موتور به سیستم برابر با $90°$ و حداقل آن برابر $-90°$ درجه می باشد. بنابراین داریم :

$$\theta = (\ -90° . 90°\ )$$

همچنین بر اساس رابطه زیر که ارتباط میان زاویه سرو موتور با زاویه میله را بیان می کند، رنج زاویه میله را بدست آورد :

$$\alpha = \frac{d}{l}\theta$$

$$\alpha = \left(\left(-90° \times \frac{0.03}{1}\right) . \left(90° \times \frac{0.03}{1}\right)\right) = (\ -2.7° .\ 2.7°\ )$$

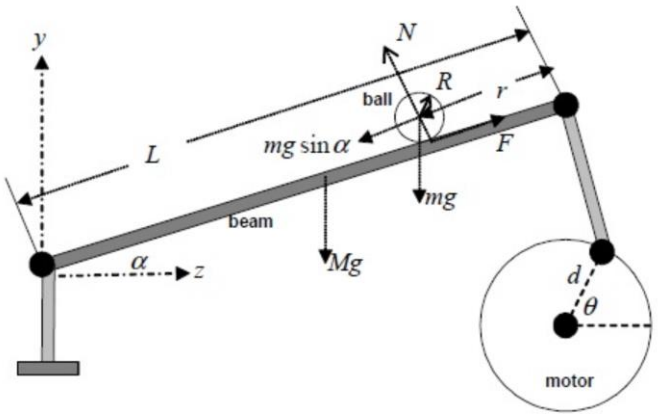
حال در این پروژه ما $\ddot{\alpha}$ را به عنوان ورودی سیستم که جنس گشتاور اعمالی به سروو موتور ما می باشد در نظر می گیریم.

همچنین در تحلیل رنج و دامنه خروجی می توان گفت موقعیت میله می تواند در سر تا سر طول میله باشد بنابراین رنج خروجی سیستم را برابر با $r\ = (0\ .\ l)$ گزارش می دهیم.

- تعیین نقطه کار :

در این تحلیل نقطه کار بر این مبنا سیستم را تحلیل می کنیم که $\ddot{\alpha} = 0$ که در واقع گشتاور یا ولتاژ اعمالی به سروو موتور برابر است و به ازای آن میله نیز در صفر مختصات قرار دارد.

$$(\ddot{\alpha} . r) = (0 . 0)$$



- خطی سازی حول نقطه کار :

ابتدا معادلات سیستم را بدست می آوریم (از طریق معادله غیرخطی گفته شده در ابتدای گزارش) :

متغیر های حالت :

r -۱

$\dot{r}$ -۲

α -۳

$\dot{\alpha}$ -۴

$$(I)\ \dot{r} = \dot{r}$$

$$(II)\ \ddot{r} = \frac{mr\dot{\alpha}^2 - mgsin(\alpha)}{\left(\frac{J}{R^2} + m\right)}$$

$$(III)\ \dot{\alpha} = \dot{\alpha}$$

$$(IV)\ \ddot{\alpha} = u$$

در نهایت با توجه به متغیر های حالت در نظر گرفته شده و معادلات بالا، به خطی سازی سیستم حول نقطه کار می پردازیم :

$$\begin{bmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\mathbf{x})}{\partial x_u} \\ \vdots & & \vdots \\ \frac{\partial f_v(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_v(\mathbf{x})}{\partial x_u} \end{bmatrix}$$

$$\begin{pmatrix} \dot{r} \\ \ddot{r} \\ \dot{\alpha} \\ \ddot{\alpha} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-mg}{(\frac{J}{R^2}+m)} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} r \\ \dot{r} \\ \alpha \\ \dot{\alpha} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} u$$

$$y = \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} r \\ \dot{r} \\ \alpha \\ \dot{\alpha} \end{pmatrix}$$

حال بر اساس ثابت های بیان شده در گزارش برای ماتریس خطی سازی شده داریم :

$$\begin{pmatrix} \dot{r} \\ \ddot{r} \\ \dot{\alpha} \\ \ddot{\alpha} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -6.9819 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} r \\ \dot{r} \\ \alpha \\ \dot{\alpha} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} u$$

$$y = \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} r \\ \ddot{r} \\ \alpha \\ \dot{\alpha} \end{pmatrix}$$

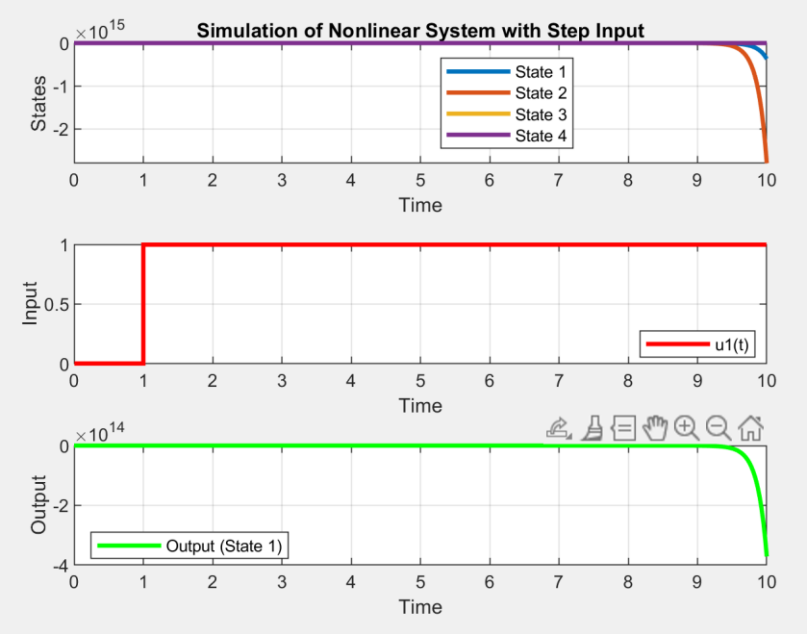
- شبیه سازی مقایسه رفتار سیستم غیرخطی و سیستم خطی سازی شده به ورودی یکسان پله :

ابتدا بر اساس کد زیر سیستم غیرخطی را شبیه سازی می کنیم و داریم :

Editor - C:\Users\Alireza\Desktop\nonlinear.m*

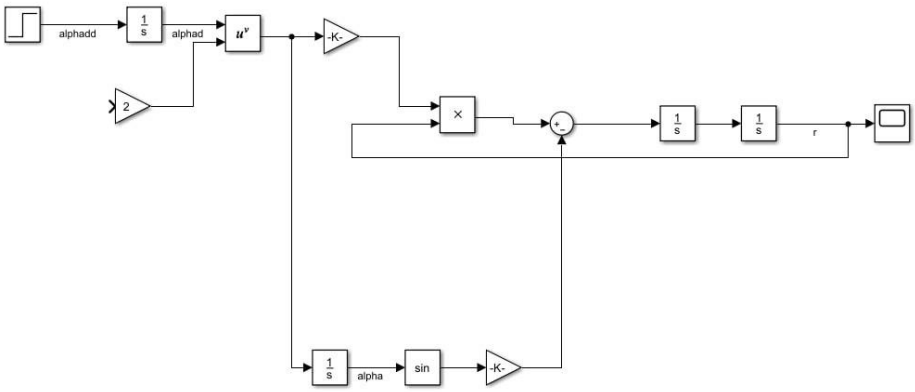
control_project.m | eleven_full_order_observer.m | stability.m | state_feedback_integrator.m | eight_state_feedback.m | nonl

```
m = 0.11;
g = 9.8;
j = 9.99*10^(-6);
R = 0.015;
d = 0.03;
l = 1;
u1 = @(t) 1 * (t >= 1)  ;
ode_function = @(t, x) [
    x(2);
   (m*x(1)*(x(4)^2)-m*g*sin(x(3)))/((j/R^2)+m);
    x(4);
    u1(t);
];
initial_conditions = [0.; 0; 0; 0];
t_span =  linspace(0, 10, 400);
[t, x] = ode45(ode_function, t_span, initial_conditions);
y = x(:, 1);

figure;
subplot(3, 1, 1);
plot(t, x(:, 1), 'LineWidth', 2, 'DisplayName', 'State 1');
hold on;
plot(t, x(:, 2), 'LineWidth', 2, 'DisplayName', 'State 2');
hold on;
plot(t, x(:, 3), 'LineWidth', 2, 'DisplayName', 'State 3');
hold on;
plot(t, x(:, 4), 'LineWidth', 2, 'DisplayName', 'State 4');
xlabel('Time');
ylabel('States');
title('Simulation of Nonlinear System with Step Input');
legend('Location', 'Best');
grid on;

subplot(3, 1, 2);
stairs(t, u1(t), 'r', 'LineWidth', 2, 'DisplayName', 'u1(t)');
hold on;
xlabel('Time');
ylabel('Input');
legend('Location', 'Best');
grid on;

subplot(3, 1, 3);
plot(t, y, 'g', 'LineWidth', 2, 'DisplayName', 'Output (State 1)');
xlabel('Time');
ylabel('Output');
legend('Location', 'Best');
grid on;
```



همچنین به طراحی شماتیک سیستم غیر خطی در محیط Simulink متلب می پردازیم :

با توجه به معادله غیر خطی حاکم بر سیستم، بلوک ها را پیاده سازی می کنیم که به شرح زیر است :

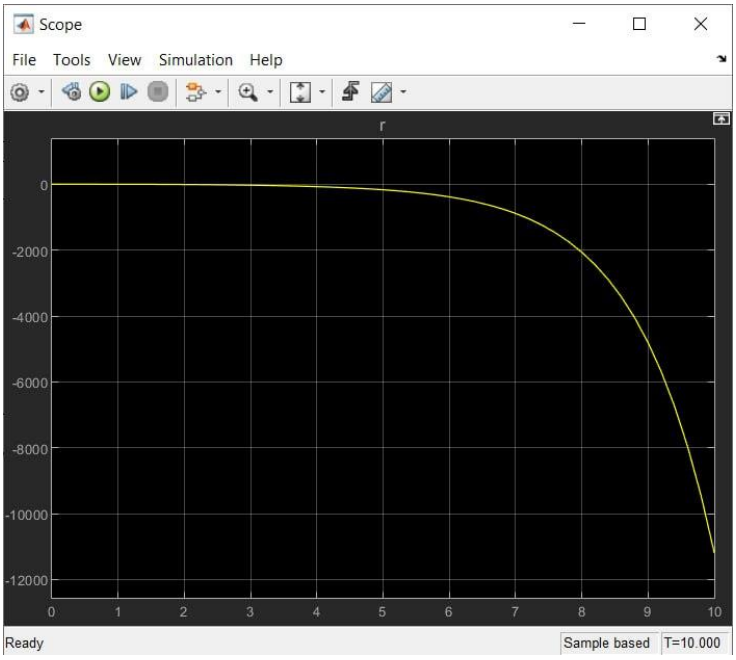


بلوک دیاگرام کلی سیستم غیر خطی

همچنین توجه داریم که :

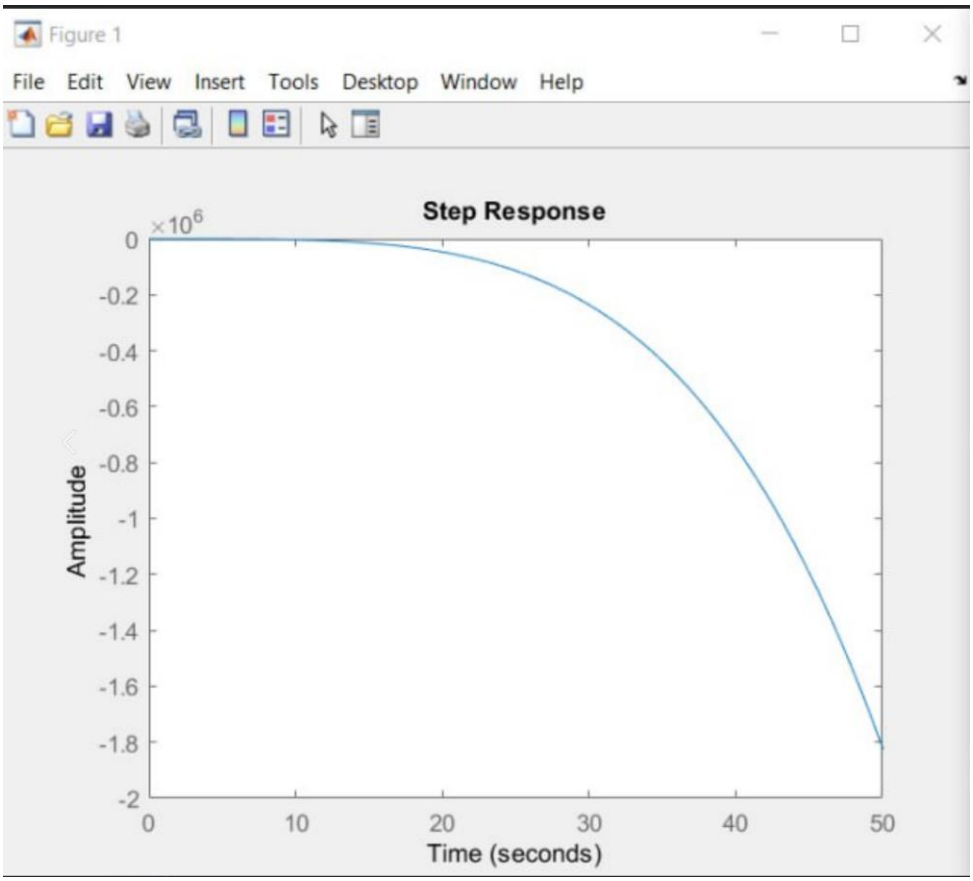
$$\ddot{r} = \frac{(mr\dot{\alpha}^2 - mg\sin\alpha)}{\left(\frac{J}{R^2} + m\right)}$$

با توجه به متغیر های حالت $r\ .\dot{r}\ .\alpha\ .\dot{\alpha}$ از انتگرالگیر برای رسیدن به حالت های مختلف سیستم استفاده میکنیم. در نهایت با دادن ورودی پله عنوان ورودی، خروجی سیستم که r است را توسط اسکوپ مشاهده می کنیم :



برای مشاهده رفتار سیستم خطی به ورودی پله، از کد step(sys) استفاده میکنیم. کد آن به صورت زیر است:

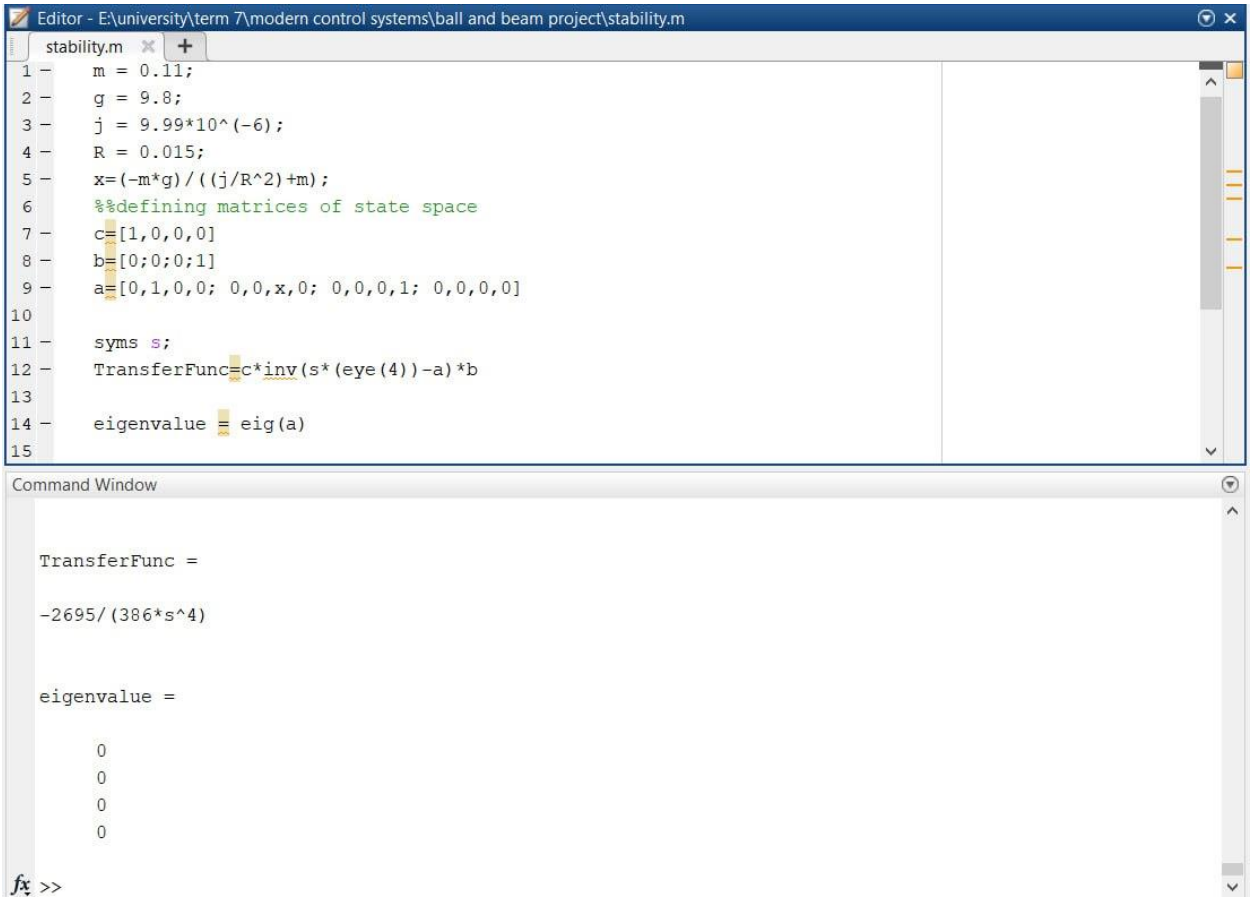
```
%%defining matrices of state space
C=[1,0,0,0]
B=[0;0;0;1]
A=[0,1,0,0; 0,0,x,0; 0,0,0,1; 0,0,0,0]

syms s;
TransferFunc=C*inv(s*(eye(4))-A)*B

pol=pole(sys)
sys=ss(A,B,C,0)

figure(1)
step(sys)
```



مقایسه رفتار دو سیستم به ورودی یکسان پله: مشاهده میشود که نمودار سیستم غیرخطی نسبت به خطی سازی شده شیب تندتری دارد. یعنی پاسخ سیستم غیرخطی نسبت به سیستم خطی سازی شده سریعتر است. درواقع مدل خطی تقریبی از سیستم غیرخطی است.

- بررسی پایداری سیستم خطی سازی شده حلقه باز :

بر مبنای تعاریف پایداری(لیاپانوف، مجانبی، total و BIBO) تابع تبدیل و مقادیر ویژه سیستم را بدست می آوریم و پایداری گزارش می دهیم :



براساس تابع تبدیل به دست آمده $(C \times (SI - A)^{-1} \times B)$ مشاهده میکنیم که چهار قطب در نقطه $s = 0$ داریم و مقادیر ویژه سیستم نیز برابر $\lambda_{1.2.3.4} = 0$ هستند. حال با این اطلاعات، انواع پایداری را بررسی میکنیم:

-پایداری لیاپانوف : داریم؛ زیرا شرط این نوع پایداری، مقادیر ویژه غیرمثبت است و در سیستم ما نیز این شرط برقرار است.

-پایداری مجانبی : نداریم؛ زیرا شرط این نوع پایداری، مقادیر ویژه منفی است و در سیستم ما با توجه به موارد بالا شرط این نوع پایداری برقرار نمی شود.

-پایداری BIBO : نداریم؛ زیرا شرط لازم برای برقراری این نوع پایداری این است که یا سیستم دارای قطب های منفی باشد و یا اگر سیستم دارای قطب در صفر می باشد، این قطب مکرر نباشد. اما با توجه به وجود چهار قطب در $s = 0$ شرط این پایداری برقرار نمی شود.

-پایداری T(Total) : نداریم؛ زیرا شرط برقراری این نوع پایداری، برقراری پایداری مجانبی و BIBO است که در سیستم بر مبنای موارد گفته شده، سیستم پایدار نیست.

- بررسی کنترل پذیری و رویت پذیری سیستم :

طبق فضای حالت خطی سازی شده، با تشکیل ماتریس های φ_o و φ_c کنترل پذیری و رویت پذیری سیستم را بررسی می کنیم :

Editor - C:\Users\Alireza\Desktop\daneshgah\term 7\پروژه\سیستم های کنترل مدرن\m file\seven_controllability_and_observebility.m

seven_controllability_and_observebility.m | control_project.m

```
clear
clc

A = [0 1 0 0;
     0 0 -6.9819 0;
     0 0 0 1;
     0 0 0 0];
B = [0; 0; 0; 1];
C = [1 0 0 0];
controllability_rank = rank( ctrb( A , B ) ) ;
observability_rank = rank( obsv( A , C ) ) ;
controllability_rank
observability_rank
ctrb( A , B )
obsv( A , C )
```

Command Window

```
controllability_rank =

     4


observability_rank =

     4


ans =

         0         0         0   -6.9819
         0         0   -6.9819         0
         0    1.0000         0         0
    1.0000         0         0         0


ans =

    1.0000         0         0         0
         0    1.0000         0         0
         0         0   -6.9819         0
         0         0         0   -6.9819
```

مشاهده می شود که هر دو ماتریس φ_o و φ_c فول رنک می باشند و سیستم خطی کنترل پذیر و رویت پذیر می باشد.

- طراحی فیدبک حالت به صورت $u = -kx + Nr$:

برای طراحی فیدبک حالت سیستم کنترلی ابتدا به تعیین ریشه های فیدبک حالت می پردازیم که به صورت زیر است و داریم :

Pole of state feed back = [-5 , -4 , -3 , -2]

حال بر اساس بدست آوردن k به مقدار عددی N می پردازیم :

$$N = (G_d(0))^{-1} = (-C \times (A - BK)^{-1} \times B)^{-1}$$

Editor - C:\Users\Alireza\Desktop\daneshgah\term 7\پروژه\سیستم های کنترل مدرن\m file\state_feedback.m

control_project.m | state_feedback.m

```
clear
clc
A = [0 1 0 0;
     0 0 -6.9819 0;
     0 0 0 1;
     0 0 0 0];
B = [0; 0; 0; 1];
C = [1 0 0 0];
state_feedback_pole = [ -5 -4 -3 -2 ] ;
k = acker ( A , B , state_feedback_pole ) ;
N = - inv ( C * inv ( A - B * k ) * B ) ;
k
N
```

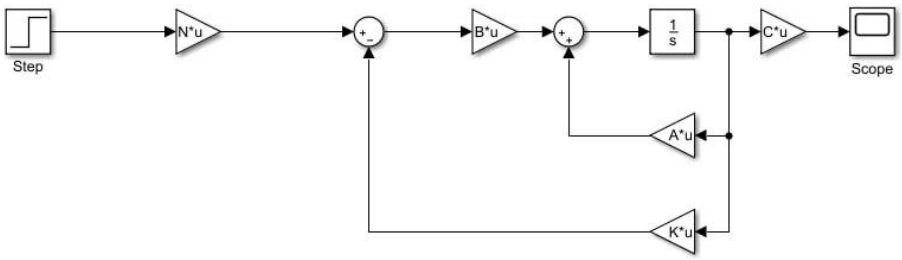
Command Window

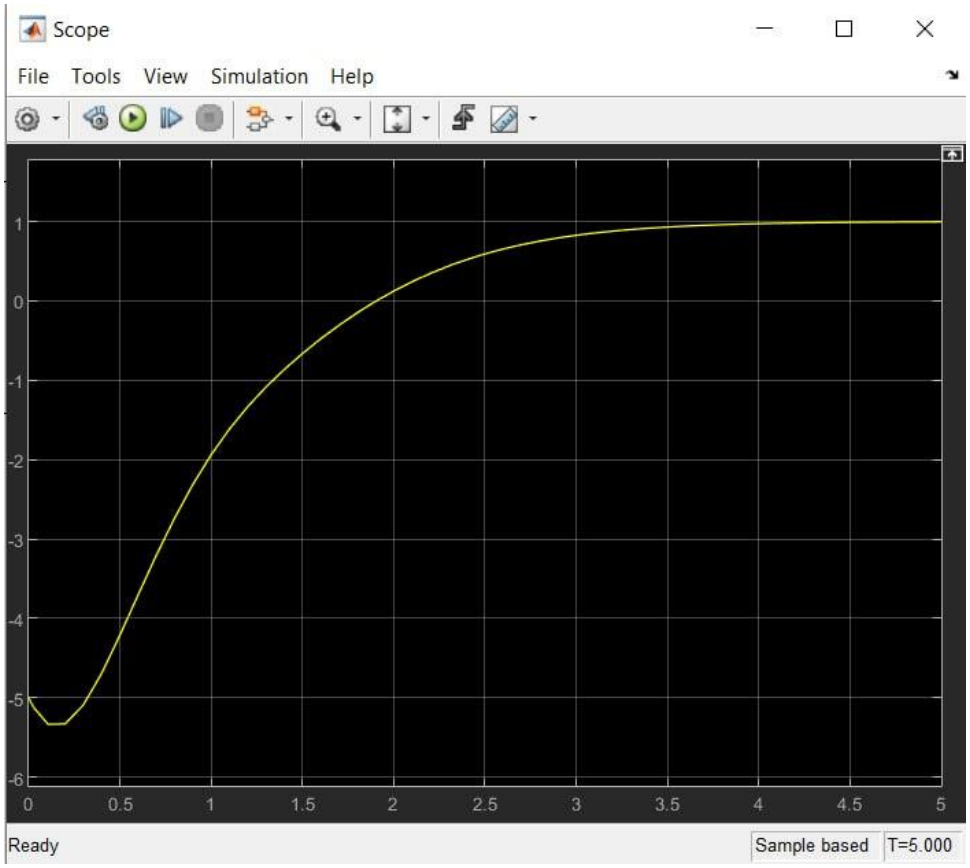
```
k =

  -17.1873  -22.0570   71.0000   14.0000


N =

  -17.1873
```

- شبیه سازی رفتار فیدبک حالت روی سیستم خطی سازی شده :

با توجه به قسمت قبل که فیدبک حالت مناسب برای سیستم را طراحی کردیم، حال به شبیه سازی آن در محیط Simulink می پردازیم :

(شرایط اولیه سیستم برابر 5- لحاظ شده است و صفر نیست)





- شبیه سازی فیدبک حالت روی سیستم غیر خطی :

در این قسمت بر اساس کد قسمت سیستم غیر خطی و با استفاده از ضرایب بدست آمده برای فیدبک حالت (k , N) به شبیه سازی فیدبک حالت بر روی سیستم غیر خطی می پردازیم : (ریشه های فیدبک حالت بر روی [-5 , -4 , -3 , -2] قرار دارد)

Editor - C:\Users\Alireza\Desktop\daneshgah\term 7\پروژه سیستم های کنترل مدرن\m file\state_feedback_for_nonlinear_system.m

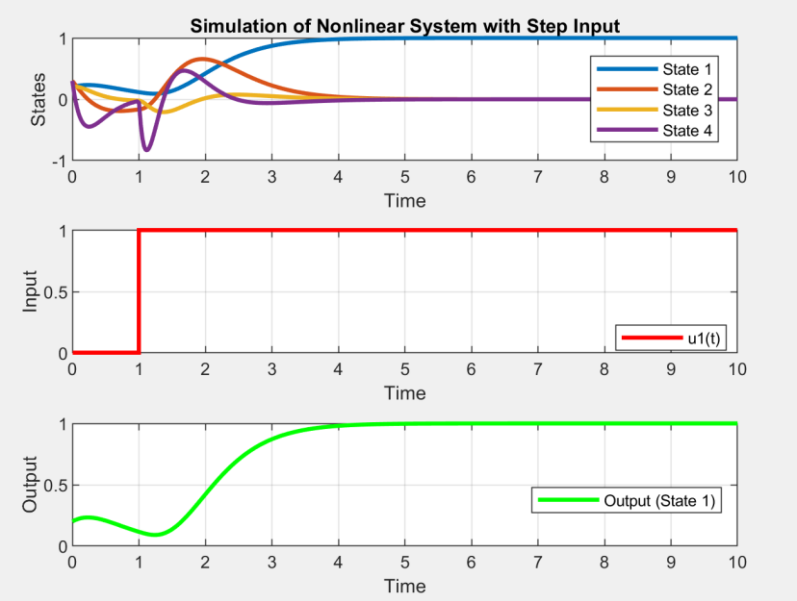
control_project.m | eleven_full_order_observer.m | stability.m | state_feedback_integrator.m | eight_state_feedback.m | state_feedback_for_nonlinear_system.m

```
m = 0.11;
g = 9.8;
j = 9.99*10^(-6);
R = 0.015;
d = 0.03;
l = 1;
u1 = @(t) 1 * (t >= 1)  ;
ode_function = @(t, x) [
    x(2);
   (m*x(1)*(x(4)^2)-m*g*sin(x(3)))/((j/R^2)+m);
    x(4);
    -k*x+N*u1(t) ;
];
initial_conditions = [0.2; 0.3; 0.2; 0.3];
t_span =  linspace(0, 10, 400);
[t, x] = ode45(ode_function, t_span, initial_conditions);
y = x(:, 1);

figure;
```

Editor - C:\Users\Alireza\Desktop\daneshgah\term 7\پروژه\سیستم های کنترل مدرن\m file\state_feedback_for_nonlinear_system.m

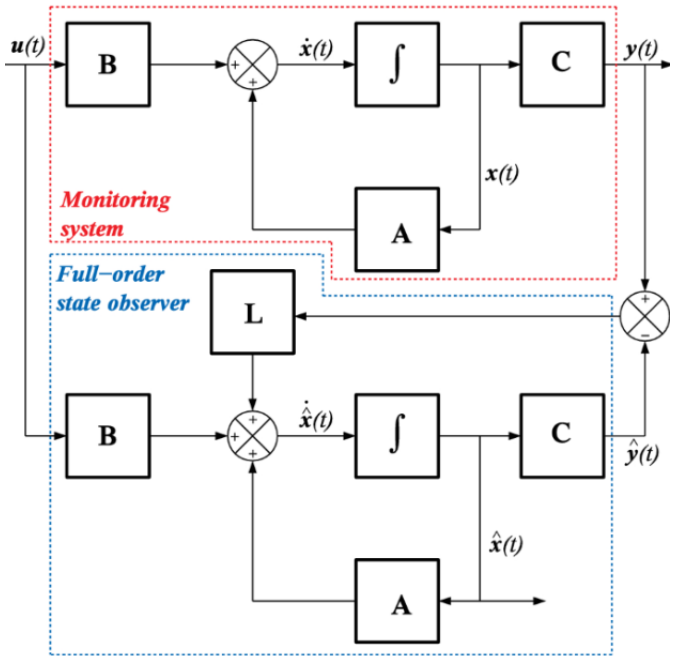
control_project.m | eleven_full_order_observer.m | stability.m | state_feedback_integrator.m | eight_state_feedback.m | state_feedback_for_nonlinear_system.m

```
subplot(3, 1, 1);
plot(t, x(:, 1), 'LineWidth', 2, 'DisplayName', 'State 1');
hold on;
plot(t, x(:, 2), 'LineWidth', 2, 'DisplayName', 'State 2');
hold on;
plot(t, x(:, 3), 'LineWidth', 2, 'DisplayName', 'State 3');
hold on;
plot(t, x(:, 4), 'LineWidth', 2, 'DisplayName', 'State 4');
xlabel('Time');
ylabel('States');
title('Simulation of Nonlinear System with Step Input');
legend('Location', 'Best');
grid on;

subplot(3, 1, 2);
stairs(t, u1(t), 'r', 'LineWidth', 2, 'DisplayName', 'u1(t)');
hold on;
xlabel('Time');
ylabel('Input');
```



- طراحی رویتگر full order :

برای طراحی رویتگر full order که به صورت بلوک دیاگرام زیر می باشد ابتدا باید رویت پذیری (که در مراحل قبل مشاهده شد که سیستم رویت پذیر است) و سپس قطب های رویتگر خود را مشخص کنیم :



Pole of full order observer = [-1 -2 -3 -4]

Editor - C:\Users\Alireza\Desktop\daneshgah\term 7\پروژه\سیستم های کنترل مدرن\m file\eleven_full_order_observer.m

control_project.m | state_feedback.m | eleven_full_order_observer.m

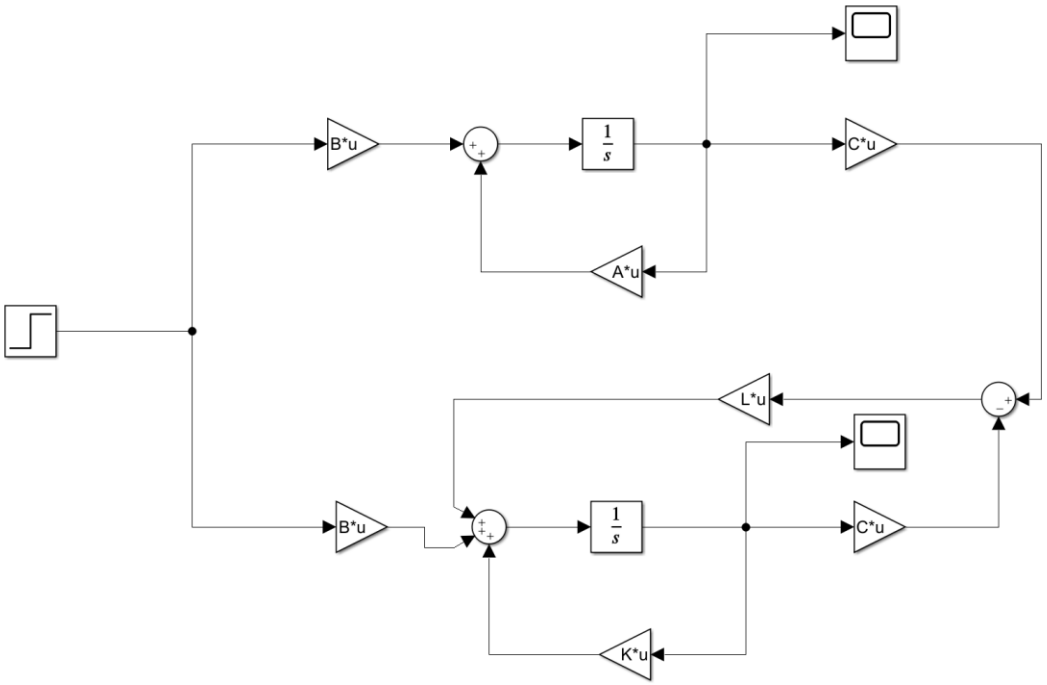
```
clear
clc
A = [0 1 0 0;
     0 0 -6.9819 0;
     0 0 0 1;
     0 0 0 0];
B = [0; 0; 0; 1];
C = [1 0 0 0];
observer_pole = [ -1 -2 -3 -4 ] ;
L_transpose = acker ( A' , C' , observer_pole ) ;
L = L_transpose'
```

Command Window

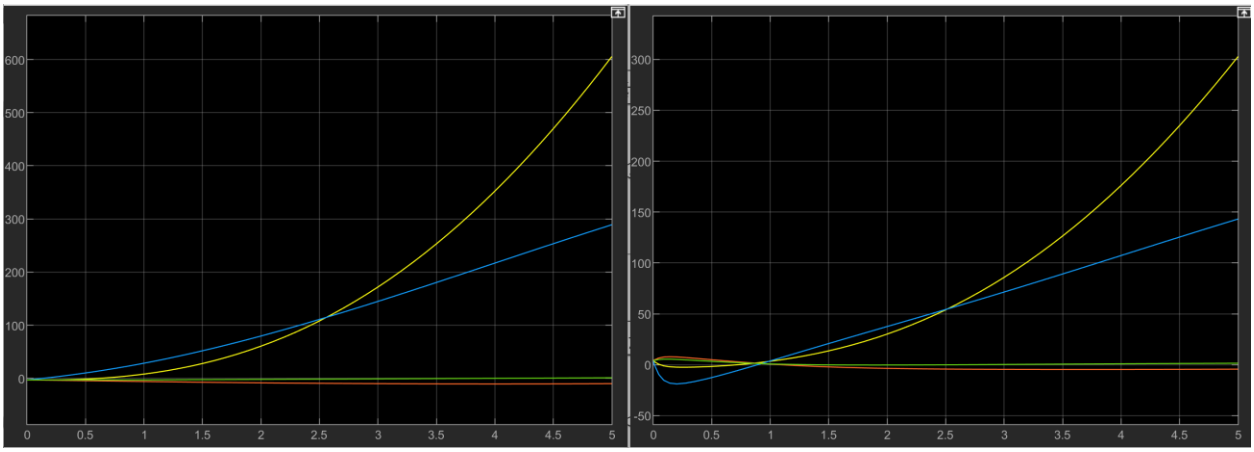
```
L =

   10.0000
   35.0000
   -7.1614
   -3.4375
```

– شبیه سازی رویتگر برای سیستم حلقه باز مقایسه حالت های سیستم خطی سازی شده و حالت های تخمینی :

بر اساس محاسبات و بلوک دیاگرام قسمت قبلی برای طراحی رویتگر full order، در این بخش به شبیه سازی و مقایسه متغیر های سیستم و تخمین متغیر های حالت توسط رویتگر در محیط simulink می پردازیم :
(شرایط اولیه سیستم برابر 3- و شرایط اولیه رویتگر برابر 5- تنظیم می شود.)



بر این اساس سیستم بالا، سیستم اصلی و سیستم پایین رویتگر می باشد که در نهایت با تفاوت در شرایط اولیه، خروجی دو اسکوپ مشاهده می کنیم :



نمودار سمت راست تخمینی از متغیر های حالت سیستم در رویتگر و نمودار سمت چپ متغیر های حالت اصلی سیستم می باشد. مشاهده می شود پس از چند ثانیه متغیر های تخمینی بر متغیر های اصلی منطبق می شوند.

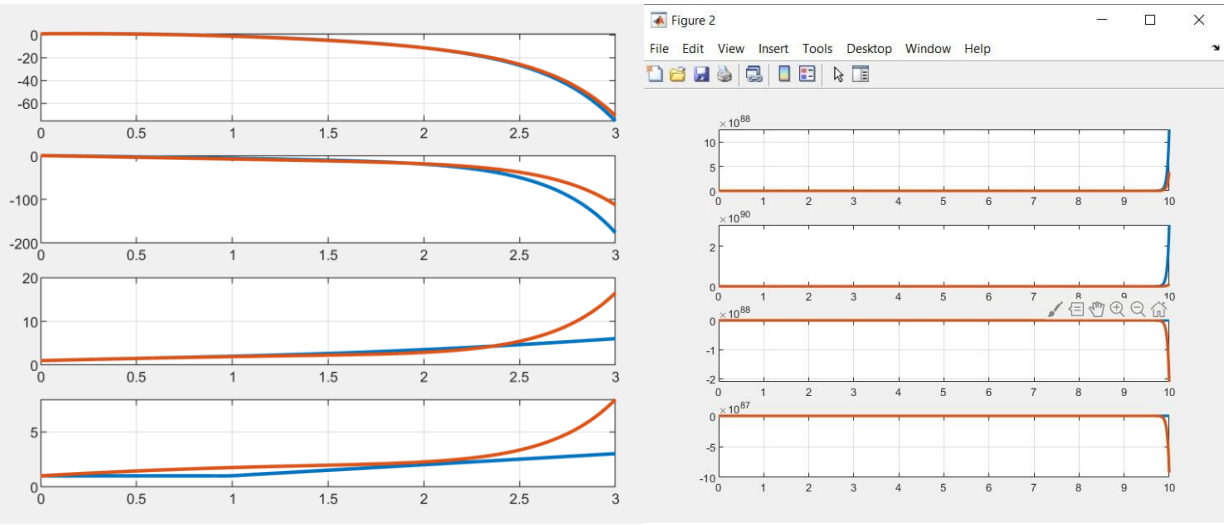
– شبیه سازی رویتگر برای سیستم حلقه باز و مقایسه متغیر های حالت سیستم غیر خطی با متغیر های تخمینی :

در این بخش بر اساس پارامتر رویتگر که در قسمت قبل (L) بدست آوردیم به شبیه سازی آن بر روی سیستم غیر خطی می پردازیم :

```
clc
clear all;

m = 0.11;
g = 9.8;
j = 9.99*10^(-6);
R = 0.015;
d = 0.03;
l = 1;
N = -17.1874;
K =[ -17.1874  -22.0571   71.0000   14.0000];

u1 = @(t) 1 * (t >= 1)  ;

ode_function = @(t, x) [
    x(2);
    ((m*x(1)*(x(4)^2)-m*g*sin(x(3)))/((j/R^2)+m));
    x(4);
    u1(t);
];

initial_conditions = [1; 1; 1; 1];

t_span =  linspace(0, 3, 4000);

```

```
[t, x] = ode45(ode_function, t_span, initial_conditions);

y = x(:, 1);

% observer
A = [0 1 0 0;
     0 0 -6.9819 0;
     0 0 0 1;
     0 0 0 0];
B = [0; 0; 0; 1];
C = [1 0 0 0];
observer_pole = [ -1 -2 -4 -3 ] ;
L_transpose = acker ( A' , C' , observer_pole ) ;
L = L_transpose' ;

Anew = A-(L*C) ;
Bnew = [ B , L ] ;
Cnew = C ;

%Initial conditions for observer

initial_conditions_observer = [1; 1; 1; 1] ;

observer = ss(Anew , Bnew , Cnew , 0);
u = ones(size(t));
```

```
u_y = [u , y];

[y_observer, t , x_observer] = lsim(observer, u_y, t_span, initial_conditions_observer);
%plots

figure;
subplot(4, 1, 1);
plot(t, x(:, 1), 'LineWidth', 2, 'DisplayName', 'State 1');
hold on;
plot(t, x_observer(:, 1), 'LineWidth', 2, 'DisplayName', 'State observer');
hold on;
grid on;

subplot(4, 1, 2);
plot(t, x(:, 2), 'LineWidth', 2, 'DisplayName', 'State 2');
hold on;
plot(t, x_observer(:, 2), 'LineWidth', 2, 'DisplayName', 'State observer');
hold on;

subplot(4, 1, 3);
plot(t, x(:, 3), 'LineWidth', 2, 'DisplayName', 'State 3');
hold on;
plot(t, x_observer(:, 3), 'LineWidth', 2, 'DisplayName', 'State observer');
hold on;
grid on;
```



مشاهده می شود که رویتگر با تقریب خوبی می تواند متغیر های حالت را تخمین بزند.

-شبیه سازی رویتگر + کنترلر برای سیستم خطی سازی شده :

در این بخش به تحلیل سیستم در حالت، رویتگر + فیدبک حالت می پردازیم و با استفاده از داده های دو مورد قبل (فیدبک حالت و رویتگر) به شبیه سازی آن در محیط Simulink می پردازیم :

Editor - C:\Users\Alireza\Desktop\daneshgah\term 7\پروژه\سیستم های کنترل مدرن\m file\eight_state_feedback.m

control_project.m | eleven_full_order_observer.m | stability.m | Untitled | eight_state_feedback.m

```
A = [0 1 0 0;
     0 0 -6.9819 0;
     0 0 0 1;
     0 0 0 0];
B = [0; 0; 0; 1];
C = [1 0 0 0];
state_feedback_pole = [ -5 -6 -7 -8 ] ;
k = acker ( A , B , state_feedback_pole ) ;
N = inv ( -C * inv ( A - B * k ) * B ) ;
k
N
```

Command Window

```
>> eight_state_feedback

k =

 -240.6222 -152.6805  251.0000   26.0000


N =

 -240.6222
```

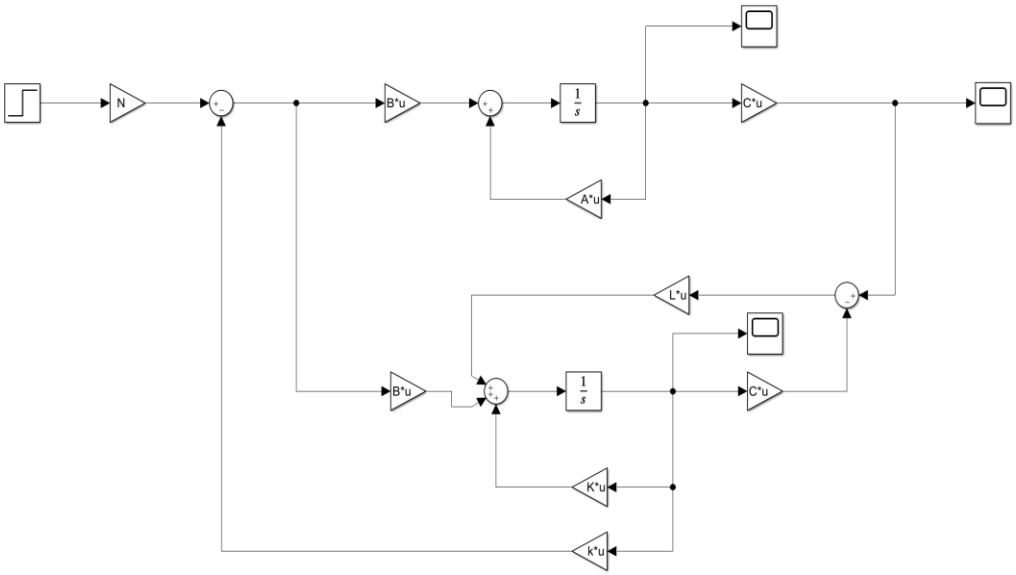
Editor - C:\Users\Alireza\Desktop\daneshgah\term 7\پروژه\سیستم های کنترل مدرن\m file\eleven_full_order_observer.m

control_project.m | eleven_full_order_observer.m | stability.m | Untitled | eight_state_feedback.m

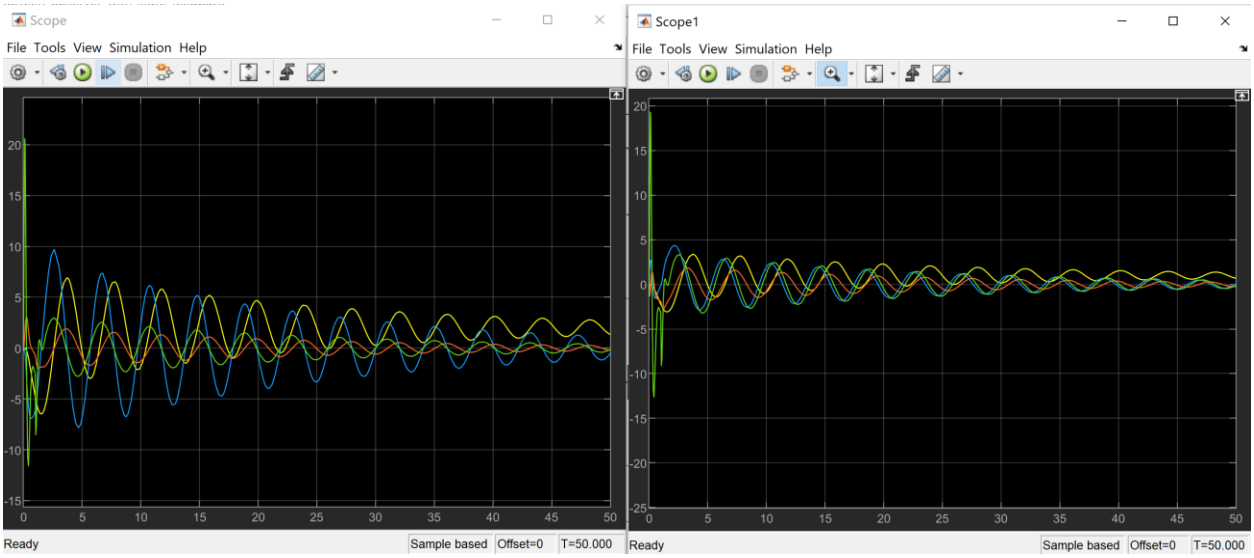
```
A = [0 1 0 0;
     0 0 -6.9819 0;
     0 0 0 1;
     0 0 0 0];
B = [0; 0; 0; 1];
C = [1 0 0 0];
observer_pole = [ -1 -2 -3 -4 ] ;
L_transpose = acker ( A' , C' , observer_pole ) ;
L = L_transpose'
```

Command Window

```
 -240.6222

>> eleven_full_order_observer

L =

   10.0000
   35.0000
   -7.1614
   -3.4375
```



شماتیک کلی رویتگر با فیدبک حالت



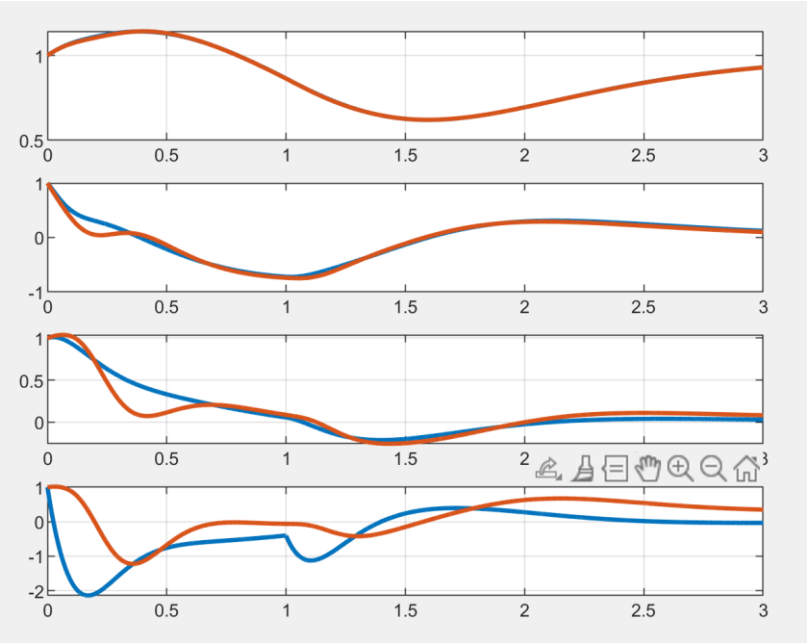
- شبیه سازی رویتگر + کنترلر برای سیستم غیر خطی :

بر اساس شبیه سازی های مربوط به سیستم غیر خطی، با الحاق رویتگر و فیدبک حالت به سیستم غیر خطی داریم :

```
clc
clear all;

m = 0.11;
g = 9.8;
j = 9.99*10^(-6);
R = 0.015;
d = 0.03;
l = 1;
N = -17.1874;
K =[ -17.1874  -22.0571   71.0000   14.0000];

u1 = @(t) 1 * (t >= 1)  ;

ode_function = @(t, x) [
    x(2);
    ((m*x(1)*(x(4)^2)-m*g*sin(x(3)))/((j/R^2)+m));
    x(4);
    (-K*x)+N*u1(t);
];

initial_conditions = [1; 1; 1; 1];

t_span =  linspace(0, 3, 4000);

[t, x] = ode45(ode_function, t_span, initial_conditions);

y = x(:, 1);

% observer
A = [0 1 0 0;
     0 0 -6.9819 0;
     0 0 0 1;
     0 0 0 0];
B = [0; 0; 0; 1];
C = [1 0 0 0];
observer_pole = [ -10 -11 -12 -13 ] ;
L_transpose = acker ( A' , C' , observer_pole ) ;
L = L_transpose' ;

Anew = A-(L*C) ;
Bnew = [ B , L ] ;
Cnew = C ;

%Initial conditions for observer

initial_conditions_observer = [1; 1; 1; 1] ;

observer = ss(Anew , Bnew , Cnew , 0);
u = ones(size(t));
```

```
u_y = [u , y];

[y_observer, t , x_observer] = lsim(observer, u_y, t_span, initial_conditions_observer);
%plots

figure;
subplot(4, 1, 1);
plot(t, x(:, 1), 'LineWidth', 2, 'DisplayName', 'State 1');
hold on;
plot(t, x_observer(:, 1), 'LineWidth', 2, 'DisplayName', 'State observer');
hold on;
grid on;

subplot(4, 1, 2);
plot(t, x(:, 2), 'LineWidth', 2, 'DisplayName', 'State 2');
hold on;
plot(t, x_observer(:, 2), 'LineWidth', 2, 'DisplayName', 'State observer');
hold on;

subplot(4, 1, 3);
plot(t, x(:, 3), 'LineWidth', 2, 'DisplayName', 'State 3');
hold on;
plot(t, x_observer(:, 3), 'LineWidth', 2, 'DisplayName', 'State observer');
hold on;
grid on;
```

```
 subplot(4, 1, 4);
 plot(t, x(:, 4), 'LineWidth', 2, 'DisplayName', 'State 4');
 hold on;
 plot(t, x_observer(:, 4), 'LineWidth', 2, 'DisplayName', 'State observer');
 hold on;
 grid on;
```



مشاهده می شود علاوه بر پایداری و دنبال کردن ورودی توسط خروجی، رویتگر نیز تخمین مناسبی از متغیر های حالت سیستم دارد.

- طراحی فیدبک حالت انتگرالی :

ابتدا ریشه های کنترلر انتگرالی خود را تعریف می کنیم و در نهایت با فرمول ها و فضای حالت انتگرالی، ضرایب کنترلر را بدست می آوریم (دقت داریم که ابتدا باید کنترل پذیری نیز باید بررسی کنیم) :

Editor - C:\Users\Alireza\Desktop\daneshgah\term 7\پروژه\سیستم های کنترل مدرن\m file\state_feedback_integrator.m

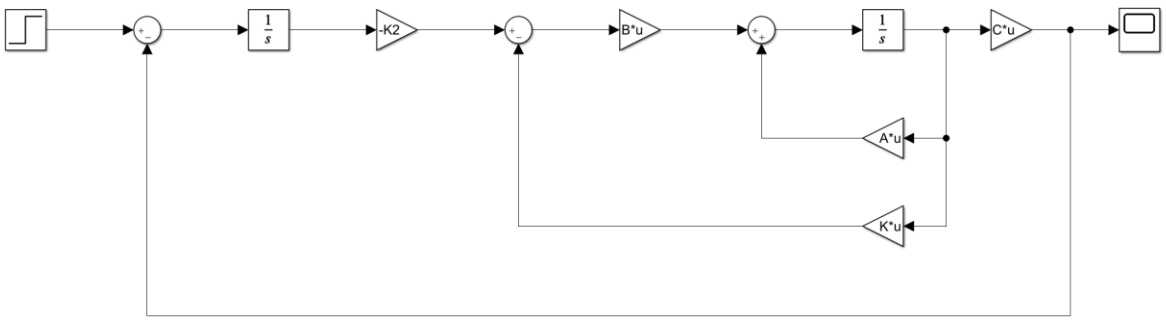
control_project.m | state_feedback.m | eleven_full_order_observer.m | state_feedback_integrator.m

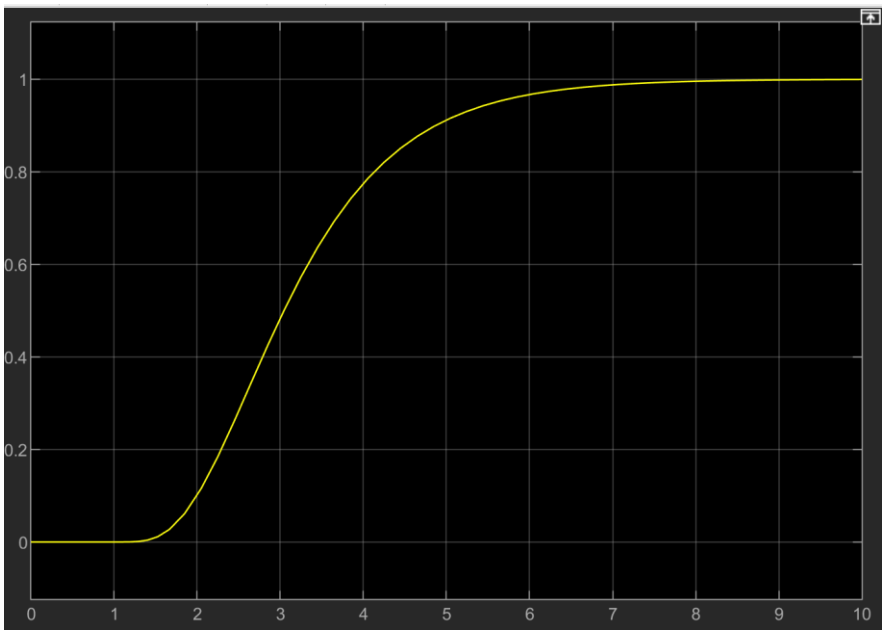
```
clear
clc
A = [0 1 0 0;
     0 0 -6.9819 0;
     0 0 0 1;
     0 0 0 0];
B = [0; 0; 0; 1];
C = [1 0 0 0];

 [ rA , cA ] = size(A) ;
 [ rB , cB ] = size(B) ;
 [ rC , cC ] = size(C) ;

 A_bar = [ A zeros(rA,1); -C zeros( rC , 1 ) ] ;
 B_bar = [ B ; zeros( rC , cB ) ] ;
 Integral_controlability_rank = rank( ctrb ( A_bar , B_bar ) )

 state_feedback_integrators = [ -1 -2 -3 -4 -5 ] ;
 K = acker ( A_bar , B_bar , state_feedback_integrators ) ;
 K1 = K ( 1 : rA )
 K2 = K ( rA + 1 : rA + rC )

```

Command Window

```
Integral_controlability_rank =

     5


K1 =

  -39.2443  -32.2262   85.0000   15.0000


K2 =

   17.1873
```

برای اطمینان از ضرایب بدست آمده، فیدبک انتگرالی در محیط Simulink شبیه سازی می کنیم و داریم :





مشاهده می شود که ضرایب بدست آمده درست هستند و در نهایت سیستم پایدار و ورودی مرجع را دنبال میکند.

- شبیه سازی رویتگر + کنترلر انتگرالی برای سیستم خطی سازی شده :

حال در این بخش با استفاده از قسمت های قبل به شبیه سازی همزمان رویتگر با کنترلر انتگرالی می پردازیم :

control_project.m | eleven_full_order_observer.m | stability.m | state_feedback_integrator.m

```
1
2 -   A = [0 1 0 0;
3          0 0 -6.9819 0;
4          0 0 0 1;
5          0 0 0 0];
6 -   B = [0; 0; 0; 1];
7 -   C = [1 0 0 0];
8 -   observer_pole = [ -1 -2 -3 -4 ] ;
9 -   L_transpose = acker ( A' , C' , observer_pole ) ;
10 -  L = L_transpose'
```

```
>> eleven_full_order_observer

L =

   10.0000
   35.0000
   -7.1614
   -3.4375
```

تنظیمات رویتگر

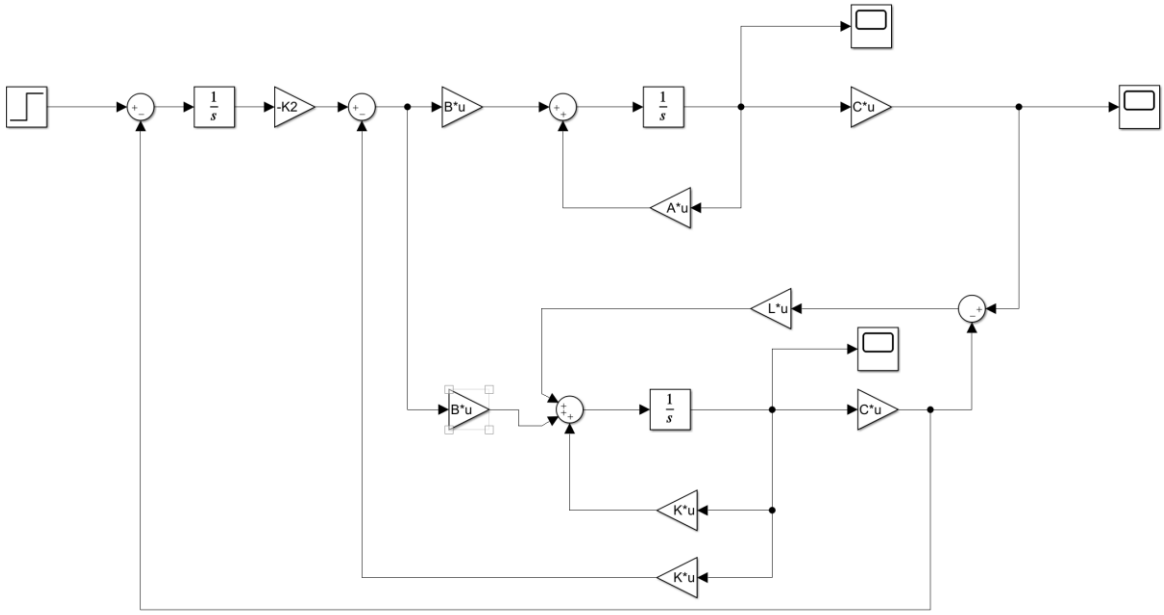
control_project.m | eleven_full_order_observer.m | stability.m | state_feedback_integrator.m

```
clear
clc
A = [0 1 0 0;
     0 0 -6.9819 0;
     0 0 0 1;
     0 0 0 0];
B = [0; 0; 0; 1];
C = [1 0 0 0];

 [ rA , cA ] = size(A) ;
 [ rB , cB ] = size(B) ;
 [ rC , cC ] = size(C) ;

 A_bar = [ A zeros(rA,1); -C zeros( rC , 1 ) ] ;
 B_bar = [ B ; zeros( rC , cB ) ] ;
 Integral_controlability_rank = rank( ctrb ( A_bar , B_bar ) )

 state_feedback_integrators = [ -10 -11 -12 -13 -14 ] ;
 K = acker ( A_bar , B_bar , state_feedback_integrators ) ;
 K1 = K ( 1 : rA )
 K2 = K ( rA + 1 : rA + rC )

```

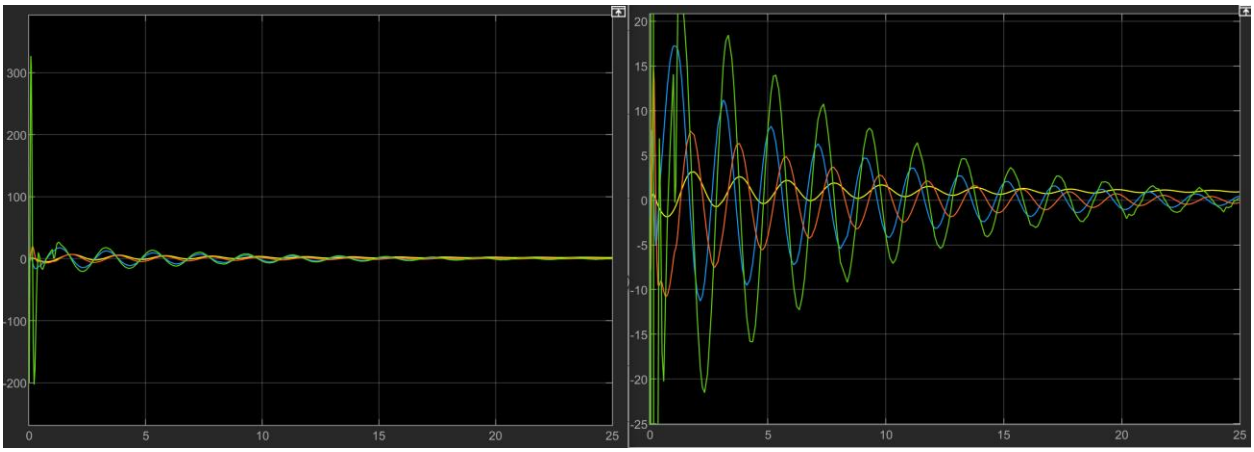
Command Window

```
Integral_controlability_rank =

     5


K1 =

   1.0e+04 *

   -1.4541   -0.2449    0.1435    0.0060


K2 =

   3.4409e+04
```

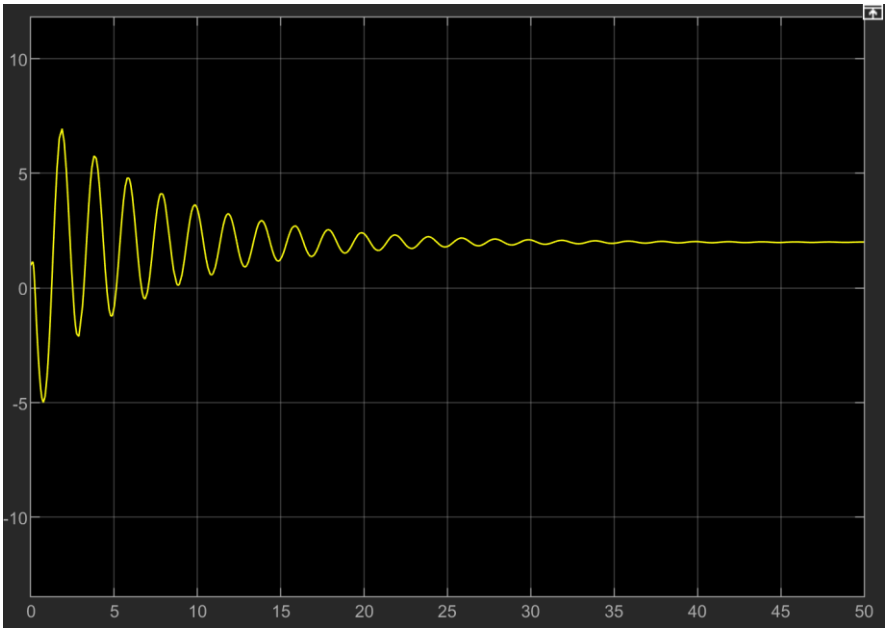
تنظیمات کنترلر انتگرالی



شماتیک کلی سیستم



متغیر های حالت (سمت چپ متغیرهای رویتگر، سمت راست متغیر های اصلی سیستم)



خروجی سیستم

مشاهده می شود که علاوه بر پایداری و دنبال ورودی توسط خروجی (بر اساس خروجی های بالا)، رویتگر نیز متغیر های حالت سیستم را با تقریب خوبی تخمین می زند.