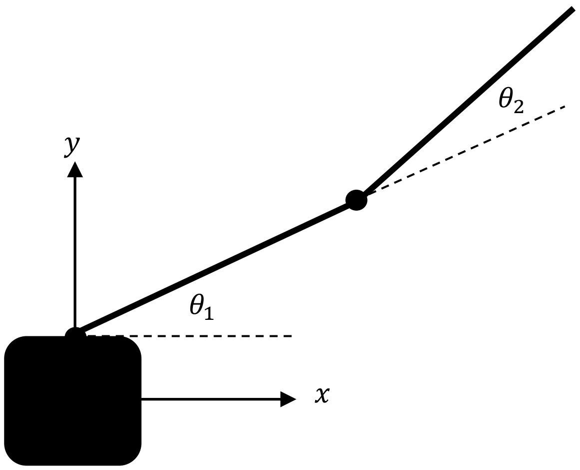
Modeling of a 2 DOF robot arm

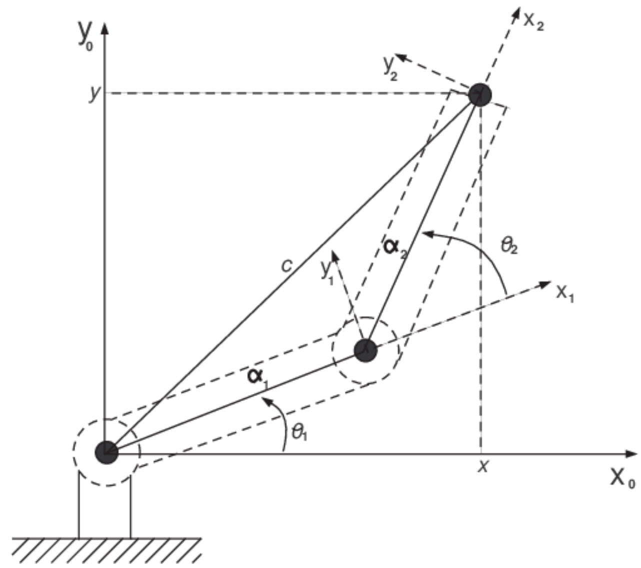
Marzieh Ghayour Najafabadi - مرضیه غیور نجف آبادی

98242112

استاد درس: دکتر آرش صادق زاده

Modern Control 401-402





- سیستم مورد نظر، یک بازوی دو لینکه می باشد، که ورودی سیستم ما، نیرو های وارده به هریک از بازوها می باشد و خروجی مکان نهایی آن هارا تعیین میکند.

1. معادلات دینامیکی سیستم

فرض میگیریم که جرم و طول هر بازو یکسان است و برابرند با:

- $\mathbf{m_1 = m_2 = m}$
- $\mathbf{l_1 = l_2 = l}$

```
% Parameters---
l = 1;
m = 1;
```

$$\mathbf{x_1 = l\cos\theta_1} \qquad \mathbf{\dot{x}_1 = -l\sin\theta_1\dot{\theta}_1}$$

$$\mathbf{y_1 = l\sin\theta_1} \qquad \mathbf{\dot{y}_1 = l\cos\theta_1\dot{\theta}_1}$$

$$\mathbf{x_2 = l\cos(\theta_1+\theta_2) + l\cos\theta_1} \qquad \mathbf{\dot{x}_2 = -l\,[(\sin\theta_1\dot{\theta}_1) + (\sin(\theta_1+\theta_2)\,(\dot{\theta}_1+\dot{\theta}_2)]}$$

$$\mathbf{y_2 = l\sin(\theta_1+\theta_2) + l\sin\theta_1} \qquad \mathbf{\dot{y}_2 = l\,[(\cos\theta_1\dot{\theta}_1) + (\cos(\theta_1+\theta_2)\,(\dot{\theta}_1+\dot{\theta}_2)]}$$

از رابطه لاگرانژین برای به دست آوردن دو ترم نیرو استفاده میکنیم:

- $\mathbf{KE = \frac{1}{2}mv_1{}^2 + \frac{1}{2}mv_2{}^2 + \frac{1}{2}I\omega^2{}_1 + \frac{1}{2}I\omega^2{}_2 = ml^2[(3.5+\cos\theta_2)\dot{\theta}_1{}^2 + \frac{3}{2}\dot{\theta}_2{}^2 + (3+\cos\theta_2)\dot{\theta}_1\dot{\theta}_2]}$

- $\mathbf{PE = mgh_1 + mgh_2 = mgl[\cos(\theta_1+\theta_2) + 2\cos\theta_1]}$

$$\bullet \quad \mathbf{L = KE - PE =}$$

$$\mathbf{ml^2[(3.5+\cos\theta_2)\dot{\theta}_1{}^2+\frac{3}{2}\dot{\theta}_2{}^2+(3+\cos\theta_2)\dot{\theta}_1\dot{\theta}_2]-\ mgl[\cos(\theta_1+\theta_2)+2\cos\theta_1]}$$

$$\mathbf{T_1 = \frac{d}{dt}\frac{\partial L}{\partial \theta_1} - \frac{\partial L}{\partial \dot{\theta}_1}}$$

$$\mathbf{T_2 = \frac{d}{dt}\frac{\partial L}{\partial \theta_2} - \frac{\partial L}{\partial \dot{\theta}_2}}$$

$$\mathbf{F = H(\ddot{\theta}) + C(\dot{\theta},\theta) + g(\theta)}$$

$$\mathbf{\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = \begin{bmatrix} ml^2(7+2\cos\theta_1) & ml^2(3+\cos\theta_2) \\ ml^2(3+\cos\theta_2) & 3ml^2 \end{bmatrix}\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} ml^2(-\sin\theta_2)\dot{\theta}_2 & ml^2(-\sin\theta_2)\dot{\theta}_2 \\ ml^2(\sin\theta_2)\dot{\theta}_1 & 0 \end{bmatrix}\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} + \begin{bmatrix} -mlg(\cos(\theta_1+\theta_2)+\cos\theta_1) \\ mlg\cos(\theta_1+\theta_2) \end{bmatrix}}$$

معادلات به دست آمده، یک سیستم غیرخطی چند ورودی چند خروجی(MIMO) را توصیف میکند، برای تحلیل این سیستم باید آن را خطی سازی کنیم.

2. معادلات فضای حالت

متغیرهای حالت سیستم به شکل زیر تعریف شده اند:

$$\mathbf{x_1 = \theta_1} \qquad \mathbf{\dot{x}_1 = \dot{\theta}_1}$$

$$\mathbf{x_2 = \theta_2} \qquad \mathbf{\dot{x}_2 = \dot{\theta}_2}$$

$$\mathbf{x_3 = \dot{\theta}_1} \qquad \mathbf{\dot{x}_3 = \ddot{\theta}_1}$$

$$\mathbf{x_4 = \dot{\theta}_2} \qquad \mathbf{\dot{x}_4 = \ddot{\theta}_2}$$

ابتدا باید نقطه کار سیستم را پیدا کنیم تا بتوانیم با استفاده از ماتریس ژاکوبین، معادلات را هول نقطه کار، نقطه خطی سازی کنیم:

$$\mathbf{T_1 = 0\,, T_2 = 0 \ \rightarrow}$$

$$\mathbf{\begin{cases} x_1 = \frac{\pi}{2}, \\ x_2 = \frac{\pi}{2}, \\ x_3 = 0, \\ x_4 = 0 \end{cases}}$$

ماتریس ژاکوبین را برای A,B,C,D تشکیل میدهیم و مقادیر نقطه کار را در آن قرار میدهیم:

$$J_u = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \\ \frac{\partial f_3}{\partial u_1} & \frac{\partial f_3}{\partial u_2} \\ \frac{\partial f_4}{\partial u_1} & \frac{\partial f_4}{\partial u_2} \end{bmatrix} \qquad J_x = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \frac{\partial f_1}{\partial x_4} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} & \frac{\partial f_2}{\partial x_4} \\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_3} & \frac{\partial f_3}{\partial x_4} \\ \frac{\partial f_4}{\partial x_1} & \frac{\partial f_4}{\partial x_2} & \frac{\partial f_4}{\partial x_3} & \frac{\partial f_4}{\partial x_4} \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -0.4568 & -0.6196 & 0 & 0 \\ 0.2485 & -6.6174 & 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0.7870 & -0.0426 \\ 0.0426 & 0.1349 \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad D = 0$$

```
% State space -----------------------
A = [0 0 1 0;0 0 0 1;-0.4568 -0.6196 0 0;0.2485 -6.6174 0 0];
B = [0 0;0 0;0.7870 -0.0426;0.0426 0.1349];
C = [1 0 0 0;0 1 0 0;0 0 1 0;0 0 0 1];
robotarm = ss(A,B,C,0);
```

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = A \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + B \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} \qquad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = C \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

3. نمایش کانونیکال جردن ماتریس های فضای حالت:

```
%Jordan canonical form matrix-------
canon(robotarm,"Modal");
```

```
A =
            x1          x2          x3          x4
 x1  -4.045e-17      0.6942           0           0
 x2     -0.6942  -4.045e-17           0           0
 x3           0           0   8.792e-17       2.568
 x4           0           0      -2.568   8.792e-17
```

```
B =
            u1          u2
 x1  -4.795e-17  -1.876e-18
 x2       1.136    -0.08159
 x3  -1.038e-17  -1.567e-17
 x4    0.008399       0.107
```

```
C =
            x1          x2          x3          x4
 y1      0.9967   4.211e-17     0.05043   5.697e-17
 y2     0.04037   5.688e-18      0.4994    7.82e-17
 y3  -6.836e-17      0.6919  -1.271e-16      0.1295
 y4  -9.911e-18     0.02802  -2.104e-16       1.282
```

4. مقادیر ویژه ماتریس A :

```
%Eigenvalues-----------
eigA_o = eig(A);
```

```
-0.0000 + 0.6942i
-0.0000 - 0.6942i
 0.0000 + 2.5675i
 0.0000 - 2.5675i
```

5. کنترل پذیری:

برای بررسی کنترل پذیری سیستم، ماتریس کنترل پذیری را تشکیل داده و رنک آن را حساب میکنیم. درصورت رنک کامل نبودن، مودهای کنترل پذیر و کنترل ناپذیر سیستم را پیدا کرده و فرم کانونیکال کنترل پذیر آن را میسازیم.

```
%Controllability--------
phi_c = ctrb(A,B);
rank(phi_c);
```

```
phi_c =

         0         0    0.7870   -0.0426         0         0   -0.3859   -0.0641
         0         0    0.0426    0.1349         0         0   -0.0863   -0.9033
    0.7870   -0.0426         0         0   -0.3859   -0.0641         0         0
    0.0426    0.1349         0         0   -0.0863   -0.9033         0         0
```

```
ans =

     4
```

رنک سیستم کامل است بنابراین، مود کنترل ناپذیر ندارد و همه مودها کنترل پذیر هستند.

6. رویت پذیری:

برای بررسی رویت پذیری سیستم، ماتریس رویت پذیری را تشکیل داده و رنک آن را حساب میکنیم. درصورت رنک کامل نبودن، مودهای رویت پذیر و رویت ناپذیر سیستم را پیدا کرده و فرم کانونیکال رویت پذیر آن را میسازیم.

```
% Observability----------
phi_o = obsv(A,C);
rank(phi_o);
```

```
phi_o =

    1.0000         0         0         0
         0    1.0000         0         0
         0         0    1.0000         0
         0         0         0    1.0000
         0         0    1.0000         0
         0         0         0    1.0000
   -0.4568   -0.6196         0         0
    0.2485   -6.6174         0         0
   -0.4568   -0.6196         0         0
    0.2485   -6.6174         0         0
         0         0   -0.4568   -0.6196
         0         0    0.2485   -6.6174
         0         0   -0.4568   -0.6196
         0         0    0.2485   -6.6174
    0.0547    4.3832         0         0
   -1.7579   43.6360         0         0
```

```
ans =

     4
```

رنک سیستم کامل است بنابراین، مود رویت ناپذیر ندارد و همه مودها رویت پذیر هستند.

7. تحقق

7.1 تحقق کانونیکال کنترل پذیری:

```
% Realization-----------------
canon(robotarm,"Companion");
```

```
A =
        x1      x2      x3      x4
  x1     0       0       0  -3.177
  x2     1       0       0       0
  x3     0       1       0  -7.074
  x4     0       0       1       0

B =
        u1      u2
  x1     1  -1.082
  x2     0       0
  x3     0  -2.097
  x4     0       0

C =
          x1        x2        x3        x4
  y1       0     0.787         0   -0.3859
  y2       0    0.0426         0  -0.08633
  y3   0.787         0   -0.3859         0
  y4  0.0426         0  -0.08633         0
```

با استفاده از دستورات متلب تنها میتوان تحقق کانونیکال کنترل پذیری را نمایش داد.

7.2 تحقق کانونیکال رویت پذیری:

```
A =
        x1      x2      x3      x4
  x1     0       1       0       0
  x2     0       0       1       0
  x3     0       0       0       1
  x4 -3.177      0  -7.074       0

B =
          u1        u2        u3        u4
  x1       0         0     0.787    0.0426
  x2   0.787    0.0426         0         0
  x3       0         0   -0.3859  -0.08633
  x4 -0.3859  -0.08633         0         0

C =
        x1      x2      x3      x4
  y1     1       0       0       0
  y2 -1.082      0  -2.097       0
```

```
% Realization---------------------------
canon_cont = canon(robotarm,"Companion");
transpose(canon_cont);
```

با transpose گرفتن از تحقق کنترل پذیری، میتوانیم تحقق رویت پذیری را نیز به دست بیاوریم.

8. تابع تبدیل (transfer function):

ماتریس G(s) ، شامل دو ستون و چهار سطر می باشد.

4 تابع تبدیل به ازای ورودی اول:

```
% Transfer function
tf(robotarm);
```

```
From input 1 to output...
                    0.787 s^2 + 5.181
 1:  ---------------------------------------------------
     s^4 - 9.494e-17 s^3 + 7.074 s^2 + 4.486e-16 s + 3.177

                   0.0426 s^2 + 0.215
 2:  ---------------------------------------------------
     s^4 - 9.494e-17 s^3 + 7.074 s^2 + 4.486e-16 s + 3.177

                   0.787 s^3 + 5.181 s
 3:  ---------------------------------------------------
     s^4 - 9.494e-17 s^3 + 7.074 s^2 + 4.486e-16 s + 3.177

                  0.0426 s^3 + 0.215 s
 4:  ---------------------------------------------------
     s^4 - 9.494e-17 s^3 + 7.074 s^2 + 4.486e-16 s + 3.177
```

4 تابع تبدیل به ازای ورودی دوم:

```
From input 2 to output...
                  -0.0426 s^2 - 0.3655
 1:  ----------------------------------------------------
     s^4 - 9.494e-17 s^3 + 7.074 s^2 + 4.486e-16 s + 3.177

                  0.1349 s^2 + 0.05104
 2:  ----------------------------------------------------
     s^4 - 9.494e-17 s^3 + 7.074 s^2 + 4.486e-16 s + 3.177

                 -0.0426 s^3 - 0.3655 s
 3:  ----------------------------------------------------
     s^4 - 9.494e-17 s^3 + 7.074 s^2 + 4.486e-16 s + 3.177

                 0.1349 s^3 + 0.05104 s
 4:  ----------------------------------------------------
     s^4 - 9.494e-17 s^3 + 7.074 s^2 + 4.486e-16 s + 3.177
```

9. پایداری

9.1. لیاپانوف :

با توجه به بخش 4 (مقادیر ویژه A)، قسمت حقیقی هر 4 مقدار ویژه، صفر می باشد، بنابراین، سیستم به دلیل داشتن بیشتر از یک صفر، پایدار لیاپانوف نمی باشد.

9.2. مجانبی:

با توجه به مقادیر ویژه A، قسما حقیقی هر 4 مقدار، صفر می باشد و به دلیل داشتن مقدار ویژه نامنفی، سیستم پایدار مجانبی نمی باشد.

9.3. پایداری BIBO:

قطب های تابع تبدیل با توجه به اینکه سیستم هم رویت پذیر و هم کنترل پذیر می‌باشد با مقادیر ویژه سیستم یکسان هستند. با توجه به اینکه هر4 تای این مقادیر، قسمت حقیقی صفر دارند، پس سیستم پایدار BIBO هم نمی‌باشد(پایداری مرزی BIBO دارد.)

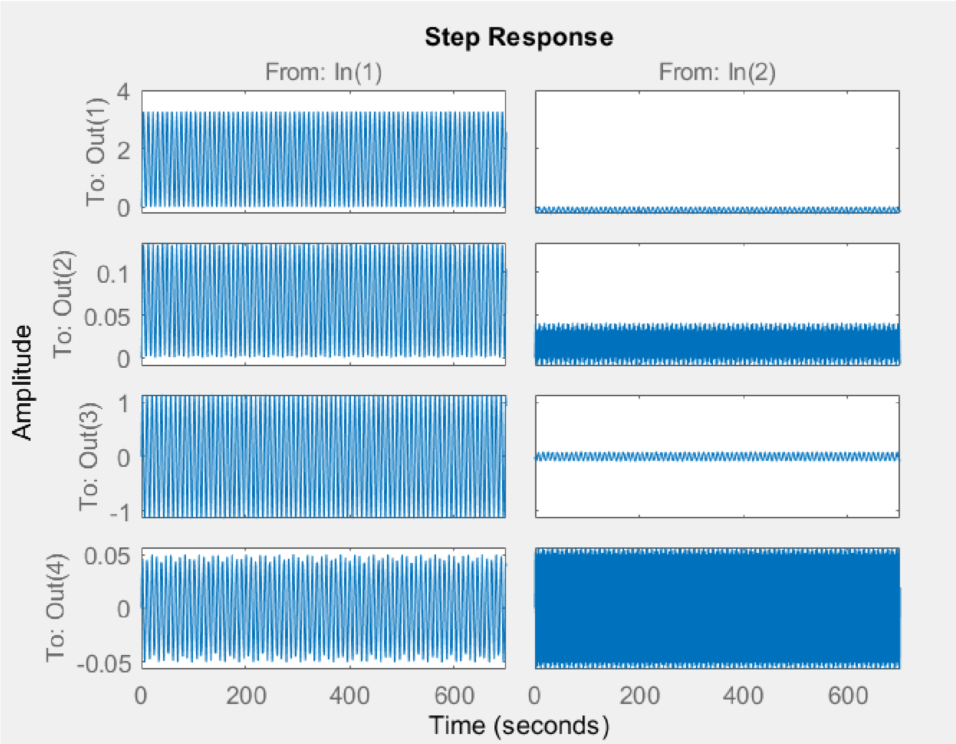
```
%Poles and zeros------------
pole(tf_o);
tzero(tf(robotarm))
```

10. طراحی فیدبک حالت

- **پاسخ پله سیستم در حالت حلقه باز:**

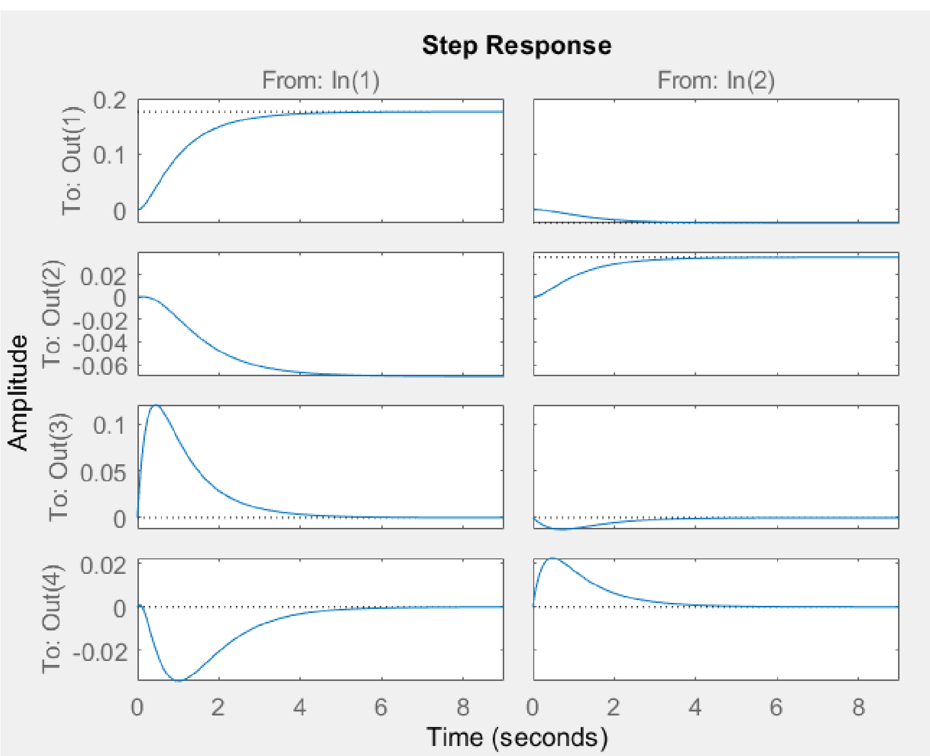
این پاسخ کاملا ناپایدار و نامیرا می باشد.

```
%Step response of the open loop system---
figure(1)
step(robotarm)
```



برای پایدار کردن این سیستم میتوانیم فیدبک حالت بسته و Pole assignment انجام بدهیم و پاسخ های آن ها را بررسی کنیم. یک بار قطب هایی را نزدیک به محور $j\omega$ انتخاب میکنیم و بار دیگر دور از آن. با توجه به اینکه تمامی مودهای این سیستم کنترل پذیر می باشند، میتوان همه آن ها را با فیدبک حالت پایدار کرد.

- **نزدیک به محور$j\omega$:**

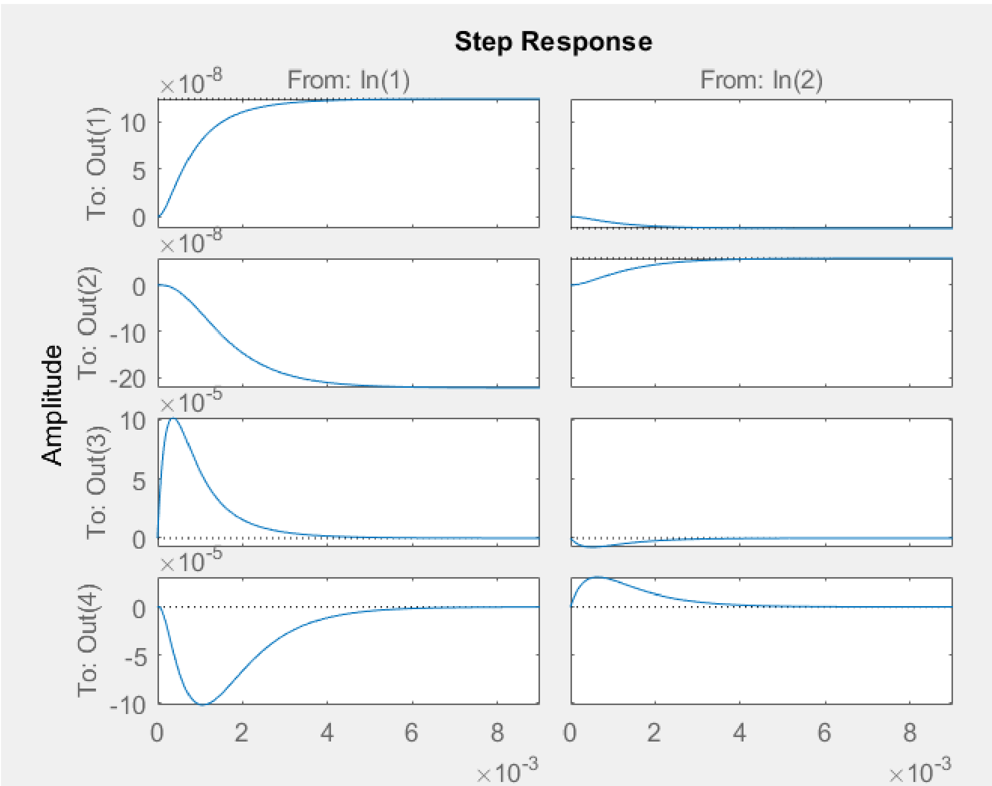



```
%Step response of the closed loop system-
%Pole assignment
p = [-1,-2,-3,-4]; %near to zero
k = place(A,B,p);
Acl = A-B*k;
syscl = ss(Acl,B,C,0);
figure(2)
step(syscl)
```

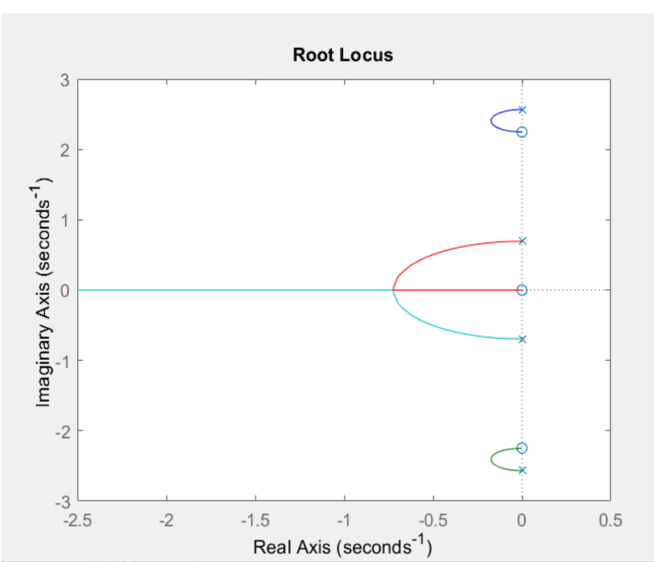
- **دور از محور $j\omega$:**

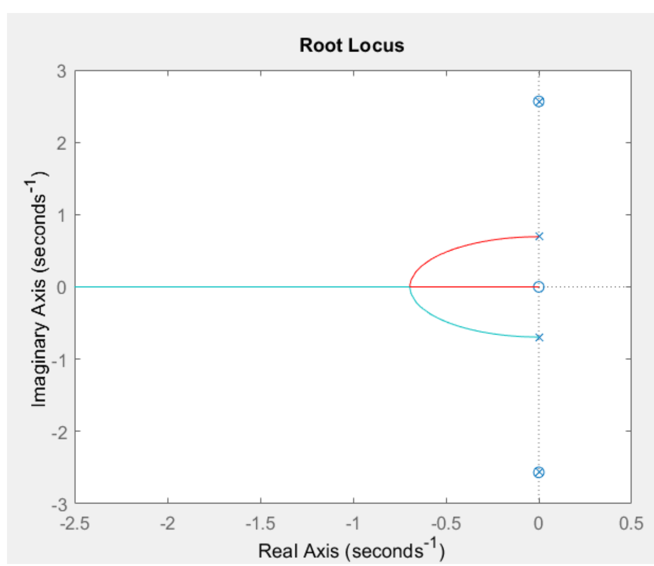
هرچقدر قطب هارا دورتر از محور به سمت منفی بی نهایت قرار دهیم، پاسخ ما سریع تر پایدار میشود.

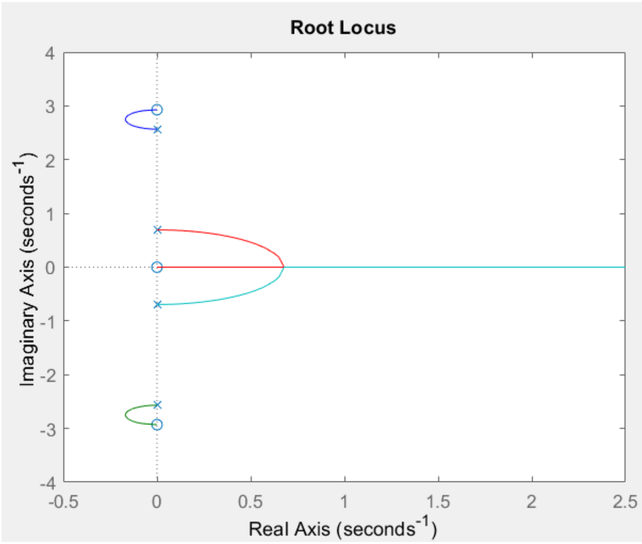
```
p = 1000*[-1,-2,-3,-4]; %far from zero
k = place(A,B,p);
Acl = A-B*k;
syscl = ss(Acl,B,C,0);
figure(3)
step(syscl)
```

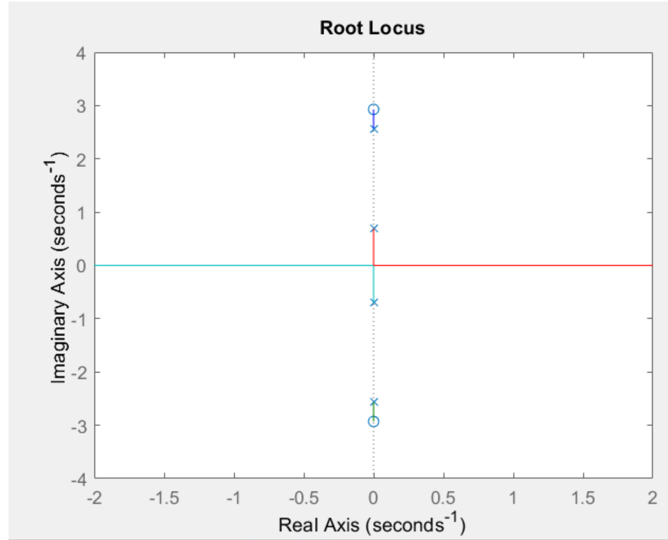


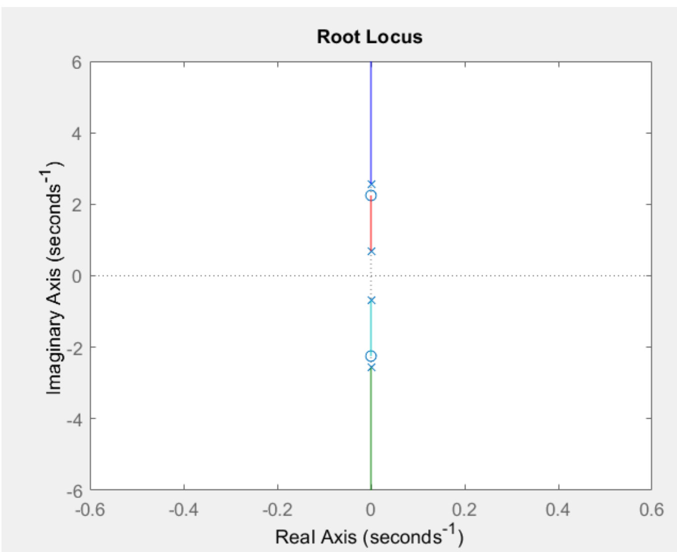
Root Locus .11

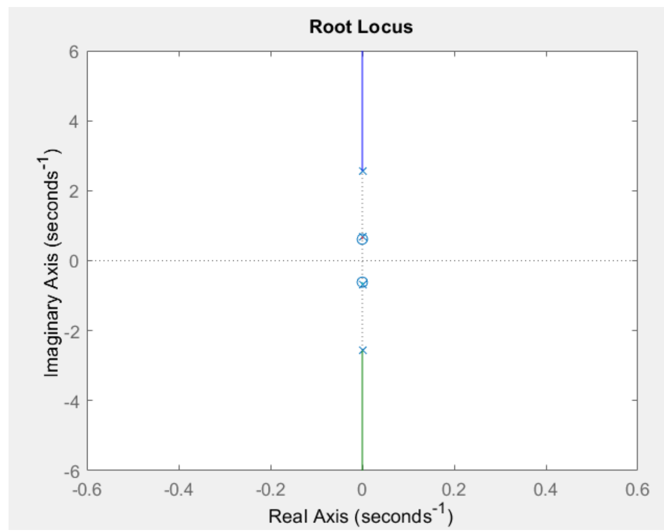


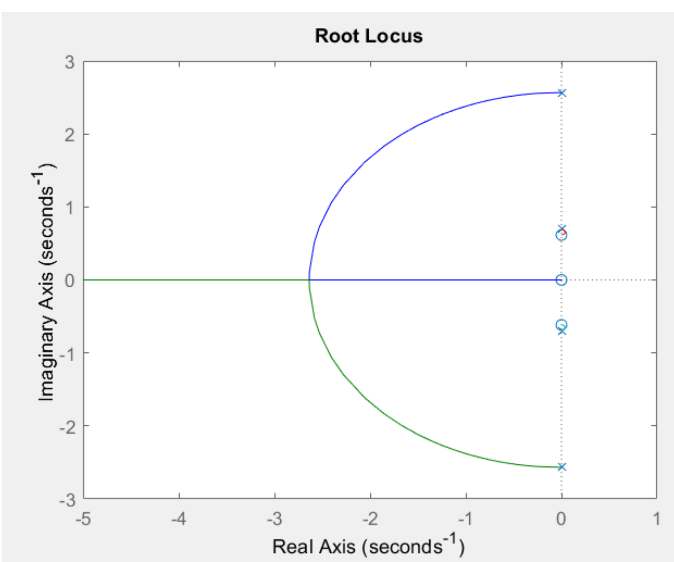


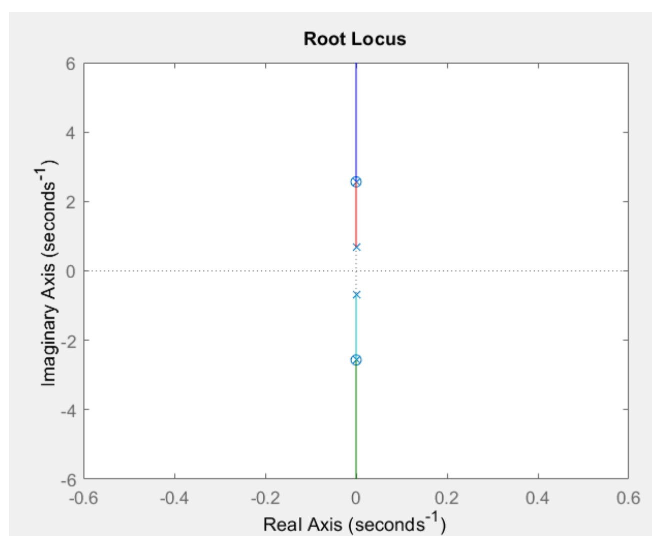

Root Locus
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})
4 3 2 1 0 -1 -2 -3 -4
-0.5 0 0.5 1 1.5 2 2.5


Root Locus
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})
4 3 2 1 0 -1 -2 -3 -4
-2 -1.5 -1 -0.5 0 0.5 1 1.5 2


Root Locus
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})
6 4 2 0 -2 -4 -6
-0.6 -0.4 -0.2 0 0.2 0.4 0.6


Root Locus
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})
6 4 2 0 -2 -4 -6
-0.6 -0.4 -0.2 0 0.2 0.4 0.6


Root Locus
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})
3 2 1 0 -1 -2 -3
-5 -4 -3 -2 -1 0 1


Root Locus
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})
6 4 2 0 -2 -4 -6
-0.6 -0.4 -0.2 0 0.2 0.4 0.6

12. طراحی رویتگر

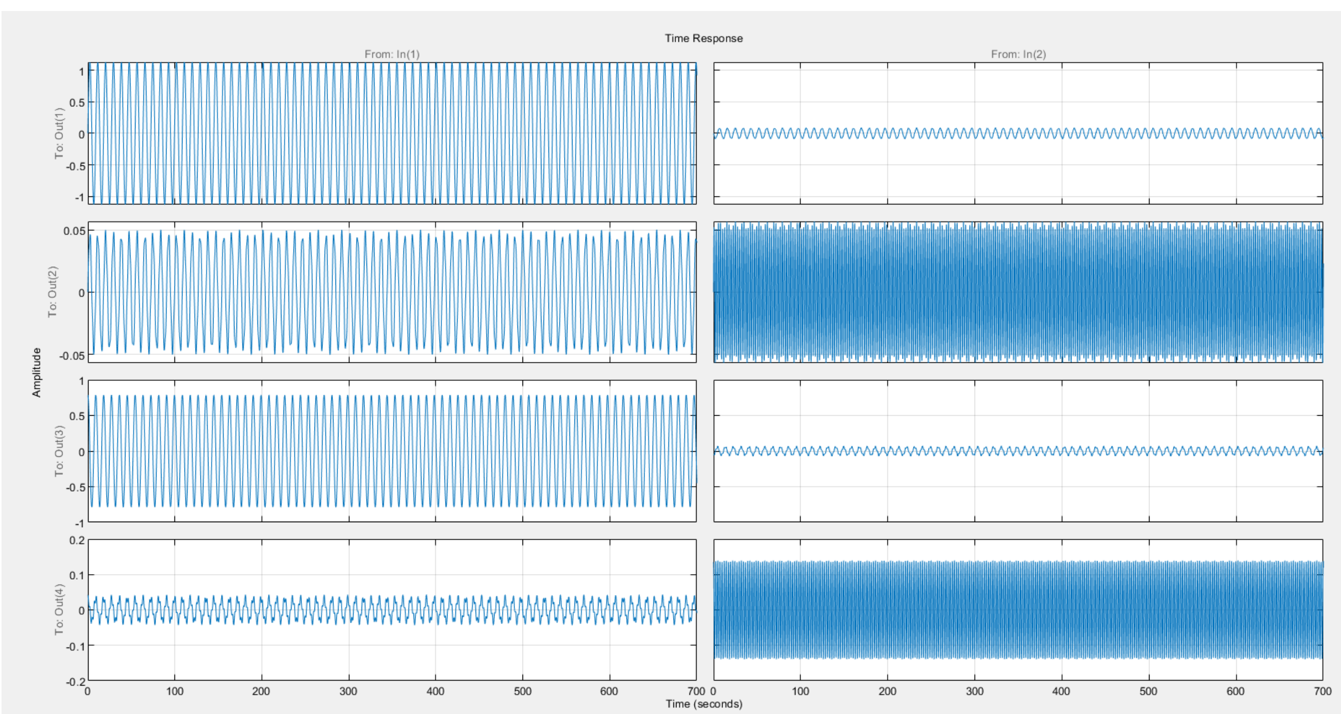
```
%Observer designing
p3 = [-1, -2, -3, -4];
L = place(A',C',p3)';
A_obs = A-L*C;
B_obs = [B, L];
C_obs = [C;eye(4)];
sysObserver = ss(A_obs,B_obs,C_obs,0);
```

```
sysObserver =

  A =
       x1  x2  x3  x4
   x1  -1   0   0   0
   x2   0  -2   0   0
   x3   0   0  -3   0
   x4   0   0   0  -4

  B =
            u1       u2       u3       u4       u5       u6
   x1        0        0        1        0        1        0
   x2        0        0        0        2        0        1
   x3    0.787  -0.0426  -0.4568  -0.6196        3        0
   x4   0.0426   0.1349   0.2485   -6.617        0        4

  C =
       x1  x2  x3  x4
   y1   1   0   0   0
   y2   0   1   0   0
   y3   0   0   1   0
   y4   0   0   0   1
   y5   1   0   0   0
   y6   0   1   0   0
   y7   0   0   1   0
   y8   0   0   0   1
```

13. پاسخ زمانی

13.1. پاسخ زمانی سیستم حلقه باز

```
%Time response
%openloop
[y,t] = step(tf_o,8);
opts = timeoptions;
opts.Grid = 'on';
impulseplot(tf_o,opts);
```



13.2. پاسخ زمانی سیستم حلقه بسته با قطب های دور از مبدا

```
%closedloop
[y,t] = step(syscl,8);
opts = timeoptions;
opts.Grid = 'on';
impulseplot(syscl,opts);
```