

آشنایی با نرم افزار
MATLAB

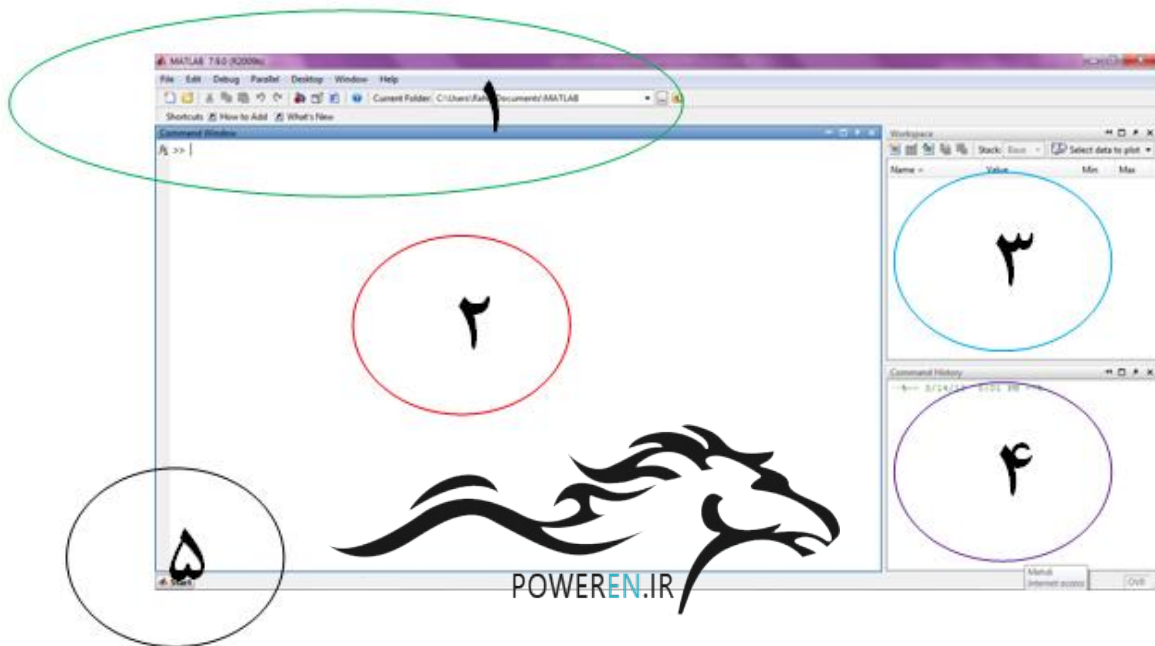


1. مقدمه:

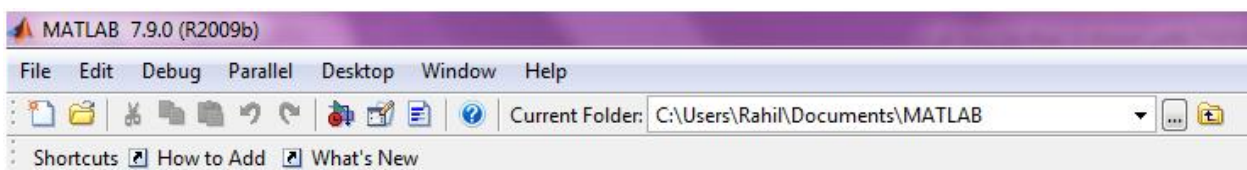
نرم‌افزار MATLAB که یکی از مهم‌ترین و قدرتمندترین ابزارهای مورد استفاده در رشته مهندسی برق می‌باشد، قابلیت‌های منحصر به فردی را برای تحلیل سیستم‌ها ارائه می‌دهد. در این متن کاربردهای این نرم‌افزار در درس کنترل خطی آموزش داده می‌شود. نسخه نرم‌افزار بکار برده شده برای این متن 9 می‌باشد.


2. معرفی بخش‌های مختلف نرم‌افزار:

با ورود به نرم‌افزار صفحه اصلی نرم‌افزار به صورت زیر باز می‌شود. همان‌طور که می‌بینید این صفحه پنج قسمت مجزا دارد:



بخش اول شامل منوهای اصلی نرم‌افزار می‌باشد:

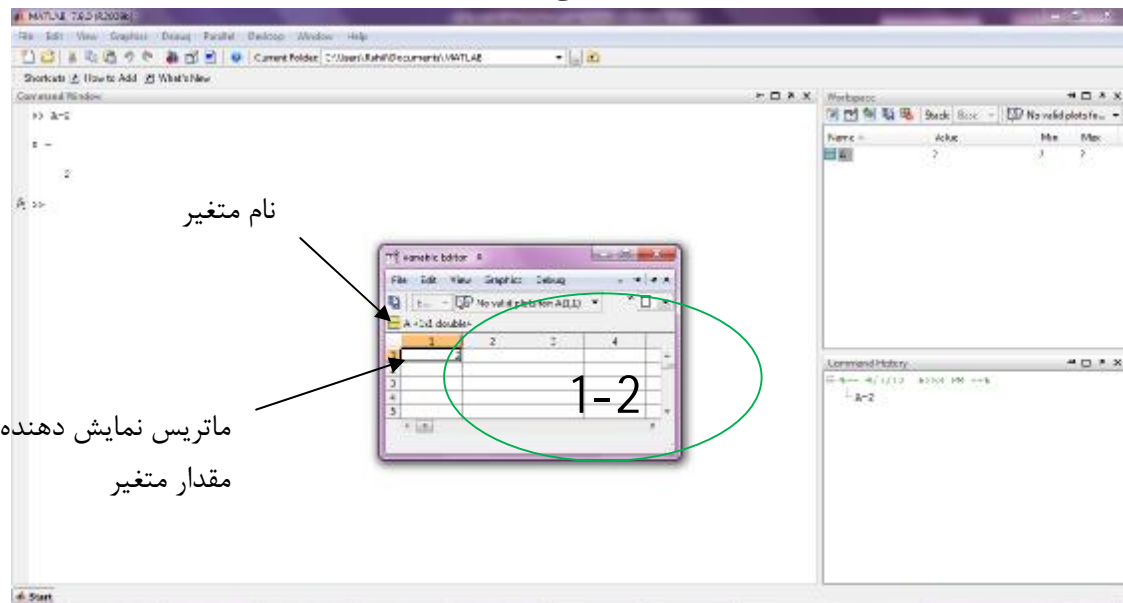


با کلیک بر روی کلید  وارد بخش Simulink نرم‌افزار خواهید شد که این بخش در فصل‌های آتی معرفی خواهد شد. منوی اصلی نیز شامل مسیرهای کاربردی است که در جای خود معرفی خواهند شد. بخش دوم Command نرم‌افزار نام دارد. جایی که دستورهای مورد نظر را تایپ کرده و پاسخ نرم‌افزار را دریافت می‌کنید.

بخش سوم Workspace نام دارد و تمام متغیرهای ذخیره‌شده توسط دستورات محیط Command در این بخش ذخیره می‌شود. به طور مثال اگر در Command تایپ کنید:

```
>>A=2
```

این متغیر در محیط Workspace به صورت زیر ذخیره می شود:



که اگر بر روی `A` در این محیط کلیک کنید؛ یک ماتریس یکه شامل مقدار `A` در پنجره `1-2` نمایش داده می شود. بخش چهارم بخش ذخیره دستوراتی است که در محیط `Command` تایپ می شود. همان طور که در شکل فوق می بینید؛ دستور `A=2` در این بخش ظاهر شده است. بخش پنجم شامل کلید `Start` است که محیط های مختلف نرم افزار را در بر دارد و برای ورود به هر یک از آنها می توان از طریق منوی باز شده توسط این کلید استفاده نمود.

▼ نکته: دستورات `Clear all` و `CLC` به ترتیب تمام متغیرهای ذخیره شده در `workspace` و محیط `Command` را پاک می کنند.

3. دستورات محاسباتی لازم برای درس کنترل خطی:

در این بخش به معرفی دستوراتی که در محاسبات مسائل کنترل خطی کاربرد دارند می پردازیم:

1- تعریف بردار و ماتریس:

ماتریس و بردار زیر

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ -1 \\ 1 \\ 2 \end{bmatrix}$$

در محیط `Command` به صورت زیر وارد می شوند:

```
>> A=[1 2 3;4 5 6;7 8 9]
A =
```

دستورکار آزمایشگاه سیستم های کنترل خطی

```
1 2 3
4 5 6
7 8 9
>> B=[0;-1;1;2]
B =
0
-1
1
2
```

توجه کنید که درایه های سطری ماتریس را می توان بجای فاصله با "،" از یکدیگر جدا کرد:

```
>> A=[1,2,3;4,5,6;7,8,9]
A =
1 2 3
4 5 6
7 8 9
```

نکته: ماتریس یکه، صفر و قطری به صورت زیر تعریف می شوند:

```
>> eye(3)
ans =
1 0 0
0 1 0
0 0 1
>> zeros(2,3)
ans =
0 0 0
0 0 0
>> diag([1,2,3])
ans =
1 0 0
0 2 0
0 0 3
```



2- عملیات ماتریسی:

عملیات ماتریسی نظیر عملیات جبری ماتریس، محاسبه معکوس ماتریس، معادله مشخصه، بردارهای ویژه و مقادیر ویژه در مثال زیر بخش معرفی می گردند.

مثال: عملیات ماتریسی زیر را برای ماتریس های A,B,C توسط نرم افزار Matlab محاسبه نمایید.

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 1 \\ 0 & 2 & 3 \end{bmatrix}, B = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- 1) $A+C$
- 2) $A-C$
- 3) A^3
- 4) A^{-1}
- 5) $A.B$
- 6) $\det(A)$

(6) مقادیر و بردارهای ویژه ماتریس A

(7) معادله مشخصه ماتریس A و محاسبه ریشه های آن

(8) ترانهاده بردار B

9) B^{-1}

```

>> A=[1 0 0;0 4 1;0 2 3]
A =
     1     0     0
     0     4     1
     0     2     3
>> B=[1;0;1]
B =
     1
     0
     1
>> C=[1 0 0;0 1 0;0 0 1]
C =
     1     0     0
     0     1     0
     0     0     1
>> A+C
ans =
     2     0     0
     0     5     1
     0     2     4
>> A-C
ans =
     0     0     0
     0     3     1
     0     2     2
>> A^3
ans =
     1     0     0
     0    86    39
     0    78    47
>> A^-1
ans =
    1.0000     0     0
         0    0.3000   -0.1000
         0   -0.2000    0.4000
>> A*B
ans =
     1
     1
     3
>> det(A)
ans =
    10
>> eig(A)
ans =
     2
     5
     1
این دستور به تنهایی تنها مقادیر ویژه ماتریس را محاسبه می‌کند:
>> [V,landa]=eig(A): این دستور هم مقادیر و هم بردارهای ویژه ماتریس را محاسبه می‌کند:
V =
     0     0     1.0000
    0.4472   -0.7071     0
   -0.8944   -0.7071     0
landa =
     2     0     0
     0     5     0
     0     0     1
>> poly(A)
ans =
     1    -8    17   -10
رزیاب معادله مشخصه ماتریس را مشخص می‌کند:
>> roots(poly(A))
ans =
    -1.0000 + 2.8284i
    -1.0000 - 2.8284i
    -0.0000 + 0.0000i
ریشه‌های معادله مشخصه را محاسبه می‌کند:

```



دستور کار آزمایشگاه سیستم های کنترل خطی

```
ans =  
5.0000  
2.0000  
1.0000
```

```
>> B'
```

```
ans =  
1 0 1
```

```
>> B^-1
```

```
??? Error using ==> mpower
```

این پیغام به این دلیل ظاهر شده است که ماتریس معکوس پذیر نیست:

برای محاسبه معکوس یک ماتریس از دستور $\text{inv}(A)$ نیز می توان استفاده نمود:

```
>> inv(A)
```

```
ans =  
1.0000 0 0  
0 0.3000 -0.1000  
0 -0.2000 0.4000
```

3- بسط به کسرهای جزئی، محاسبه قطب و صفر تابع تبدیل:

اگر تابع تبدیل یک سیستم به صورت زیر باشد:

$$G(s) = \frac{b_0 s^m + b_1 s^{m-1} + \dots + b_{mn}}{s^n + a_1 s^{n-1} + \dots + a_n}$$

که در آن $m \leq n$ است. با دستورات زیر ابتدا کسر تابع تبدیل به صورت بردارهای صورت و مخرج تعریف می کنیم:

```
num_G=[b_0 b_1 ... b_m]
```

```
den_G=[1 a_1 ... a_n]
```

(بجای num_G و den_G می توان از هر اسم دیگری نیز استفاده کرد. سپس آن را به کسرهای جزئی بسط می دهیم:

```
[r,p,k]=residue(num,den)
```

که در آن r, p, k به ترتیب ریشه های مخرج، ضرایب صورت و عدد ثابت می باشند.

اگر بخواهیم تابع تبدیل $G(s)$ را به تنهایی نشان دهیم از دستور

```
num_G=[b_0 b_1 ... b_m]
```

```
den_G=[1 a_1 ... a_n]
```

```
G=tf(num_G,den_G)
```

استفاده می کنیم¹.

برای محاسبه صفر و قطب های تابع دستورات زیر را اجرا می کنیم:

```
[z,p,k]=tf2zp(num_G,den_G)
```

دستور

¹ روش دیگر تعریف تابع تبدیل $G(s) = \frac{b_0 s^m + b_1 s^{m-1} + \dots + b_{mn}}{s^n + a_1 s^{n-1} + \dots + a_n}$ بصورت زیر است:

$$s = \text{tf}('s')$$

$$G = (b_m * s^m + \dots + b_0) / (a_n * s^n + \dots + a_0)$$

آشنایی با نرم افزار MATLAB

[num_G,den_G]=tf2zp(z,p,k)

عکس این عملیات را انجام می‌دهد. Tf مخفف Transfer Function و zp مخفف Zero/pole می‌باشد. (بجای اینکه بردارهای صورت و مخرج را به صورت جداگانه تعریف کنیم می‌توانیم مستقیم بردارها را در دستور مورد نظر قرار دهیم.)

مثال: عملیات با نرم‌افزار MATLAB نشان دهید که $G(s)$ بسطی به صورت زیر دارد:

$$G(s) = \frac{2s^3 + 5s^2 + 3s + 6}{s^3 + 6s^2 + 11s + 6} = \frac{-6}{s+3} + \frac{-4}{s+2} + \frac{3}{s+1} + 2$$

سپس قطب و صفرهای آن را مشخص نمایید:

```
>> num_G=[2 5 3 6]
num_G =
     2     5     3     6
>> den_G=[1 6 11 6]
den_G =
     1     6    11     6
>> G=tf(num_G,den_G)
Transfer function:
 2 s^3 + 5 s^2 + 3 s + 6
-----
s^3 + 6 s^2 + 11 s + 6
یا از دستور

>> s=tf('s');
>> G=(2*s^3+5*s^2+3*s +6)/(s^3 +6*s^2+11*s+6)

Transfer function:
 2 s^3 + 5 s^2 + 3 s + 6
-----
s^3 + 6 s^2 + 11 s + 6
استفاده می‌کنیم.

>> [r,p,k]=residue (num_G,den_G)
r =
   -6.0000
   -4.0000
    3.0000

p =
   -3.0000
   -2.0000
   -1.0000

k =
     2

*****

>> [z,p,k]=tf2zp(num_G,den_G)
z =
   -2.3965
   -0.0518 + 1.1177i
   -0.0518 - 1.1177i
p =
   -3.0000
   -2.0000
   -1.0000
k =
     2
```

4- تبدیل تابع تبدیل به فضای حالت و بالعکس:

اگر تابع تبدیل یک سیستم به صورت زیر باشد:

$$G(s) = \frac{b_0 s^m + b_1 s^{m-1} + \dots + b_{mm}}{s^n + a_1 s^{n-1} + \dots + a_n}$$

که در آن $m \leq n$ است. با دستورات زیر معادلات حالت سیستم به دست می آید:

[A,B,C,D]=tf2ss(num_G,den_G)

تبدیل تابع تبدیل به فضای حالت:

[num_G,den_G]=ss2tf(A,B,C,D)

تبدیل فضای حالت به تابع تبدیل:

مثال: برای تابع تبدیل مثال فوق معادلات حالت را مشخص نمایید:

```
>> [A,B,C,D]=tf2ss(num_G,den_G)
A =
    -6    -11     -6
     1     0     0
     0     1     0
B =
     1
     0
     0
C =
    -7    -19     -6
D =
     2
```

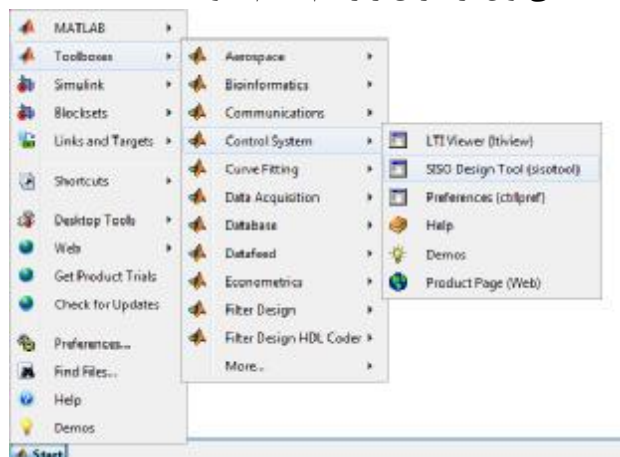


4. آشنایی با محیط SISO:

برای بررسی پایداری و همچنین رسم پاسخ یک سیستم که به صورت دیاگرام بلوکی و یا تابع تبدیل بیان شده است، نرم افزار MATLAB یک جعبه ابزار خاص به نام sisotool دارد. برای وارد شدن به این محیط در بخش Command نرم افزار تایپ کنید: sisotool

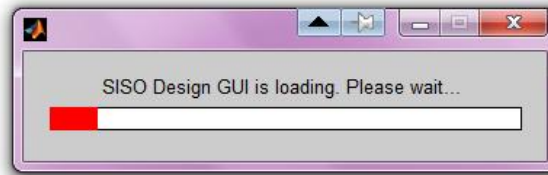
```
>> sisotool
```

البته برای وارد شدن به این محیط می توان از طریق زیر هم اقدام نمود:

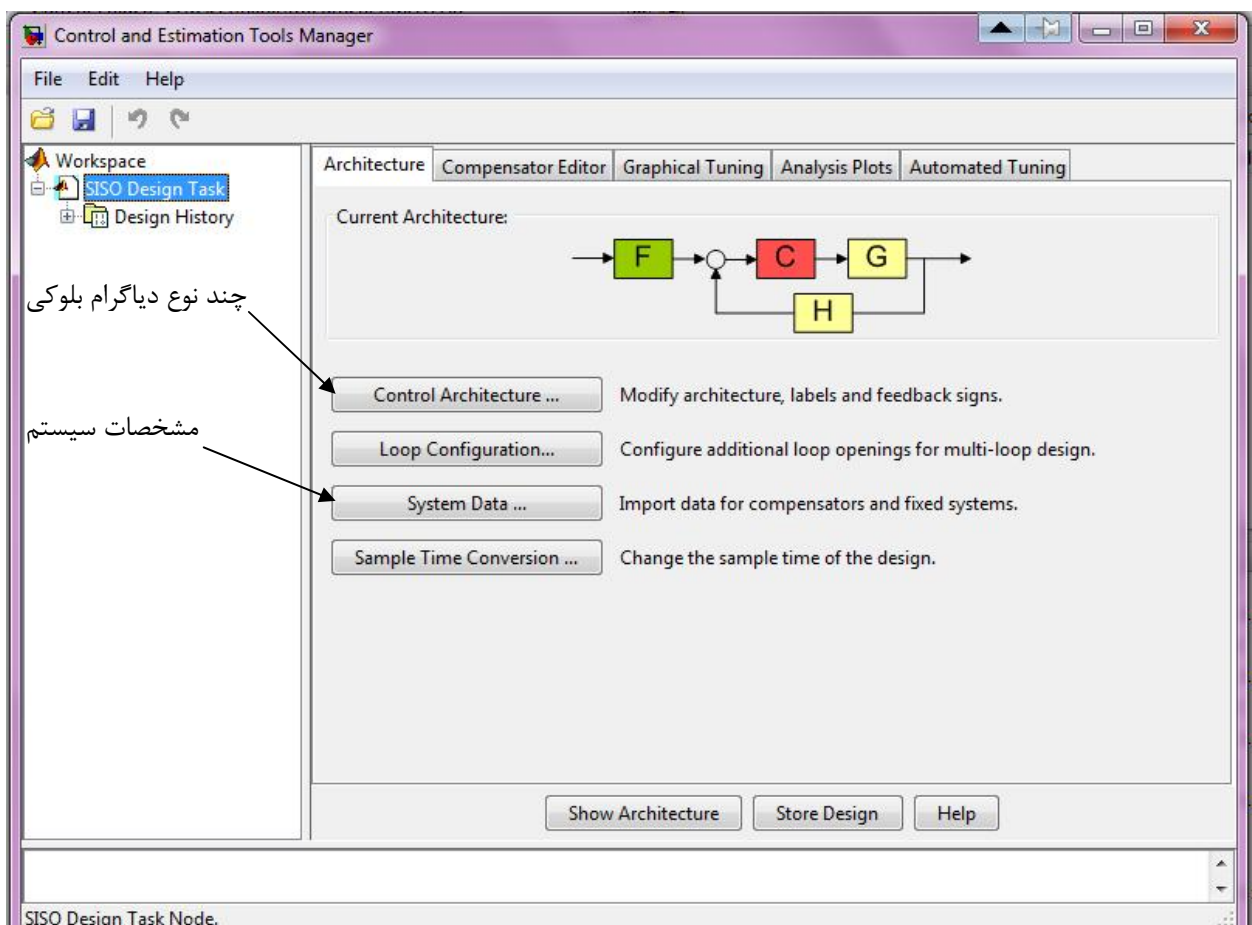


آشنایی با نرم افزار MATLAB

پس از انجام این کار پنجره زیر ظاهر می شود:



جعبه ابزار SISO دارای امکانات بسیار زیادی است که معرفی تمام آنها از حوزه درس کنترل خطی خارج است. بنابراین در این متن تنها به معرفی امکانات مورد نیاز درس کنترل خطی می پردازیم. برای این کار یک سیستم با تابع تبدیل مشخص را بررسی کرده و انواع پاسخ و نمودارهای پایداری آن را رسم می کنیم.

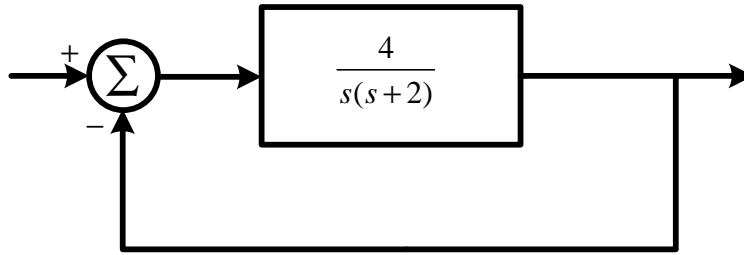


مثال: برای سیستم زیر با استفاده از جعبه ابزار SISO نرم افزار MATLAB یک دیاگرام حلقه بسته ایجاد کرده و پاسخ های پله، ضربه، مشخصات پاسخ پله، قطب و صفرها و نمودارهای مکان ریشه، نایکوئیست و بود سیستم حلقه بسته را رسم نمایید.

$$G(s) = \frac{4}{s(s+2)}$$

سیستم حلقه بسته به صورت زیر است:

دستورکار آزمایشگاه سیستم‌های کنترل خطی

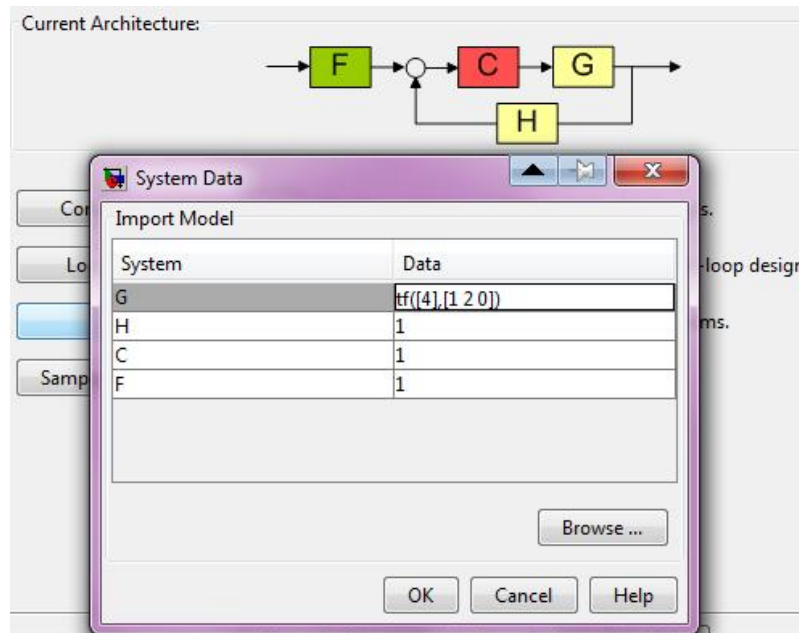


پس از ورود به بخش siso ابتدا باید تابع تبدیل را وارد نماییم. برای وارد کردن تابع تبدیل فوق ابتدا باید بردارهای صورت و مخرج را مشخص نماییم.

بردار صورت: [4]

بردار مخرج: [1 2 0]

اکنون بر روی کلید system data کلیک نموده و تابع تبدیل و سیستم حلقه بسته را به صورت زیر مشخص می‌کنیم:

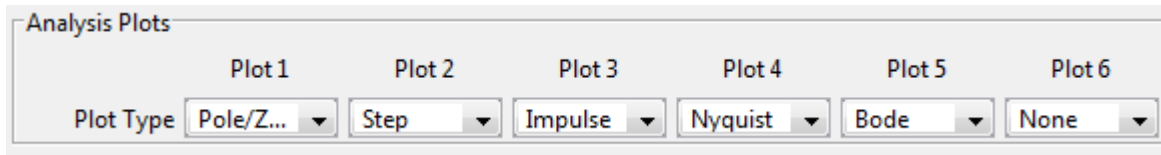


با کلیک بر روی دکمه OK سیستم وارد می‌شود. توجه کنید که تابع تبدیل پس از وارد شدن و کلیک بر روی دکمه OK دیگر قابل اصلاح نیست و برای تغییر آن باید مجدداً کل جمله تایپ شود. اکنون تب analysis plots را انتخاب کرده و چون سیستم حلقه بسته مورد نظر است گزینه زیر را انتخاب می‌کنیم:

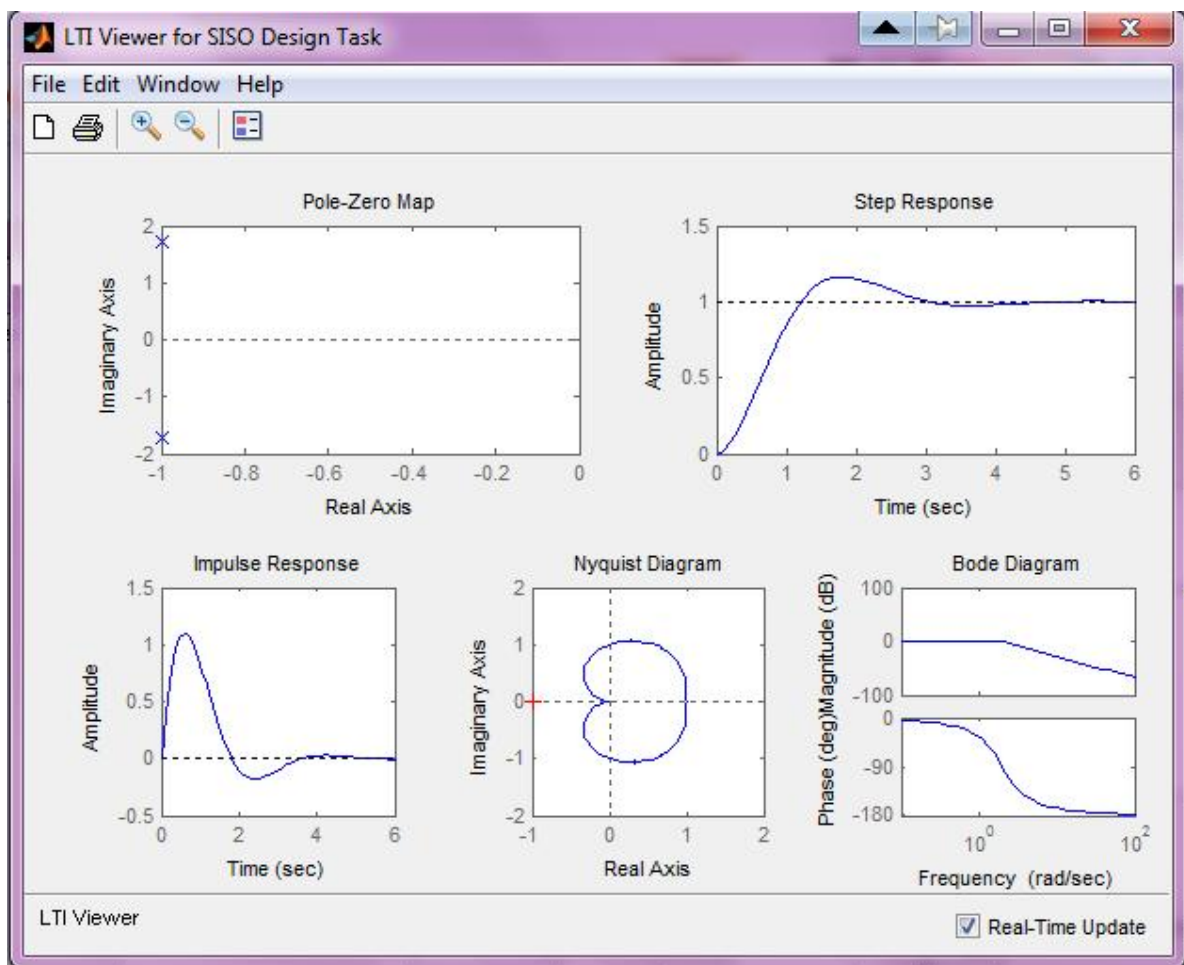
Contents of Plots						
Plots						
1	2	3	4	5	6	All
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

سپس برای رسم نمودارهای مربوطه گزینه‌های زیر را انتخاب می‌کنیم:

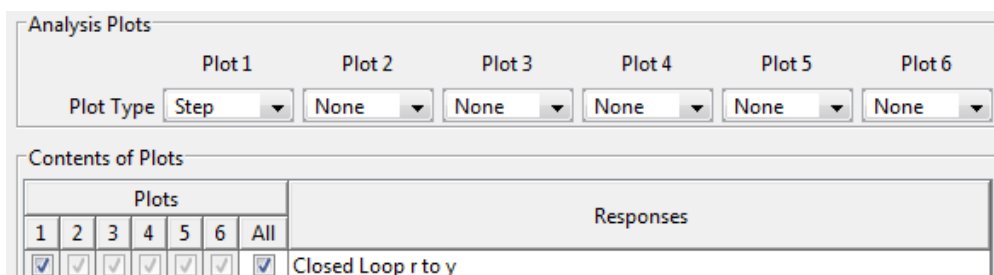
آشنایی با نرم افزار MATLAB



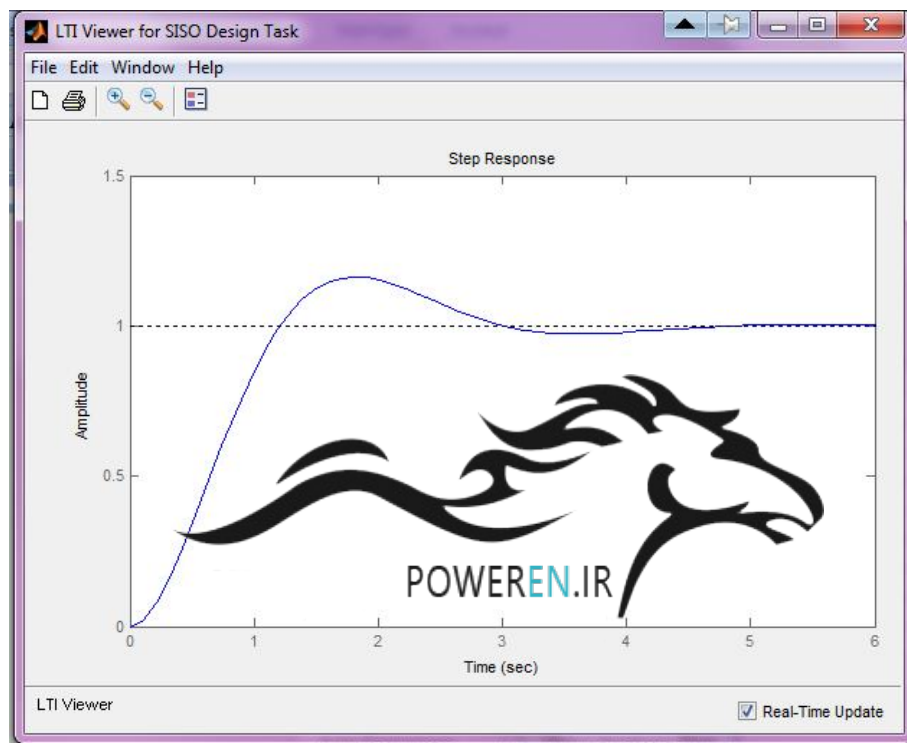
اگر بخواهیم هر یک از نمودارها به صورت جداگانه رسم شوند باید تنها plot1 را مشخص کرده و بقیه را روی none قرار دهیم. سپس گزینه show analysis plots را انتخاب می‌کنیم تا نمودارها ظاهر شوند:



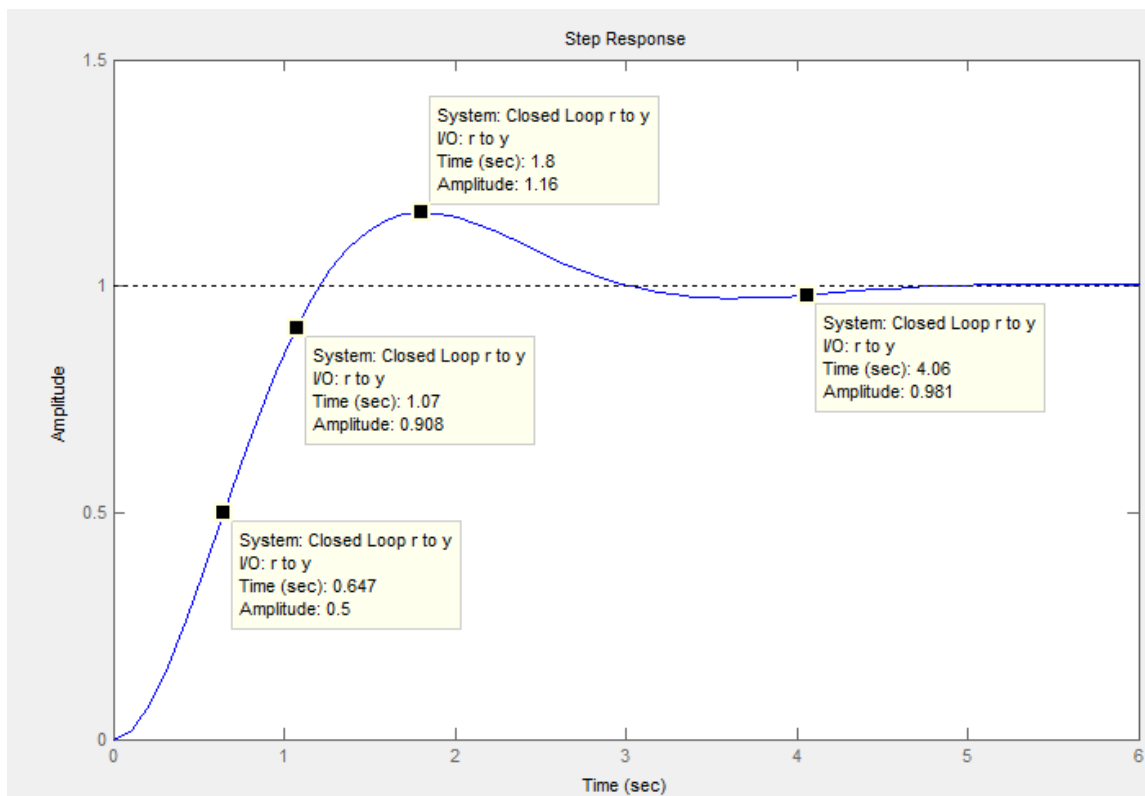
اکنون برای مشخص کردن مشخصات پاسخ مرتبه دوم یک‌بار به صورت تنها پاسخ پله را رسم می‌کنیم:



دستورکار آزمایشگاه سیستم های کنترل خطی



اکنون که پاسخ را داریم با کلیک بر روی قسمت های مختلف پاسخ مشخصات را به دست آوریم:

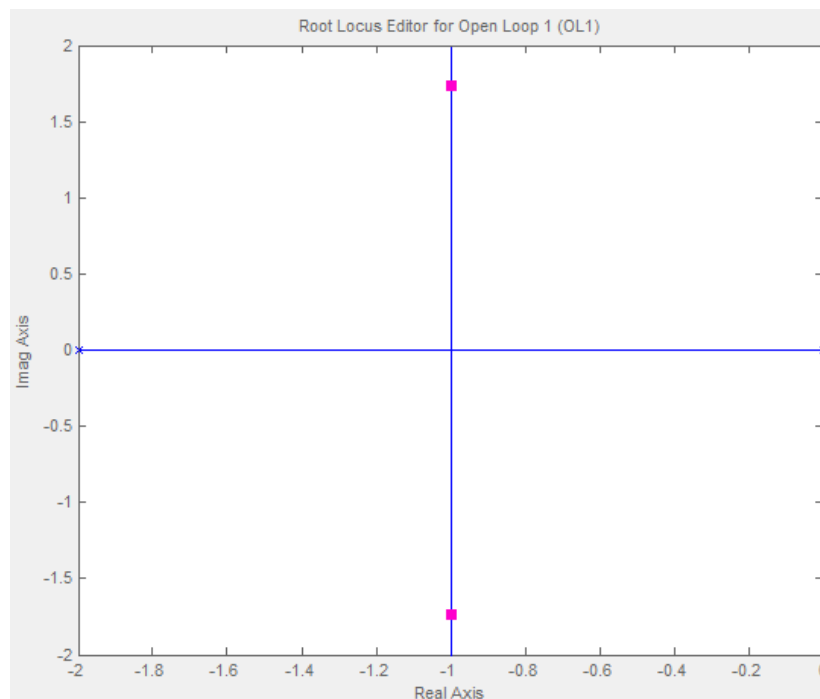


آشنایی با نرم افزار MATLAB

برای رسم نمودار مکان ریشه باید به تب graphical tuning می‌رویم. Plot 1 را بر روی root locus قرار داده و بقیه را روی none قرار می‌دهیم:

Architecture	Compensator Editor	Graphical Tuning	Analysis Plots	Automated Tuning
Design plots configuration				
Plot	Available Open/Closed Loop to Tune		Plot Type	
Plot 1	Open Loop 1	▼	Root Locus	▼
Plot 2	Open Loop 1	▼	None	▼
Plot 3	Closed Loop 1	▼	None	▼
Plot 4	Open Loop 1	▼	None	▼
Plot 5	Open Loop 1	▼	None	▼

اکنون گزینه show design plot را انتخاب می‌کنیم تا نمودار مکان ریشه مشاهده شود:



برای آموزش ترسیم نمودار مکان ریشه‌های یک سیستم، از یک مثال استفاده می‌کنیم. سیستم حلقه باز زیر را در نظر بگیرید:

$$G(s) = \frac{s+5}{s(s+1)^2(s+4)}$$

برای ترسیم نمودار مکان ریشه در قسمت command نرم‌افزار دستوره‌های زیر را اجرا می‌کنیم:

دستورکار آزمایشگاه سیستم های کنترل خطی

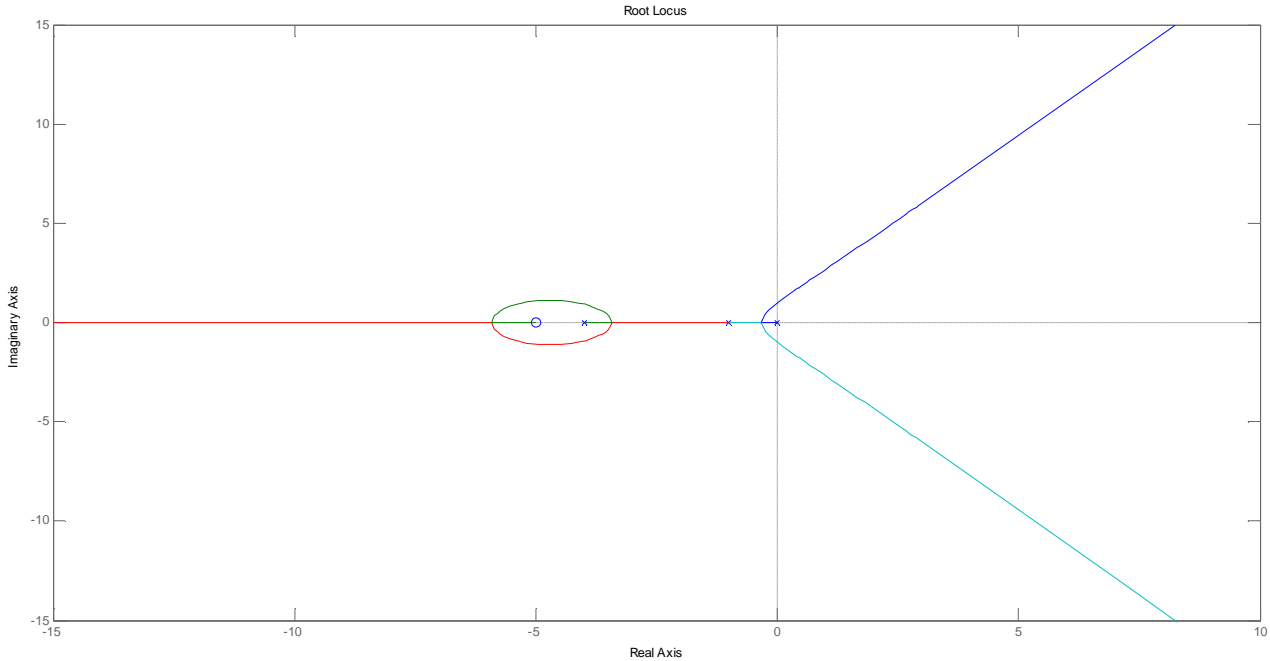
```
>> sys=tf([1 5],[1 6 9 4 0])
```

Transfer function:

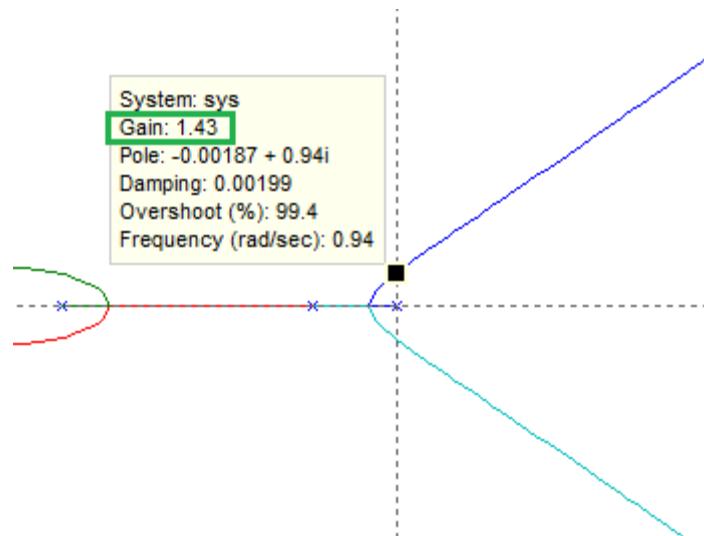
$$\frac{s + 5}{s^4 + 6s^3 + 9s^2 + 4s}$$

$$s^4 + 6s^3 + 9s^2 + 4s$$

```
>> rlocus(sys)
```



برای به دست آوردن بهره مرزی، بر روی محل برخورد نمودار با محور موهومی کلیک می کنیم. تا جای ممکن damping را به صفر (مثبت) نزدیک نموده و مقدار بهره را یادداشت نمایید.



برای ترسیم نمودار مکان ریشه های یک سیستم، از یک مثال استفاده می کنیم. سیستم حلقه باز زیر را در نظر بگیرید:

آشنایی با نرم افزار MATLAB

$$G(s) = \frac{s+5}{s(s+1)^2(s+4)}$$

برای ترسیم نمودار مکان ریشه در قسمت command نرم افزار دستوره های زیر را اجرا می کنیم:

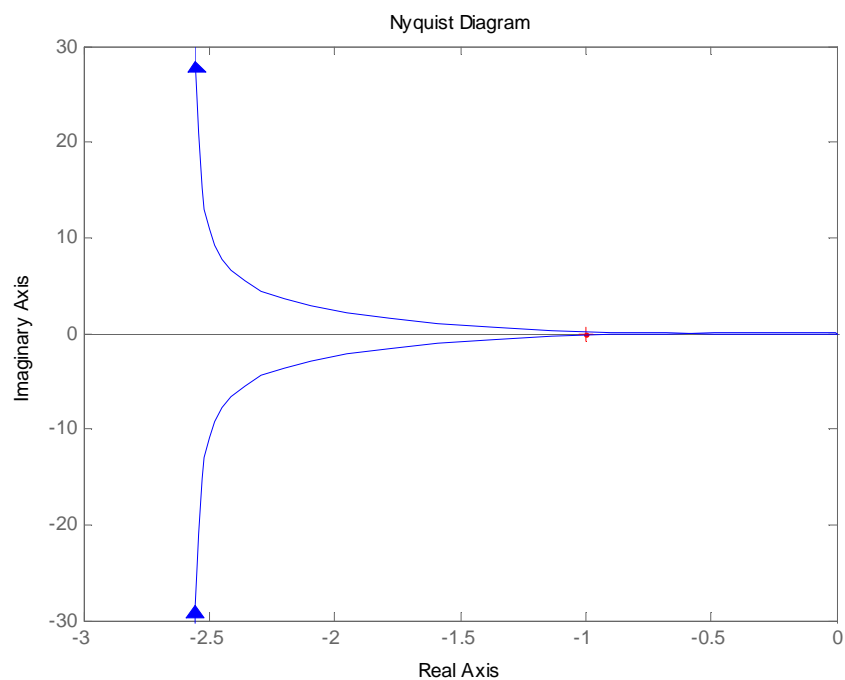
```
>> sys=tf([1 5],[1 6 9 4 0])
```

Transfer function:

s + 5

s^4 + 6 s^3 + 9 s^2 + 4 s

```
>> nyquist(sys)
```



برای ترسیم نمودارهای بودی یک سیستم، از یک مثال استفاده می کنیم. سیستم حلقه باز زیر را در نظر بگیرید:

$$G(s) = \frac{s+5}{s(s+1)^2(s+4)}$$

برای ترسیم نمودار بودی در قسمت command نرم افزار دستوره های زیر را اجرا می کنیم:

```
>> sys=tf([1 5],[1 6 9 4 0])
```

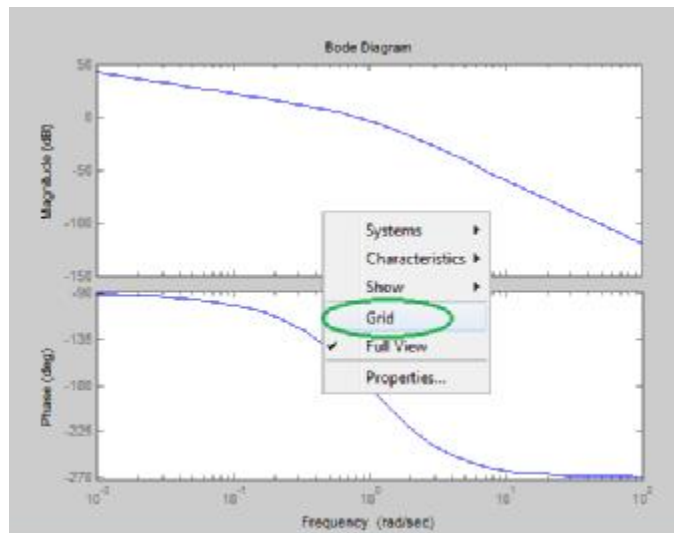
Transfer function:

s + 5

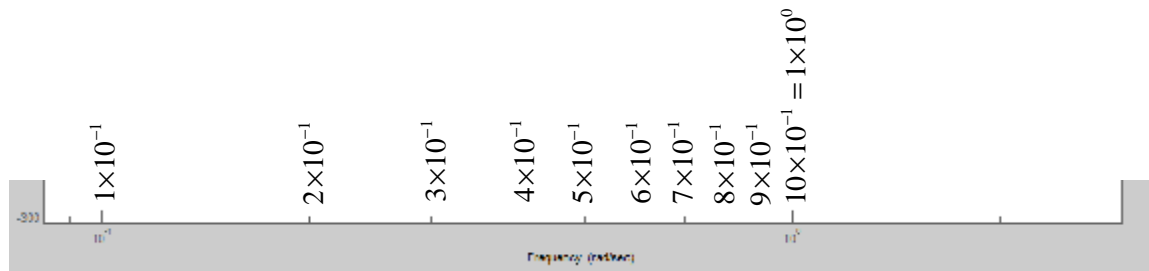
s^4 + 6 s^3 + 9 s^2 + 4 s

```
>>bode(sys)
```

دستور کار آزمایشگاه سیستم های کنترل خطی

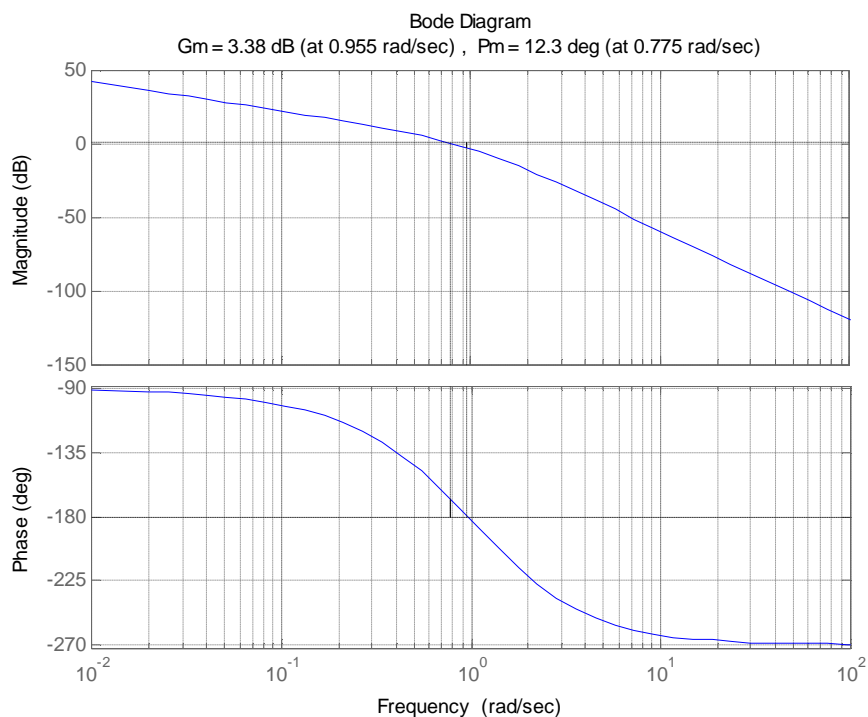


محور افقی، محور لگاریتمی فرکانس زاویه ای می باشد، برای خواندن اعداد روی این محور به شکل زیر توجه کنید:



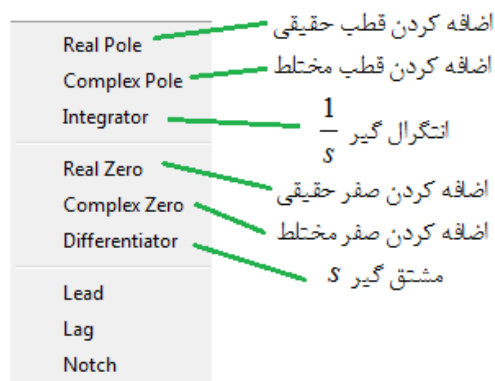
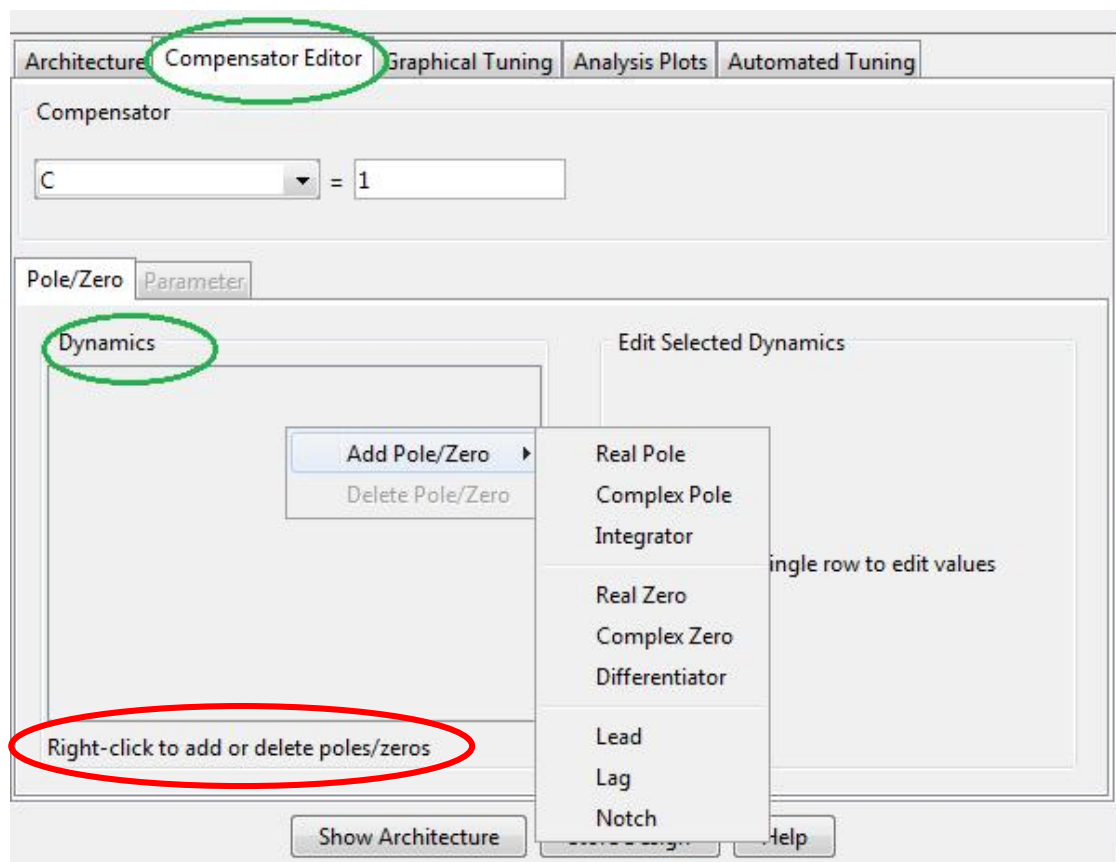
برای مشاهده مشخصات پایداری سیستم در محیط command دستور زیر را اجرا می کنیم:

```
>> margin(sys)
```

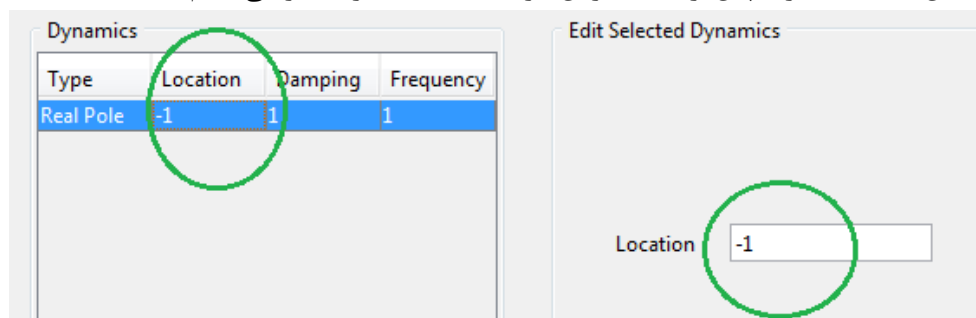


آشنایی با نرم افزار MATLAB

برای بررسی تأثیر اضافه نمودن صفر و قطب بر روی نمودارها به بخش Sisotool رفته و مراحل زیر را طی می کنیم:

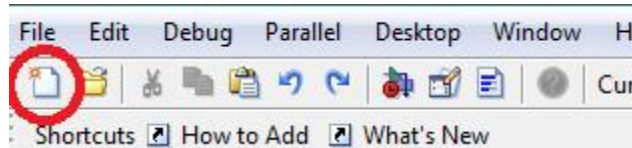


برای تعیین مکان قطب یا صفرها پس از اضافه کردن گزینه location را تغییر می دهیم:

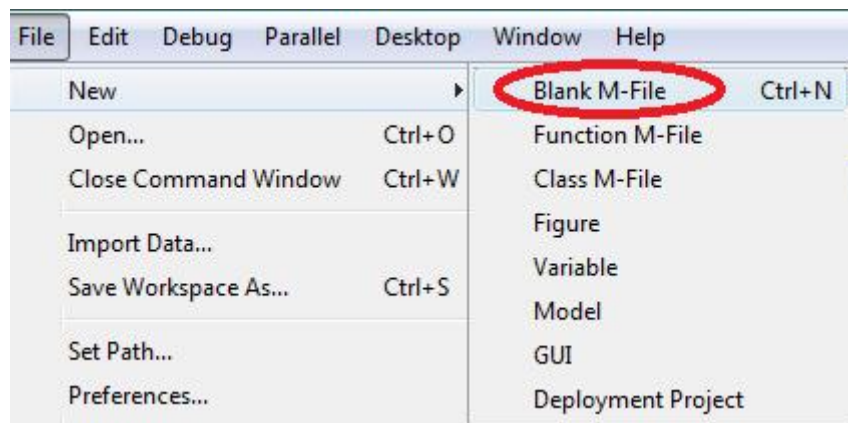


5. آشنایی با محیط برنامه نویسی:

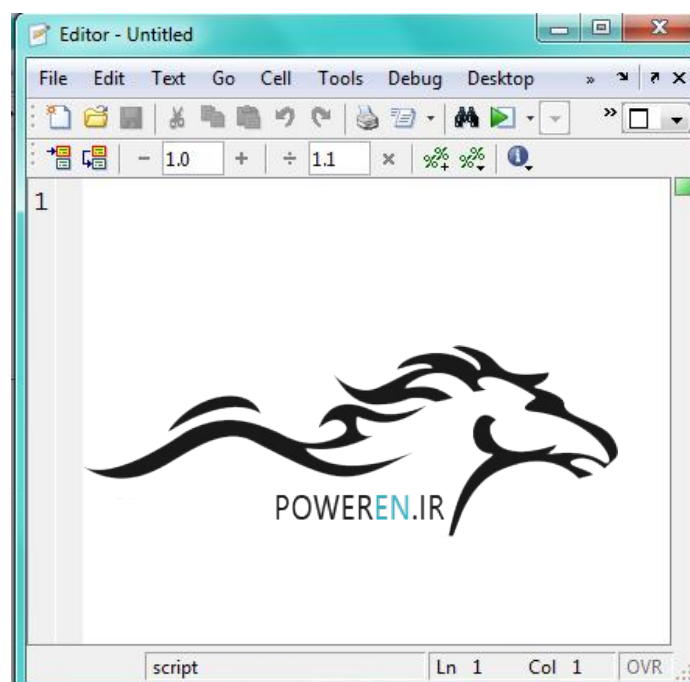
همان طور که در بخش های قبل دیدیم، اجرای دستورات در محیط Command به صورت تکی می باشد، یعنی بعد از نمایش اجرای هر دستور، می توان دستور بعدی را اجرا نمود. برای دستوراتی که باید پشت سر هم اجرا گردند، استفاده از محیط Command راحت نیست. چرا که اگر یکی از دستورات نیاز به اصلاح داشته باشد، مجدد تمام دستورات بعد از آن هم باید تایپ و اجرا گردد. در چنین مواردی استفاده از محیط برنامه نویسی MATLAB ضروری می باشد. برای باز کردن محیط برنامه نویسی نرم افزار ابتدا بر روی گزینه



یا مسیر



در صفحه اصلی کلیک می کنیم تا پنجره برنامه نویسی به شکل زیر باز شود:



آشنایی با نرم افزار MATLAB

برای شروع برنامه نویسی تایپ دو کد زیر در ابتدای برنامه ضروری می باشد:

```
clc
clear all
```

در صورتی که برنامه شامل دستورات ترسیمی است، لازم است تا کد

```
close all
```

نیز به ابتدای برنامه اضافه گردد.

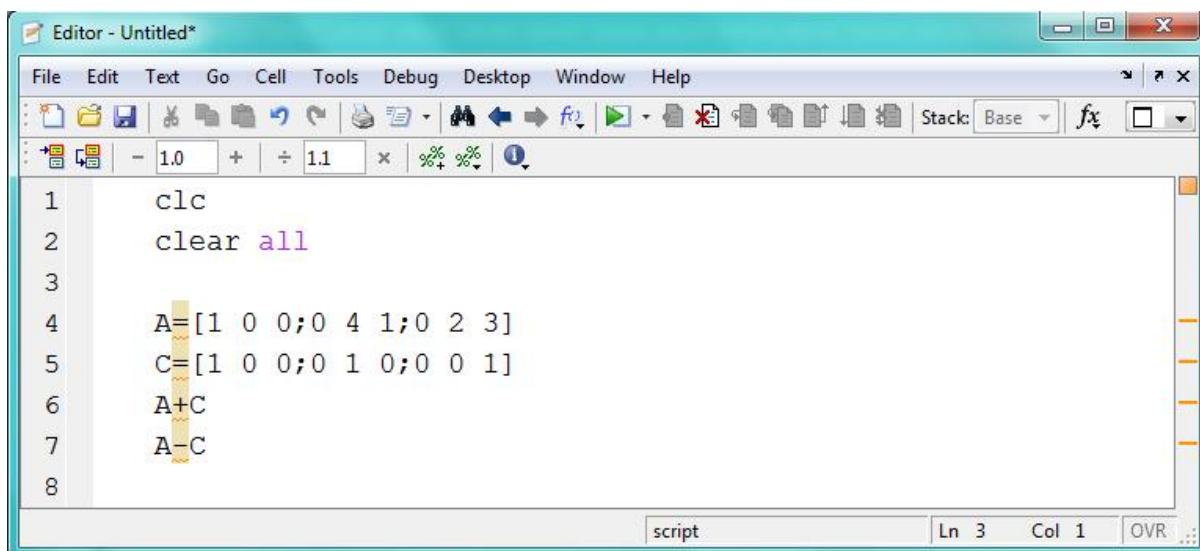
برای درک بهتر کار با این بخش به مثال زیر توجه کنید:

مثال: عملیات ماتریسی زیر را برای ماتریس های A, C توسط نرم افزار Matlab محاسبه نمایید.


$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 1 \\ 0 & 2 & 3 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

1) $A+C$


2) $A-C$



```
Editor - Untitled*
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base fx
- 1.0 + ÷ 1.1 x % % % % !
1 clc
2 clear all
3
4 A=[1 0 0;0 4 1;0 2 3]
5 C=[1 0 0;0 1 0;0 0 1]
6 A+C
7 A-C
8
script Ln 3 Col 1 OVR
```

برای اجرای برنامه ابتدا باید برنامه را ذخیره نمود (). برای ذخیره برنامه در نرم افزار MATLAB، نام فایل تنها می تواند شامل حروف الفبا و یا ترکیبی از حروف و اعداد باشد با این شرط که حرف اول اسم عدد نباشد. استفاده از اعداد به تنهایی برای ذخیره برنامه صحیح نیست و تنها کاراکتر مجاز در MATLAB کاراکتر underline یا _ می باشد. همچنین از فاصله در نام گذاری برنامه نباید استفاده کرد:

P1	صحیح	I_1	ناصحیح
lp	ناصحیح	P-1	ناصحیح
P_1	صحیح	P 1	ناصحیح

پس از ذخیره برنامه را اجرا می کنیم (). نتیجه اجرای برنامه در MATLAB در محیط Command نمایش داده می شود:

دستورکار آزمایشگاه سیستم‌های کنترل خطی

```
A =
    1     0     0
    0     4     1
    0     2     3

C =
    1     0     0
    0     1     0
    0     0     1

ans =
    2     0     0
    0     5     1
    0     2     4

ans =
    0     0     0
    0     3     1
    0     2     2
```

برای حذف نمایش اجرای کدهای غیرضروری می‌توان از `;` در انتهای خط مربوطه استفاده نمود. اگر بخشی از کد اشتباه نوشته شده باشد، برنامه پیغام خطا را در محیط Command نمایش می‌دهد:

```
clc
clear all

A=[1 0 0;0 4 1;0 2 3];
C=[1 0 0;0 1 0;0 0 1];
A+C
A-C
```

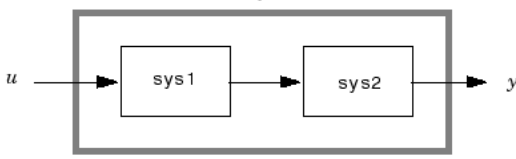
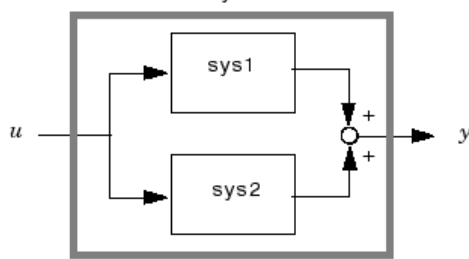
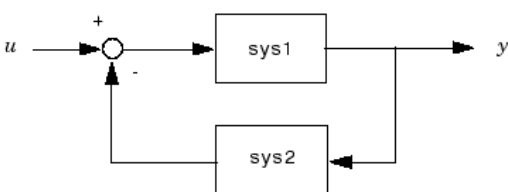
نتیجه اجرا:

```
??? Error using ==> pl at 5
Error using ==> vertcat
CAT arguments dimensions are not consistent.
```

با کلیک بر روی بخشی از پیغام که خط زیر آن کشیده شده است؛ برنامه شما را به خط مربوطه هدایت می‌کند. لیست زیر مجموعه‌ای از دستورات لازم برای درس کنترل خطی را به صورت یکجا نمایش می‌دهد:

دستورات ماتریسی	
eig(A)	محاسبه مقادیر ویژه یک ماتریس
[landa,vector]= eig(A)	محاسبه مقادیر ویژه و یک ماتریس
poly(A)	محاسبه معادله مشخصه ماتریس
roots(poly(A))	محاسبه ریشه‌های معادله مشخصه ماتریس
inv(A)	محاسبه معکوس ماتریس
tf(x,y)	اگر X و Y بردارهای صورت و مخرج یک تابع تبدیل باشد این دستور تابع تبدیل را نمایش می‌دهد.
tf2ss(x,y)	اگر X و Y بردارهای صورت و مخرج یک تابع تبدیل باشد این دستور معادلات حالت را نمایش می‌دهد.
tf2zp(x,y)	اگر X و Y بردارهای صورت و مخرج یک تابع تبدیل باشد این دستور قطب و صفر تابع

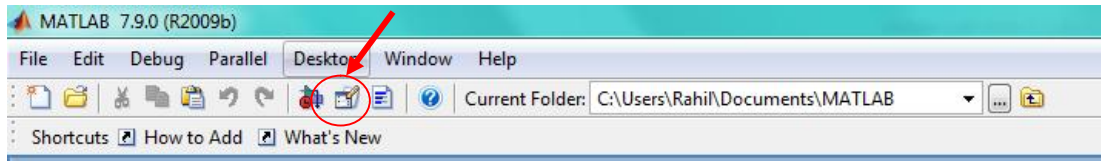
آشنایی با نرم افزار MATLAB

	تبدیل را نمایش می دهد.
<code>ss2tf(A,B,C,D)</code>	اگر ماتریس های A,B,C,D ماتریس های معادلات حالت باشند، این دستور تابع تبدیل معادل را نشان می دهد.
اگر x و y بردارهای صورت و مخرج یک تابع تبدیل باشند:	
<code>[r,f,k]=residue(x,y)</code>	این دستور بسط تابع تبدیل را نشان می دهد.
<code>step(x,y)</code> <code>step(A,B,C,D)</code>	این دستورات پاسخ پله سیستم را نمایش می دهند.
<code>impulse(x,y)</code> <code>impulse(A,B,C,D)</code>	این دستورات پاسخ ضربه سیستم را نمایش می دهند.
<code>lsim(x,y,u,t)</code> <code>lsim(A,B,C,D,u,t)</code>	این دستورات پاسخ به ورودی u سیستم را نمایش می دهند. ورودی u باید به صورت تابع زمانی تعریف گردد.
<code>rlocus(x,y)</code> <code>rlocus(A,B,C,D)</code>	این دستورات مکان ریشه های سیستم را ترسیم می کند.
<code>bode(x,y)</code> <code>bode(A,B,C,D)</code>	این دستورات دیاگرام بودی سیستم را ترسیم می کند.
<code>nyquist(x,y)</code> <code>nyquist(A,B,C,D)</code>	این دستورات دیاگرام نایکوئیست سیستم را ترسیم می کند.
<code>nichols(x,y)</code> <code>nichols(A,B,C,D)</code>	این دستورات دیاگرام زیگلر نیکولز سیستم را ترسیم می کند.
دستورات مخصوص وارد نمودن بخش های بلوک دیاگرام یک سیستم	
<code>sys = series(sys1,sys2)</code>	سری نمودن دو تابع تبدیل مطابق شکل زیر: 
<code>sys = parallel(sys1,sys2)</code>	سری نمودن دو تابع تبدیل مطابق شکل زیر: 
<code>sys = feedback(sys1,sys2)</code>	شاخه فیدبک: 

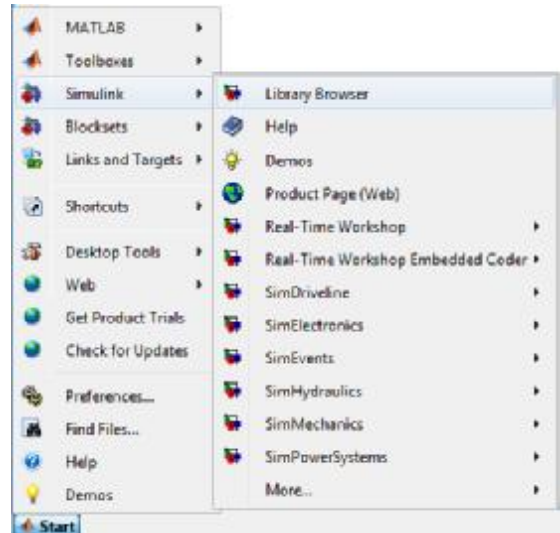
6. آشنایی با محیط شبیه سازی MATLAB:

دستور کار آزمایشگاه سیستم های کنترل خطی

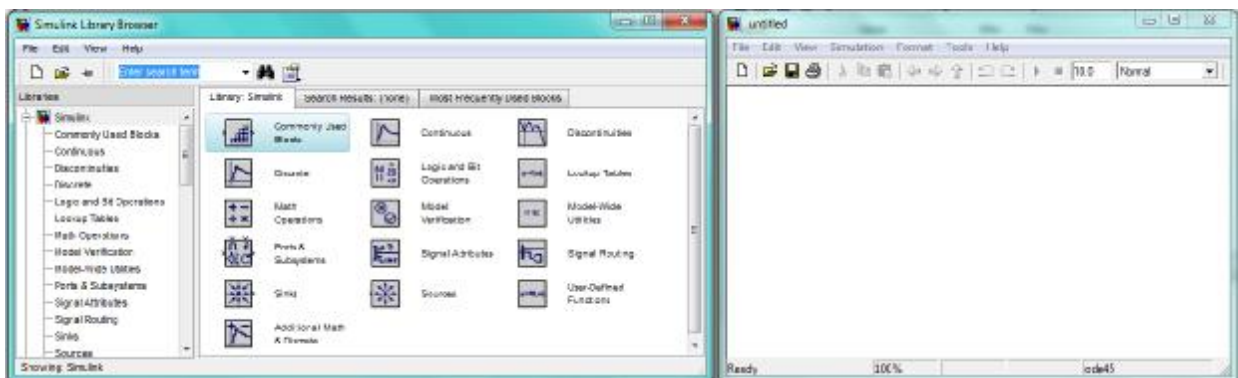
برای استفاده از محیط شبیه سازی نرم افزار، بر روی دکمه `simulink` در نوار بالای محیط `Command` کلیک می کنیم:



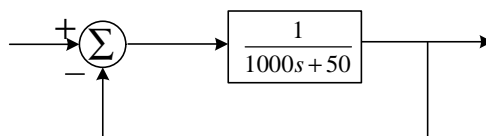
برای ورود به این محیط از طریق زیر نیز می توان عمل نمود:



پس از باز شدن کتابخانه بلوکی این محیط، یک فایل جدید برای شبیه سازی از طریق `File>New>New Model` ایجاد می کنیم:

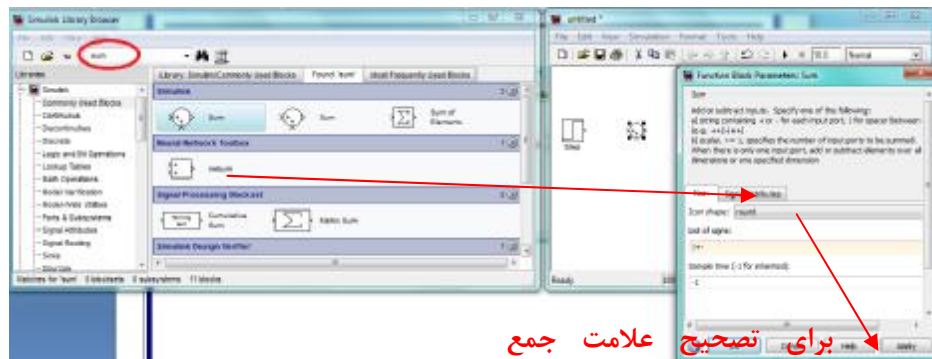
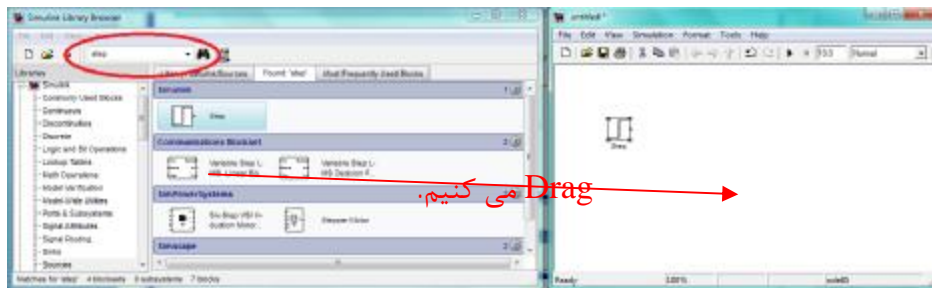


اکنون سیستم مورد نظر را به ورودی پله یکبار بصورت حلقه باز و بار دیگر بصورت حلقه بسته ترسیم می کنیم:

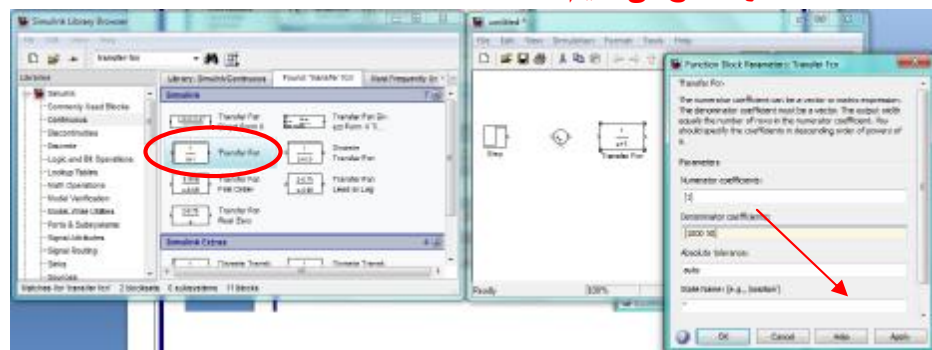


به مراحل زیر توجه کنید:

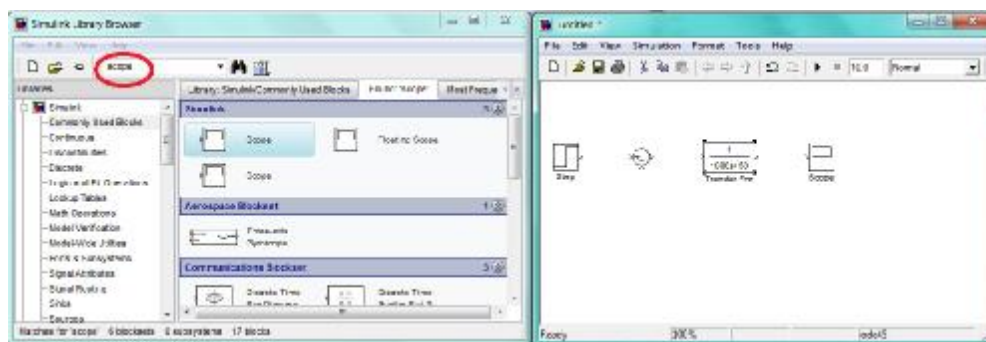
آشنایی با نرم افزار MATLAB



برای تصحیح علامت جمع کننده بر روی بلوک جمع کننده دو بار کلیک می کنیم و علامت را اصلاح می کنیم.



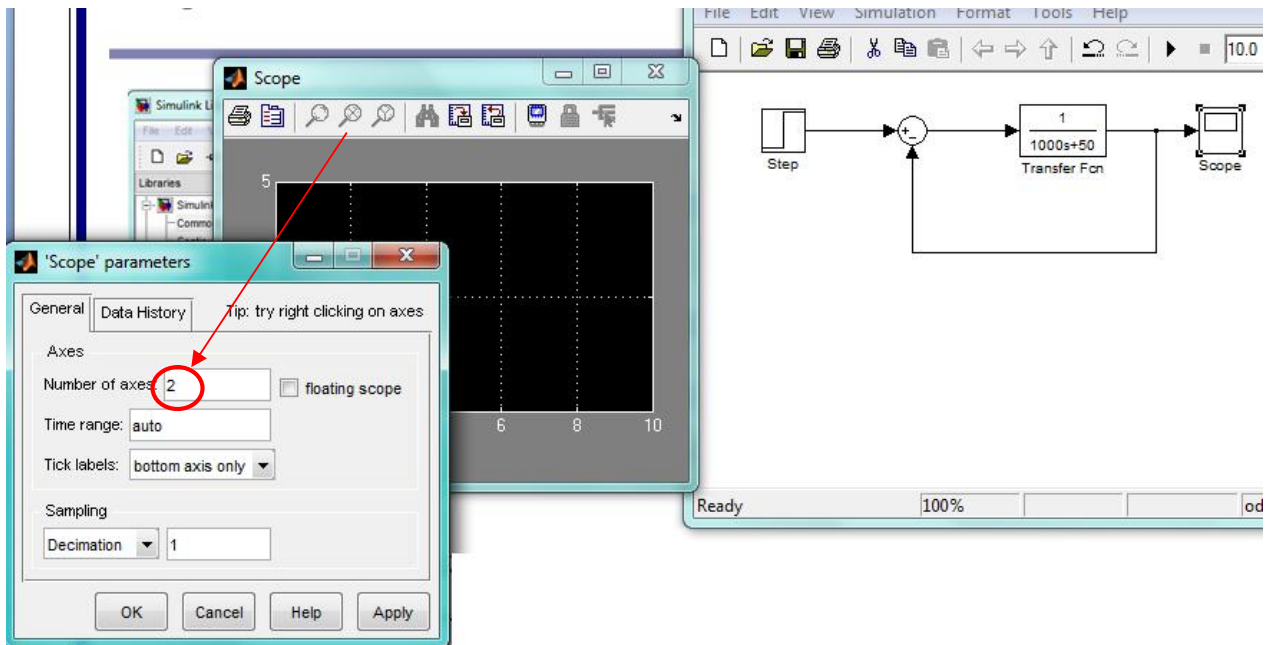
برای تصحیح تابع تبدیل بلوک بر روی آن دو بار کلیک نموده و بردارهای صورت و مخرج را وارد می کنیم.



برای مشاهده خروجی از بلوک Scope.

دستورکار آزمایشگاه سیستم های کنترل خطی

برای اتصال بلوک ها به هم می توان با کلیک بر روی یکی و نگداشتن دکمه Ctrl و کلیک بر روی بلوک دیگر آن دو را به هم متصل نمود. همچنین برای اتصال بلوک به سیم از سیم از ماوس استفاده می شود. برای مشاهده پاسخ حلقه باز و حلقه بسته بر روی یک اسیلوسکوپ بر روی آن کلیک نموده و تعداد ورودی آن را اصلاح می کنیم:



اکنون بار دیگر دیاگرام بلوکی حلقه باز را رسم می کنیم. با تغییر زمان سیمولیشن و کلیک بر روی دکمه start simulation سیستم ها شبیه سازی شده و پس از کلیک بر روی اسیلوسکوپ و کلیک راست بر روی صفحه و انتخاب گزینه Auto Scale می توان هر دو پاسخ را مشاهده نمود:

