

معرفی

متلب یکی از زبان های سطح بالا با تمرکز بر تکنیک های محاسباتی است. این نرم افزار محیطی مناسب برای انجام عملیات های ریاضی ، تحلیل های آماری ، بهینه سازی های ایجاد محیط های ویژوال و برنامه نویسی آن را همزمان فراهم کرده است.

نام متلب از حروف ابتدایی Matrix Laboratory آمده است.

آشنایی با برخی از قابلیت های متلب:

تعریف متغیرها:



- در متلب نیازی به تعریف متغیر ها (data type) نیست.
- متلب به کوچکی و بزرگی حروف حساس است.
- همه متغیرها از نوع double (۸ بایت) فرض می شوند.
- با دستور clear می توان متغیرها را از حافظه پاک کرد.
- دستور clc برای پاک کردن پنجره command بکار می رود.
- در نامگذاری فایل ها در متلب ، بهتر است که از _ (under line) و حروف کوچک استفاده شود. مثال :
a_new
- دستور whos همه متغیر های موجود در work space را با اطلاعاتی از قبیل اندازه ، تعداد بایت ها ، کلاس و ... نشان می دهد.
- دستور who همه ی متغیر های موجود را لیست می کند.
- در نامگذاری فایل ها از دستورات متلب استفاده نمی کنیم.
- با دستور doc صفحه help متلب باز می شود.
مثال: >> doc erf
- با دستور demo تعدادی از مثال های متلب نمایش داده می شود.
- متلب مابین کوچکترین عدد (10^{+311}) و بزرگترین عدد (10^{-311}) محدود است.

برخی از علائم و نشانه های پرکاربرد در متلب:

- NaN (not a number) مبهم
- i و j مختلط $i = \sqrt{-1}$
- pi $\pi=3.14$ → در متلب $\sin(\pi/4)$ را به صورت $\sin(\pi/4)$ می نویسیم.
- eps (epsilon) مقدار بسیار کوچک → $1+\text{eps}*e(-1)$
- error
- erf (error function)
- inf → بینهایت

- $\text{real min} \rightarrow 1*10^{-311}$ کوچکترین مقدار
- $\text{real min} \rightarrow 1*10^{+311}$ بزرگترین مقدار
- به جای کاما می توان از فاصله استفاده کرد. $\rightarrow \text{space}$ کاما ,
- رفتن به خط بعد \rightarrow سمی کالن ;
- ایجاد یک رشته از اعداد- ایجاد آرایه \rightarrow کالن یا دونقطه :
- " quote \rightarrow "ورودی"

مثال:

>> x=0:pi/2:2*pi; نرم افزار این خط را اجرا نمی کند.

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

a =

1 2 3

4 5 6

7 8 9

>> a=1:2:10 از ۱ تا ۱۰ با گام ۲

a = 1 3 5 7 9

>> a=1:10

a = 1 2 3 4 5 6 7 8 9 10

معنی رنگ ها در متلب:

فرمان ها ← مشکی

دستورات ← آبی

رشته ها ← بنفش

توضیحات ← سبز



ترتیب حق تقدم: - + > * / \ < ^

■ با پیمودن مسیر زیر می توان برنامه ها را در یک m-file ذخیره کرد و بعد با زدن Run آن را اجرا کرد.

File → New → script

- برای فراخوانی یک m-file ذخیره شده، از طریق پنجره command ، فایل باید از current خوانده شود.
- برای دیدن version متلب از منوی (نوار ابزار) help استفاده می شود.
- برای شناسایی خطاهای برنامه از نرم افزار debug استفاده می شود.
- برای دیدن مثال های حل شده در متلب به help و demo مراجعه می کنیم.

مثال:

```
>> a=[1 2;3 4];
```

```
>> b=[5 6;7 8];
```

```
>> c=b.a
```

Attempt to reference field of non-structure array.



خطا

```
>> c=b*a
```

```
c =
```

```
23 34
```

```
31 46
```

```
>> c=b.*a
```

```
c =
```

```
5 12
```

```
21 32
```

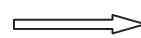
جواب ها متفاوتند

```
>> a=[1 2;3 4];
```

```
>> b=5;
```

```
>> c=b.a
```

Attempt to reference field of non-structure array.



پیغام خطا

```
>> c=b*a
```

```
c =
```

```
5 10
15 20
```

```
>> c=b.*a
```

```
c =
```

```
5 10
15 20
```

جواب ها یکسان

عملگر نقطه:

- ضرب بصورت درایه - درایه → ماتریس * اسکالر
- ضرب ماتریس → ماتریس * ماتریس
- ضرب درایه - درایه → ماتریس .* ماتریس

برخی از انواع فرمت ها در متلب:

```
>> format short
```

عدد را تا ۴ رقم اعشار نشان می دهد.

مثال:

```
>> pi
```

```
ans = 3.1416
```

```
>> format long
```

تا ۱۵ رقم اعشار محاسبه می کند.

```
>> pi
```

```
ans = 3.141592653589793
```

```
>> format rat
```

نشان دهنده اعداد اعشاری به صورت یک تقسیم گویا

```
>> pi
```

```
ans = 355/113
```

```
>> format hex
```

در این فرمت عدد و حروف با هم هستند.

```
>> pi
```

```
ans = 400921fb54442d18
```

```
>> format bank
```

اعداد را گرد می کند.

```
>> pi
```

```
ans = 3.14
```

دستور **vpa** : با این دستور فرمان می دهیم که تا چند رقم اعشار نشان داده شود.

```
>> vpa(pi,11)
```

```
ans = 3.1415926536
```

نمایش عدد موهومی: $(e=a+jb)$

```
>>a= complex(1,3)
```

```
a = 1.0000 + 3.0000i
```

```
>> real(complex(1,3))
```

قسمت حقیقی

```
ans = 1
```

```
>> imag(complex(1,3))
```

قسمت موهومی

```
ans = 3
```

abs: قدر مطلق

```
>> abs(-1)
```

```
ans = 1
```

angle : زاویه $(\tan^{-1} \frac{b}{a})$

atan2 ← زاویه

```
>> a=complex(1,3)
a = 1.0000 + 3.0000i
>> angle(a)
ans = 1.2490
>> atan2(imag(a),real(a))
ans = 1.2490
```

conj : مزدوج

```
>> conj(a)
ans = 1.0000 - 3.0000i
```

exp : تابع نمایی

```
>> exp(x) →  $e^x$ 
```

```
>> atan(x) →  $\tan^{-1}(x)$ 
```

ones(n) : ماتریسی که همه ی درایه های آن برابر با ۱ است.

```
>> ones(3)
ans =
    1.00    1.00    1.00
    1.00    1.00    1.00
    1.00    1.00    1.00
```

eye(n) : ماتریس واحد

```
>> eye(3)
```

```
ans =
```

```
    1.00    0    0
    0    1.00    0
    0    0    1.00
```

zeros(n) : ماتریس صفر

```
>> zeros(3)
```

```
ans =
```

```
    0    0    0
    0    0    0
    0    0    0
```

Magic(n) : مجموع درایه های سطر، ستون و قطر با هم برابرند.

```
>> magic(3)
```

```
ans =
```

```
    8.00    1.00    6.00
    3.00    5.00    7.00
    4.00    9.00    2.00
```

دستور length :

نشان دهنده بیشترین طول یک ماتریس یا بردار

مثال:

```
>> X = [5, 3.4, 72, 28/4, 3.61, 17, 94, 89];
```

```
>> length(X)
```

```
ans = 8
```

دستور `:trace`

مجموع عناصر روی قطر اصلی یک ماتریس را نشان می دهد.

```
>> a=magic(3);
```

```
>> trace(a)
```

```
ans = 15
```

دستور `triu`: ماتریس بالا مثلثی

```
>> triu(ones(4),0)
```

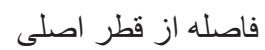
```
ans =
```

```
1 1 1 1
```

```
0 1 1 1
```

```
0 0 1 1
```

```
0 0 0 1
```



فاصله از قطر اصلی

```
>> triu(ones(3),2)
```

```
ans =
```

```
0 0 1
```

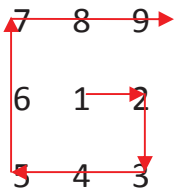
```
0 0 0
```

```
0 0 0
```


معرفی ماتریس حلزونی spiral :

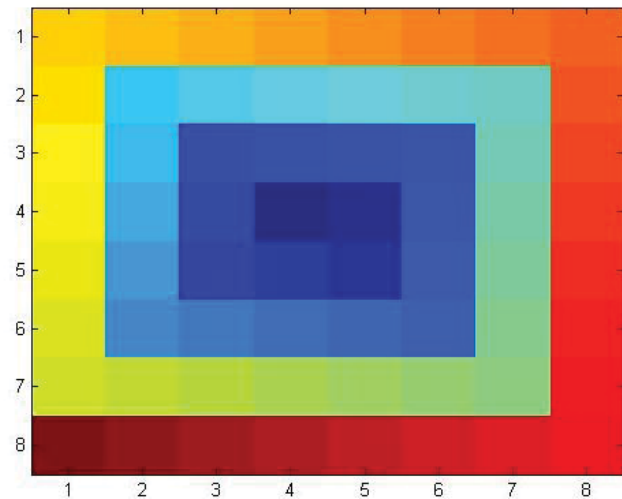
```
>> a=spiral(3)
```

```
a =
```



```
>> a=spiral(8);
```

```
>> image(a)
```



معرفی ماتریس پاسکال - خیام: pascal

```
>> pascal(4)
```

```
ans =
```

```
1 1 1 1
1 2 3 4
1 3 6 10
1 4 10 20
```

دستور **diag**: این دستور روی قطر اصلی ماتریس کار می کند.

```
>> a=1:3;           → ابتدا تعریف آرایه
```

```
>> diag(a)
```

```
ans =
```

```
1 0 0
0 2 0
0 0 3
```

```
>> a=cell(2,2)
```

```
a =
```

```
 [] []
```


```
 [] []
```

```
>> a{1,1}=[1 2;3 4]
```

```
a =
```

```
 [2x2 double] []
```

```
 [] []
```

```
>> a{1,2}='Ali'  رشته یا string
```

```
a =
```

```
 [2x2 double] 'Ali'
```

```
 [] []
```

تعریف سلول در یک خط:

```
>> a={ [1 2,3 4] 'Ali';85 2}
```

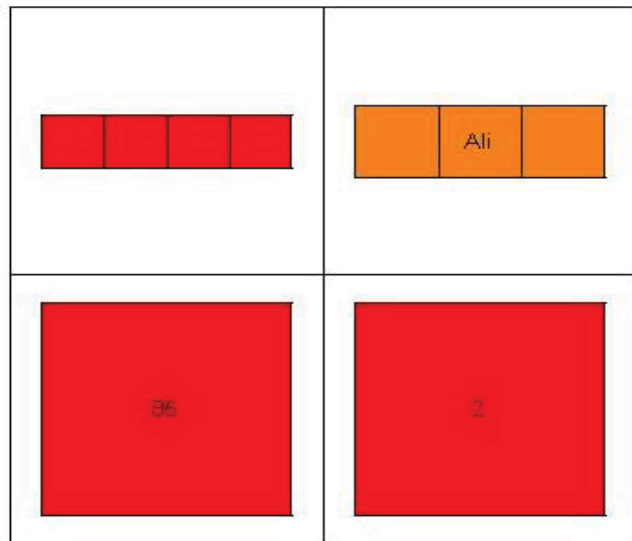
```
a =
```

```
 [1x4 double] 'Ali'
```

```
 [ 85] [ 2]
```

دستور رسم سلولی: `cellplot` و `celldisp`

```
>> a=cell(2,2);
>> a={1 2,3 4 'Ali';85 2}
>> cellplot(a)
```



دستور مفید `repmat`: این دستور می تواند قسمتی از یک ماتریس را به آن بچسباند.

`repmat(a,m,n)`

```
>> a=[1 2 3 4];
>> b=repmat(a,2,3)
```

b =

```
1 2 3 4 1 2 3 4 1 2 3 4
1 2 3 4 1 2 3 4 1 2 3 4
```

اجتماع دو مجموعه: $(A \cup B)$

```
>> setdiff(A,B)
```

اشتراک دو مجموعه: $(A \cap B)$

```
>> intersect(A,B)
```

تفاضل متقارن دو مجموعه: $(A \Delta B)$

```
>> setxor(A,B)
```

تفاضل دو مجموعه: (A-B)

```
>> setdiff(A,B)
```

برخی تبدیل های مهم و دستوره های مربوط به آنها:

 $\sqrt{\sinh(0.125)}$ →

```
>> sqrt(sinh(0.125))
```

 $\cos(35^\circ)$ →

```
>> cosd(35)
```

 $\sqrt[6]{\ln(125)}$ →

```
>> nthroot(log(125),6)
```

 $\binom{10}{3}$ →

```
>> nchoosek(10,3)
```

 $\tan 17^\circ$ →

```
>>[tand(17)]
```

 $\log_7 23$ →

```
>> log(23)/log(7)
```

قضیه جردن برای ساده کردن ماتریس به صورت قطری:

```
>> a=magic(3);
```

```
>> b=jordan(a)
```

b =

15.0000	0	0
0	4.8990	0
0	0	-4.8990

با این روش می توان به راحتی دترمینان ماتریس را محاسبه کرد.

■ با زدن کلید tab در کنار دستور نوشته شده help کوچکی باز می شود.

ماتریس تصادفی n در m : (rand و randn)

>> rand(2) \Rightarrow اعداد تصادفی بین [0,1]

ans =

0.0573 0.7962

0.6295 0.6912

>> randn(2) \Rightarrow اعداد تصادفی بین [-2,2]

ans =

-0.6126 0.0723

2.2444 0.8655

مثال:

ماتریس تصادفی 3*3 بین (2,5)

>> a=2+3*rand(3)

a =

2.2208	4.7426	2.4555
2.6211	4.3477	4.5437
4.3251	2.8866	4.3546

ابتدای بازه

5-2

دستگاه معادلات مجهول:

$$\begin{cases} 2x+3y-5z=12 \\ -x-y+3z=25 \\ y+z=70 \end{cases}$$

$$[A][x]=[B]$$

$$\bar{A}\bar{x} = \bar{B} \rightarrow \bar{x} = \frac{\bar{B}}{\bar{A}} = \bar{A}^{-1} * \bar{B}$$

```
>> a=[2 3 -5;-1 -1 3;0 1 1];
```

```
>> b=[12;25;70];
```

```
>> x=inv(a)*b
```

Warning: Matrix is singular to working precision.

```
x =
```

```
Inf
```

```
Inf
```

```
Inf
```

```
>> linsolve(a,b)
```

ضرایب

مجهولات

معکوس ماتریس A یا A^{-1} : $inv(A)$

توابع گرد کردن:

```
>> a=2.15692;
```

```
>> fix(a)
```

```
ans = 2
```

```
>> round(a)
```

```
ans = 2
```

```
>> floor(a) → گرد به سمت پایین
```

```
ans = 2
```

```
>> ceil(a) → گرد به سمت بالا
```

```
ans = 3
```

مثال:

```
>> sqrt(floor(abs(-4.7))) ans = 2
```

بزرگترین مقسوم علیه مشترک:

```
>> gcd(x,y) → ب . م . م
```

```
>> gcd(2,10)          ans = 2
```

کوچک ترین مضرب مشترک:

```
>>>> lcm(x,y)      → ک . م . م
```

```
>> lcm(2,10)       ans = 10
```

```
>> rem(x,y) = mod(x,y)    باقیمانده ۲ عدد x و y
```

■ متلب log را فقط با مبنای ۱۰ و ۲ می گیرد.

```
>> log10(10)      → گرفتن log
```

```
ans = 1
```

```
>> log(10)        → گرفتن ln
```

```
ans = 2.3026
```

```
>> log2(8)
```

```
ans = 3
```

```
>> log(100)/log(5)
```

```
ans = 2.8614
```

$$\log_5 100 = \frac{\log 100}{\log 5}$$

تابع علامت: **sign(x)**

دستور **disp** : نشان دهنده ورودی و رشته ها است.

```
>>disp('ورودی')
```

دستور **input** :

```
>> a=input('enter a= ')
```

```
enter a=
```

دستور **pause** : توقف در حین اجرا

مثال: برنامه ای بنویسید که یک عدد را از کاربر بگیرد و آن را در متغیری مانند x ذخیره کند. با استفاده از آن عبارت زیر را محاسبه کند و مقدار y را با پیغام مناسب نمایش دهد.

$$y = x^3 + 3x^2 + 6x + 6$$

file/new/scrip

1. `x=input('enter x = ');`
2. `y=x^3+3*x^2+6*x+6;`
3. `disp('result')`
4. `disp(y)`

دستور linspace:

```
>> a=linspace(0,10,5)
```

```
a =      0    2.5000    5.0000    7.5000   10.0000
```

بین ۰ و ۱۰، پنج عدد با فاصله یکسان ایجاد می کند.

دستور logspace:

```
>> a=logspace(1,3,3)
```

```
a =      10      100     1000
```

چون log دارد پس به معنی $10^1, 10^2, 10^3$

مثال: برنامه ای بنویسید که عدد صحیح n را از ورودی بگیرد و برداری 100 عنصری بین 0 و $2n\pi$ ایجاد نموده و در متغیر x قرار دهد. سپس مقادیر y را از رابطه زیر محاسبه کرده و نمایش دهد.

$$y = |\sin(x)| * x^2$$

file/ new/ scrip

1. `n=input('enter n= ');`
2. `x=linspace(0,2*n*pi,100);`
3. `y=abs(sin(x)).*(x.^2)`

دستور strcmp:

دستور برای برابری رشته ها

دستور char:

تابعی برای تبدیل متغیر از نوع double به رشته

دستور `mat2str` :

دستوری برای تبدیل ماتریس از اعداد به رشته

دستور `eval` :

دستوری برای اجرای فرمانی از متلب که به صورت رشته وارد شده

برای نمایش کد اسکی یک رشته:

```
>> abs  
>> nthroot
```

■ این برنامه رشته `hello` را از آخر به اول نشان می دهد.

```
>> s='hello';  
>> num=s(end:-1:1);  
>> disp(num)  
olleh
```

■ $\text{Sin}(\pi/6) \rightarrow \text{sind}(30)$

رادیان

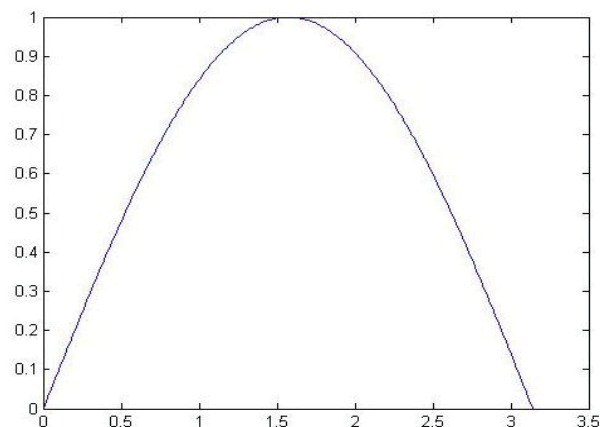
درجه

رسم نمودار:

دستورات متعددی برای رسم نمودار وجود دارد.

دستور `plot` : دستور `plot(x,y)` نمودار `y` را بر حسب `x` رسم می کند.

```
>> x=0:0.01:3.14;  
>> y=sin(x);  
>> plot(x,y)
```



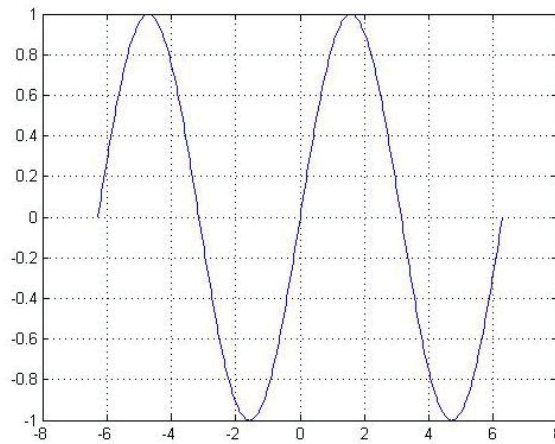
برنامه فوق نمودار `y=sin(x)` را در بازه `[0,3.14]` رسم می کند.

■ نقاط داده شده گسسته هستند اما تابع plot با اتصال نقاط رسم شده به یکدیگر یک نمودار پیوسته را نشان می دهد.

مثال:

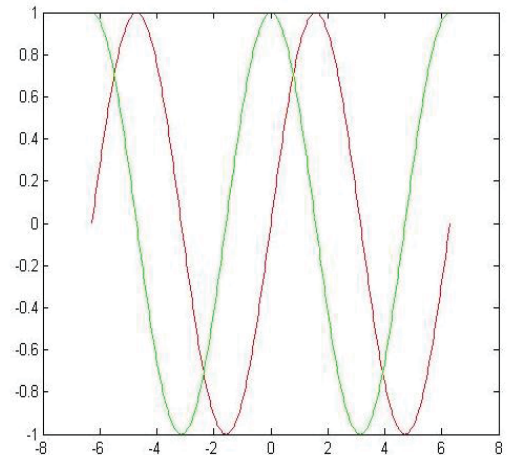
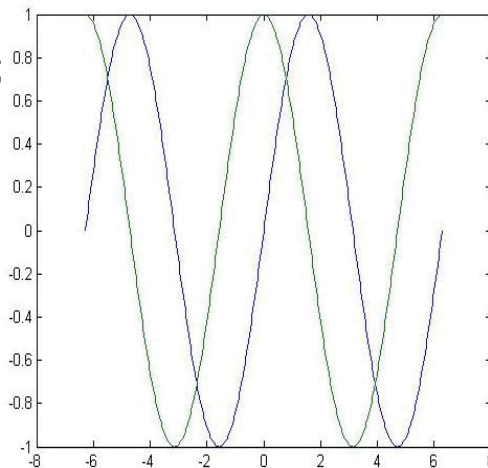
```
>> x=[-2*pi:pi/100:2*pi];
>> y=sin(x);
>> plot(x,y)
>> grid on
```

تابع رسم
شبكة بندی



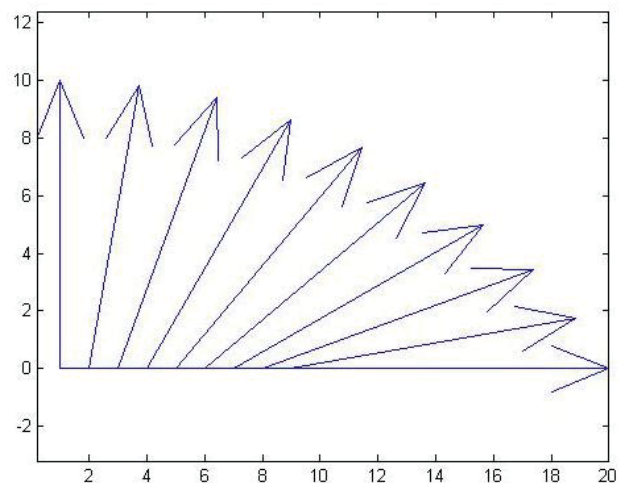
```
>> x=[-2*pi:pi/100:2*pi];
>> y1=sin(x);
>> y2=cos(x);
>> plot(x,y1,x,y2)
>> plot(x,y1,'r',x,y2,'g')
```

رنگ خطوط



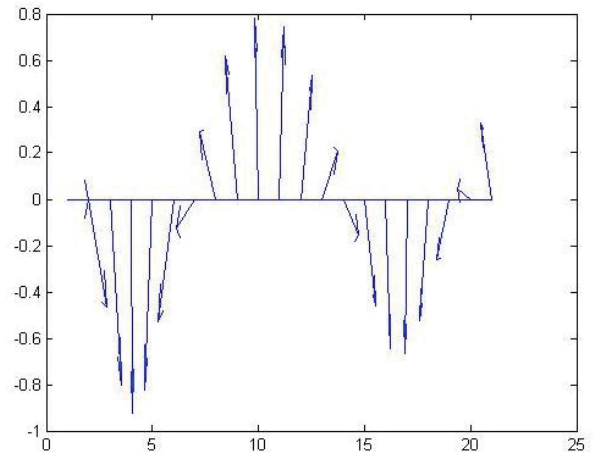
دستور رسم نمودار میدانی و برداری: (feather)

```
>> theta=90:-10:0;
>> r=ones(size(theta));
>> [u,v]=pol2cart(theta*pi/180,r*10);
>> feather(u,v)
>> axis equal
```



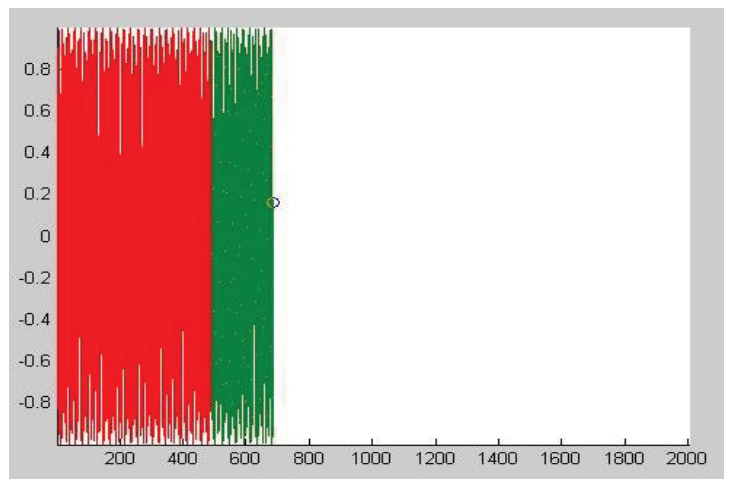
اگر ورودی مختلط داشته باشیم:

```
>> t=0:0.5:10;
>> s=0.05+i;
>> z=exp(-s*t)
>> feather(z)
```



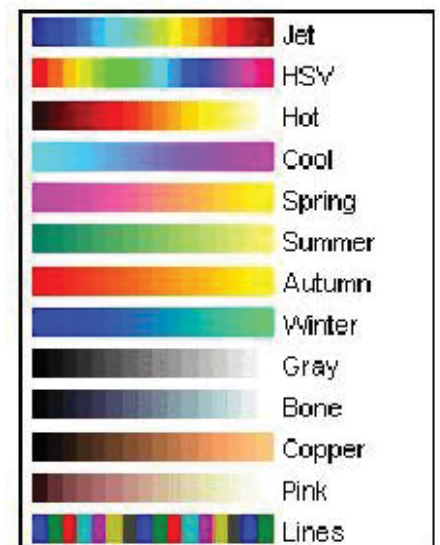
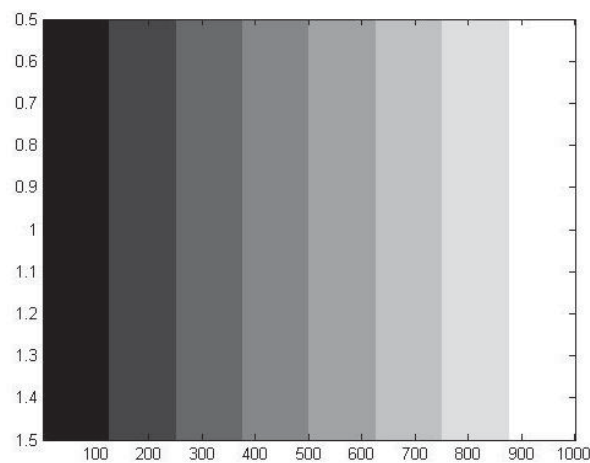
دستور رسم همراه با نمایش گرافیکی: (comet)

```
>> t=-1000:1000;
>> x=sin(t);
>> comet(x)
```

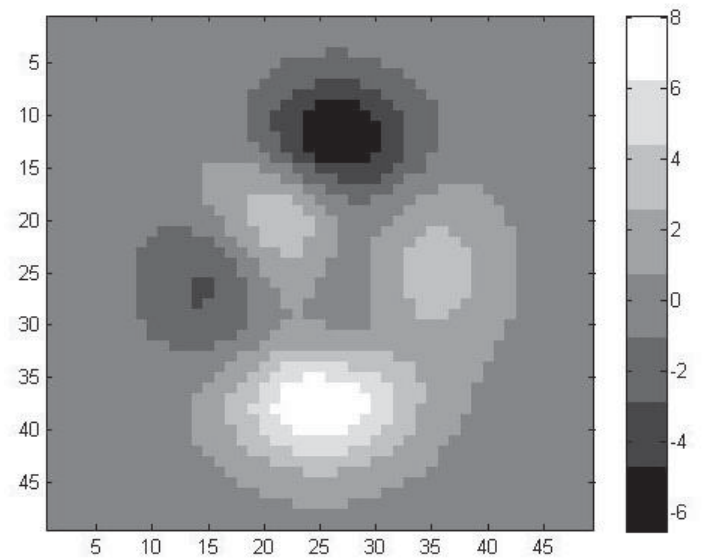


معرفی جعبه ابزار رنگ: colormap

```
>> m=gray(8);
>> colormap(m)
>> imagesc(1:1000)
```

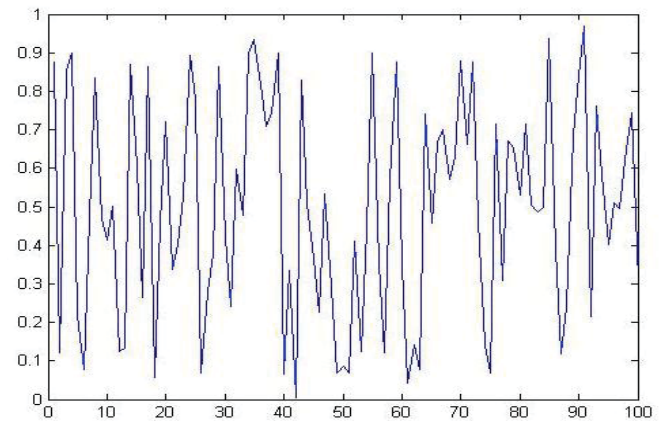


```
>> z=peaks;
>> colormap(gray(8))
>> imagesc(z)
>> colorbar
```



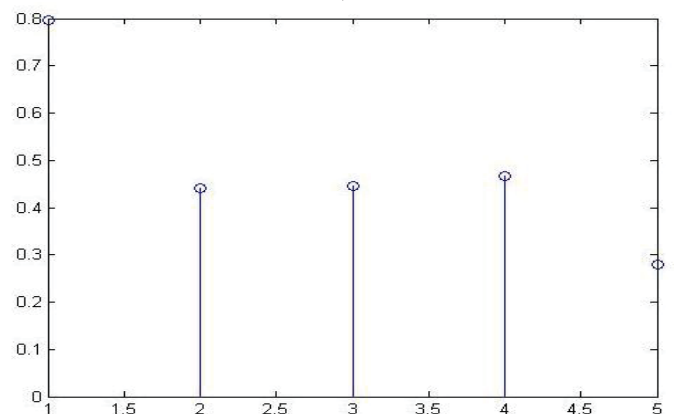
ایجاد یک سیگنال درهم:

```
>> plot(rand(100,1))
```



دستور stem : برای رسم نمودارهای گسسته

```
>> y=rand(1,5);
>> stem(y)
```



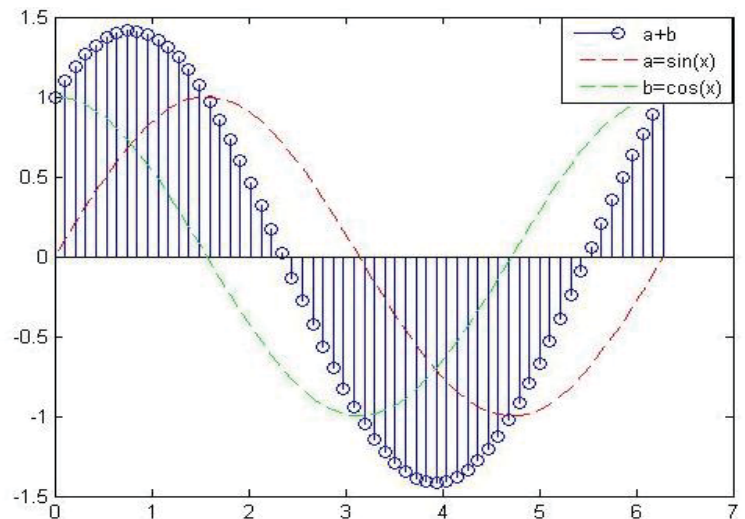
تمرین: فرض کنید می خواهیم دو نمودار a و b را با هم جمع کنیم.

```
>> x=linspace(0,2*pi,60);
>> a=sin(x); b=cos(x);
>> stem_handles=stem(x,a+b)
>> hold on
```

```
stem_handles =
```

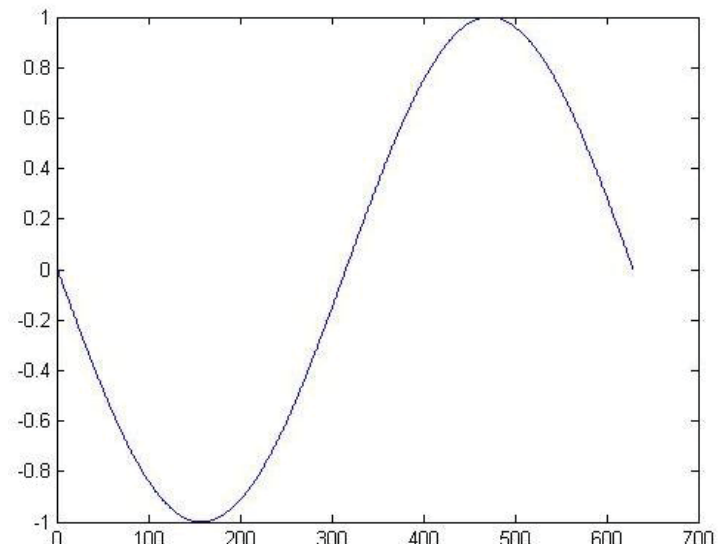
```
174.0138
```

```
>> plot_handles=plot(x,a,'--r',x,b,'--g');
>> hold off
>> legend_handles=[stem_handles(1);plot_handles];
>> legend(legend_handles,'a+b','a=sin(x)','b=cos(x)')
```



تمرین: می خواهیم مینیمم و ماکزیمم یک تابع مثل $\sin(x)$ را بیابیم.

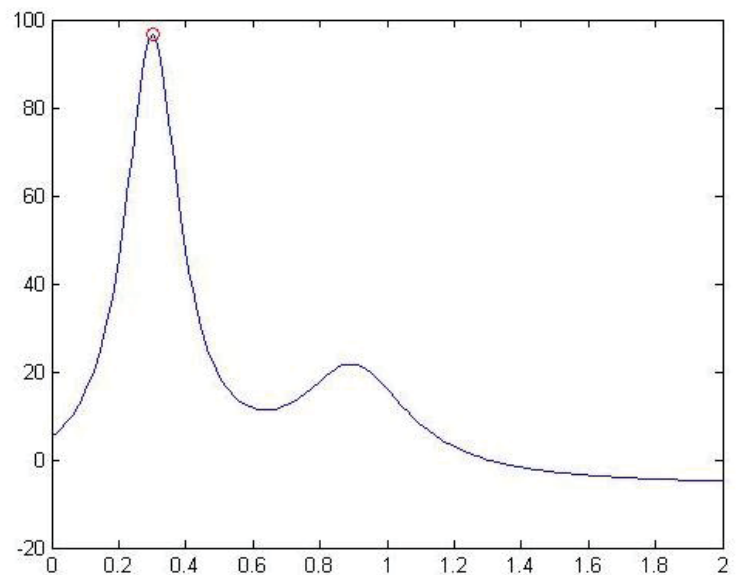
```
>> x=-pi:.01:pi;
>> f=sin(x);
>> plot(f)
>> max=max(f)
max = 1.0000
>> min=min(f)
min = -1.0000
```



دستور **tic/toc**: اگر در ابتدای برنامه **tic** و در انتهای آن **toc** بنویسیم زمان اجرای برنامه نشان داده می شود.

نمایش نقاط ماکزیمم و مینیمم:

```
>> x=0:.01:2;
>> y=humps(x);
>> plot(x,y)
>> [v,ind]=max(y)
v = 96.5000
ind = 31
>> hold on
>> plot(x(ind),y(ind),'ro');
>> x(ind)
ans = 0.3000
>> y(ind)
ans = 96.5000
```



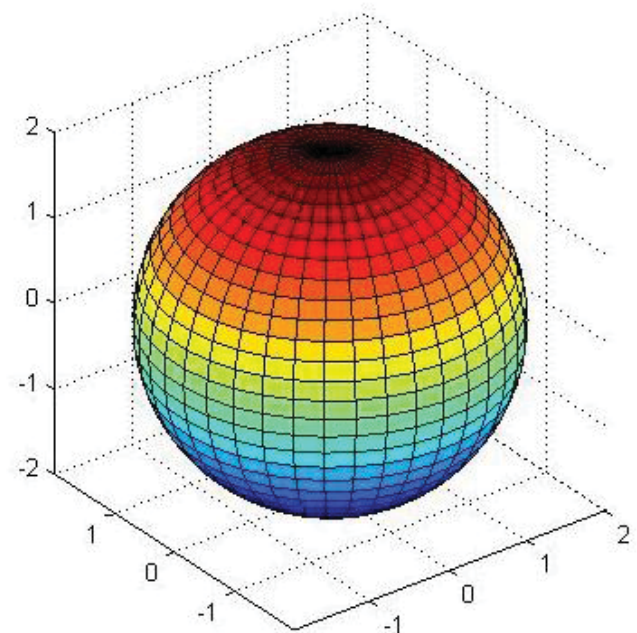
دستور **type**: برای مشاهده source برنامه از این دستور استفاده می شود.

```
>> type humps
```

تمرین: رسم یک کره

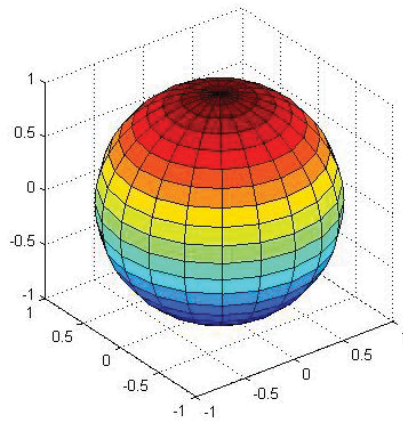
$$\begin{cases} X= 2\cos(v)\cos(u) & 0 < u < 2\pi \\ Y= 2\cos(v)\sin(u) & -\pi/2 < v < \pi/2 \\ Z= 2\sin(v) \end{cases}$$

```
>> u=linspace(0,2*pi,40);
>> v=linspace(-pi/2,pi/2,31);
>> [u,v]=meshgrid(u,v);
>> x=2.*cos(v).*cos(u);
>> y=2.*cos(v).*sin(u);
>> z=2.*sin(v);
>> surf(x,y,z)
>> axis image
```

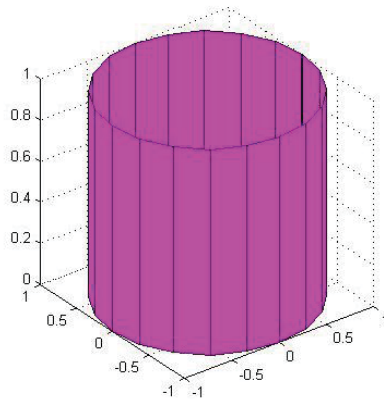


رسم کره با دستور `sphere` :

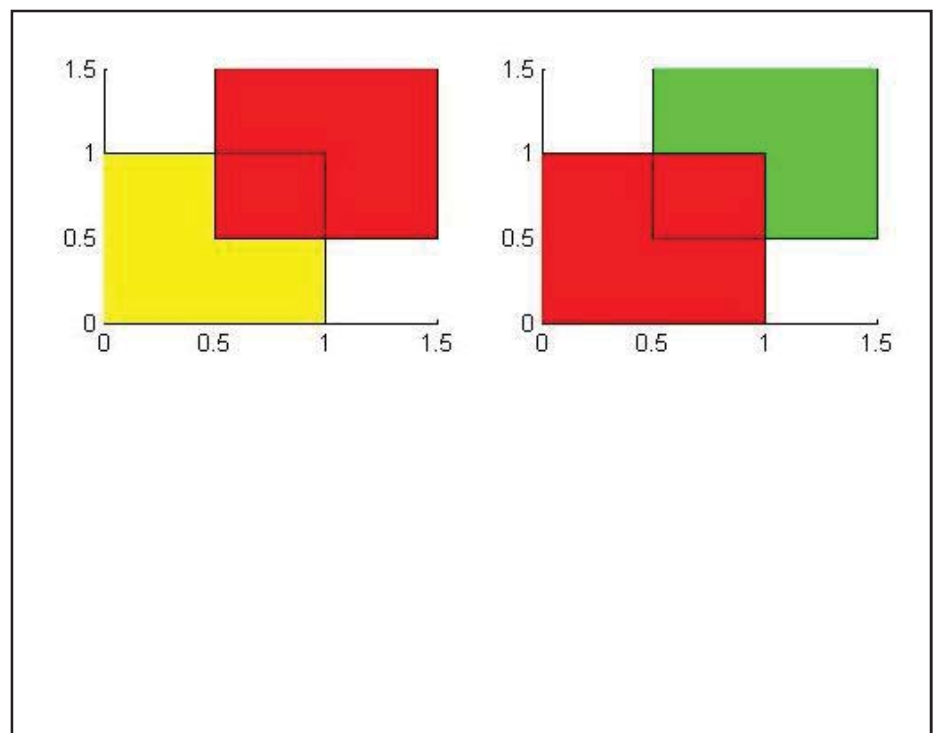
```
>> sphere
>> axis image
```

رسم استوانه با دستور `cylinder` :

```
>> cylinder
>> axis square
>> colormap(spring)
```

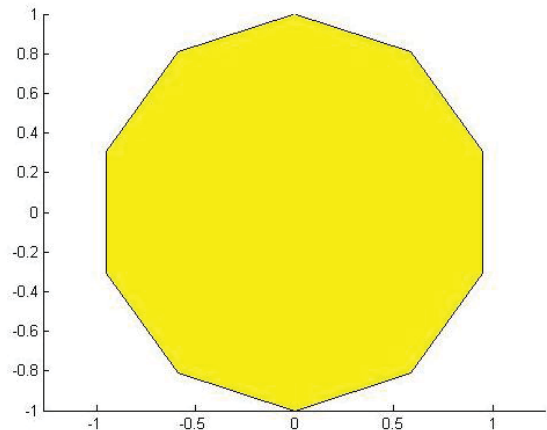
دستور رسم نواحی روی هم `patch` :

```
>> x1=[0 1 1 0];
>> y1=[0 0 1 1];
>> x2=x1+0.5;
>> y2=y1+0.5;
>> clf
>> subplot(221)
>> patch(x1,y1,'y')
>> patch(x2,y2,'r')
>> subplot 222
>> patch(x2,y2,'g')
>> patch(x1,y1,'r')
```



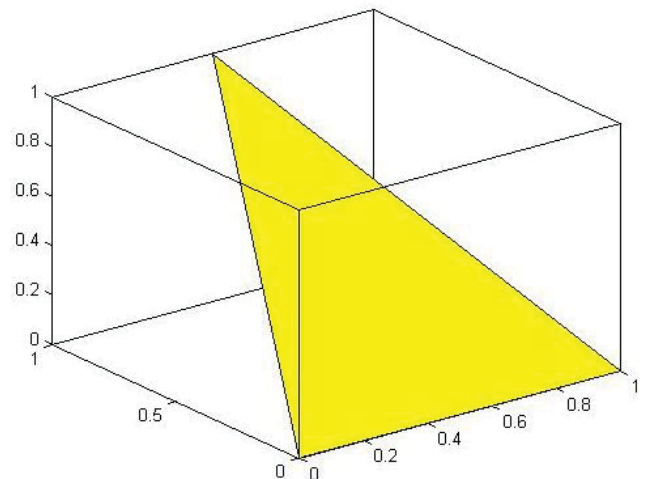
تولید چندضلعی

```
>> t=0:pi/5:2*pi;
>> figure
>> patch(sin(t),cos(t),'y')
>> axis equal
```



تمرین: دستورات زیر را اجرا کرده و نتیجه را بررسی کنید.

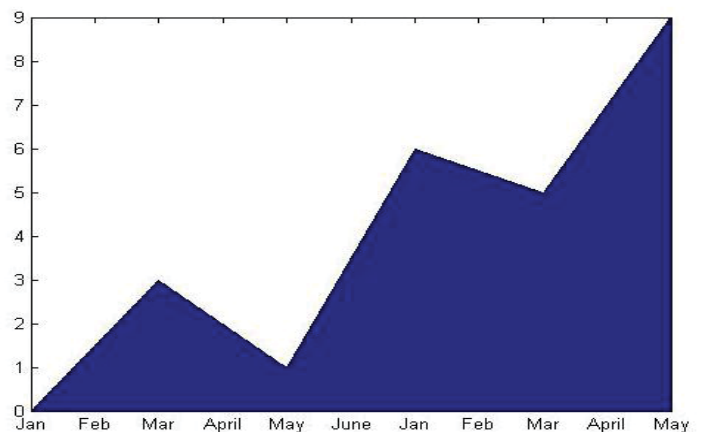
```
>> xt=[0 1 0.5];
>> yt=[0 0 1];
>> zt=[0 0 1];
>> patch(xt,yt,zt,'y')
>> patch(xt,yt,zt,'y')
>> view(3)
>> box
```



نکته: fillm و fill3m را با patch مقایسه کنید.

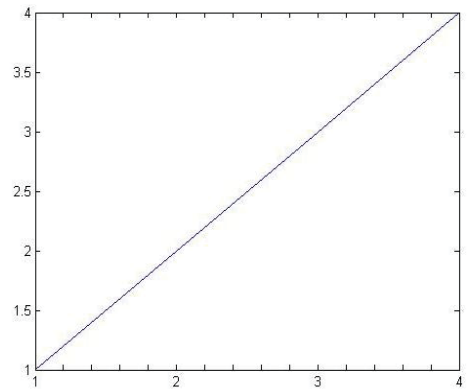
کنترل محورهای رسم: XDir ، XTick ، XTicklabel

```
>> y=[0 3 1 6 5 9];
>> area(y)
>> str='Jan|Feb|Mar|April|May|June';
>> set(gca,'XTicklabel',str)
```



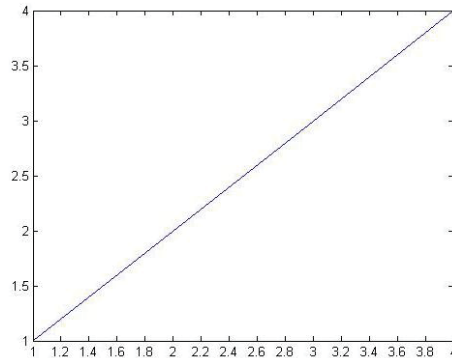

```
>> plot(1:4)
```

```
>> set(gca,'XTick',1:0.2:4,'XTicklabel','1|||||2|||||3|||||4')
```



```
>> plot(1:4)
```

```
>> set(gca,'XTick',1:0.2:4)
```



یک مثال دیگر:

```
>> x=linspace(0,pi,99);
```

```
>> plot(x,sin(x))
```

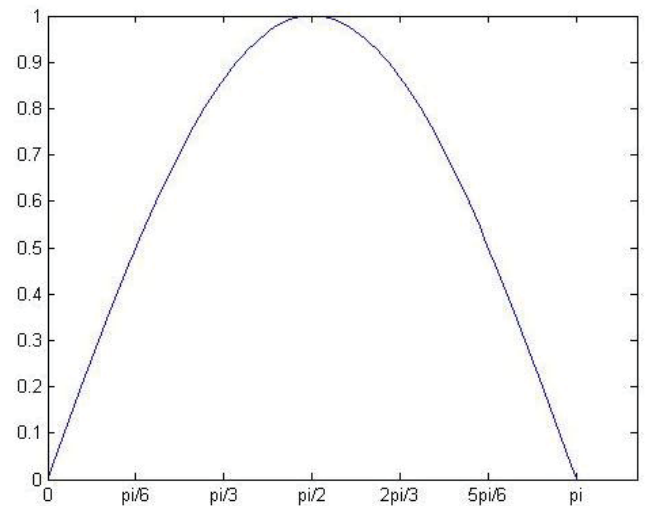
```
>> xt=0:pi/6:pi;
```

```
>> set(gca,'XTick',xt)
```

```
>> axis tight
```

```
>> tikstr={'0','pi/6','pi/3','pi/2','2pi/3','5pi/6','pi'};
```

```
>> set(gca,'XTicklabel',tikstr)
```



مختصری درباره نوشتن در نمودارها:

توجه کنید که نحوه نوشتن فرمول ها در متلب با روش زیر همان روش Latex یا Zpersian می باشد. برای فعال کردن زبان Latex در متلب از `\` استفاده می شود.

```
>>text(x,y,'string')
```

```
>>gtext('string')
```

```
>>title('string')
```

```
 $\alpha=30^\circ \rightarrow \backslash\alpha=30\backslash\text{circ}$ 
```

```
>>gtext(\alpha=30\circ)
```

```
 $\cos(t_{i,j}^{2m+1}) \rightarrow \cos(t\{i,j\}^{2m+1})$ 
```

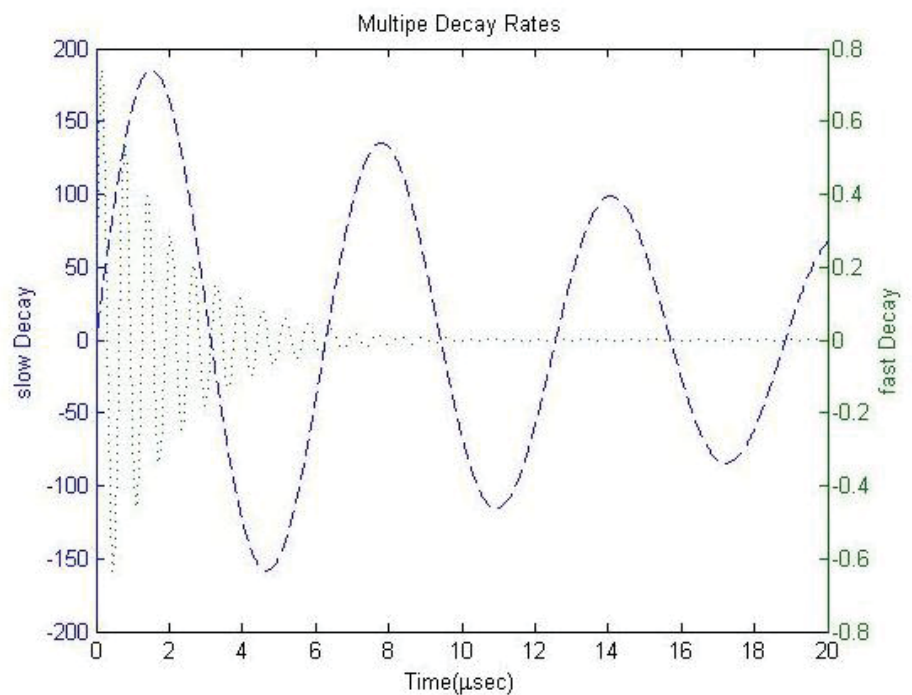
$y \rightarrow \pm\infty \rightarrow y \rightarrow \pm\infty$

equation: \rightarrow \it equation

دستور رسم در هر دو طرف محور x ها:

دستور `plotyy`:

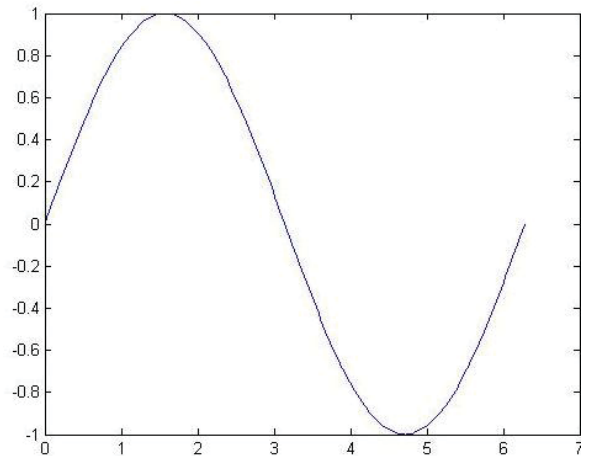
```
>> x=0:0.01:20;
>> y1=200*exp(-0.05*x).*sin(x);
>> y2=0.8*exp(-0.5*x).*sin(10*x);
>> [A,H1,H2]=plotyy(x,y1,x,y2,'plot')
A = 173.0067 175.0068
H1 = 174.0083
H2 = 176.0073
>> set(get(A(1),'ylabel'),'string','slow Decay')
>> set(get(A(2),'ylabel'),'string','fast Decay')
>> xlabel('Time(\musec)')
>> title('Multiple Decay Rates')
>> set(H1,'linestyle','--')
>> set(H2,'linestyle',':')
```



تمرین: می خواهیم یکبار دیگر max و min را بیابیم اما روی نمودار.

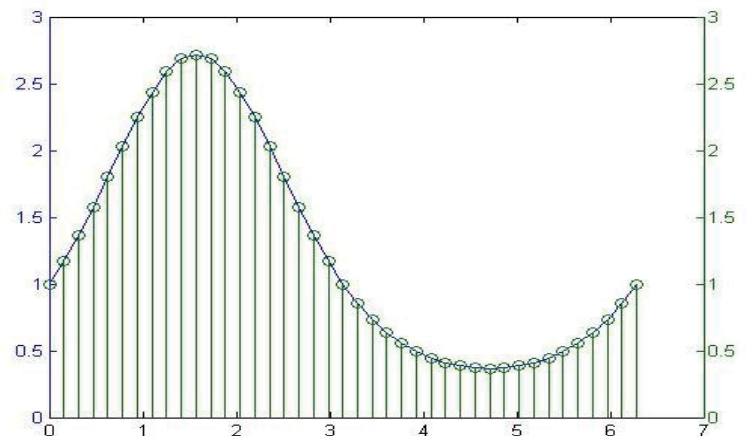
```
>> x=0:pi/100:2*pi;
>> y=sin(x);
>> plot(x,y)
>> ymin=find(max(y)==y)
>> ymax=find(min(y)==y)
X(ymin)      ans =  1.5708
Y(ymin)      ans =  1
```

```
ymin =  51
ymax =  51
```



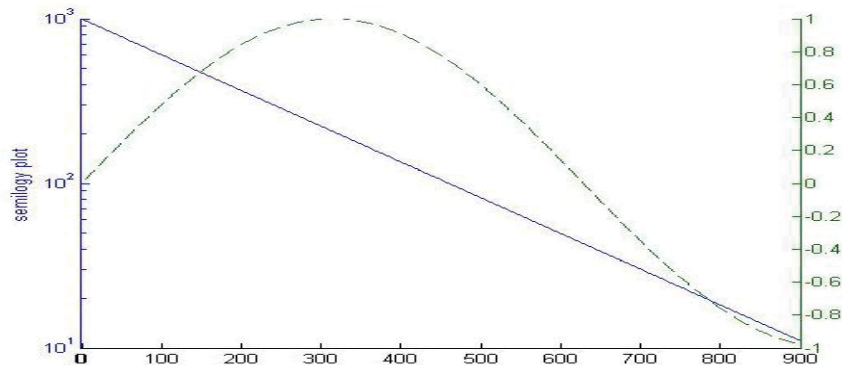
مثال:

```
>> x=0:pi/100:2*pi;
>> y=sin(x);
>> plot(x,y)
>> ymin=find(max(y)==y)
```



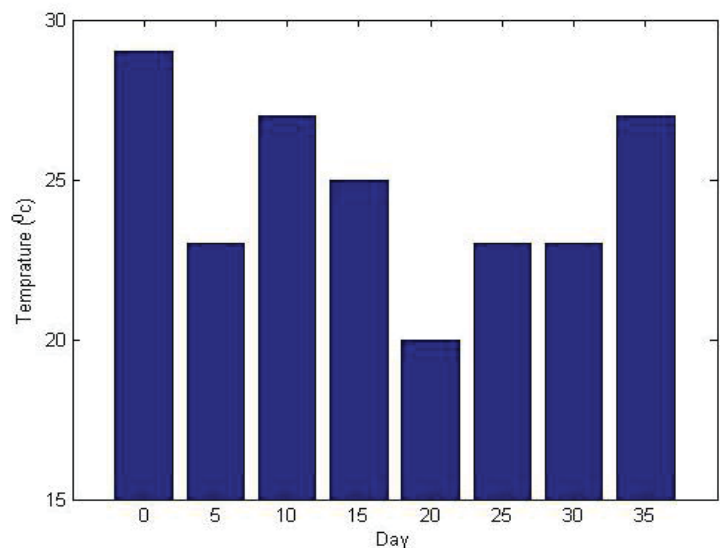
مثال:

```
>> t=0:900; A=1000; a=0.005; b=0.005;
>> z1=A*exp(-a*t);
>> z2=sin(b*t);
>> [haxes,hline1,hline2]=plotyy(t,z1,t,z2,'semilogy','plot');
>> ylabel('semilogy plot')
>> set(hline2,'linestyle','--')
```



تمرین: می خواهیم خطا را روی نموداری آماری نشان دهیم، نمودار دمای هوا- آلودگی هوا- ذرات معلق

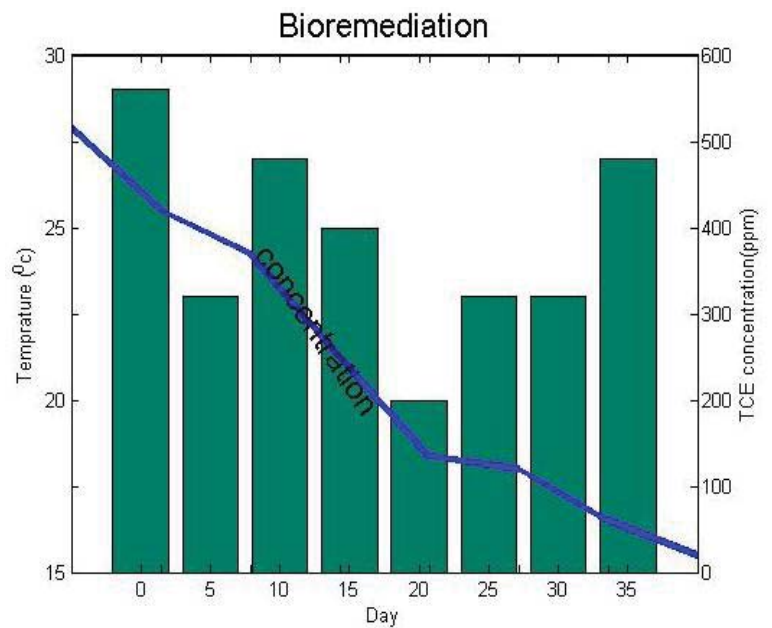
```
>> temp=[29 23 27 25 20 23 23 27];
>> days=0:5:35;
>> colormap summer
>> bar(days,temp)
>> xlabel('Day')
>> ylabel('Temperature (^{0}c)')
>> set(gca,'ylim',[15 30],'layer','top')
```



به عنوان مثال اگر وجود ذرات غبار موجود در هوا برای دماهای مختلف (TCE) باشد. داریم:

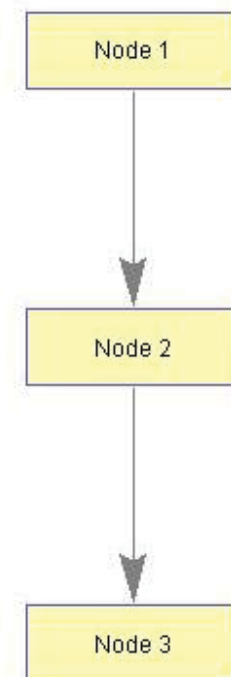
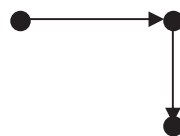
```
>> TCE=[515 420 370 250 135 120 60 20];
>> h2=axes('position',get(gca,'position'))
h2 = 179.0089
>> plot(days,TCE,'linewidth',3)
>> set(h2,'YAxislocation','right','color','none','XTicklabel',[])
>>set(h2, 'xlim',get(h2,'xlim'), 'layer','top')
>>text(11,380,'concentration','Rotation',-55,'FontSize',16)
```

```
>>ylabel('TCE concentration (ppm)')
>>title('Bioremediation','FontSize',16)
>>set(gcf,'PaperPositionMode','auto')
```



رسم گراف ها: biograph

```
      1 2 3
1  ( 0 1 0 )
2  ( 0 0 1 )
3  ( 0 0 0 )
```



```
>> cm=[0 1 0;0 0 1;0 0 0]
```

cm =

```
0 1 0
0 0 1
0 0 0
```

```
>> t=biograph(cm)
```

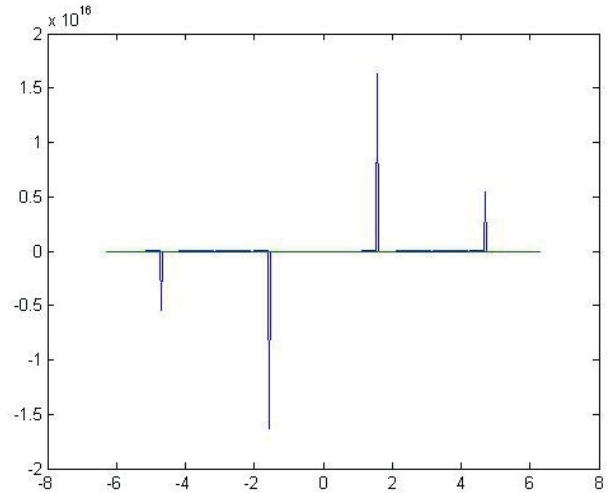
Biograph object with 3 nodes and 2 edges.

```
>> t.view  $\Rightarrow$  گراف را نشان می دهد
```

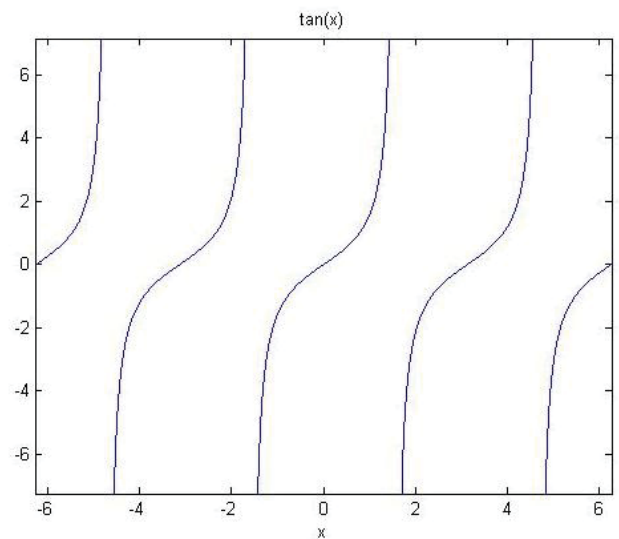
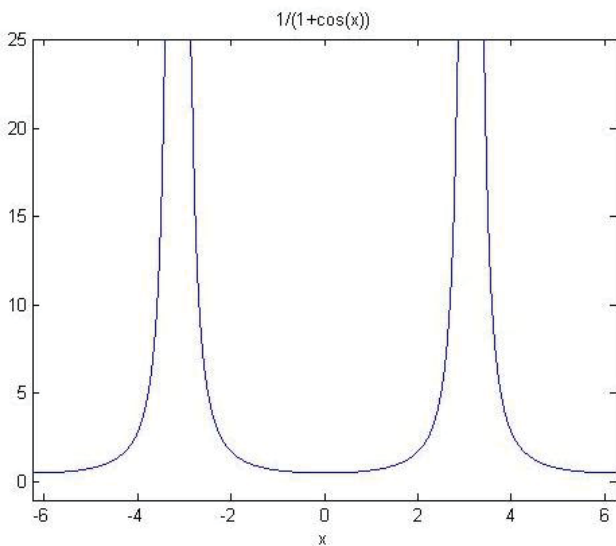
```
>> isdag(t)  $\Rightarrow$  تعداد حلقه ها را در صورت وجود نشان می دهد.
```

مثال: در این مثال چون دو تابع داریم نمودار را در figure نمایش می دهیم.

```
>> x=[-2*pi:pi/100:2*pi];
>> y1=tan(x);
>> y2=1./(1+cos(x));
>> plot(x,y1,x,y2)
>> figure
>> ezplot('tan(x)')
>> figure
```

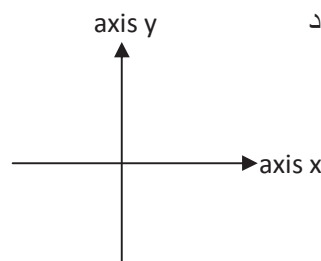


>> ezplot('1./(1+cos(x))') \implies برای رسم راحت تر منحنی هایی که مجانب دارند به کار می رود.



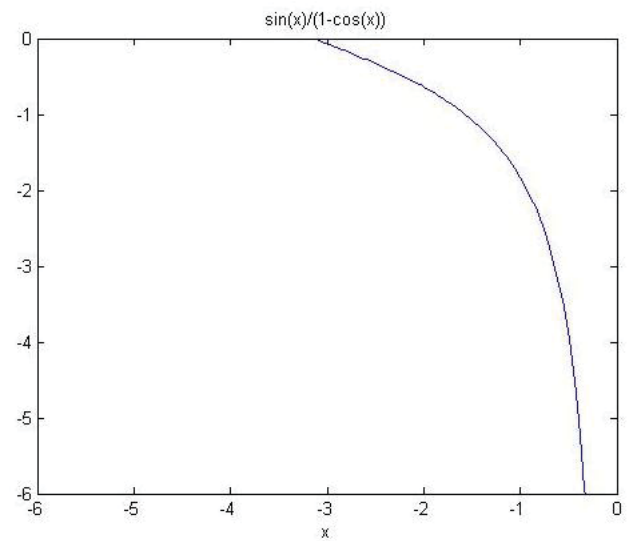
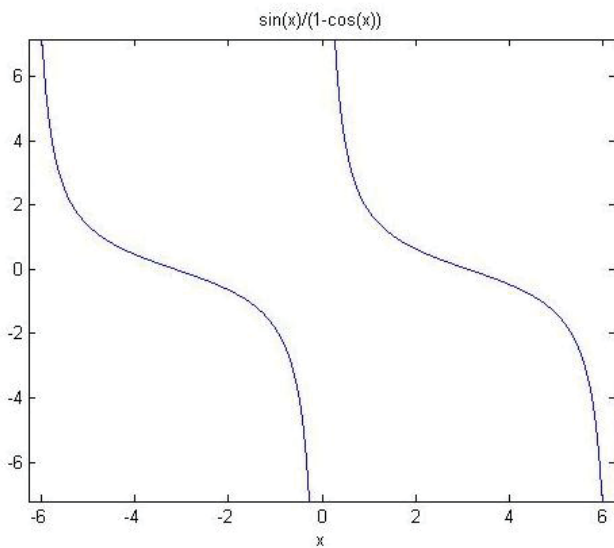
دستور axis (محور): تعیین دامنه و برد

```
axis([x_min x_max y_min y_max])
```



مثال:

```
>> ezplot('sin(x)/(1-cos(x))')
>> axis([-6 0 -6 0])
```



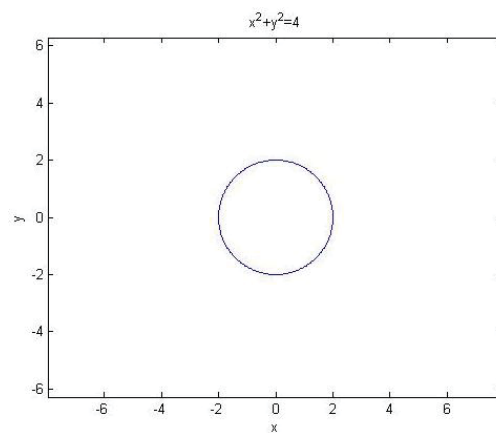
دستور `axis equal` : x و y را برابر هم قرار می دهد.

دستور `ezplot` :

$$x^2+y^2=4$$

```
>> ezplot('x^2+y^2=4')
```

```
>> axis equal
```

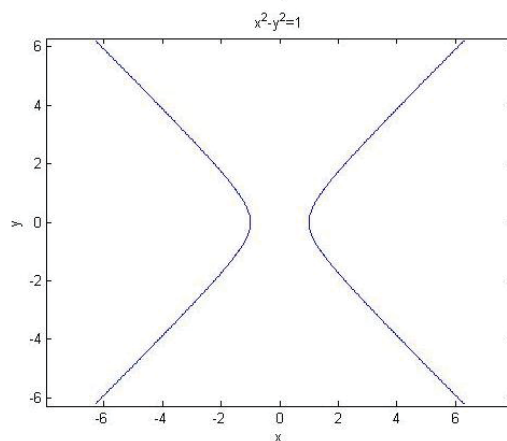


مثال

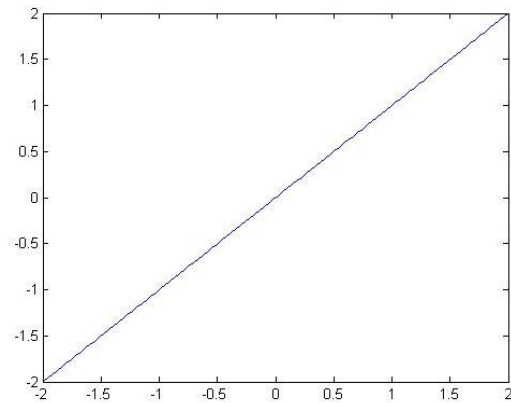
$$x^2-y^2=1 \rightarrow \text{هذلولی}$$

```
>> ezplot('x^2-y^2=1')
```

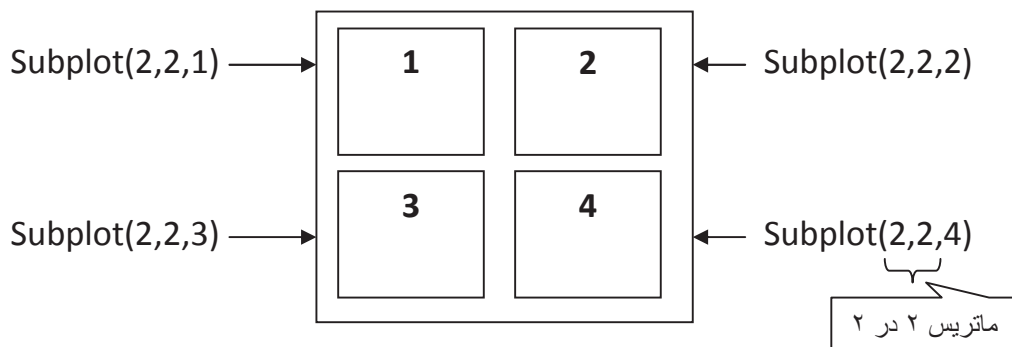
```
>> axis equal
```



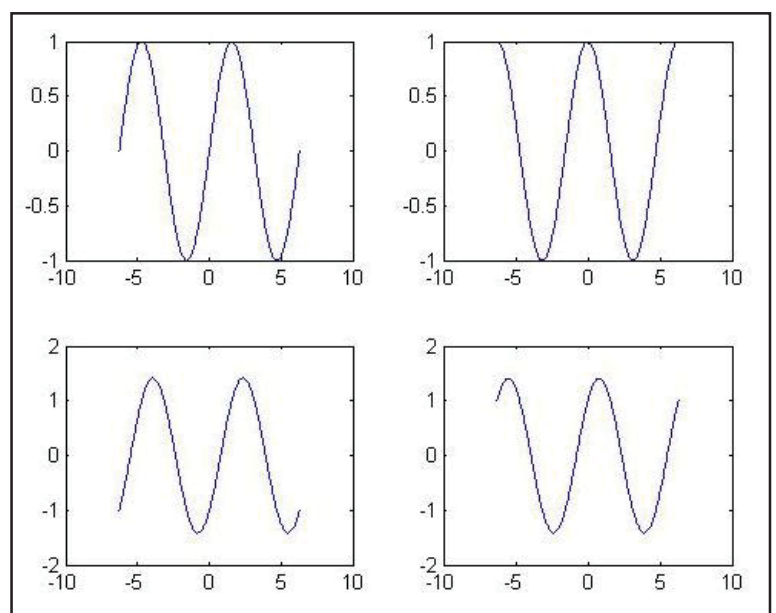
```
>> x=linspace(-2,2,100);
>> y=linspace(-2,2,100);
>> plot(x,y)
```

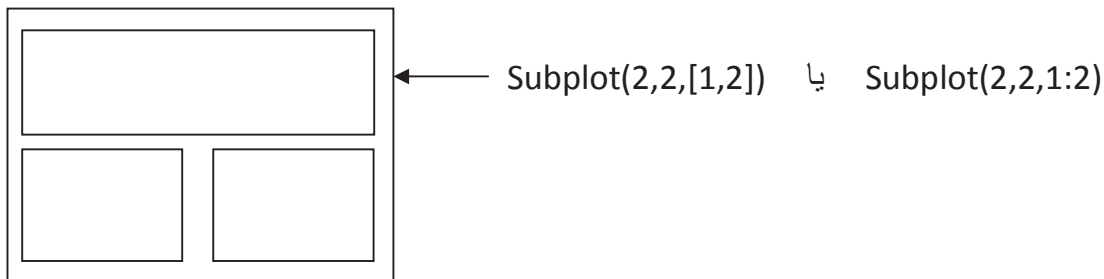


دستور **subplot** : با استفاده از این دستور می توان چند نمودار را به طور همزمان در یک figure مشاهده کرد.



```
>> x=[-2*pi:pi/100:2*pi];
>> subplot 221 یا subplot(2,2,1)
>> y1=sin(x);
>> plot(x,y1)
>> subplot 222
>> y2=cos(x);
>> plot(x,y2)
>> subplot 223
>> plot(x,y1-y2)
>> subplot 224
>> plot(x,y1+y2)
```





') و xlabel(') : با این دستور می توان نام محور و نمودار را عوض کرد.

') title(' : با این دستور می توان برای نمودارها عنوان گذاشت.

' رشته ') gtext(' : با این دستور می توان مکان متن را به دلخواه تغییر داد.

' رشته ') text(x , y , ' : با این دستور مختصات مکان متن نیز داده می شود.

' رشته ۱ ' , ' رشته ۲ ') legend(' : با این دستور جعبه راهنما در کنار نمودار ساخته می شود.

مثال:

```
>> x=0:pi/100:2*pi;
```

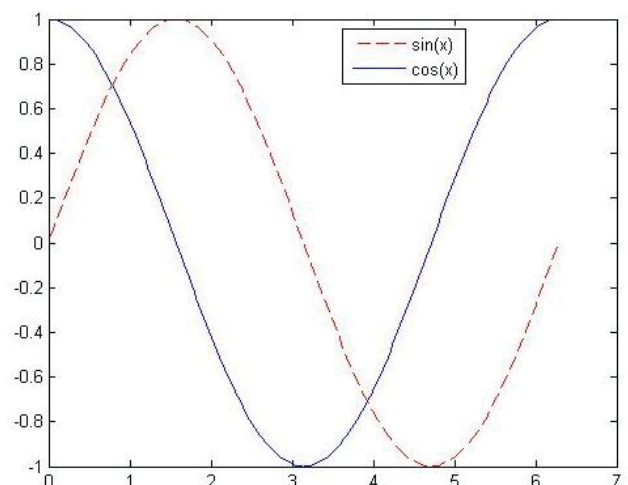
```
>> plot(x,sin(x),'-r',x,cos(x),'b')
```

```
>> legend('sin(x)','cos(x) ',2)
```

با تغییر این عدد می توان مکان legend را تغییر

```
>> legend('sin(x)','cos(x)',-1)
```

مکان legend در بیرون نمودار



نمودارهای آماری

۱- میله ای : **bar-bar3-barh**

۲- هیستوگرام (افقی - عمودی) : **hist- hist3**

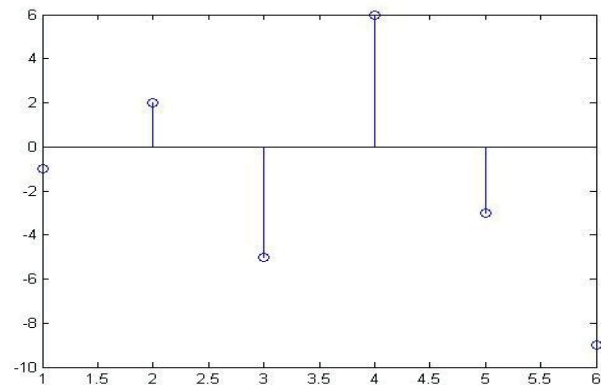
۳- دایره ای : **pie – pie3**

۴- پله ای : **stairs**

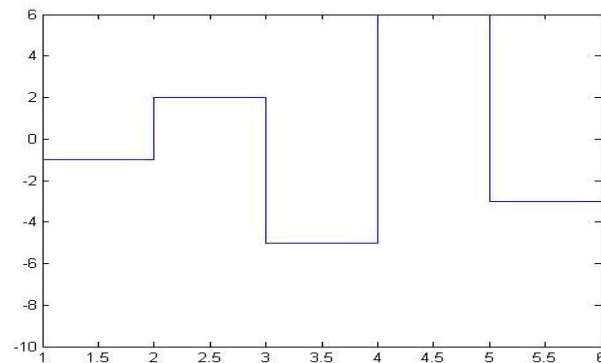
۵- ساقه ای **stem** : همان کار **plot** را انجام می دهد با این تفاوت که نقاط را به هم وصل نکرده و نمودار گسسته ایجاد می کند.

```
>> x=[-1 2 -5 6 -3 -9];
```

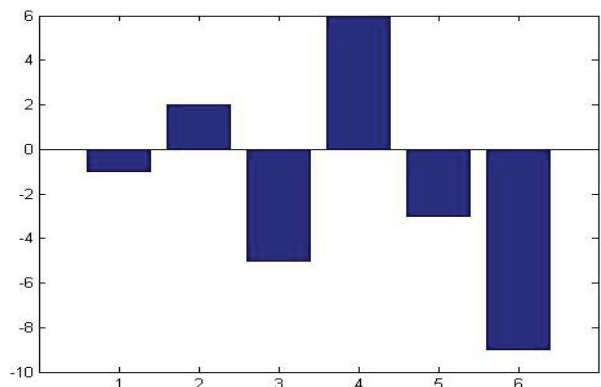
```
>> stem(x)
```



```
>> stairs(x)
```

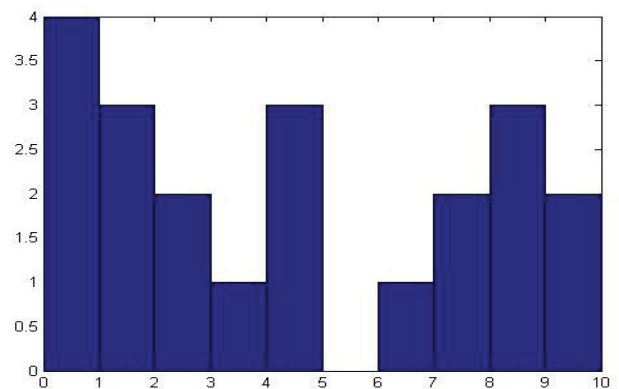


```
>> bar(x)
```



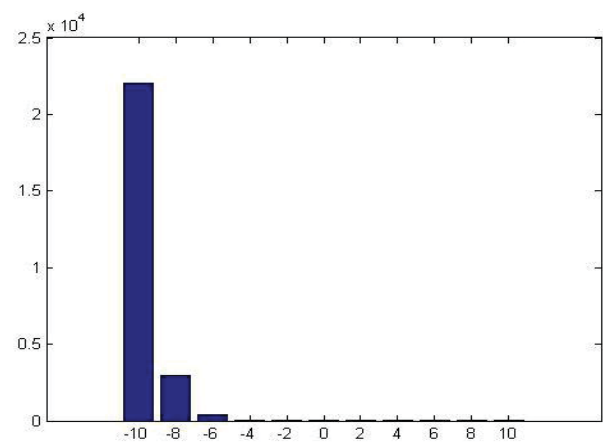
```
>> data = [0,2,9,2,5,8,7,3,1,9,4,3,5,8,10,0,1,2,9,5,10];
```

```
>> hist(data)
```



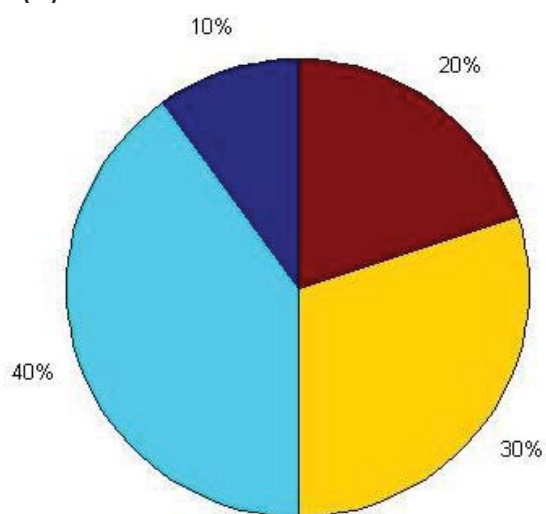
```
>> x=[-10:2:10];
```

```
>> bar(x,exp(-x))
```



```
>> a=[1 4 3 2];
```

```
>> pie(a)
```

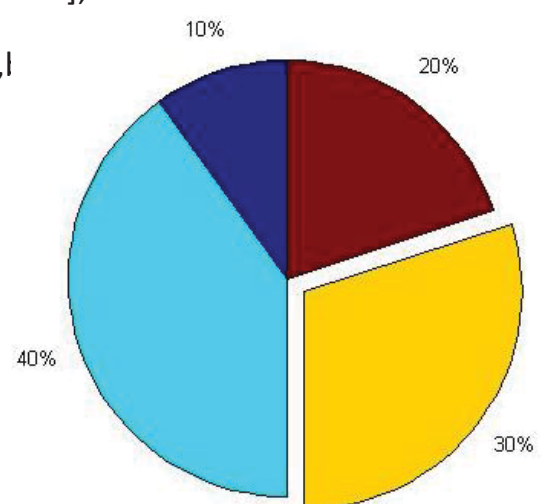


```
>> a=[1 4 3 2];
```

```
>> b=[0 0 1 0];
```

```
>> pie(a,b)
```

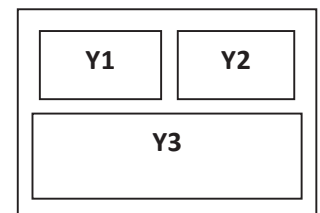
جدا کننده تکه ها



تمرین: نموداری رسم کنید که صفحه را به ۳ قسمت به صورت زیر تقسیم کرده و ۳ نمودار زیر را رسم کنید.

$$y_1 = 1/\cot(x) \quad y_2 = 1/x^2 \quad y_3 = \tan(x) + \cot(x)$$

figure



```
>> syms x
```

```
>> subplot(2,2,1)
```

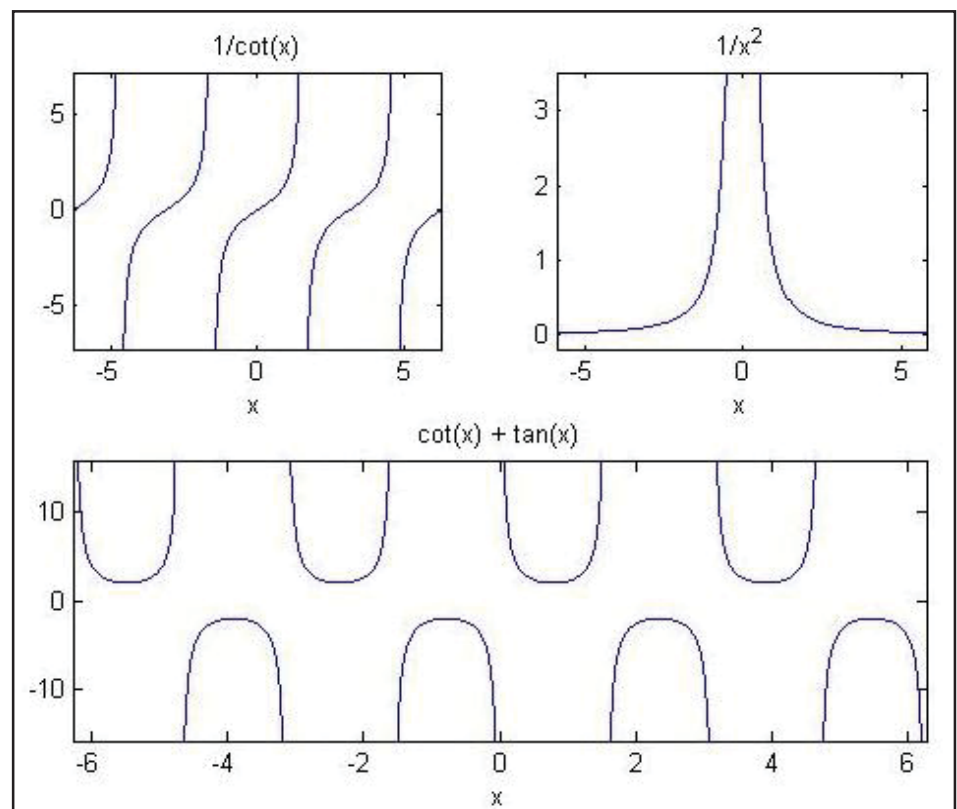
```
>> ezplot(1./cot(x))
```

```
>> subplot(2,2,2)
```

```
>> ezplot(1./(x.^2))
```

```
>> subplot(2,2,[3 4])
```

```
>> ezplot(tan(x)+cot(x))
```



رسم نمودارهای سه بعدی:

با متلب می توان توابع به فرم $z=f(x,y)$ را رسم نمود. نکته ای که در این خصوص باید به آن اشاره کرد این است که اگر مثلا $x=1:3$ و $y=1:2$ و z به ازای همه ی (x,y) ها یعنی $(1,1)$ ، $(2,1)$ ، $(3,1)$ و ... تعداد داشته باشد؛ دستور **meshgrid** برای ساخت این نقاط به کار می رود.

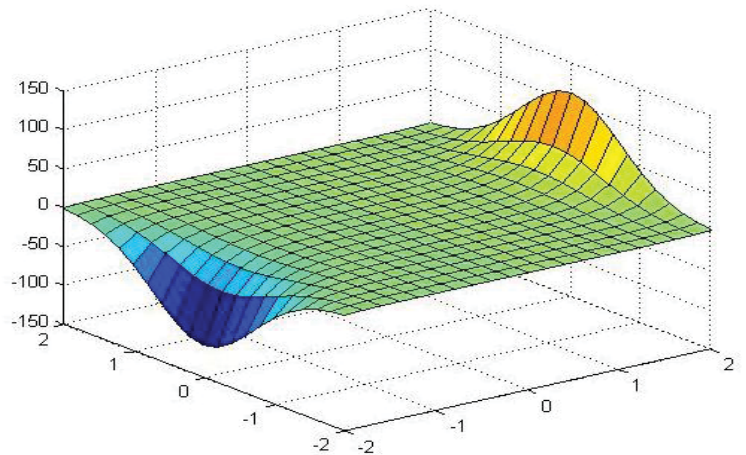
دستور **surf**:

$$z = xe^{(x^2-y^2)}$$

```
>> [x,y]=meshgrid(-2:.2:2,-2:.2:2);
```

```
>> z=x.*exp(x.^2-y.^2);
```

```
>> surf(x,y,z)
```

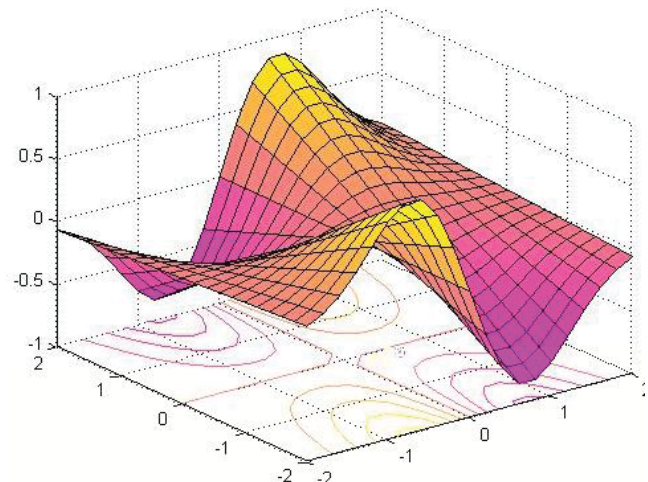


```
>> [x,y]=meshgrid(-2:.2:2,-2:.2:2);
```

```
>> z=x.*y.*exp(-x.^2);
```

```
>> surfc(x,y,z)
```

```
>> colormap spring
```

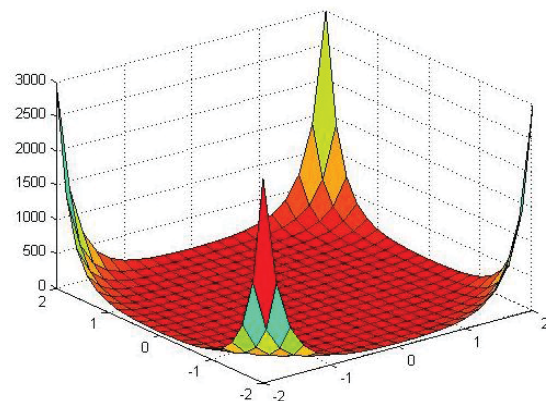


```
>> [x,y]=meshgrid(-2:.2:2,-2:.2:2);
```

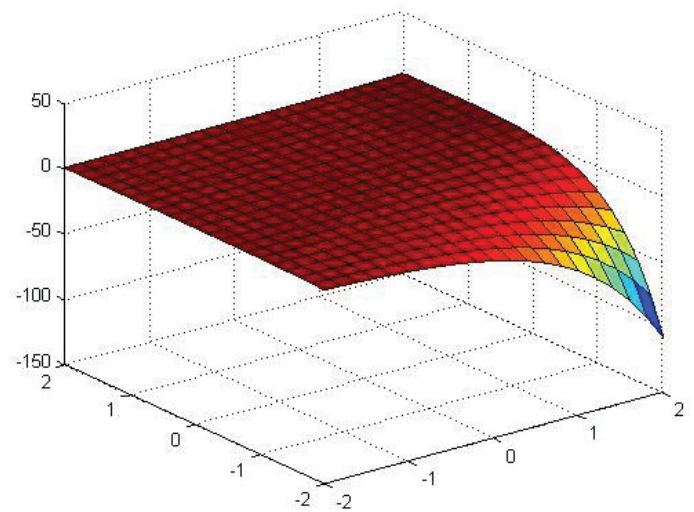
```
>> z=exp(x.^2+y.^2);
```

```
>> surf(x,y,z)
```

```
>> colormap hsv
```

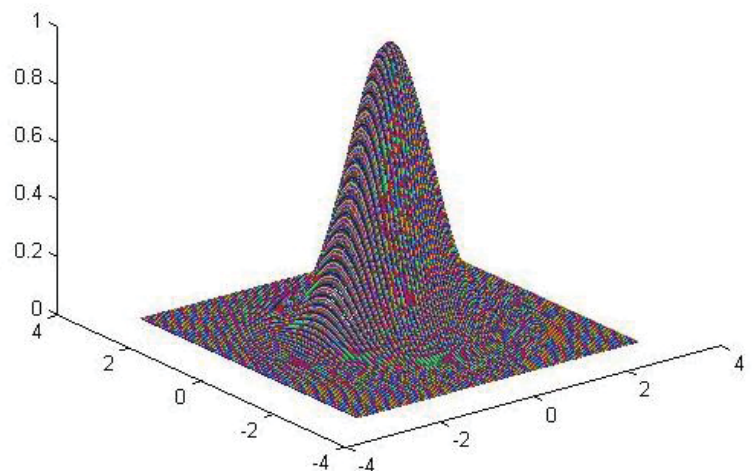


```
>> [x,y]=meshgrid(-2:.2:2,-2:.2:2);
>> z=y.*exp(-y+x);
>> surf(x,y,z)
>> colormap jet
```



دستور **plot3** : این دستور نیز برای رسم $z=f(x,y)$ بکار می رود.

```
>> [x,y]=meshgrid(-3:.01:3,-3:.01:3);
>> z=exp(-x.^2-y.^2);
>> plot3(x,y,z)
```



دستورات حلقه و شرط:

در متلب مشابه زبان های برنامه نویسی دستورات حلقه و شرط وجود دارد.

دستور if :

```
if شرط
> دستورات <
end
```

حلقه : برای ایجاد حلقه از دو دستور **for** و **while** استفاده می شود.

دستور **for** : تعداد گام های مشخص دارد.

دستور **while** : یک شرط خروجی دارد که یک حلقه بینهایت می سازد.

عملگر مقایسه ای در متلب:

> < >= <= == (مساوی) ~ = (نامساوی) & (و منطقی) | (یا منطقی) ~ (تقیض)

دستور **break** : برای قطع برنامه ناشی از حلقه **for** از دستور **break** استفاده می شود.

نوشتن تابع (function) در Matlab :

- یک فایل بصورت مقابل ایجاد می کنیم: `function y=afun(a,b)`
- `afun` اسم تابع است. `y` برگشتی تابع است.
- `afun` اسم تابع است. `y` برگشتی تابع است. `a` و `b` ورودی های تابع هستند.
- بعد از نوشتن `m` فایل ، آن را با نام تابع (در اینجا `afun`) ذخیره نمائید.
- سپس می توان در پنجره `command` تابع را اجرا کرد. مثلا اگر بنویسیم `afun(3,4)` ، 3 به جای `a` و 4 به جای `b` قرار می گیرد. تابع اجرا می شود و مقدار `y` بدست آمده نشان داده می شود.
- در مثال زیر ، تابع مقدار قدرمطلق عدد ورودی `x` را محاسبه و در متغیر `y` بر می گرداند.

```
function y= abs(x)
if(x>0)
    y=x;
else
    y=-x;
end
```

نکات:

- نام تابع باید هم نام با `m-file` باشد.
- می توان در یک `m-file` چندین تابع نوشت.
- متغیرهای موجود در یک تابع محلی هستند و توسط توابع دیگر قابل دسترسی نمی باشند.

توابع کتابخانه ای:

- توابع مثلثاتی: \sin , \cos , \tan , \cot , asin , acos , atan
- Exp (نمایی) – \log (لگاریتم طبیعی) – \log_{10} (لگاریتم در مبنای ۱۰) – factorial (محاسبه فاکتوریل) – floor (جزء صحیح) – fix (قسمت صحیح) – ceil (سقف) – sqrt (جذر) – abs (قدر مطلق)
- Inv (وارون ماتریس) – eye (ایجاد ماتریس واحد) – zeros (ایجاد ماتریس حاوی فقط صفر) – ones (ماتریس با عناصر فقط یک)

تابعی برای تبدیل مختصات دکارتی به قطبی:

$$r = \sqrt{x^2 + y^2} \quad \theta = \tan^{-1} \frac{y}{x}$$

```
function [r,theta]=r2p(x,y)
theta =(180/pi)*atan2(x,y)
r=sqrt(x.^2+y.^2);
end
```

180/pi: تبدیل θ به درجهArc tan x \leftarrow atan(x)Arc tan(x/y) \leftarrow Atan2(x,y)

برای اجرای این تابع پس از ذخیره کردن mfile، در پنجره command به صورت زیر می نویسیم.

```
>> r2p(3,4)
theta = 36.8699
ans = 5
```

تابعی برای تبدیل مختصات قطبی به دکارتی:

$$x = r \cos\theta \quad y = r \sin\theta \quad x^2 + y^2 = r^2$$

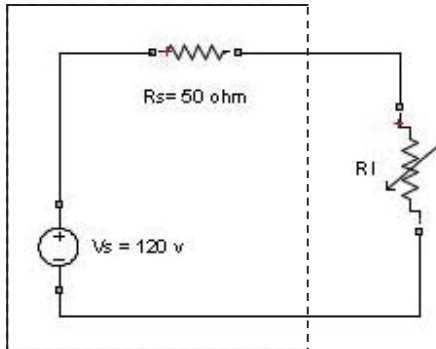
```
function[x,y]=p2r(r,theta)
x=r.*cos(theta*(pi/180))
y=r.*sin(theta*(pi/180))
end
```

برای اجرا در پنجره command دستور زیر را می نویسیم.

```
>> p2r(5,36)
x = 4.0451
y = 2.9389
ans = 4.0451
```


قضیه انتقال توان ماکزیمم

تمرین: مطلوب است ماکزیمم توان رسیده به مقاومت بار در مدار شکل زیر.

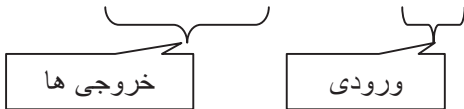


$$\begin{cases} I = \frac{V_s}{R_s + R_L} \\ P_L = \frac{V_s}{R_L} = R_L \cdot I^2 & 0\Omega < R_L < 100\Omega \\ R_s = 50\Omega \quad \text{مقاومت منبع} \end{cases}$$

```

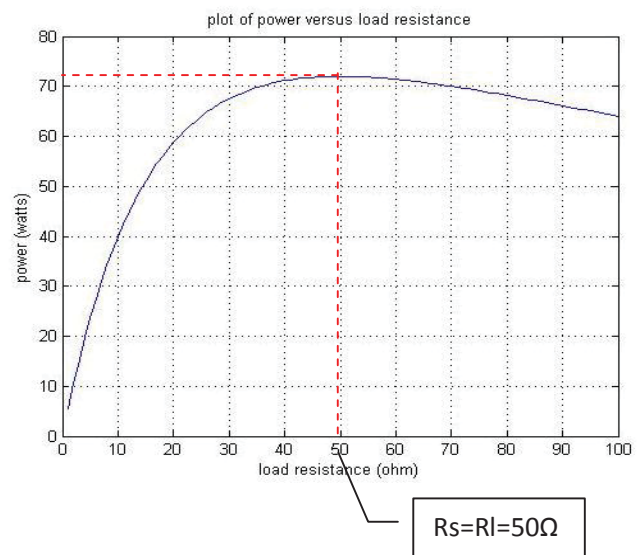
1. Rs=50;
2. Rl=1:100;
3. amper=120./(Rs+Rl);
4. Pl=(amper.^2).*Rl;
5. plot(Rl,Pl);
6. title('plot of power versus load resistance');
7. xlabel('load resistance (ohm)');
8. ylabel('power (watts)');
9. grid on
10. [Pmax,Rl]= max(Pl)

```



$P_{max} = 72$

$R_L = 50$



مثال: برنامه ای بنویسید که نمرات دانشجویان را بگیرد . اگر بالاتر از ۹۵ بود نمره A و اگر بین 85-95 بود نمره B و در غیر این صورت نمره C بدهد.

```
1.n=input('nomre:')
2.if n>95
3. disp('nomre A')
4.elseif n>86
5. disp('nomre B')
6.else
7. disp('nomre c')
8.end
```

دنباله فیبوناچی (fibonacci) :

```
1.function f = fibbo(n)
2.f(1)=1;
3.f(2)=1;
4.for k=3:n
5. f(k)=f(k-1)+f(k-2);
6.end
7.end
```

به صورت زیر برنامه را در پنجره command اجرا می کنیم:

```
>> fibbo(8)
```

```
ans =
```

```
1 1 2 3 5 8 13 21
```

دستور **fprintf** : برای چاپ خروجی از این دستور استفاده می کنیم.

مثال: برنامه ای بنویسید که ضرایب را از یک معادله درجه ۲ به شکل $Ax^2 + Bx + c$ دریافت نموده و با نمایش توضیحات مناسب ریشه های آن را بدست آورد.

```
% calc roots
a=input('A=');
b=input('B=');
c=input('C=');
delta=b.^2-(4*a.*c);
if delta>0
    x1=(-b+sqrt(delta))/(2*a);
    x2=(-b-sqrt(delta))/(2*a);
    disp('two real roots')
    fprintf('x1=%f\n',x1)
    fprintf('x2=%f\n',x2)
elseif delta==0
    x1=(-b)/(2*a);
    disp('two identical real roots')
    fprintf('x1=x2=%f\n',x1)
else
    real_part=(-b)/(2*a);
    image_part=sqrt(abs(delta))/(2*a);
    disp('complex roots')
    fprintf('x1=%f+i%f\n',real_part,image_part)
    fprintf('x2=%f-i%f\n',real_part,image_part)
end
```

(در ابتدای توضیحات برنامه از % استفاده می شود)

f\n یعنی جواب را چاپ کرده و به خط بعد برود.

اجرا در پنجره command :

```
A=8
B=4
C=-6
two real roots
x1=0.651388
x2=-1.151388
```

ترکیب ماتریس ها و ساخت آرایه سلولی:

مثال:

```
>> a=[1 2] 'hello';3 [5;6];
>> b={spiral(3) eye(2);'good' 'bad'};
>> c=cat(3,a,b)
```

```
c(:,:,1) =
```

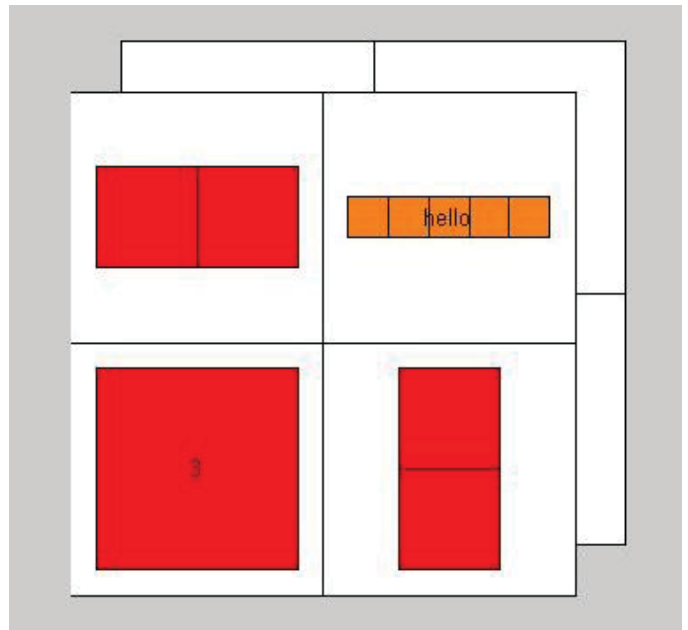
```
[1x2 double] 'hello'
```

```
[ 3] [2x1 double]
```

```
c(:,:,2) =
```

```
[3x3 double] [2x2 double]
```

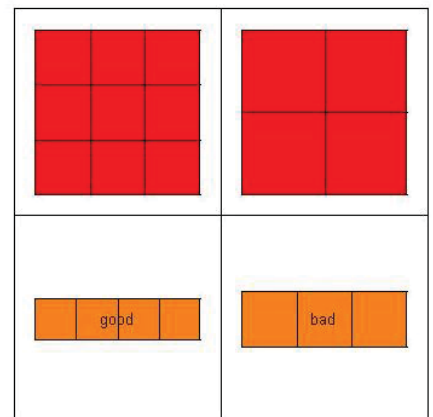
```
'good' 'bad'
```



```
>> cellplot(c)
```

```
>> cellplot(c(:,:,2))
```

برای دستیابی به صفحات دیگر از این دستور استفاده می کنیم.

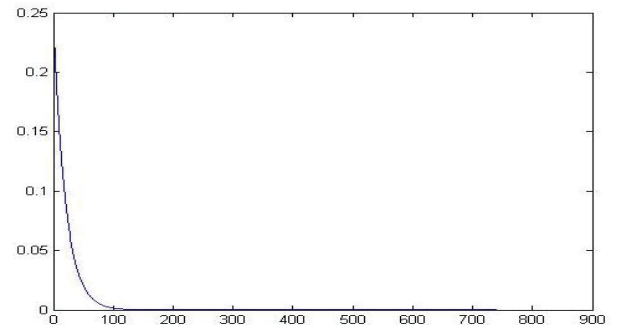


معکوس کردن محورها:

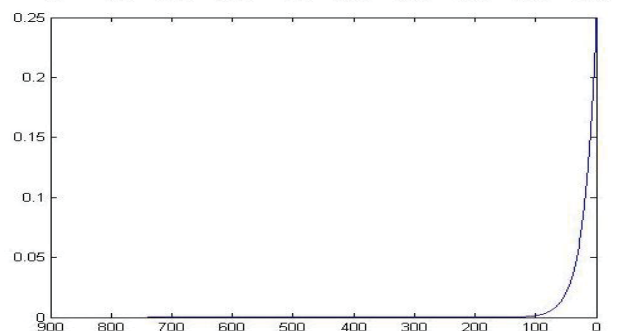
مثال

```
>> t=0:900;
```

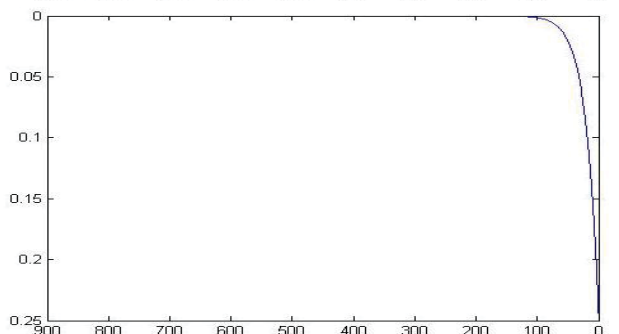
```
>> plot(t,0.25*exp(-0.05*t))
```



```
>> set(gca,'XDir','reverse')
```



```
>> set(gca,'XDir','rev','YDir','rev')
```



$$n! = n * (n-1) * (n-2) * \dots * 2 * 1$$

برنامه فاکتوریل:

```
n=input('enter n =');
fact=1;
for k=1:n
    fact=fact*k;
end
disp([n fact])
```

تمرین : برنامه ای بنویسید که تعدادی ستاره را بصورت یک مثلث قائم الزاویه رسم کند.

```
a=input('enter a =');
for s=1:a;
    for c=1:s;
        fprintf('*');
    end
    fprintf('\n');
end
```

```
enter a =5
*
**
***
****
*****
```

تمرین: این برنامه را با while هم بنویسید.

```
a=input('enter a =');
s=1;
while s<=a
    c=1;
    while c<=s
        fprintf('*')
        c=c+1;
    end
    s=s+1;
    fprintf('\n')
end
```

```
enter a =5
*
**
***
****
*****
```

رسم تابع همراه با مجانب هایش :

توجه شود که متلب نمی تواند مجانب رسم کند اما برای رسم مجانب راهی پیشنهاد می کنیم و با مثال زیر بررسی می کنیم.

$$f(x) = \frac{3x^2+6x-1}{x^2+x-3} \quad \text{مجانب افقی : } y=3 \quad \text{مجانب قائم : } x = -2.3, x = 1.3$$

```
>> syms x
>> f=(3*x^2+6*x-1)/(x^2+x-3);
>> roots=solve('x^2+x-1')
```

roots =

$$13^{(1/2)}/2 - 1/2$$

- 13^(1/2)/2 - 1/2

```
>> vpa(roots,3)
```

ans =

1.3

-2.3

```
>> ezplot(f)
```

```
>> hold on
```

```
>> plot([-2*pi 2*pi],[3 3],'g')
```



```
>> x1=-2.3; x2=1.3;
```

```
>> y=-100:0.1:100;
```

```
>> plot(x1,y,'r',x2,y,'r')
```

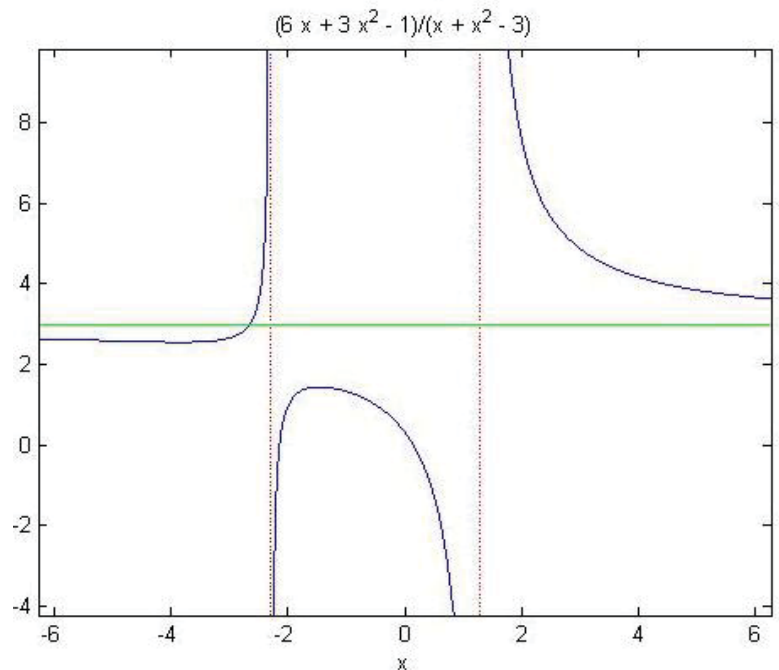
```
>> hold off
```

```
>> y=3;
```

```
>> x=-10:.01:10;
```

```
>> hold on
```

```
>> plot(x,y,'g')
```



تمرین: تابعی که مجانب مایل دارد را بررسی کنید.

$$f(x) = \frac{x^2 + x - 1}{x + 1}$$

مجانب قائم : $x=-1$

مجانب مایل : $y=x$

```
>> syms x
```

```
>> f=(x^2+x-1)/(x+1);
```

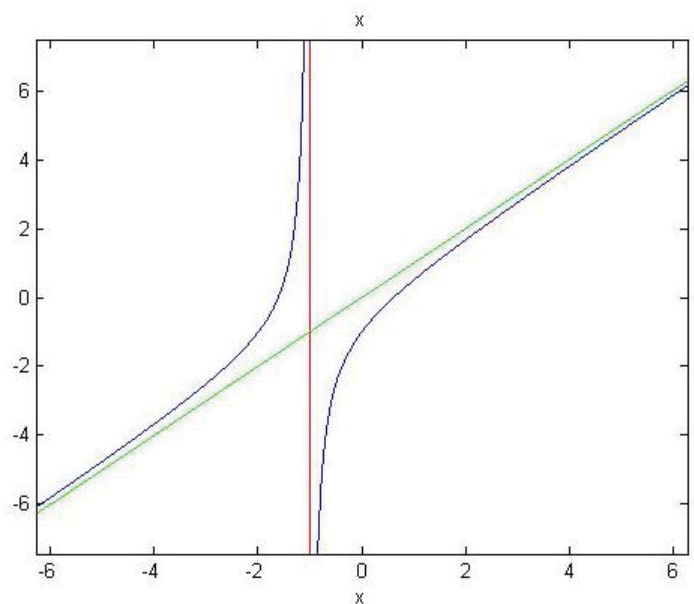
```
>> ezplot(f)
```

```
>> hold on
```

```
>> x1=-1;
```

```
>> y=-20:.01:20;
```

```
>> plot(x1,y,'r')
```



```
>> remain=rem((x^2+x-1),(x+1))
```

```
remain = -1
```

```
>> div=((x^2+x-1)-(-1))/(x+1)
```

```
div = (x^2 + x)/(x + 1)
```

```
>> simplify(div)
```

```
ans = x
```

```
>> h=ezplot('x');
```

```
>> set(h,'Color','g');
```

مجانِب مایل $y=x$

دستور rem : باقیمانده را محاسبه می کند.

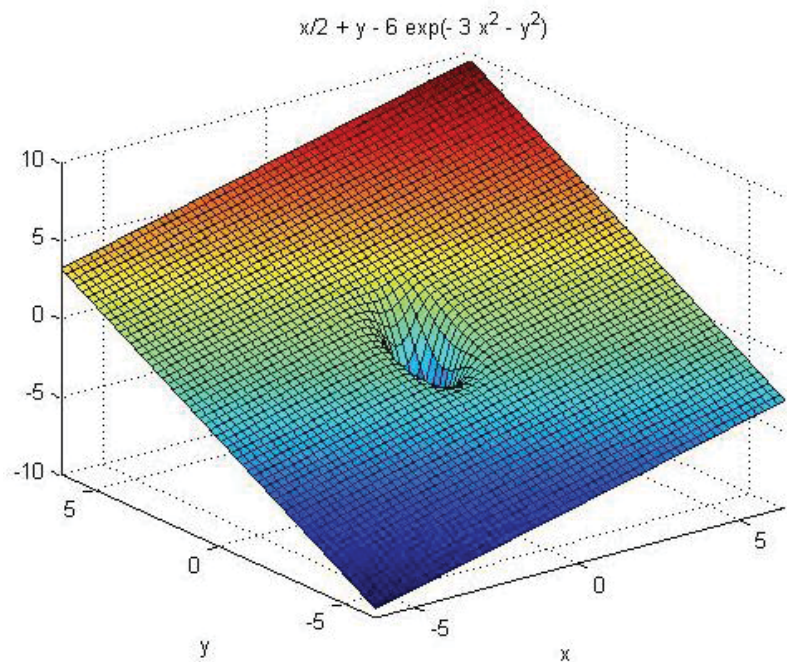
رسم توابع دو متغیره

مثال:

```
>> syms x y
```

```
>> f=-6*exp(-3*x^2-y^2)+0.5*x+y
```

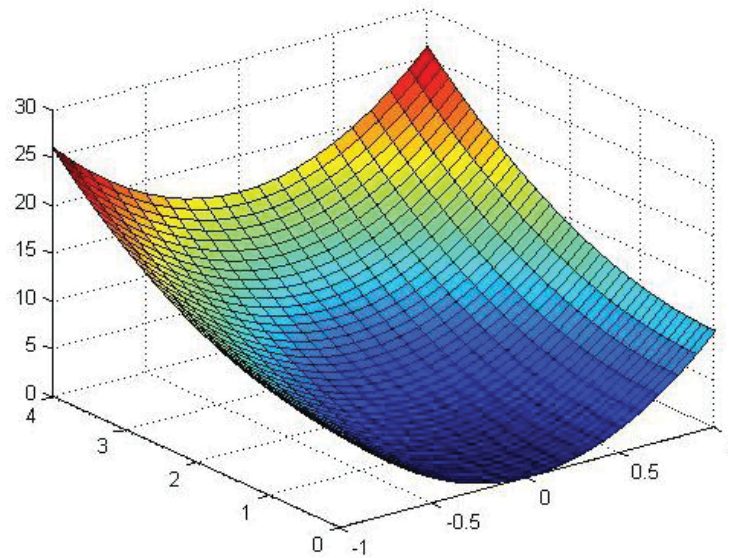
```
>> ezsurf(f)
```



مثال:

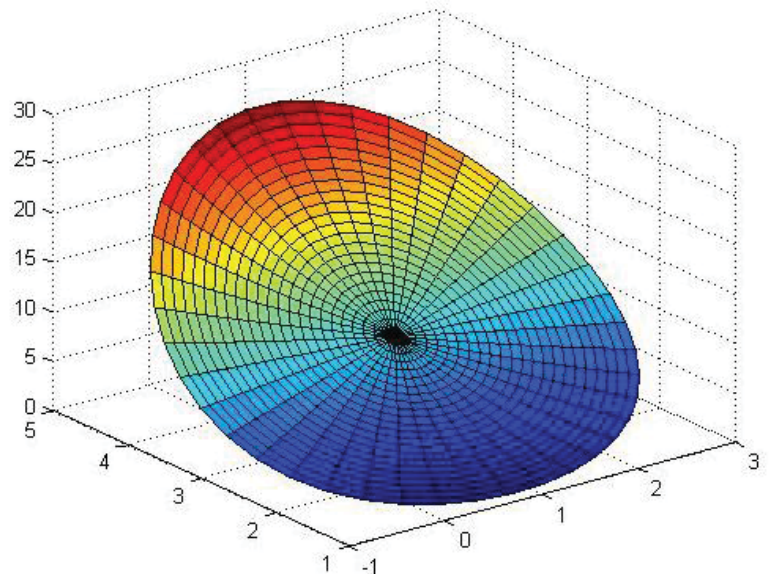
$$f(x, y) = 10x^2 + y^2$$

```
>> x=-1:1:1;
>> y=0:1:4;
>> [x,y]=meshgrid(x,y);
>> f=inline('10*x.^2+y.^2','x','y')
>> surf(x,y,f(x,y))
```



مثال: رسم $f(x, y) = x - 1 + y^2$ بر روی ناحیه مدور $(x - 1)^2 + (y - 3)^2 = 4$

```
r=linspace(0,2,24);
theta=linspace(0,2*pi,41);
[R,TH]=meshgrid(r,theta);
x=1+R.*cos(TH);
y=3+R.*sin(TH);
z=x-1+y.^2;
surf(x,y,z)
```

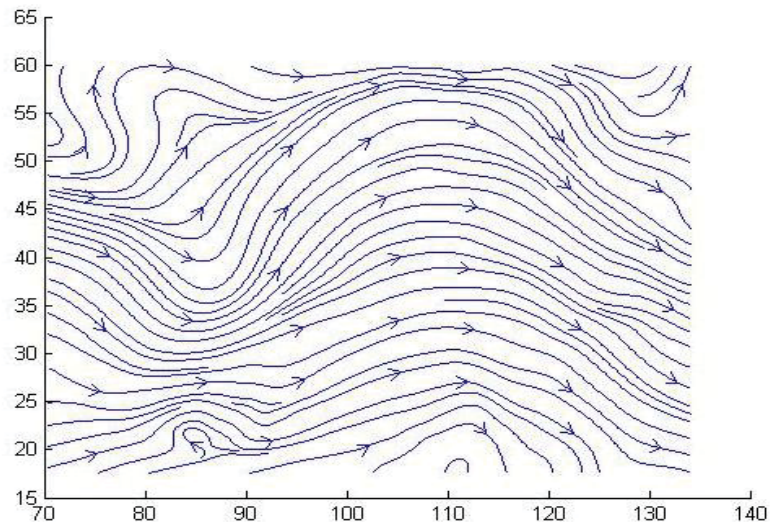


نکته: توجه شود که در این مثال از مختصات قطبی استفاده کردیم.

نمایش داده ها به صورت سیال:

```
>> load wind
>> zmax=max(z(:))
zmax = 16
```

```
>> zmin=min(z(:))
zmin = -0.0020
```



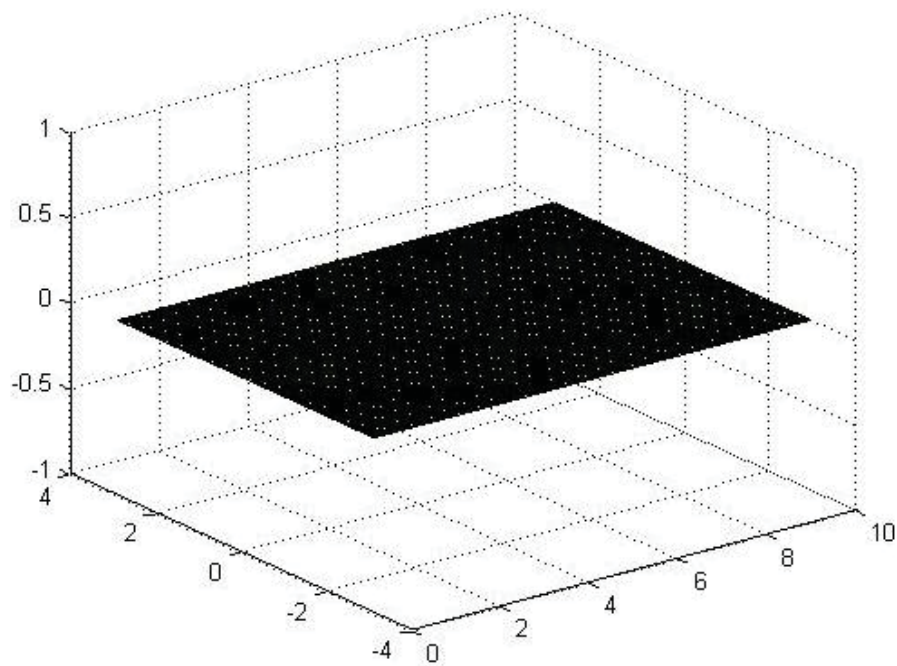
```
>> streamslice(x,y,z,u,v,w,[],[],(zmax-zmin)/2)
```

دستور **countorslice** : این دستور را نیز بررسی کنید.

مثال: با برنامه زیر می توانید یک نوع نمایش سیال داده ها را مشاهده کنید.

Slicing Fluid Flow

```
>> [x,y,z,v] = flow;
>> xmin = min(x(:));
>> ymin = min(y(:));
>> zmin = min(z(:));
>> xmax = max(x(:));
>> ymax = max(y(:));
>> zmax = max(z(:));
>> hslice = surf(linspace(xmin,xmax,100),...
    linspace(ymin,ymax,100),zeros(100));
```

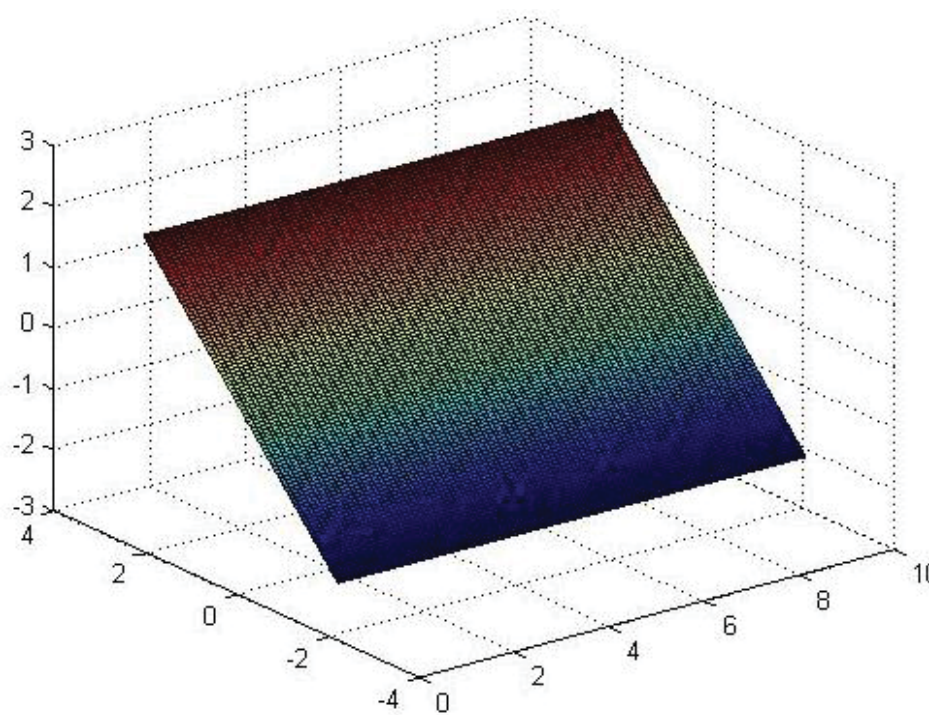


```
>> rotate(hslice,[-1,0,0],-45)
```

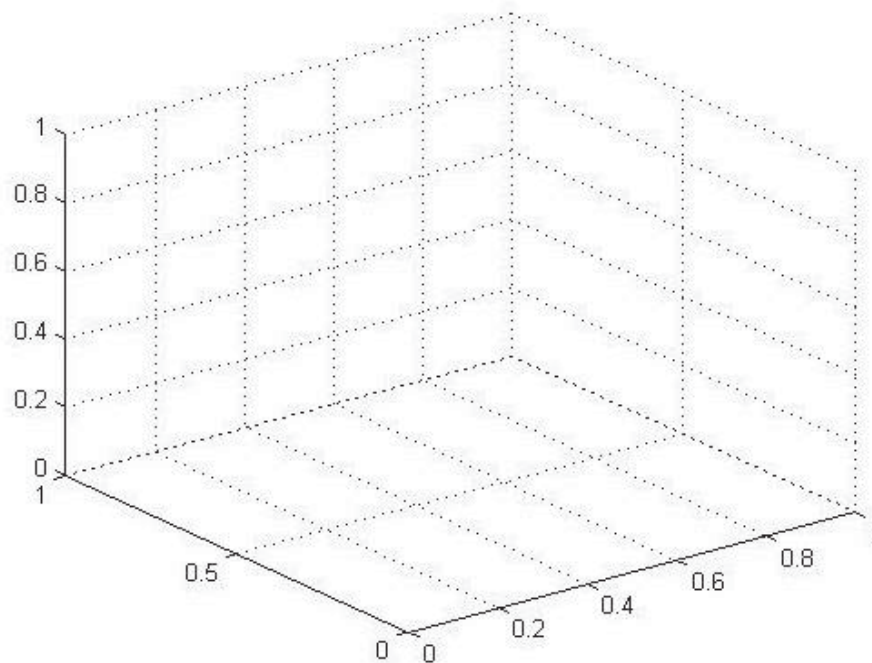
```
>> xd = get(hslice,'XData');
```

```
>> yd = get(hslice,'YData');
```

```
>> zd = get(hslice,'ZData');
```



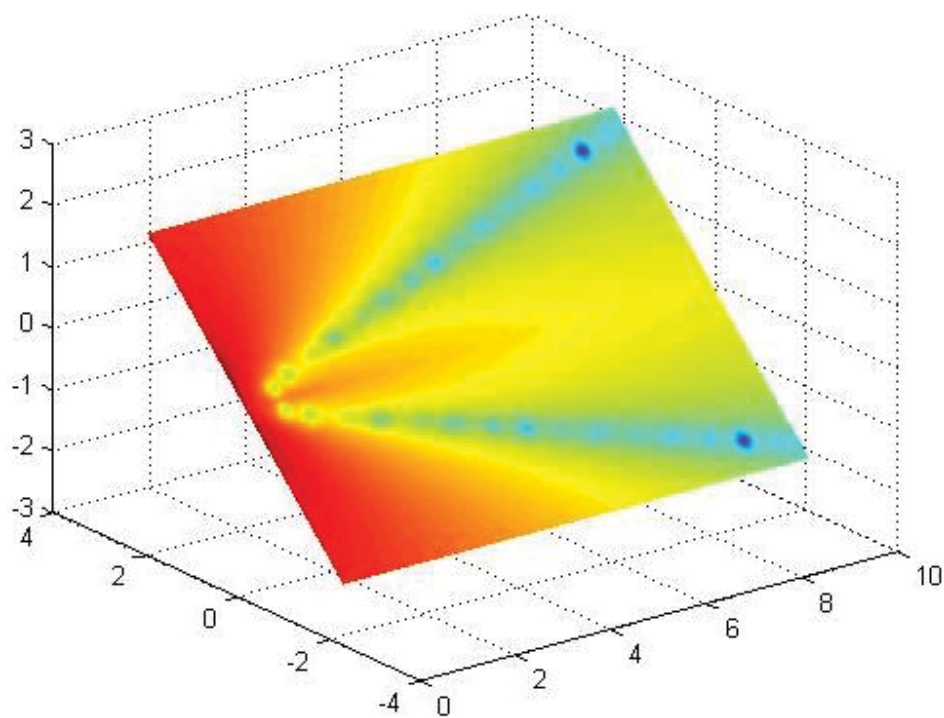
```
>> delete(hslice)
```



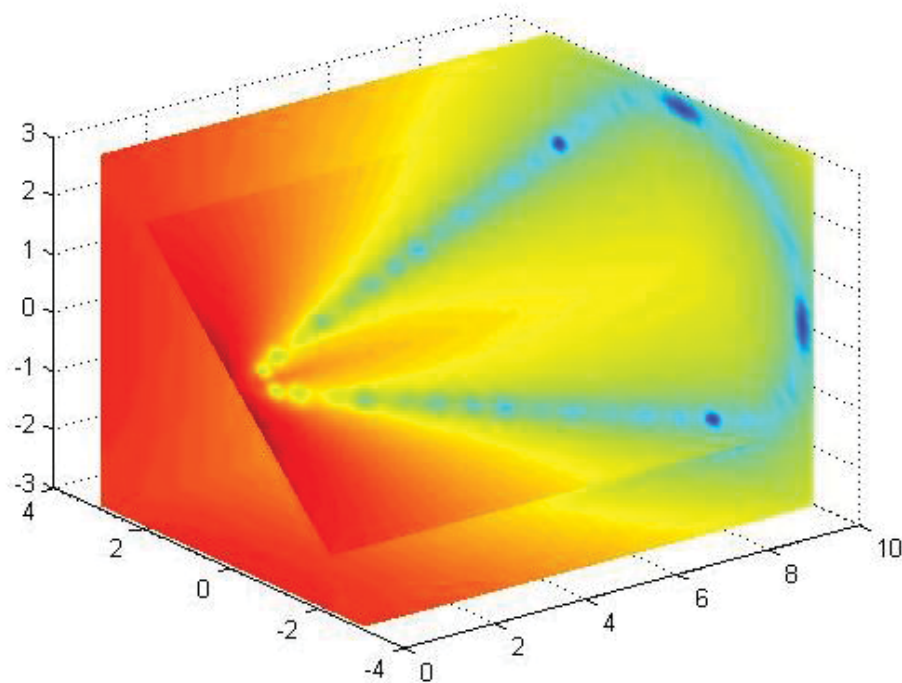
```
>> h = slice(x,y,z,v,xd,yd,zd);
```

```
>> set(h,'FaceColor','interp',...
```

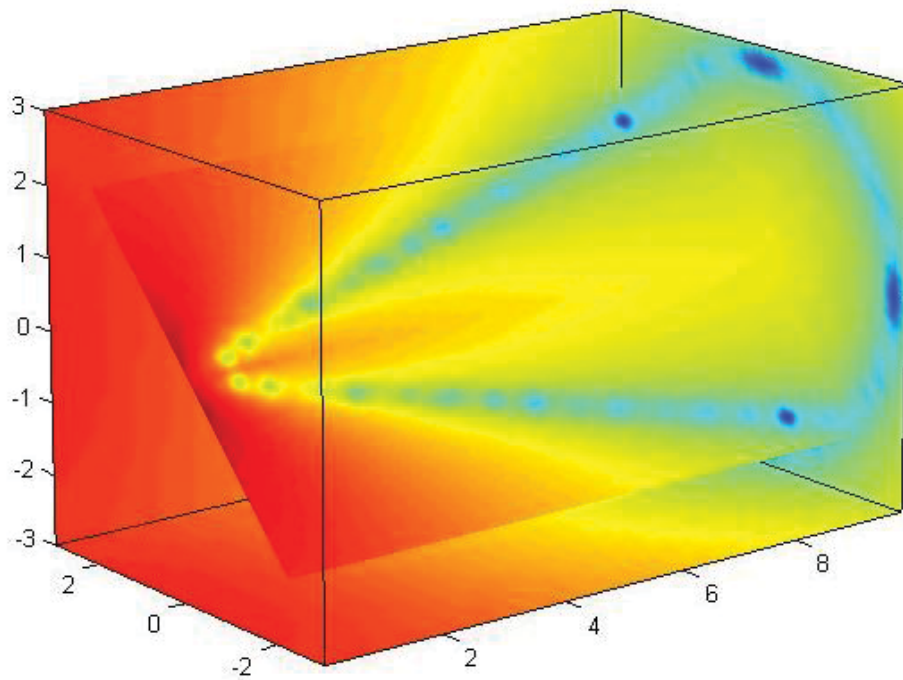
```
'EdgeColor','none','DiffuseStrength',.8)
```



```
>> hold on
>> hx = slice(x,y,z,v,xmax,[],[]);
>> set(hx,'FaceColor','interp','EdgeColor','none')
>> hy = slice(x,y,z,v,[],ymax,[]);
>> set(hy,'FaceColor','interp','EdgeColor','none')
>> hz = slice(x,y,z,v,[],[],zmin);
>> set(hz,'FaceColor','interp','EdgeColor','none')
```



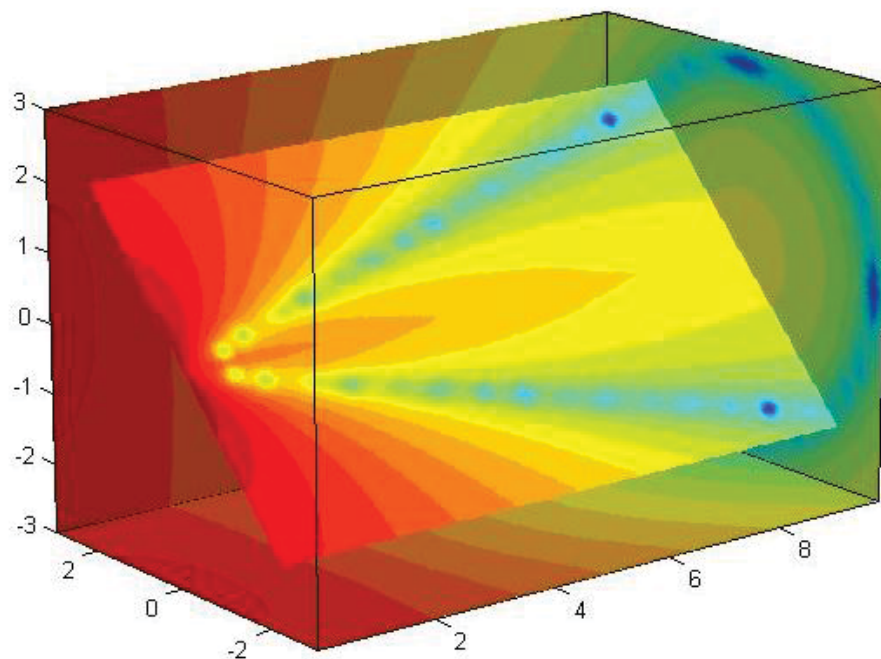
```
>> daspect([1,1,1])
>> axis tight
>> box on
>> view(-38.5,16)
>> camzoom(1.4)
>> camproj perspective
```



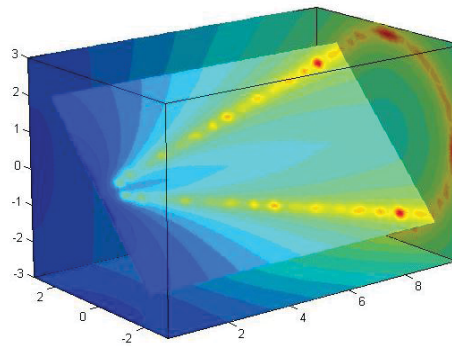
```
>> lightangle(-45,45)
```

```
>> colormap (jet(24))
```

```
>> set(gcf,'Renderer','zbuffer')
```



```
>> colormap (flipud(jet(24)))
```



نمودار رسم میدان: **quiver**

فرض کنید می خواهیم میدان اطراف یک دو قطبی را رسم کنیم.

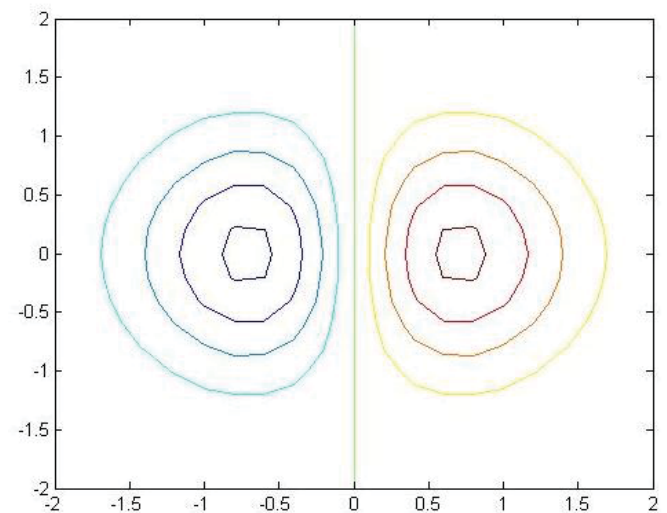
$$z = x e^{-(x^2+y^2)}$$

```
>> [x,y]=meshgrid(-2:.2:2);
```

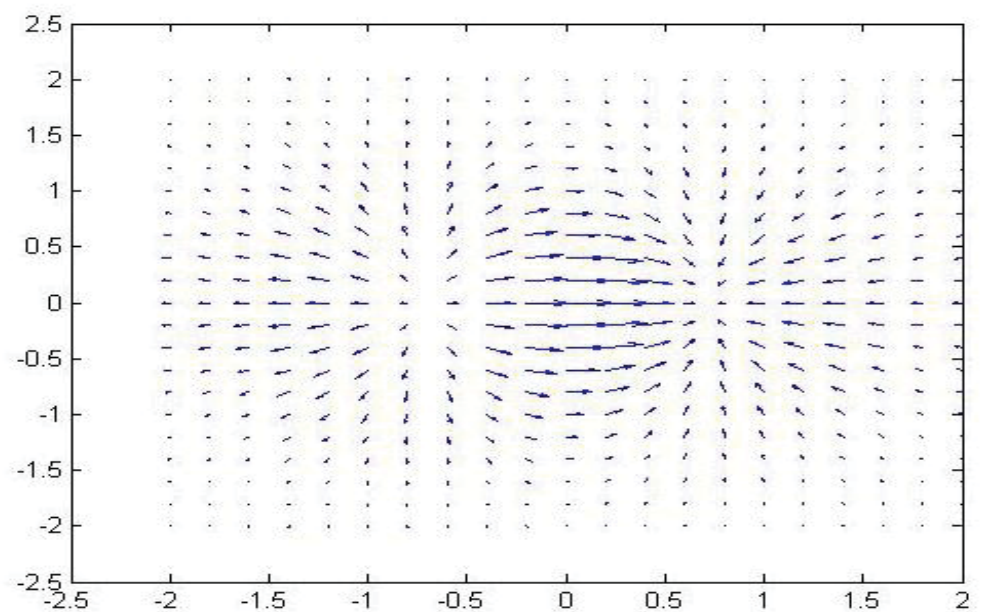
```
>> z=x.*exp(-x.^2-y.^2);
```

```
>> [Dx,Dy]=gradient(z,.2,.2);
```

```
>> contour(x,y,z)
```



```
>> quiver(x,y,Dx,Dy)
```



تمرین : برای یک تک قطبی الکتریکی با بار منفی همین کار را تکرار کنید:

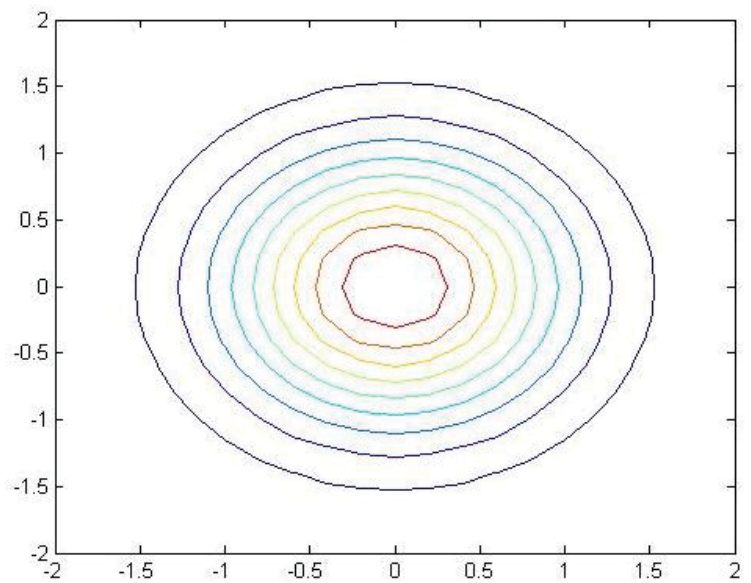
```
Z=exp(-x^2-y^2)
```

```
>> [x,y]=meshgrid(-2:.2:2);
```

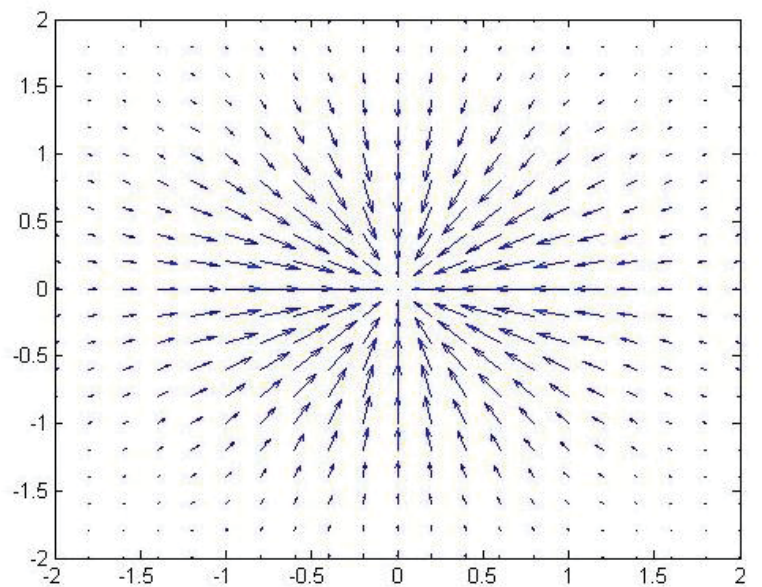
```
>> z=exp(-x.^2-y.^2);
```

```
>> [Dx,Dy]=gradient(z,.2,.2);
```

```
>> contour(x,y,z)
```

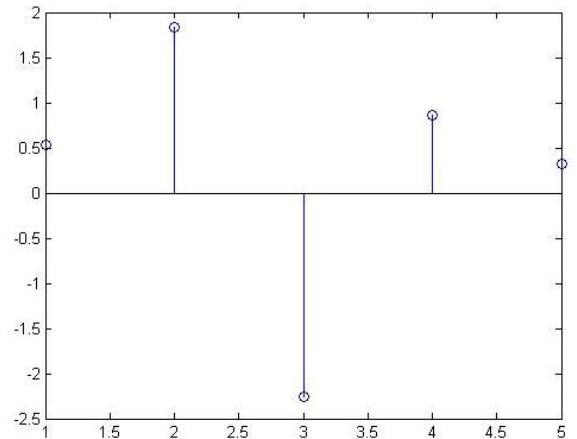


```
>> quiver(x,y,Dx,Dy)
```



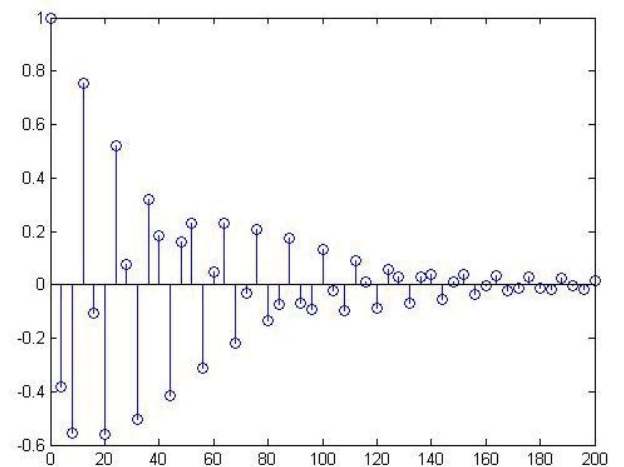
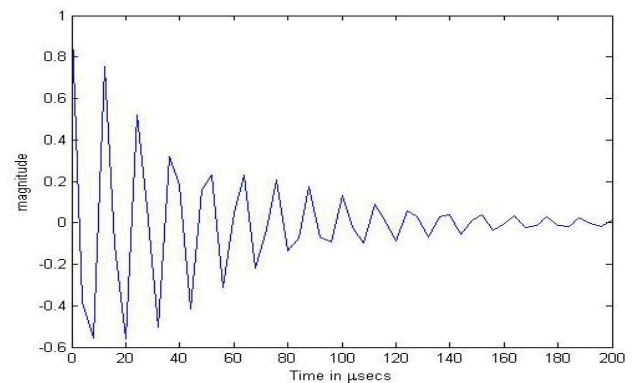
رسم گسسته

```
>> y=randn(1,5)
y = 0.5377 1.8339 -2.2588 0.8622 0.3188
>> stem(y)
```

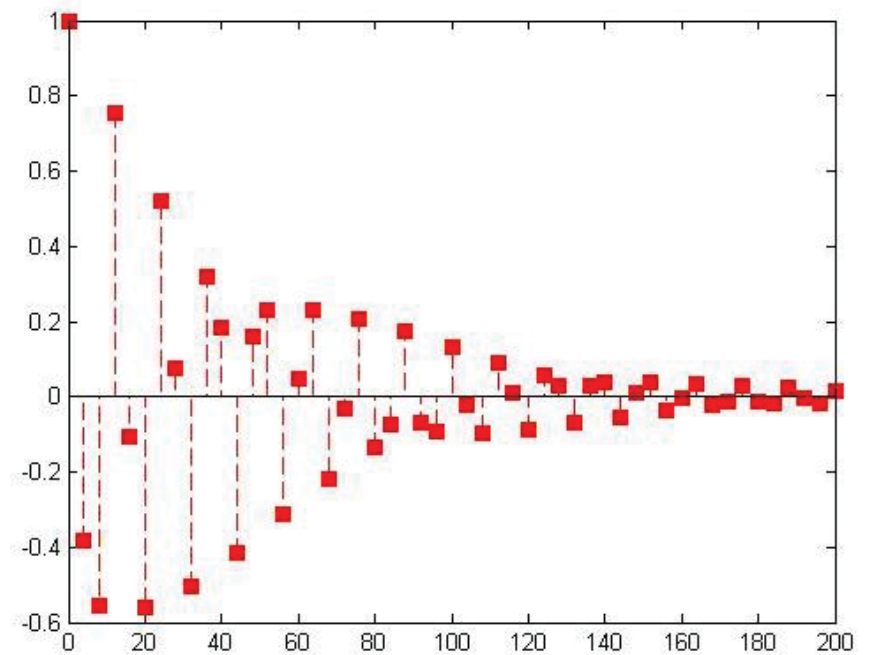


رسم گسسته فوریه:

```
>> alpha=0.02;
>> beta=0.5;
>> t=0:4:200;
>> y=exp(-alpha*t).*cos(beta*t);
>> plot(t,y)
>> figure
>> stem(t,y)
>> xlabel('Time in \musecs')
>> ylabel('magnitude')
```



```
>> stem(t,y,'--sr','fill')
```



مثال: جمع دو تابع a و b

```
>> x=linspace(0,2*pi,60);
```

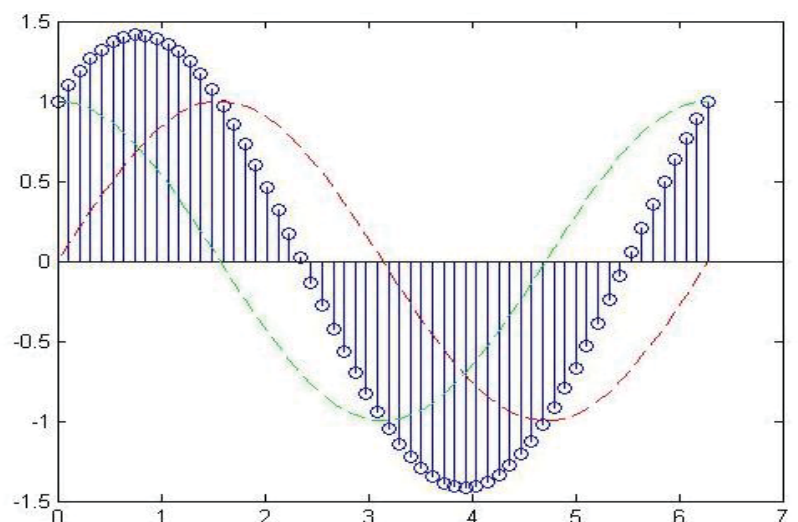
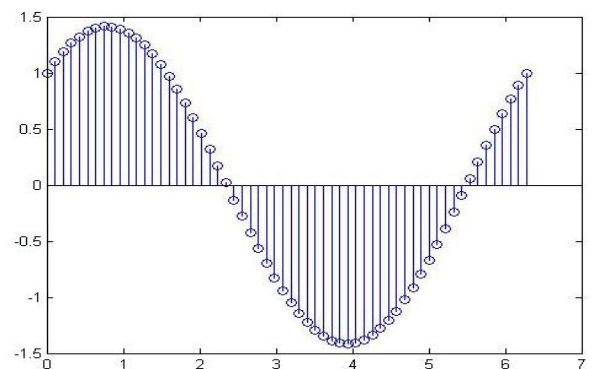
```
>> a=sin(x); b=cos(x);
```

```
>> st=stem(x,a+b);
```

```
>> hold on
```

```
>> pt=plot(x,a,'--r',x,b,'--g');
```

```
>> hold off
```



ریاضیات در متلب:

مثال: برنامه ای بنویسید که میانگین و انحراف معیار را محاسبه کند.

$$\bar{X} = \sum_{i=1}^n \frac{x_i}{n} \qquad \delta = \sqrt{\frac{\bar{X} - (\sum x_i)^2}{n(n-1)}}$$

```
n=0; sum_x1=0; sum_x2=0;
x=input('enter first value:');
while x>=0
    n=n+1
    sum_x1=sum_x1+x;
    sum_x2=sum_x2+x^2;
    x=input('enter next value:');
end
x_bar=sum_x2./n;
std_dev=sqrt((n*sum_x2-sum_x1^2)/n*(n-1));
average=sum_x1./n
disp('results')
disp(' ')
fprintf('average=%f\n',average)
fprintf('mean_data=%f\n',x_bar)
fprintf('std_dev=%f\n',std_dev)
fprintf('number_data=%f\n',n)
```

حلقه تو در تو

جدول ضرب

```
m=input('m=');
n=input('n=');
for ii=1:m
    for jj=1:n
        product=ii*jj
        fprintf('%d*%d=%d\n',ii,jj,product)
    end
end
```

محاسبات و عملیات ریاضی در متلب

۱- محاسبات سمبلیک

۲- محاسبات عددی

چند جمله ای ها:

$$P(x)=ax^n+bx^{n-1}+\dots+c$$

```
>> p=[a b ... c]
```

مثال: دو چند جمله ای مقابل را با هم جمع کنید.

$$p(x)=x^3+2x^2-1 \quad \text{و} \quad t(x)=-x^2+1$$

```
>> p=[1 2 0 -1]
```

```
>> t=[0 -1 0 1]
```

```
>> sum_1=p+t
```

```
sum_1 =
```

```
1 1 0 0
```

دستور poly2sym

```
>> poly2sym([1 1 0 0])
```

```
ans=
```

```
x^3 + x^2
```

ضرب دو عبارت سمبولیک

```
>> conv(p,t)
```

```
ans =
```

```
0 -1 -2 1 3 0 -1
```

```
>> a=[1 1 2 3 0];
```

```
>> b=[2 1 0];
```

```
>> [q,r]=deconv(a,b)
```

```
q =
```

```
0.5000 0.2500 0.8750
```

```
r =
```

```
0 0 0 2.1250 0
```

یافتن ریشه های چندجمله ای: دستور **roots**

$$P(x)=x^3+2x^2+3x+4$$

```
>> p=[1 2 3 4]
```

```
>> roots(p)
```

```
ans =
```

```
-1.6506 + 0.0000i
```

```
-0.1747 + 1.5469i
```

```
-0.1747 - 1.5469i
```

■ تعداد ریشه های مختلط یک معادله زوج است.

تمرین: قضیه زیر را اثبات کنید:

● تعداد ریشه های موهومی یک چند جمله ای همواره عدد زوج است.

تمرین: اگر چند جمله ای بسیار بزرگ داشته باشیم چگونه باید ریشه هایش را یافت؟

■ برای اینکه عددهای داده شده در متلب به صورت عمودی گذاشته شود ، مابین عددها از ; استفاده می کنیم.

```
>> p=[1;-1;0]
```

```
p =
```

```
1
```

```
-1
```

```
0
```

■ دستور poly2sym نمایش سمبولیک یک چندجمله ای را که ضرایب آن داده شده است نشان می دهد. چند جمله ای به صورت قرار دادی با متغیر x نمایش داده می شود.

```
>> poly2sym([1 0 -1 0])
```

```
ans = x^3 - x
```

برای تغییر متغیر :

```
>> poly2sym([1 0 -1 0],'t')
```

```
ans = t^3 - t
```

دستور polyval

برای یافتن مقدار تابع چندجمله ای از این دستور استفاده می کنیم:

```
f(x)=x2-1    f(0)=-1
```

```
>> f=[1 0 -1];
```

```
>> polyval(f,0)
```

```
ans = -1
```

دستور subs

برای یافتن مقدار تابع غیرخطی از دستور subs استفاده می کنیم.

مثال:

تابع $f(x) = \sin x + x^2\sqrt{x-1}$ مفروض است. مقدار تابع را در نقطه $x=1$ بیابید.

```
>> syms x
>> f=sin(x)+x.^2*(sqrt(x-1));
>> subs(f,x,1)
ans = sin(1)
>> sin (1)                ans = 0.8415
```

دستور **syms** : با **syms** متغیر را به متلب می شناسانیم.

مثال: تابع $f(x)=x^2+2x$ مفروض است. مطلوب است مقدار $f(x-1)$ ؟

```
>> syms x
>> f=x.^2+2*x ;
>> subs(f,x,x-1)
ans = 2*x + (x - 1)^2 - 2
```

مثال: مقدار تابع $f(x,y)=\sin xy + xy(x+1)$ را در نقاط $x=1$ ، $x=1$ و $y=1$ بدست آورید.

```
>> syms x y
>> f=sin(x.*y)+x.*y.*(x+1);
>> subs(f,x,1)                ans = 2*y + sin(y)
>> subs(f,y,1)                ans = sin(x) + x*(x + 1)
>> subs(f,x,0)                ans = 0
```

■ **Sym** نشان دهنده مقدار سمبولیک است

■ برای تبدیل عبارت سمبولیک به جبری از دستور **sym2poly** استفاده می کنیم.

چند جمله ای نیوتن

$$(x + y)^n = \binom{n}{0} x^{n-1} + \binom{n}{1} x^{n-2} y + \dots$$

```
>> syms x y
```

```
>> expand((x+y)^3)
```

```
ans = x^3 + 3*x^2*y + 3*x*y^2 + y^3
```

ضرب جملات نیوتن:

```
>> factor(ans)
```

```
ans = (x + y)^3
```

```
expand≠collect=factor
```

توجه:

ساده سازی چندجمله ای ها:

```
>> syms x
```

```
>> f(x)=x.^4+3.*x.^2-3.*x.^4+5.*x.^2+x-1;
```

```
>> simplify(f)
```

```
ans(x) = - 2*x^4 + 8*x^2 + x - 1
```

```
>> f=tan(x)+cot(x)+(1/tan(x));
```

```
>> simplify(f)
```

```
ans = (tan(x)^2 + 2)/tan(x)
```

■ برای ساده سازی و قشنگ تر نوشتن جواب از دستور `pretty(ans)` استفاده می شود.

```
>> pretty(ans)
```

$$\frac{\tan^2(x) + 2}{\tan(x)}$$



دستور simple با تشریحات جواب را نشان می دهد:

$$f(x) = \frac{\sin x + \sin 3x + \sin 5x}{\cos x + \cos 3x + \cos 5x}$$

```
>> syms x
```

```
>> f(x)=(sin(x)+sin(3.*x)+sin(5.*x))./(cos(x)+cos(3.*x)+cos(5.*x));
```

```
>> simple(f)
```

```
simplify: sin(3*x)/cos(3*x)
```

```
radsimp: (sin(3*x) + sin(5*x) + sin(x))/(cos(3*x) + cos(5*x) + cos(x))
```

```
simplify(Steps = 100): tan(3*x)
```

```
combine(sincos): (sin(3*x) + sin(5*x) + sin(x))/(cos(3*x) + cos(5*x) + cos(x))
```

```
combine(sinhcosh): (sin(3*x) + sin(5*x) + sin(x))/(cos(3*x) + cos(5*x) + cos(x))
```

```
combine(ln): (sin(3*x) + sin(5*x) + sin(x))/(cos(3*x) + cos(5*x) + cos(x))
```

```
factor: (sin(3*x) + sin(5*x) + sin(x))/(cos(3*x) + cos(5*x) + cos(x))
```

```
expand:
```

```
sin(x)/(3*cos(x) - 16*cos(x)^3 + 16*cos(x)^5) - (8*cos(x)^2*sin(x))/(3*cos(x) - 16*cos(x)^3 + 16*cos(x)^5) + (16*cos(x)^4*sin(x))/(3*cos(x) - 16*cos(x)^3 + 16*cos(x)^5)
```

```
combine:
```

```
(sin(3*x) + sin(5*x) + sin(x))/(cos(3*x) + cos(5*x) + cos(x))
```

```
rewrite(exp):
```

```
((exp(-x*i)*i)/2 - (exp(x*i)*i)/2 + (exp(-x*3*i)*i)/2 - (exp(x*3*i)*i)/2 + (exp(-x*5*i)*i)/2 - (exp(x*5*i)*i)/2)/(exp(-x*i)/2 + exp(x*i)/2 + exp(-x*3*i)/2 + exp(x*3*i)/2 + exp(-x*5*i)/2 + exp(x*5*i)/2)
```

```
rewrite(sincos):
```

```
(sin(3*x) + sin(5*x) + sin(x))/(cos(3*x) + cos(5*x) + cos(x))
```

```
rewrite(sinhcosh):
```

```
-(sinh(x*i)*i + sinh(x*3*i)*i + sinh(x*5*i)*i)/(cosh(x*i) + cosh(x*3*i) + cosh(x*5*i))
```

```
rewrite(tan):
```

```
-(2*tan(x/2))/(tan(x/2)^2 + 1) + (2*tan((3*x)/2))/(tan((3*x)/2)^2 + 1) +  
(2*tan((5*x)/2))/(tan((5*x)/2)^2 + 1)/((tan(x/2)^2 - 1)/(tan(x/2)^2 + 1) + (tan((3*x)/2)^2 -  
1)/(tan((3*x)/2)^2 + 1) + (tan((5*x)/2)^2 - 1)/(tan((5*x)/2)^2 + 1))
```

```
mwcossin:
```

```
-(sin(3*x) + sin(5*x) + sin(x))/(2*sin(x/2)^2 + 2*sin((3*x)/2)^2 + 2*sin((5*x)/2)^2 - 3)
```

```
collect(x):
```

```
(sin(3*x) + sin(5*x) + sin(x))/(cos(3*x) + cos(5*x) + cos(x))
```

```
ans(x) = tan(3*x)
```

```
>> simplify(f)
```

```
ans(x) =
```

```
sin(3*x)/cos(3*x)
```

```
>> pretty(f)
```

```
sin(3 x) + sin(5 x) + sin(x)
-----
cos(3 x) + cos(5 x) + cos(x)
```

تجزیه کسرهای جزئی:

$$\frac{b(s)}{a(s)} = \frac{b_1 s^m + b_2 s^{m-1} + \dots + b_n}{a_1 s^n + a_2 s^{n-1} + \dots + a_n} = \frac{r_1}{s - p_1} + \frac{r_2}{s - p_2} + \dots + k(s)$$

■ عملگر لاپلاس $s = \sigma + j\omega$

$$\frac{b(s)}{a(s)} = \frac{2s^3 + s^2 - 1}{s^2 + 1}$$

مثال:

$$\frac{b(s)}{a(s)} = \frac{-1+i}{s+i} + \frac{-1-i}{s-i} + k(s)$$

```
>> b=[2 1 0 -1];
```

```
>> a=[1 0 1];
```

```
>> [r p k]=residue(b,a)
```

```
r =
```

```
-1.0000 + 1.0000i
```

```
-1.0000 - 1.0000i
```

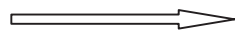
```
p =
```

```
0.0000 + 1.0000i
```

```
0.0000 - 1.0000i
```

```
k =
```

```
2 1
```


 $K(s)=2s+1$

```
>> r=[1 2 3];
```

```
>> p=[-1 -3 -5];
```

```
>> k=1;
```

```
>> [b,a]=residue(r,p,k)
```

```
b = 1 15 55 49
```

```
a = 1 9 23 15
```



$$\frac{b(s)}{a(s)} = \frac{s^3 + 15s^2 + 55s + 49}{s^3 + 9s^2 + 23s + 15}$$

دستور solve

این دستور برای حل معادلات غیر خطی به کار می رود.

$$ax^2+bx^2+c=0$$

فرض کنید می خواهیم معادله درجه ی دوم را در حالت کلی حل کنیم:

```
>> syms a b c x
```

```
>> solve('a*x^2+b*x+c=0');
```

```
>> syms a b c x
```

```
>> solve('a*x^2+b*x+c=0')
```

```
ans =
```

$$-(b + (b^2 - 4*a*c)^{(1/2)})/(2*a)$$

$$-(b - (b^2 - 4*a*c)^{(1/2)})/(2*a)$$

```
>> pretty(ans)
```

$$\begin{array}{r} \frac{-b + \sqrt{b^2 - 4ac}}{2a} \\ \frac{-b - \sqrt{b^2 - 4ac}}{2a} \end{array}$$

$$x^2-4x+2=0$$

مثال:

```
>> syms x
```

```
>> solve('x^2-4*x+2');
```

```
>> syms x
```

```
>> solve('x^2-4*x+2')
```

```
ans =
```

$$2^{1/2} + 2$$

$$2 - 2^{1/2}$$

```
>> pretty(ans)
```

$$\begin{array}{r} +- \qquad \qquad \qquad -+ \\ | \quad 1/2 \qquad \qquad | \\ | \quad 2 \quad + \quad 2 \quad | \\ | \qquad \qquad \qquad \qquad | \\ | \qquad \qquad \qquad 1/2 \quad | \\ | \quad 2 \quad - \quad 2 \quad | \\ +- \qquad \qquad \qquad -+ \end{array}$$

حل دستگاه با دستور solve

$$\begin{cases} x - y = 5 \\ x + y = 10 \end{cases}$$

```
>> syms x y
```

```
>> s=solve('x-y=5','x+y=10');
```

```
>> s=[s.x s.y]
```

```
s = [ 15/2, 5/2]
```

$$\begin{cases} x - \sin y = x \\ xy - \sin x = 1 \end{cases}$$

```
>> syms x y
```

```
>> s= solve('x-sin(y)=x','(x*y)-sin(x)=1');
```

```
>> s=[s.x s.y]
```

```
s =
```

```
[-1.5707963267948966192312084391505, 0]
```

دستور **dsolve** : دستور dsolve برای حل معادلات دیفرانسیل به کار می رود.

$$y' \rightarrow Dy \quad y'' \rightarrow D2y \quad y''' \rightarrow D3y$$

$$y' = xy$$

مثال:

```
>> dsolve('Dy=x*y','x')
```

```
ans = C5*exp(x^2/2)
```

$$y'' - 2y' + y = xe^{-x}, \quad y(0) = 1, \quad y'(-1) = -1$$

```
>> dsolve('D2y-2*Dy+y=x*exp(-x)','y(0)=1','Dy(-1)=-1')
```

```
ans =
```

```
piecewise([x*exp(-x) == exp(1) + 1, {x*exp(-x) - exp(1)*exp(t) + C4*t*exp(t)}], [x*exp(-x) ~ = exp(1) + 1, {}])
```

دستور **diff** : دستور مشتق در متلب diff می باشد.

```
>> syms x
```

```
>> diff(sin(x),5)  $\longrightarrow$  sin(5)(x)
```

```
ans = cos(x)
```

```
>> diff(sin(x))
```

```
ans = cos(x)
```

مشتق توابع دو متغیره

$$\frac{\partial}{\partial x} \rightarrow \text{diff}(f, x)$$

$$\frac{\partial}{\partial y} \rightarrow \text{diff}(f, y)$$

مثال: تابع $f(x,y) = \sin(xy) - x/y$ مفروض است، $\frac{\partial}{\partial x}$ ، $\frac{\partial}{\partial y}$ ، $\frac{\partial}{\partial x}$ و $\frac{\partial}{\partial y}$ را برای آن محاسبه کنید.

■ برای تعریف تابع از دستورات روبرو استفاده می کنیم:

```
@ f(x) - inline(f) - function
```

```
>> syms x y
```

```
>> f=@(x,y)(sin(x.*y)-x./y);
```

```
>> diff(f,x)
```

```
ans = y*cos(x*y) - 1/y
```

```
>> diff(f,y)
```

```
ans = x*cos(x*y) + x/y^2
```

```
>> diff(f,x,4)
```

```
ans = y^4*sin(x*y)
```

```
>> diff(f,y,3)
```

```
ans = (6*x)/y^4 - x^3*cos(x*y)
```

دستور **limit** : دستور حد در متلب ، **limit** می باشد.

$$\lim_{x \rightarrow \infty} \left(1 + \frac{1}{x}\right)^x = e$$

```
>> syms x
```

```
>> limit((1+1/x)^x,x,inf)
```

```
ans = exp(1)
```

$$\lim_{x \rightarrow 0} \frac{\sin x}{x} = 1$$

```
>> limit(sin(x)/x,x,0)
```

```
ans = 1
```

$$\lim_{x \rightarrow 0^-} \frac{|x|}{x} = -1$$

```
>> limit(abs(x)./x,x,0,'left')
```

```
ans = -1
```

دستورات انتگرال گیری در متلب به شرح زیر است:

- انتگرال معین و نامعین ← `int`
- انتگرال معین (به روش عددی) ← `quad-quadl`

$$\int \sin x$$

```
>> syms x
```

```
>> int(sin(x))                    یا                    >> int(sin(x),x)
```

```
ans = -cos(x)
```

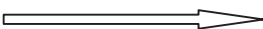
$$\int_0^{\pi} \sin(x) = 2$$

```
>> int(sin(x),0,pi)
```

```
ans = 2
```

یا

```
>> f=inline(sin(x));                    یا                    >> quad(f,0,pi);
```

```
>> quad(f,x)                                        جواب نمی دهد
```

```
>> quad(f,0,pi)
```

```
ans = 2.0000
```

■ دستور تعیین تابع نامشخص: `@(x)` - `inline`

■ معرفی تابع دو متغیره: `@(x,y)`

مثال:

$$\int x e^{ax} dx = ?$$

```
>> syms a x
```

```
>> int(x*exp(a*x),x)
```

```
ans = (exp(a*x)*(a*x - 1))/a^2
```

$$\int e^{ax} \sin(x) dx = ?$$

```
>> syms a x
```

```
>> int(exp(a*x)*sin(x),x)
```

```
ans = -(exp(a*x)*(cos(x) - a*sin(x)))/(a^2 + 1)
```

```
>> pretty(ans)
```

$$-\frac{\exp(ax) (\cos(x) - a \sin(x))}{a^2 + 1}$$

$$\int_1^{e^4} \int_0^x \frac{dy dx}{(x+y)^2} = ?$$

```
>> syms x y
```

```
>> int(int(1./(x+y)^2,y,0,x),x,1,exp(4))
```

```
ans = log(960500813064011/17592186044416)/2
```

```
>> vpa(ans,4)  $\Longrightarrow$  جواب را تا ۴ رقم اعشار نشان می دهد.
```

```
ans = 2.0
```

$$\int_0^1 \int_0^1 \int_0^1 xyz \, dx \, dy \, dz = ?$$

```
>> syms x y z
```

```
>> int(int(int(x*y*z,x,0,1),y,0,1),z,0,1)
```

```
ans = 1/8
```

سری ها:

دستور `symsum`

$$s = \sum_{k=0}^5 k^2 = 0^2 + 1^2 + 2^2 + 3^2 + 4^2 + 5^2$$

```
>> syms k
```

```
>> symsum(k^2)
```

```
ans = k^3/3 - k^2/2 + k/6
```

```
>> symsum(k^2,k,0,5)
```

```
ans = 55
```

دستور `sym`

```
>> sym(2/3)
```

```
ans = 2/3
```

```
>> sym(2/3,'d')
```

→ d: double

```
ans = 0.66666666666666662965923251249478
```

```
>> sym(2/3,'f')
```

→ f: float

```
ans = 6004799503160661/9007199254740992
```

- مجموع اعداد فرد: n^2
- مجموع اعداد زوج: n^2+n
- مجموع n عدد متوالی: $\sum_{x=1}^n x = \frac{n(n+1)}{2}$

تمرین: مجموع ۲۰ عدد فرد متوالی و ۲۰ عدد زوج متوالی را بدست آورید.

سری تیلور:

$$s = \sum_{k=0}^{\infty} \frac{x^k}{k!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

```
>> syms k x
```

```
>> symsum((x^k)/sym('k!'),k,0,inf)
```

```
ans = exp(x)
```

$$f(z) = \sum_{n=0}^{\infty} a_n (z - z_0)^n = a_0 + a_1 (z - z_0) + \dots$$

بسط مک لورن:

$$a_n = \frac{f^{(n)}(z)}{n!} \quad z_0=0$$

- $e^z = 1 + z + \frac{z^2}{2!} + \frac{z^3}{3!} + \dots = \sum_{n=0}^{\infty} \frac{z^n}{n!}$
- $\sin z = z - \frac{z^3}{3!} + \frac{z^5}{5!} - \dots = \sum_{n=0}^{\infty} \frac{(-1)^n z^{2n+1}}{(2n+1)!}$
- $\cos z = 1 - \frac{z^2}{2!} + \frac{z^4}{4!} - \dots = \sum_{n=0}^{\infty} \frac{(-1)^n z^{2n}}{(2n)!}$

```
>> syms x
```

```
taylor(sin(x))
```

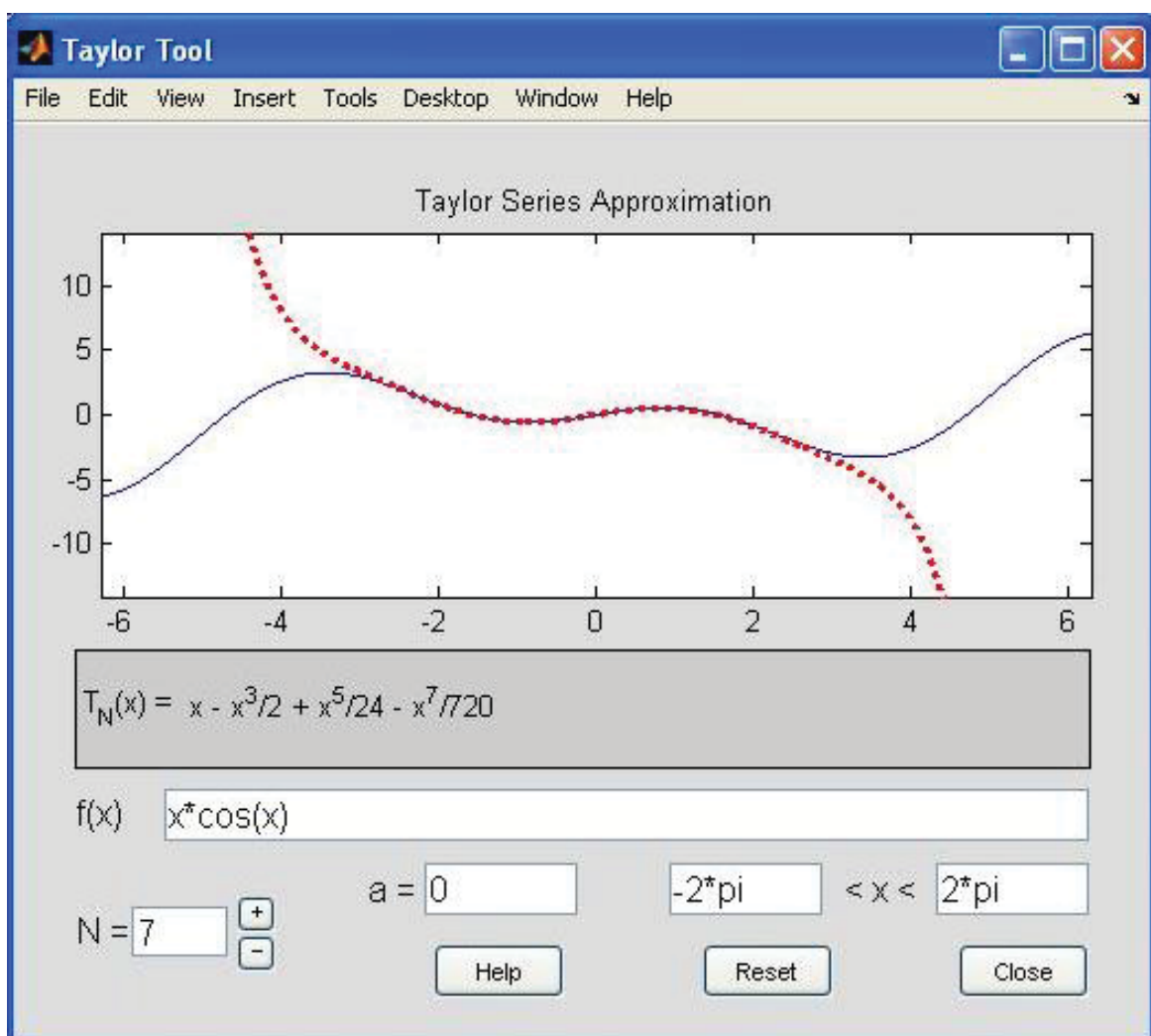
```
ans = x^5/120 - x^3/6 + x
```

```
>> taylor(1/5+4*cos(x))
```

```
ans = x^4/6 - 2*x^2 + 21/5
```

taylor tool: این دستور را در پنجره `command` نوشته و اجرا کنید. پنجره ای باز می شود که در آن خط آبی، نمودار اصلی و خط قرمز، تقریب را نشان می دهد.

```
>> taylor tool
```



طول یک خط در فضا:

$$l = \int_a^b \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2 + \left(\frac{dz}{dt}\right)^2}$$

نمایش پارامتری خط در فضا

$$l: \frac{x-x_0}{a} = \frac{y-y_0}{b} = \frac{z-z_0}{c} = t \quad \rightarrow \quad \begin{cases} x = at + x_0 \\ y = bt + y_0 \\ z = ct + z_0 \end{cases}$$

مثال: برای $1 < t < 2$ و $n=25$ طول خط زیر را بیابید:

$$\begin{cases} x = 2t \\ y = t^2 \\ z = \ln t \end{cases}$$

```
>> t=linspace(1,2,25);
```

```
>> sum(sqrt(diff(2.*t).^2+diff(t.^2).^2+diff(log(t).^2)))
```

```
ans = 4.9149
```

■ دستورات از داخل به خارج داده می شود.

لاپلاس:

```
>> laplace(f)
```

$$F(s) = \int_0^{\infty} f(x) e^{-st} dx$$

معکوس لاپلاس:

```
>> ilaplace(f)
```

مثال:

$$f=1/t \xrightarrow{L} ?$$

```
>> syms t
```

```
>> f=1/t;
```

```
>> laplace(f)
```

```
ans = laplace(1/t, t, s)
```

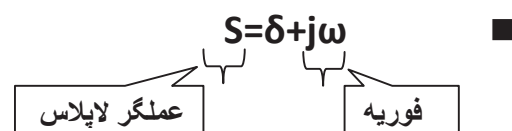
$$f(s)=1/s^2+4 \xrightarrow{L^{-1}} ?$$

```
>> syms s
```

```
>> f=1/(s^2+4);
```

```
>> ilaplace(f)
```

```
ans = sin(2*t)/2
```



تبدیل فوریه :

```
>> fourier(f)
```

معکوس فوریه:

```
>> ifourier(f)
```

مثال:

$$f(x) = e^{-x^2}$$

```
>> syms x
```

```
>> f=exp(-x^2);
```

```
>> fourier(f)
```

```
ans = pi^(1/2)*exp(-w^2/4)
```

مثال: معکوس تبدیل فوریه تابع زیر را بیابید.

$$F(\omega) = e^{-\frac{\omega^2}{4}}$$

```
>> syms w
```

```
>> f=exp(-w^2/4);
```

```
>> ifourier(f)
```

```
ans = exp(-x^2)/pi^(1/2)
```

تابع گاما:

```
>> gamma(x)
```

$$\Gamma(x) = \int_0^{\infty} e^{-x} \cdot x^{a-1} \cdot dx$$

$$\begin{cases} \Gamma(n+1) = n! & n \in \mathbf{N} \\ \Gamma(n+1) = n\Gamma(n) \end{cases}$$

مثال:

```
>> syms t
```

```
>> u=diff(gamma(t^2-1))
```

```
u = 2*t*gamma(t^2 - 1)*psi(t^2 - 1)
```

```
>> subs(t,1)
```

```
ans = 1
```

نکته: $\Gamma\left(\frac{1}{2}\right) = \sqrt{\pi}$

>> gamma(1/2) ans = 1.7725

>> sqrt(pi) ans = 1.7725

تبدیل فوریه گسسته: DFT

$$F_k = \sum_{n=0}^{N-1} f_n e^{-ik\omega n} \quad \text{for } k = 0 \sim N-1$$

$$f_n = \frac{1}{N} \sum_{k=0}^{N-1} F_k e^{ik\omega n} \quad \text{for } n = 0 \sim N-1$$

تبدیل فوریه FFT : (Fast fourier transform)

$$F_k = \sum_{n=0}^{(N/2)-1} f_n e^{-i(2\pi/\omega)kn} + \sum_{n=N/2}^{N-1} f_n e^{-i(2\pi/N)kn}$$

$$F_{2k} = \sum_{n=0}^{(N/2)-1} (f_n + f_{(n+N/2)}) \omega^{2kn} \quad \text{زوج}$$

$$F_{2k+1} = \sum_{n=0}^{(N/2)-1} (f_n - f_{(n+N/2)}) \omega^n \omega^{2kn} \quad \text{فرد}$$

توان سیگنال:

$$P = \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} f(t)^2 dt$$

تبدیل فوریه گسسته:

$$X(k) = \sum_{j=1}^N x(j) \omega_N^{(j-1)(k-1)}$$

$$X(j) = \frac{1}{N} \sum_{k=1}^N x(k) \omega_N^{-(j-1)(k-1)}$$

$$\omega_N = e^{(2\pi i)/N}$$

حل دستگاه معادلات دیفرانسیل خطی:

مثال:

$$\begin{cases} y'_1 = 2y_1 - 5y_2 \\ y'_2 = 5y_1 - 6y_2 \end{cases}$$

```
>> syms y1 y2
```

```
>> [y1,y2]=dsolve('Dy1=2*y1-5*y2','Dy2=5*y1-6*y2')
```

$$y_1 = -(\exp(-2*t)*(3*C2*\cos(3*t) - 4*C1*\cos(3*t) + 3*C1*\sin(3*t) + 4*C2*\sin(3*t)))/5$$

$$y_2 = \exp(-2*t)*(C1*\cos(3*t) - C2*\sin(3*t))$$

روش اویلر در حل مسائل:

$$\begin{cases} \frac{dy}{dt} = \frac{y(t+h)-y(t)}{h} \\ f(y, t) = \frac{dy}{dt} \end{cases} \implies y_{i+1} = y_i + hf(y_i, f_i)$$

h: طول گام حرکت

مثال:

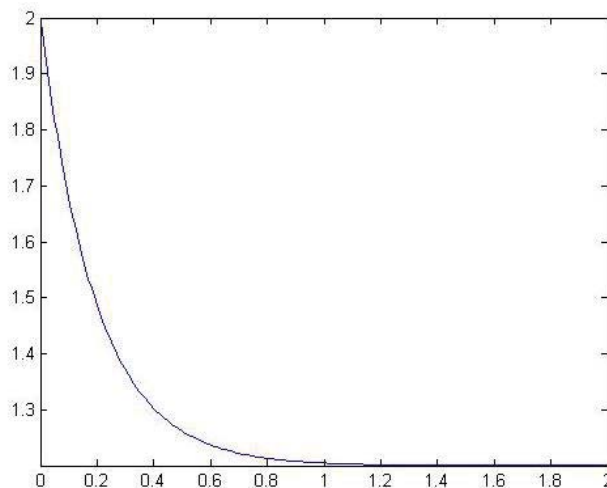
$$Y(t+1)=2 ; \quad dy/dt+5y=6 ; \quad h=1/100$$

```

h=0.01;
t0=0;
tf=2;
t=t0:h:tf;
y(1)=2;
N=length(t)
for i=1:N-1
    y(i+1)=y(i)+h*(6-5*y(i));
end
plot(t,y)

```

این دستور بیشترین طول یک ماتریس/ بردار را نشان می دهد. \longrightarrow



روش رانگه کوتا: (runge-kutta)

■ دقت این روش از روش اویلر بیشتر می باشد.

■ الگوریتم محاسباتی این روش ode45 و ode23 و خطا از مرتبه $o(h^3)$ یا $o(h^3)$

$$y_{i+1} = y_i + h/4(k_1 + 2k_2 + 2k_3 + k_4)$$

$$k_1 = f(t_i, y_i)$$

$$k_2 = f\left(t_i + \frac{h}{2}, y_i + \frac{h}{2} * k_1\right)$$

$$k_3 = f\left(t_i + \frac{h}{2}, y_i + \frac{h}{2} * k_2\right)$$

$$k_4 = f(t_i + h, y_i - h * k_2)$$

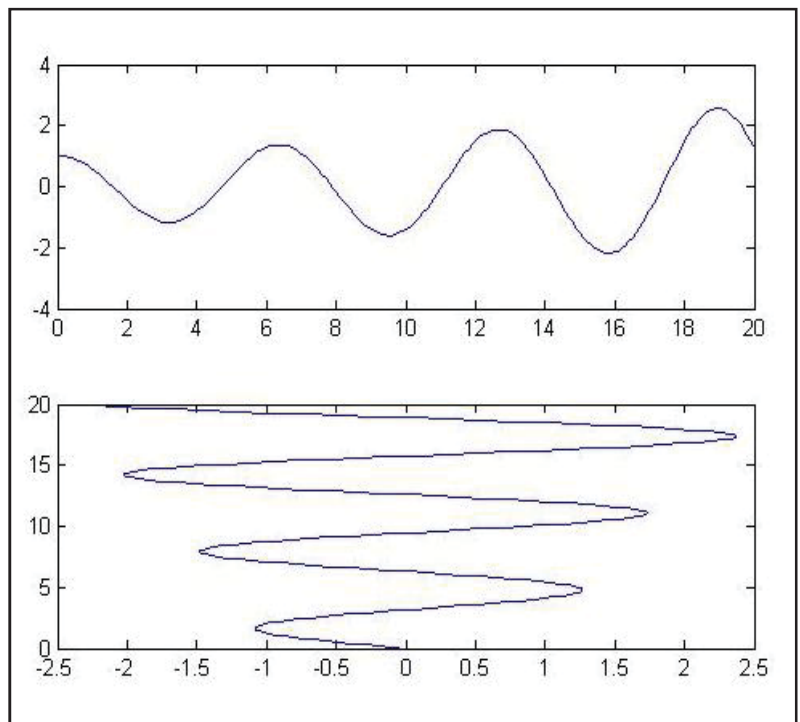
■ قانون هوك $\omega = \sqrt{k/m}$ فرکانس حرکت زاویه ای

حل مسئله به روش اویلر:

$$\text{معادله مکان: } \frac{d^2x}{dt^2} + \frac{k}{m}x = 0$$

در scrip می نویسیم:

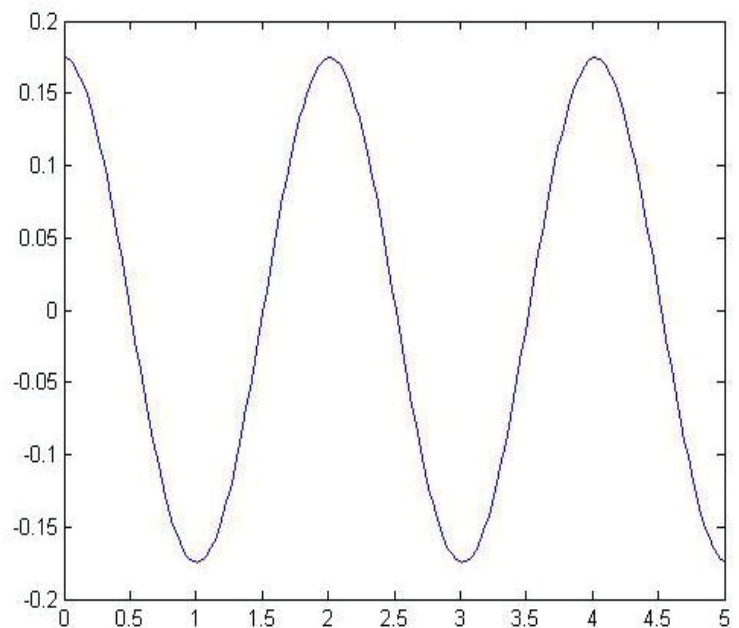
```
%d^2(x)/dt^2=-w^2*x
%w=sqrt(k/m)
%i,cxo=x(t,0)=1,r(0)=r(t,0)=0
%simple harmonic oscillator-euler's method colsall
k=1;m=1;
w=sqrt(k/m);
h=0.1;
t=0:h:20;
n=length(t);
x=zeros(1,n);
r=zeros(1,n);
x(1)=1;
r(1)=0;
for i=1:n-1;
    x(i+1)=x(i)+h*r(i);    معادله مکان
    r(i+1)=r(i)-h*w^2*x(i);    معادله سرعت
end
subplot(2,1,1)
plot(t,x)
subplot(2,1,2)
plot(r,t)
```



```

%harkate arang
%runge-kutta o(h3)
g=9.8; l=1; n=200;
w=sqrt(g/l);
t=linspace(0,5,n); %time between 0-5 second
h=t(2)-t(1); %time stop
f=@(alpha)alpha;
g=@(theta)-w^2*sin(theta);
theta=zeros(1,n);
alpha=zeros(1,n);
theta(1)=10*pi/180; %initial angle
alpha(1)=0; %initial angle speed
for i=1:n-1;
    k1=f(alpha(i));
    kp1=g(theta(i));
    k2=f(alpha(i)+h/2*kp1);
    kp2=g(theta(i)+h/2*k1);
    k3=f(alpha(i)+h/2*kp2);
    kp3=g(theta(i)+h/2*k2);
    k4=f(alpha(i)+h*kp2);
    kp4=g(theta(i)+h*k2);
    theta(i+1)=theta(i)+h/6*(k1+2*k2+2*k3+k4);
    alpha(i+1)=alpha(i)+h/6*(kp1+2*kp2+2*kp3+kp4);
end
plot(t,theta)

```



$F(x)=0$ ریشه حدسی $x_0=1$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

این رابطه تکرار می شود پس حلقه داریم و شرط خروج از حلقه $|x_{n+1} - x_n| < \varepsilon$ می باشد.

File → new → script

```
function [xn,k,err]=newton (f,x0,e)
syms x h
g=inline(limit(((f(x+h)-f(x))/h),h,0));
k=1;
xn=x0-(f(x0)/g(x0));
while abs(xn-x0)>e
    disp(xn)
    x0=xn;
    xn=x0-(f(x0)./g(x0));
    k=k+1;
end
err=abs(xn-x0);
disp(xn)
```

در پنجره command :

```
>> f=inline('sin(x)-x+0.5');
```

```
>> [xn,k,err]=newton(f,pi/2,0.0001)
```

```
xn=1.49
```

```
K=3
```

$$x_2 = \frac{x_0 f(x_1) - x_1 f(x_0)}{f(x_1) - x_1 f(x_0)}$$

File → new → script

```
function [x2,k,err]=vatari(f,x0,x1,e)
k=1;
x2=((x0*f(x1)-x1*f(x0))/(f(x1)-f(x0)));
while abs(x2-x1)>e
    disp(x2);
    x0=x1;
    x1=x2;
    x2=((x0*f(x1)-x1*f(x0))/(f(x1)-f(x0)));
    k=k+1;
end
disp(x2)
err=abs(x2-x1)
```

این روش را برای تابعی با شرایط زیر در پنجره command اجرا می کنیم.

$f(x)=x^2-2$ [1,2] $err=10^{-3}$

```
>> f=inline('x^2-2')
```

```
>> [x2,k,err]=vatari(f,1,2,0.001)
```

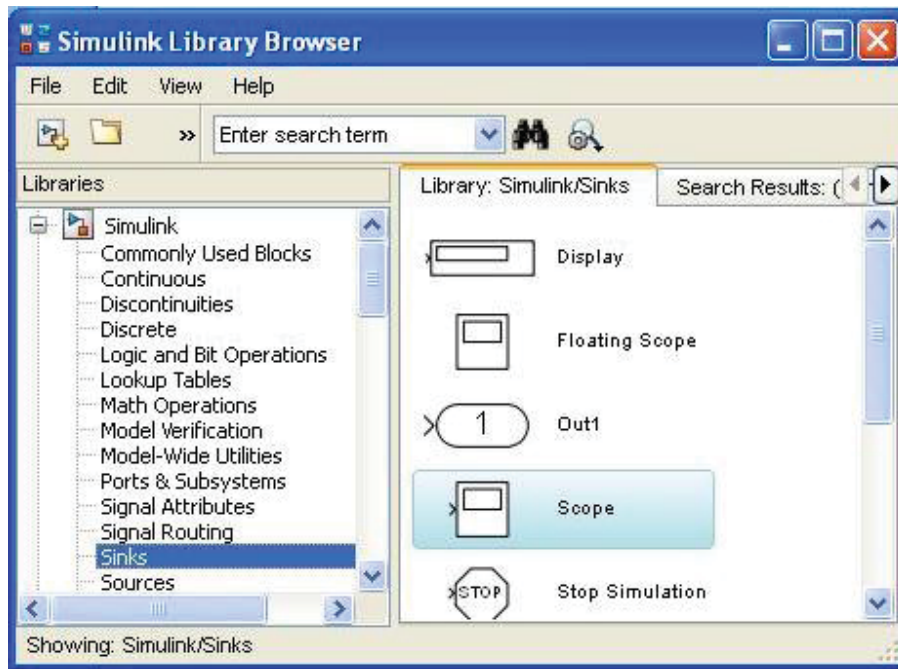
```
x2 = 1.4142
```

```
k = 4
```

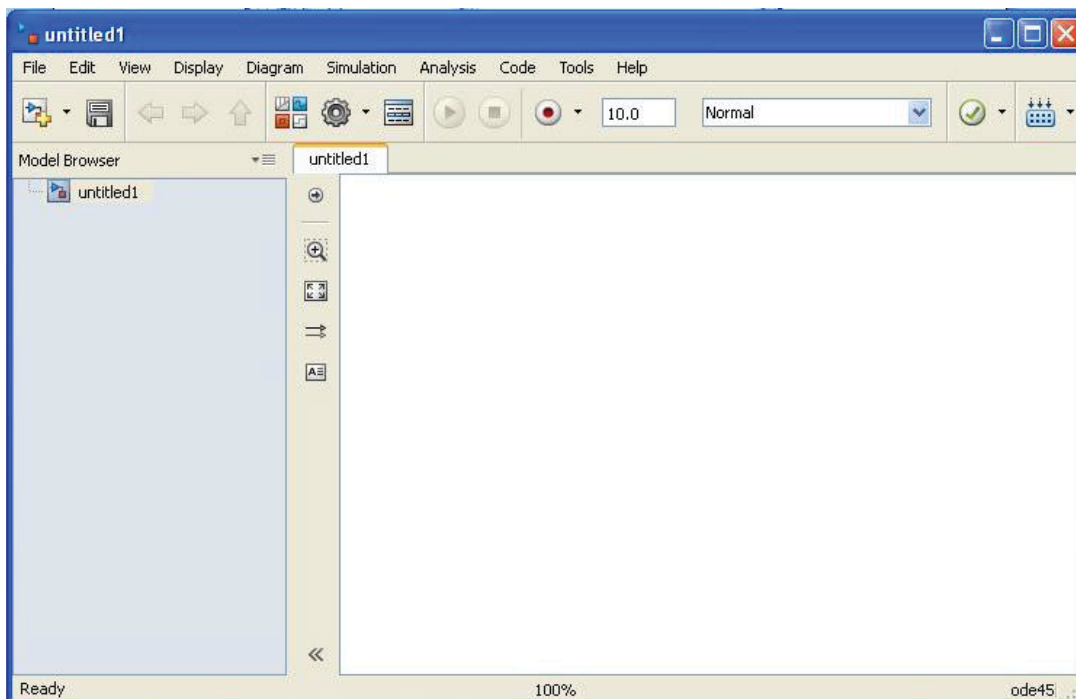
```
err = 4.2271e-04
```

سیمولینک (simulink) :

ابتدا بر روی آیکن  کلیک می‌کنیم تا پنجره ی Simulink Library Browser باز شود.



سپس از مسیر file/ new/ model پنجره ی سفیدی برای ایجاد مدل باز می‌شود.

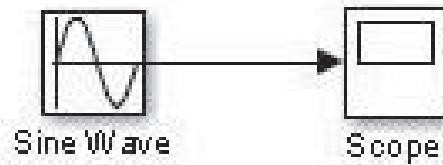


حال می توانیم بلوک های مورد نیاز را از قسمت Simulink Library Browser یافته و در این صفحه درگ کنیم.


مثال: می خواهیم موج ایجاد شده توسط یک منبع سینوسی را به وسیله ی اسکوپ مشاهده کنیم. بدین منظور ابتدا از طریق مسیرهای زیر بلوک های مورد نیاز را یافته و مدار مورد نظر را طراحی می کنیم.

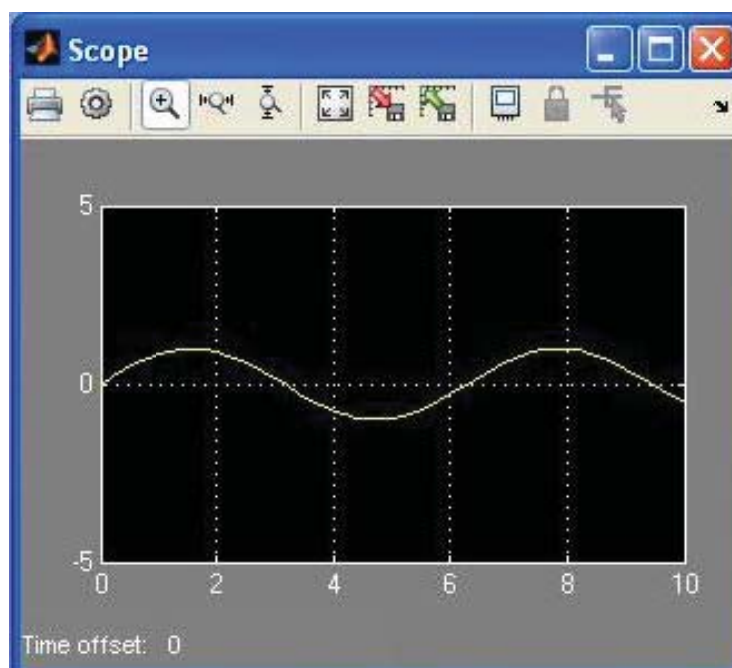
Simulink/ sources/ sine wave


Simulink/ sinks/ scope

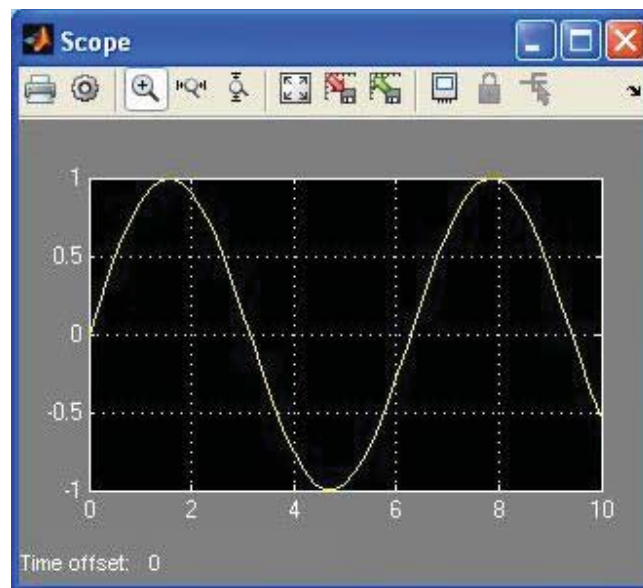


نکته: برای تغییر تنظیمات هرکدام از بلوک ها بر روی آن دابل کلیک می کنیم تا پنجره ی میانبر properties مربوط به آنها باز شود و سپس می توانیم برخی از تنظیمات را تغییر دهیم.

برای اجرای مدل طراحی شده از نوار بالای صفحه ی model بر روی آیکن  (Run) کلیک می کنیم. سپس روی scope دابل کلیک کرده و بدین ترتیب موج ایجاد شده مشاهده می شود.



نکته: با کلیک روی آیکن  (Autoscale) واقع بر روی صفحه ی scope می توان موج ظاهر شده را واضح تر مشاهده کرد.

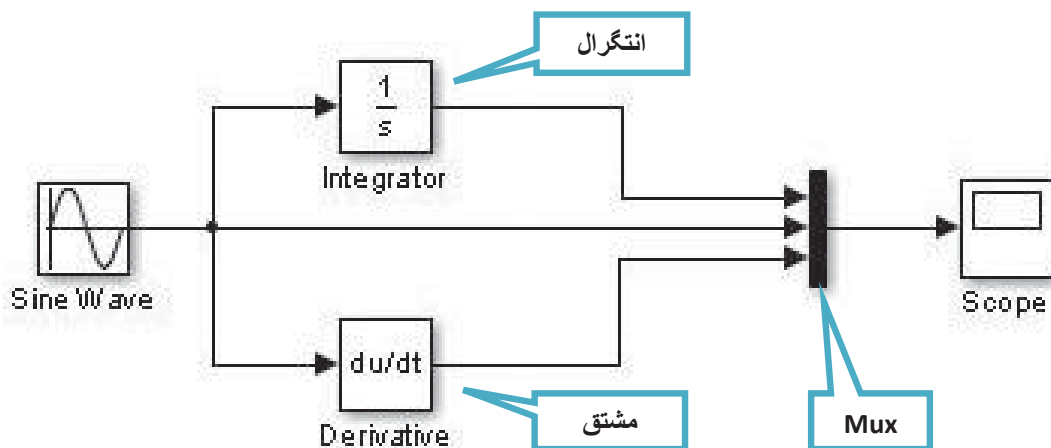


برخی از ویژگیهای سیمولینک:

- برای انتخاب اسیلوسکوپ (scope) در سیمولینک از بلوک sinks استفاده می شود.
- برای انتخاب صفحه نمایش (display) سیمولینک از بلوک sinks استفاده می شود.
- برای انتخاب یک مقدار ثابت (constant) در سیمولینک به بلوک commonly Used Blocks مراجعه می شود.
- برای کار با اعداد مختلط از بلوک math operations در سیمولینک استفاده می شود.
- مدهای STL - Accelerator – Normal : simulations
- پسوند فایل های simulink برای ذخیره سازی mdl می باشد.
- زمان شبیه سازی به صورت پیش فرض 10 ثانیه می باشد.
- برای رفتن به تنظیمات مدل در سیمولینک از دستور میانبر `ctrl + E` استفاده می شود.
- برای تحلیل معادلات حالت در سیمولینک از بلوک Stateflow استفاده می شود.
- برای ایست دادن به شبیه سازی در سیمولینک به بلوک sinks مراجعه می شود.
- برای ایجاد یک زیرسیستم در سیمولینک (subsystem) از گزینه های منو Edit انتخاب می شود.
- برای ایجاد یک مدل جدید در سیمولینک از دستور میانبر `ctrl + N` استفاده می شود.

مثال:

می خواهیم مدار زیر را در simulink طراحی و اجرا کنیم.



بلوک های مورد نیاز را از مسیرهای زیر می یابیم.

Simulink/ Sources/ Sine Wave

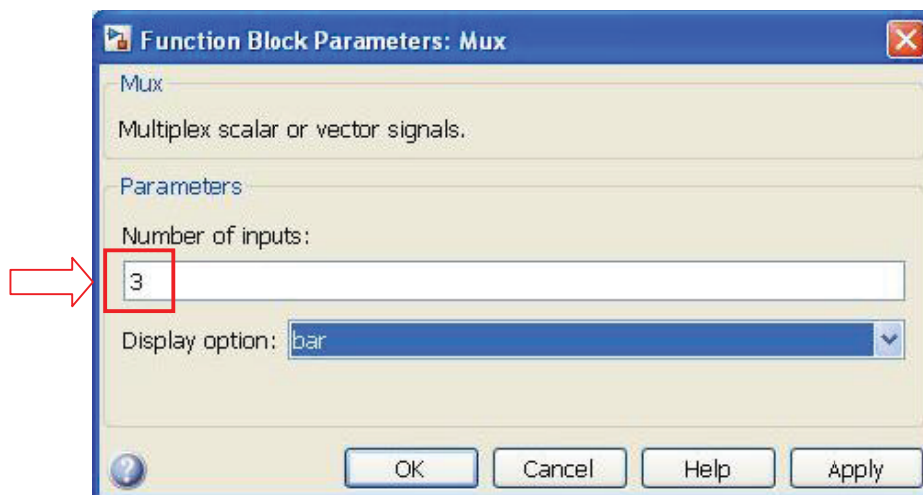
Simulink/ Continuous/ Integrator


Simulink/ Continuous/ Derivative

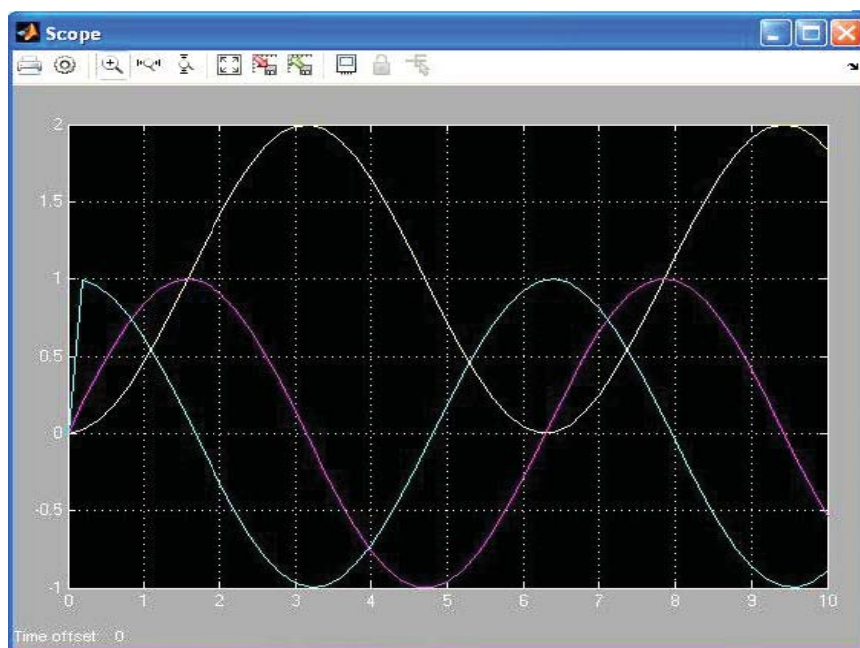
Simulink/ Signal Routing/ Mux

Simulink/ Sinks/ Scope

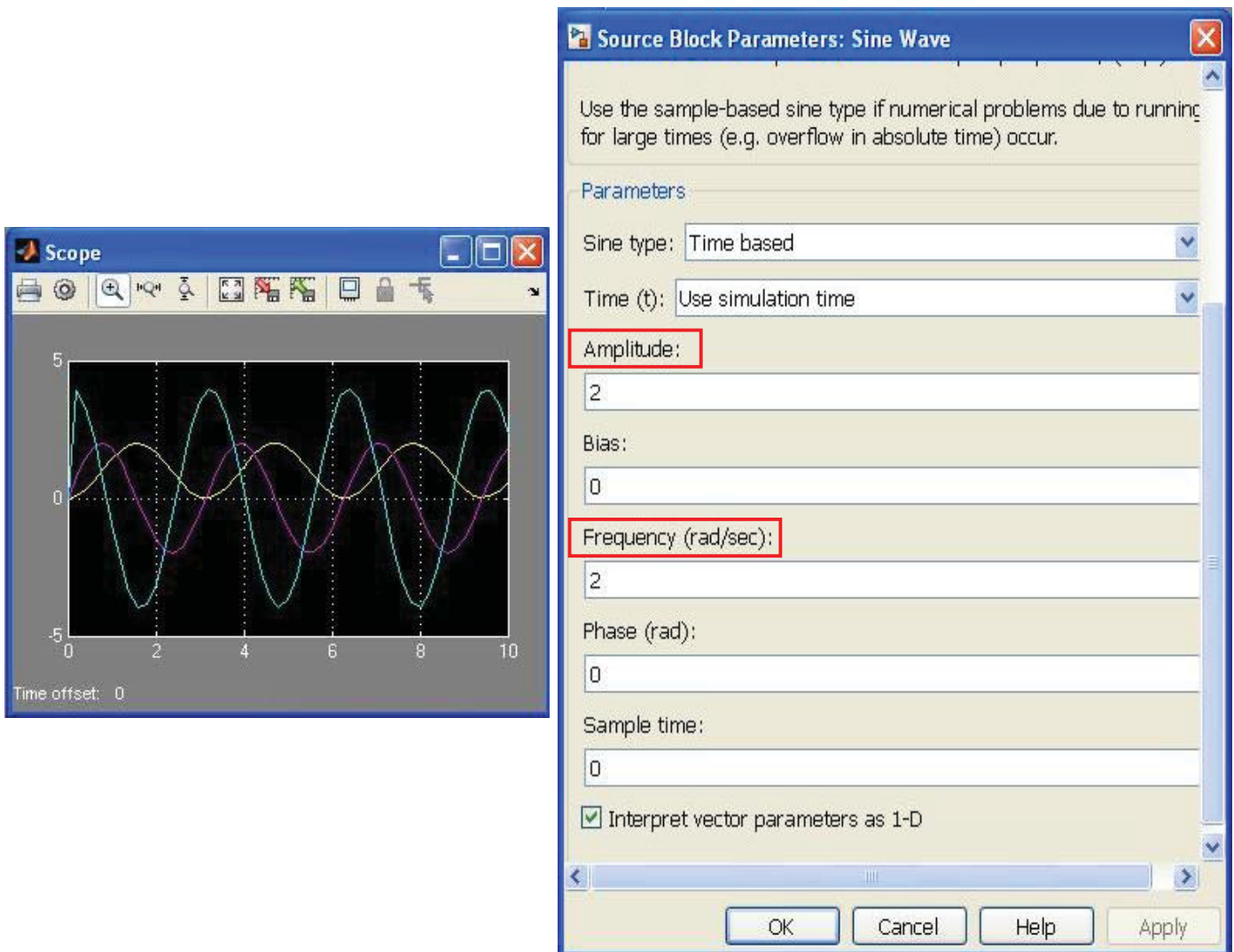
توجه: Mux دارای دو ورودی است. برای ایجاد ورودی های بیشتر، در اینجا ۳ ورودی، بر روی آن دابل کلیک کرده و در پنجره ی باز شده پارامتر Number of inputs را از عدد ۲ به عدد ۳ تغییر می دهیم.



سپس با کلیک روی آیکون  آن را اجرا کرده و بر روی اسکوپ دابل کلیک می کنیم تا هر سه موج ظاهر شود. برای مشاهده بهتر روی گزینه Autoscale کلیک می کنیم.



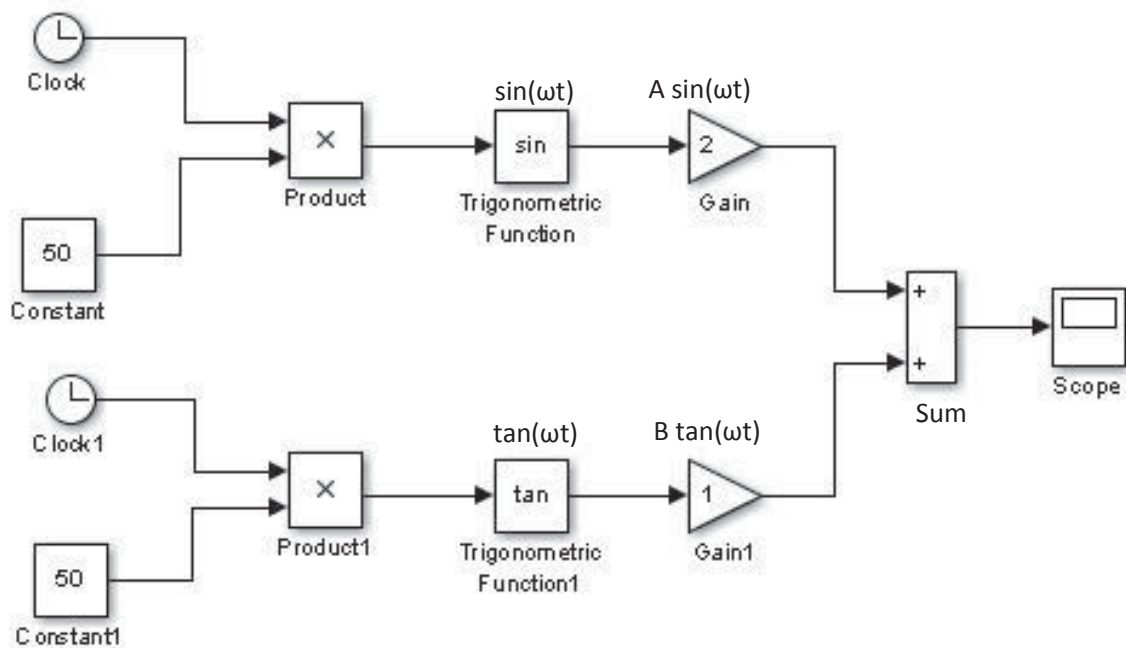
می توان فرکانس و دامنه منبع را تغییر داده و تغییرات نمودار را مشاهده و بررسی کنیم. برای این منظور روی منبع سینوسی دابل کلیک کرده و دامنه (Amplitude) و فاز (Frequency) آن را تغییر می دهیم.



نکته: با فعال کردن تیک legend در تنظیمات scope ، راهنمای legend بر روی صفحه scope نمایش داده میشود.

مثال: برای رابطه ی $y=A \sin(\omega t) + B \tan(\omega t)$ مداری طراحی کرده و آن را حل کنید.

به عنوان مثال رابطه ی $y=2 \sin(50t) + \tan(50t)$ را در نظر می گیریم. برای حل چنین مسائلی ابتدا ضرب و تقسیم را طراحی و ایجاد کرده و سپس به جمع و تفریق می پردازیم.



بلوک های مورد نیاز را می توان از مسیرهای زیر انتخاب کرد.

Simulink/ Sources/ Clock

Simulink/ Sources/ Constant

Simulink/ Math Operations/ Product

Simulink/ Math Operations/ Trigonometric Function

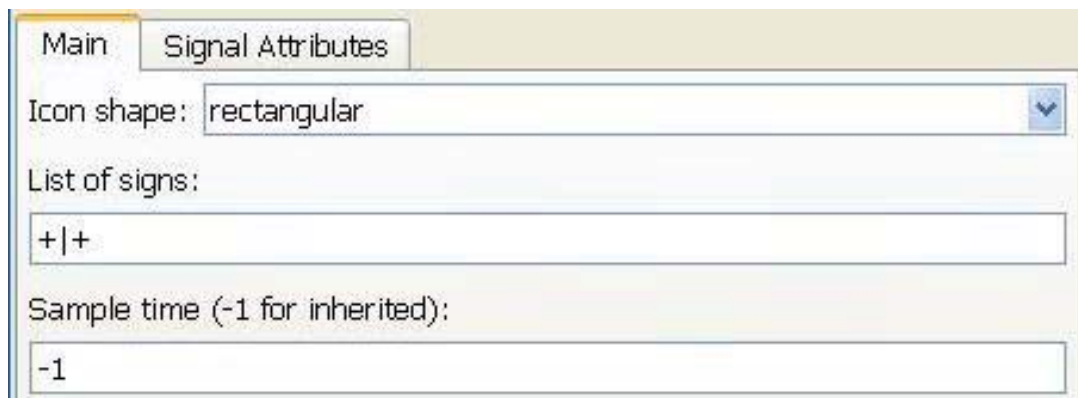
Simulink/ Math Operations/ Gain

Simulink/ Math Operations/ Sum

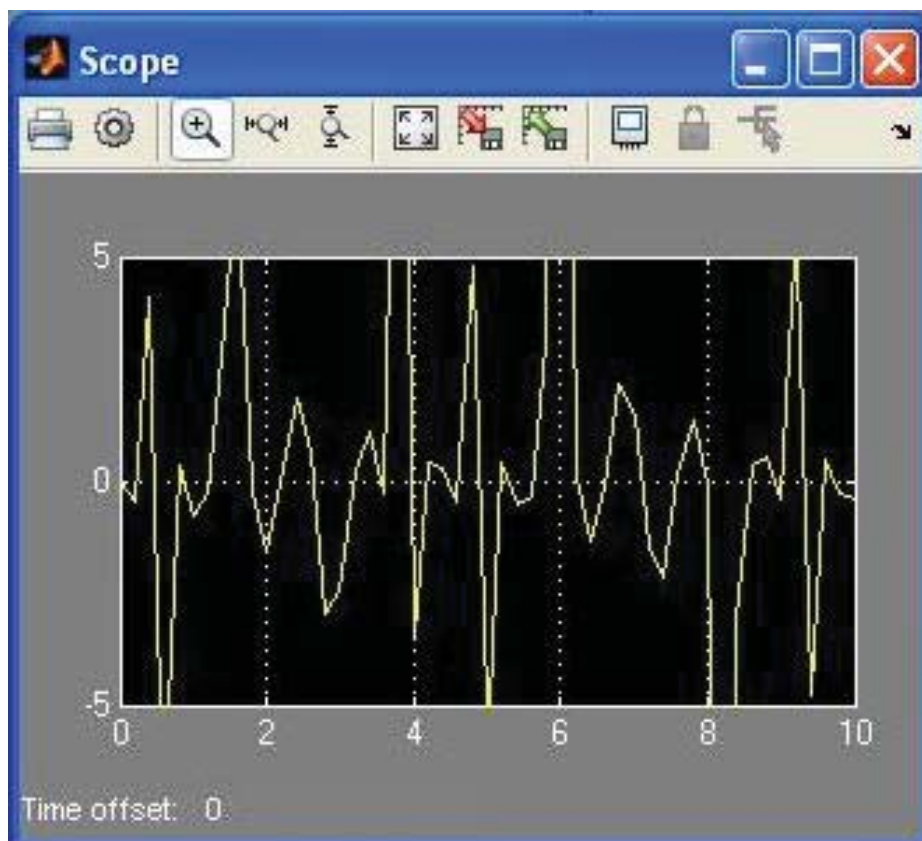
Simulink/ Sinks/ Scope

برای ایجاد tan بر روی Trigonometric function دابل کلیک کرده و در قسمت function تابع tan را انتخاب می کنیم.

می توان با دابل کلیک بر روی sum برخی از تنظیمات آن را به صورت زیر تغییر داد:

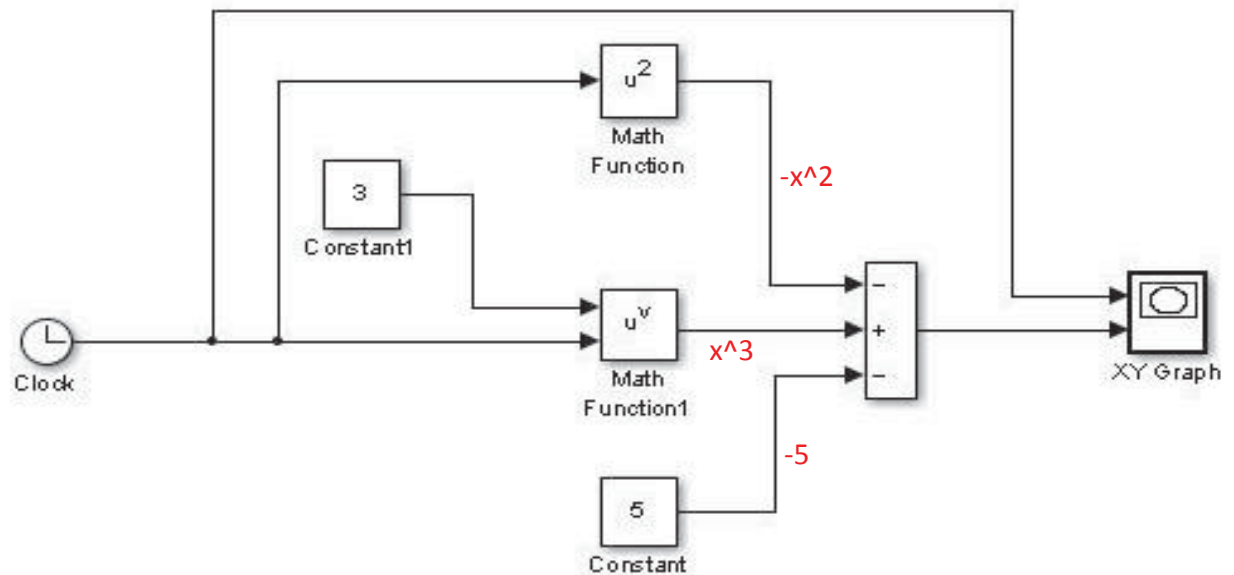


برای ایجاد کردن ۲ روی Gain متصل به sin دابل کلیک کرده و در قسمت Gain عدد ۲ را وارد می کنیم. برای وارد کردن مقدار ω که برابر با ۵۰ است بر روی constant دابل کلیک کرده و در قسمت constant value عدد ۵۰ را وارد می کنیم. حال می توانیم این مدار را Run کنیم.



مثال: معادله مقابل را شبیه سازی کنید.

$$y = x^3 - x^2 - 5$$



با پیمودن مسیرهای زیر بلوک های مورد نیاز را بیابید.

Simulink/ Sources/ Clock

Simulink/ Sources/ Constant

Simulink/ Math Operations/ Sum

Simulink/ Math Operations/ Math Function

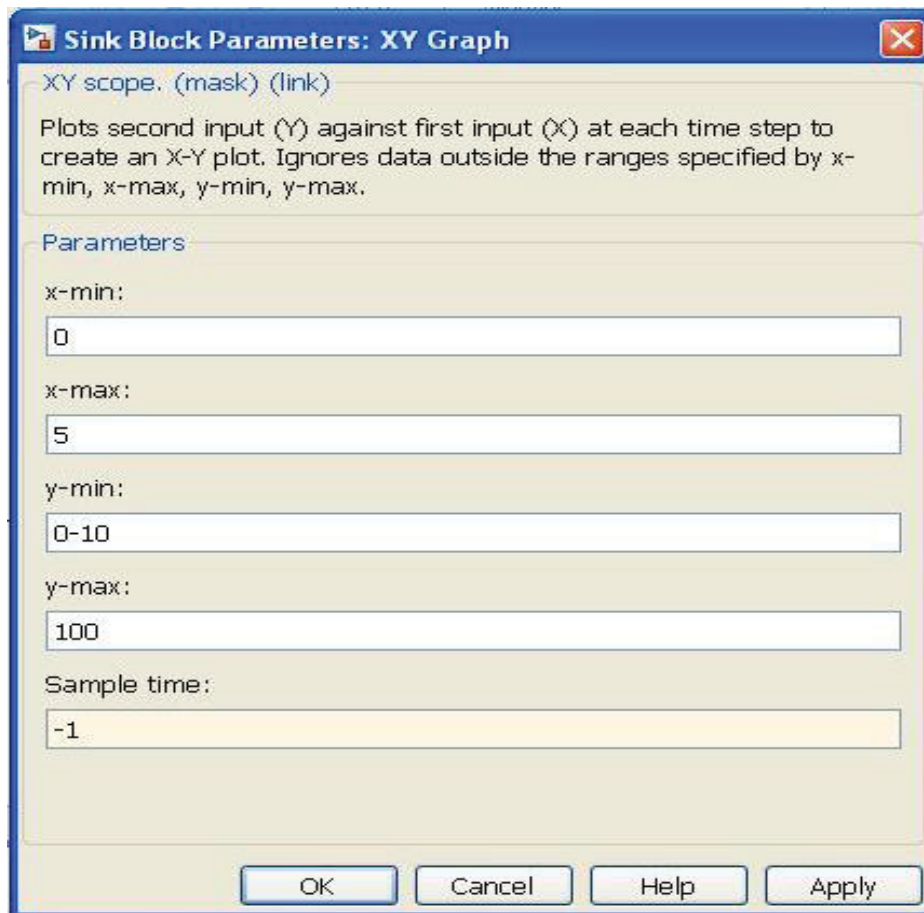
Simulink/ Sinks/ XY Graph

باید قبل از اجرای مدل برخی از تنظیمات را به صورت زیر تغییر دهید.

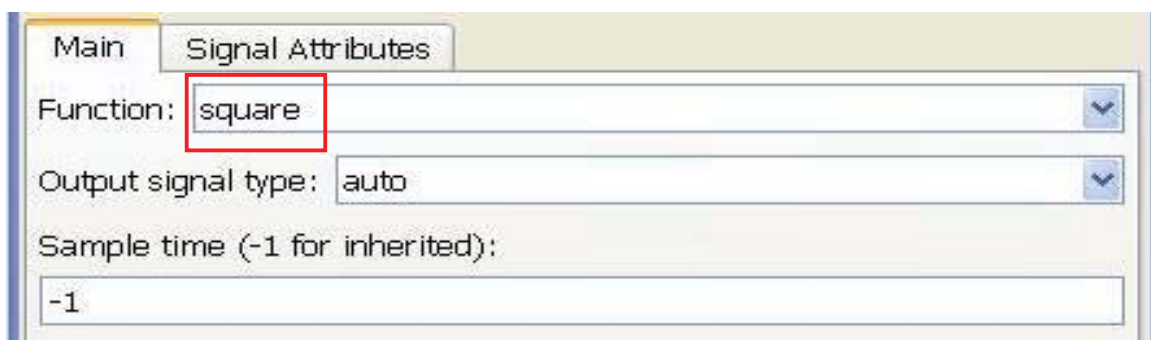
- روی sum دابل کلیک کرده و برخی از تنظیمات را به صورت زیر تغییر دهید.



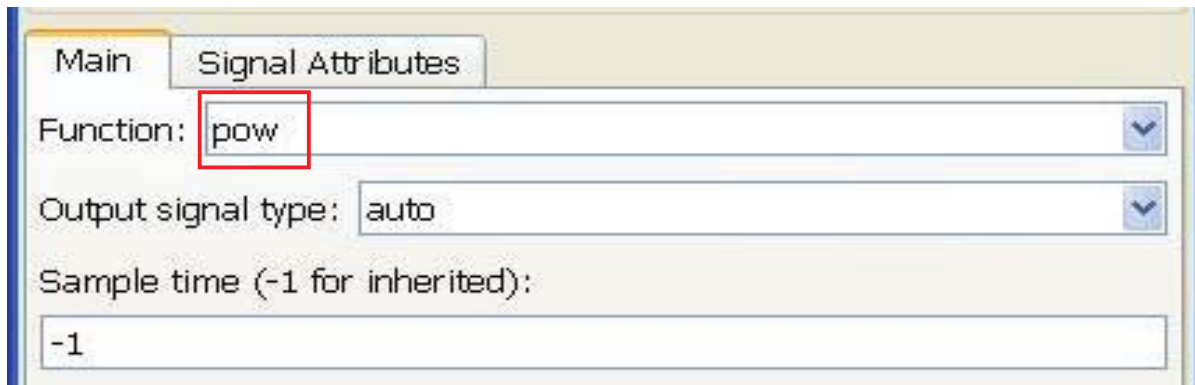
- بر روی XY Graph دابل کلیک کرده و تنظیمات آنرا به صورت زیر تغییر دهید.



- بر روی math function دابل کلیک کرده و مقدار function را به صورت زیر تغییر می دهیم.



- روی math function1 دابل کلیک کرده و مقدار function را به صورت زیر تغییر می دهیم.



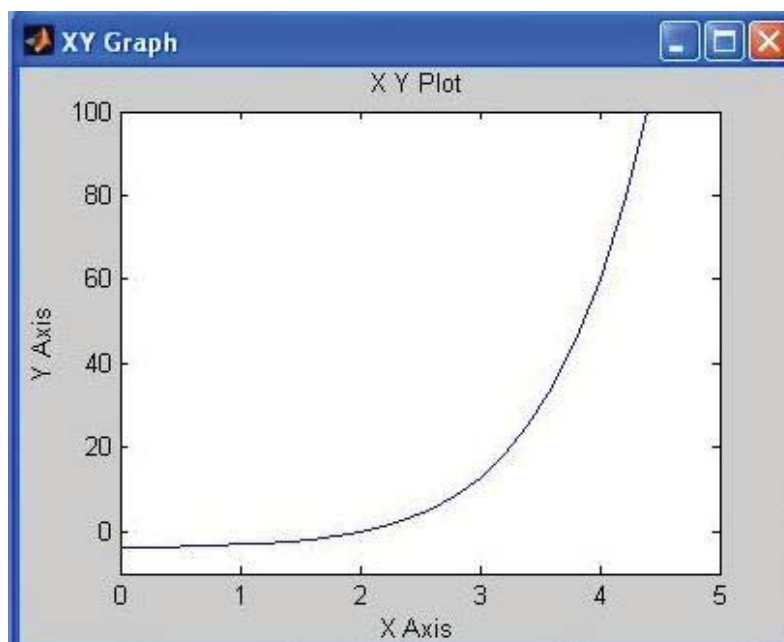
توجه :

Pow → نمای ۳

Square → نمای ۲

- روی constant دابل کلیک کرده و مقدار constant value را برابر با 5 قرار دهید.
- روی constant1 دابل کلیک کرده و مقدار constant value را برابر با 3 قرار دهید.

پس از اجرا نمودار زیر مشاهده می شود.



مثال:

$F=ma$

$F-cv^2 = ma$

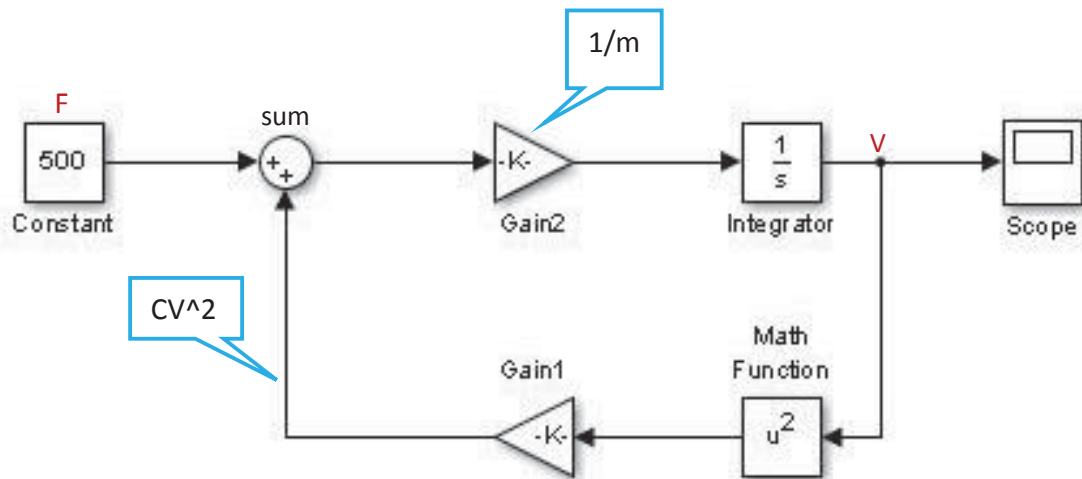
$F-cv^2=m(dv/dt)$

$(F-cv^2)/m=dv/dt$

$M=1000\text{kg}$

$c=0.06$

$F=500\text{N}$



• Gain را می توان با $R + ctri$ برگرداند.

می توانید با پیمودن مسیرهای زیر بلوک های مورد نیاز را بیابید.

Simulink/ Sources/ Constant

Simulink/ Math Operations/ Sum

Simulink/ Math Operations/ Gain

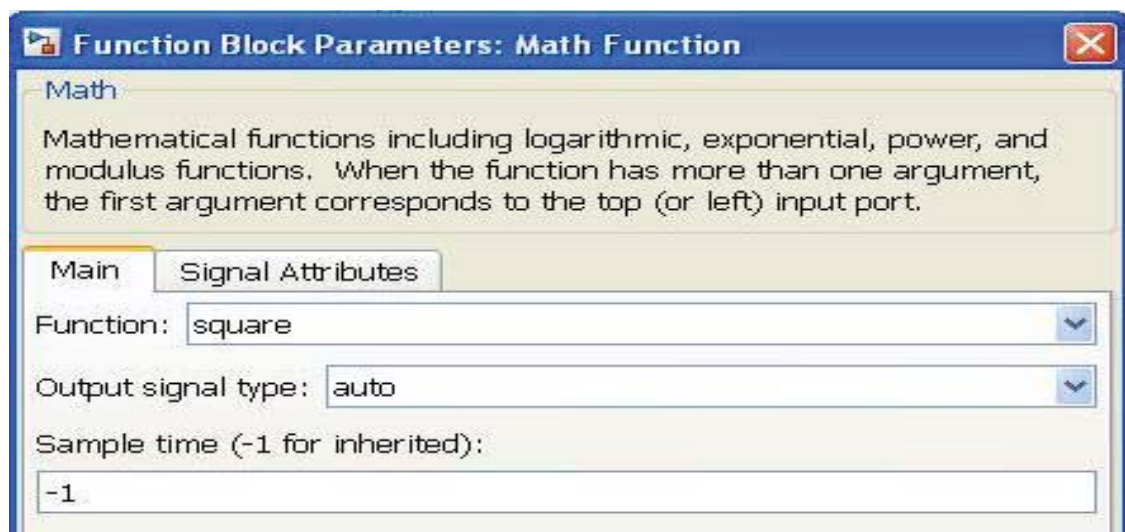
Simulink/ Math Operations/ Math Function

Simulink/ Sinks/ Scope

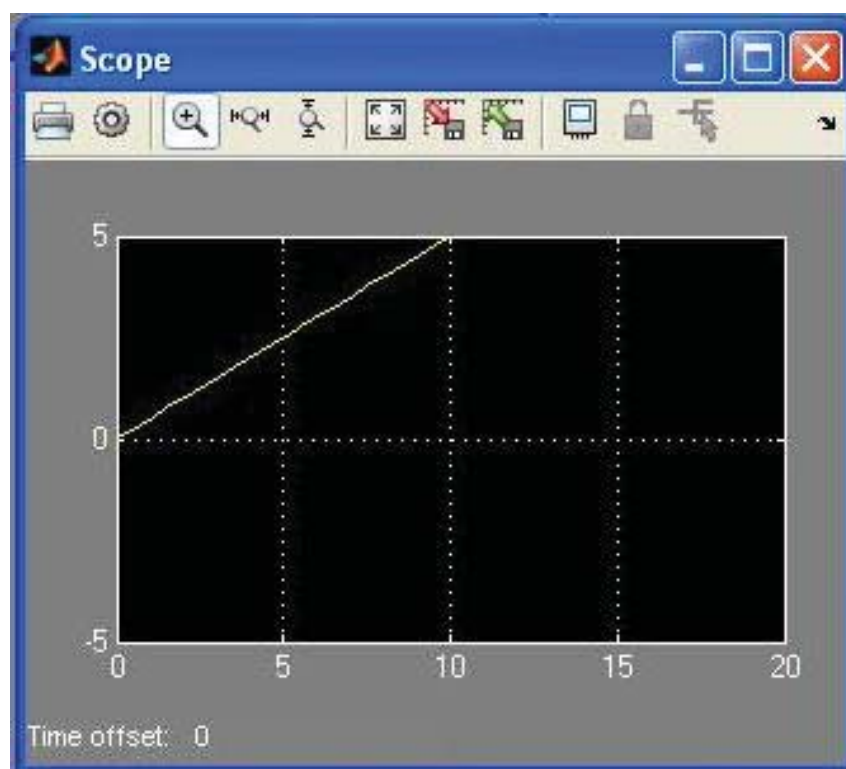
Simulink/ Continuous/ Integrator

قبل از اجرای مدل ، تنظیمات زیر لازم است:

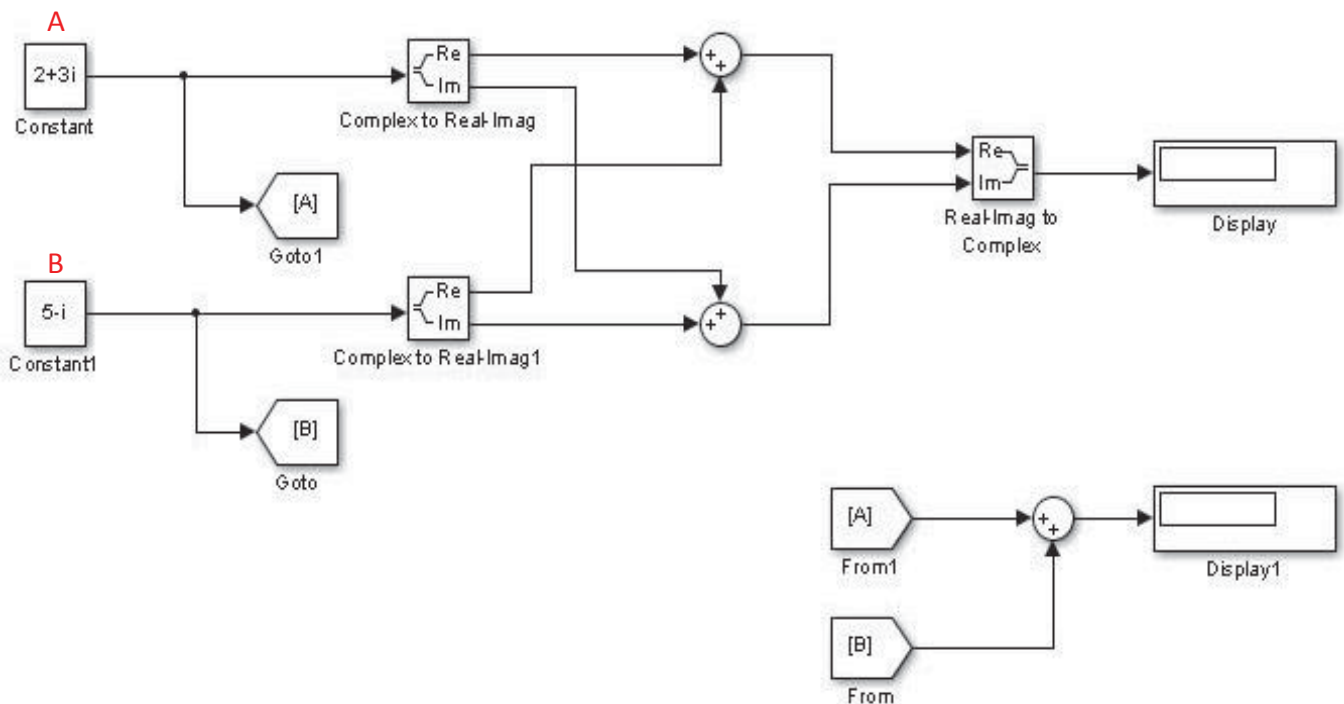
- روی constant دابل کلیک کرده و مقدار constant value را برابر با 100 قرار دهید.
- روی Gain1 دابل کلیک کرده و مقدار Gain را برابر با -0.06 قرار دهید. (می توان منفی را در sum وارد کرد.)
- روی Gain2 دابل کلیک کرده و مقدار Gain را برابر با 1/1000 قرار دهید.
- روی function math دابل کلیک کرده و function را برابر با square قرار دهید.



پس از اجرا می توان نمودار زیر را در صفحه اسکوپ مشاهده کرد.



مثال: برنامه زیر دو عدد مختلط را گرفته، قسمت موهومی و حقیقی را جدا کرده و با هم جمع کرده و نشان می دهد.



Simulink/ Sources/ Constant

Simulink/ Math Operations/ Sum

Simulink/ Math Operations/ Complex to Real-Imag

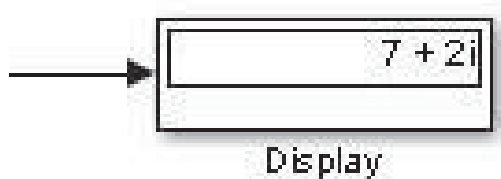
Simulink/ Math Operations/ Real-Imag to Complex

Simulink/ Signal Routing/ From

Simulink/ Signal Routing/ Goto

Simulink/ Sinks/ Display

پس از اجرا داریم:

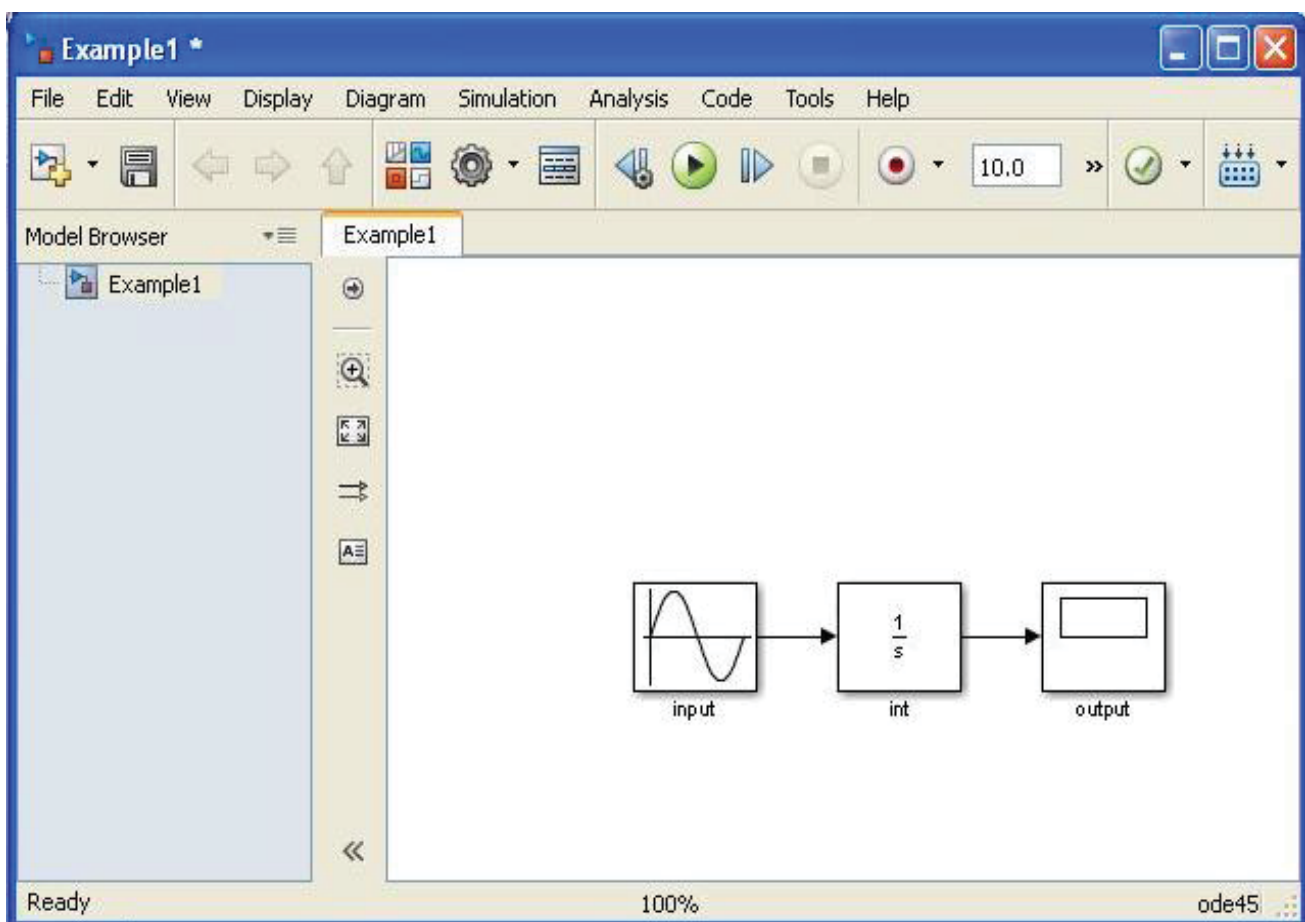


روش حل مسئله : ode45

در پنجره Editor دستورات زیر را بنویسید:

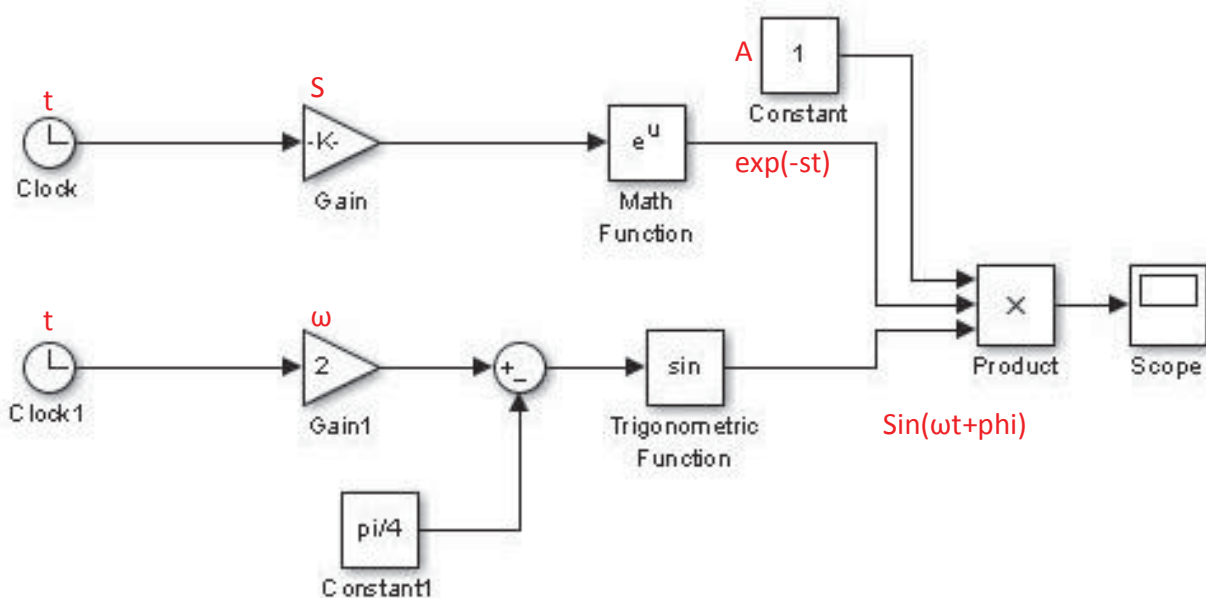
```
new_system('Example1')
open_system('Example1')
add_block('Simulink/Sources/Sine Wave','Example1/input')
add_block('Simulink/Continuous/Integrator','Example1/int')
add_block('Simulink/Sinks/Scope','Example1/output')
set_param('Example1/input','position',[120 150 180 200])
set_param('Example1/int','position',[220 150 280 200])
set_param('Example1/output','position',[320 150 380 200])
add_line('Example1','input/1','int/1')
add_line('Example1','int/1','output/1')
sim('Example1')
```

پس از اجرا داریم:



$$y=A\exp(-st).\sin(\omega t-\phi)$$

$$A=1 \quad S=-0.5 \quad \omega=2 \quad \phi=\pi/4$$



Simulink/ Sources/ Clock

Simulink/ Sources/ Constant

Simulink/ Math Operations/ Product

Simulink/ Math Operations/ Trigonometric Function

Simulink/ Math Operations/ Math Function

Simulink/ Math Operations/ Gain

Simulink/ Math Operations/ Sum

Simulink/ Sinks/ Scope

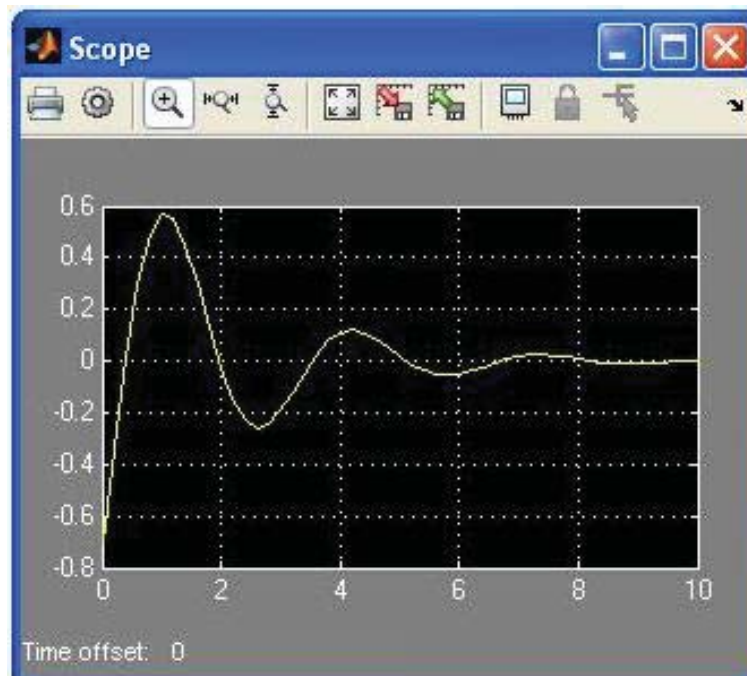


تنظیمات مورد نیاز:

- روی Gain دابل کلیک کرده و مقدار gain را برابر با -0.5 قرار دهید.
- روی Gain1 دابل کلیک کرده و مقدار Gain را برابر با 2 قرار دهید.
- روی product دابل کلیک کرده و مقدار number of inputs را برابر با 3 قرار دهید.
- روی constant1 دابل کلیک کرده و مقدار constant value را برابر با pi/4 قرار دهید.

- روی sum دابل کلیک کرده و list of signs را به صورت +- تغییر دهید

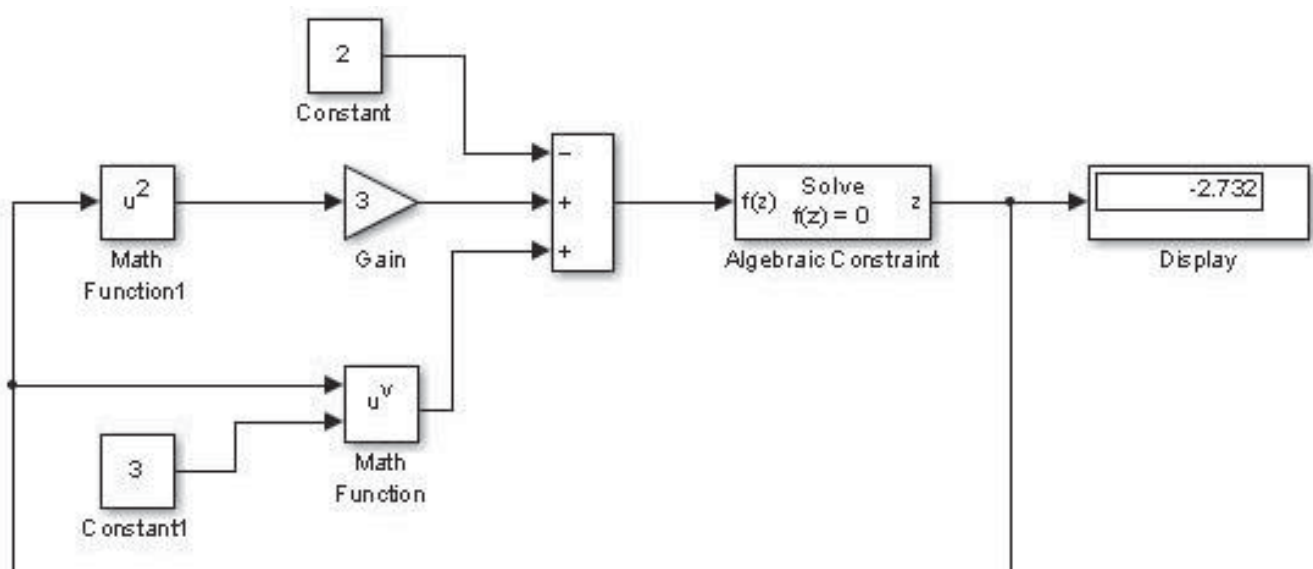
پس از اجرا نمودار زیر مشاهده می شود.



مثال: حل معادله $f=0$

$$x^3 + 3x^2 - 2 = 0$$

ریشه ها : -2.732 , 0.73 , -1



Simulink/ Sources/ Constant

Simulink/ Math Operations/ Algebraic Constraint

Simulink/ Math Operations/ Math Function

Simulink/ Math Operations/ Gain

Simulink/ Math Operations/ Sum

Simulink/ Sinks/ Display

- روی بلوک Algebraic constraint دابل کلیک کرده و در قسمت initial guess یک حدس اولیه برای مثال عدد -5 را وارد می کنیم و اجرا می کنیم تا ریشه نمایان شود. برای یافتن دیگر ریشه ها مقدار حدس اولیه را تغییر داده و اجرا می کنیم.

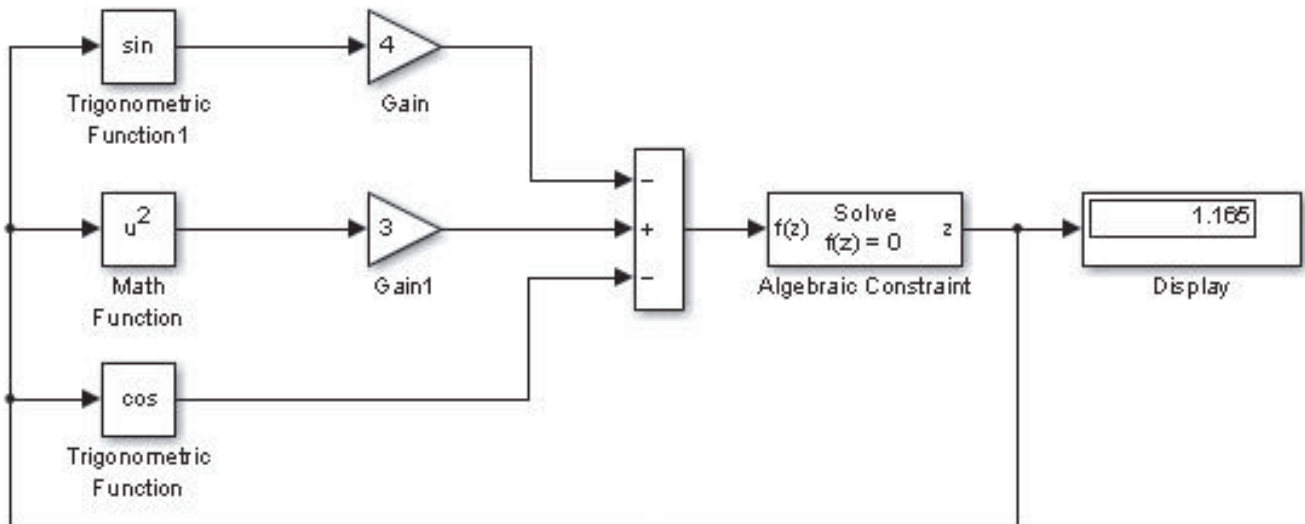
Initial guess : -1 → $x=-1$

Initial guess : -5 → $x=-2.732$

Initial guess : 5 → $x=0.7321$

مثال:

$$3x^2 - 4\sin(x) - \cos(x) = 0$$



Simulink/ Math Operations/ Algebraic Constraint

Simulink/ Math Operations/ Math Function

Simulink/ Math Operations/ Gain

Simulink/ Math Operations/ Trigonometric Function

Simulink/ Math Operations/ Sum

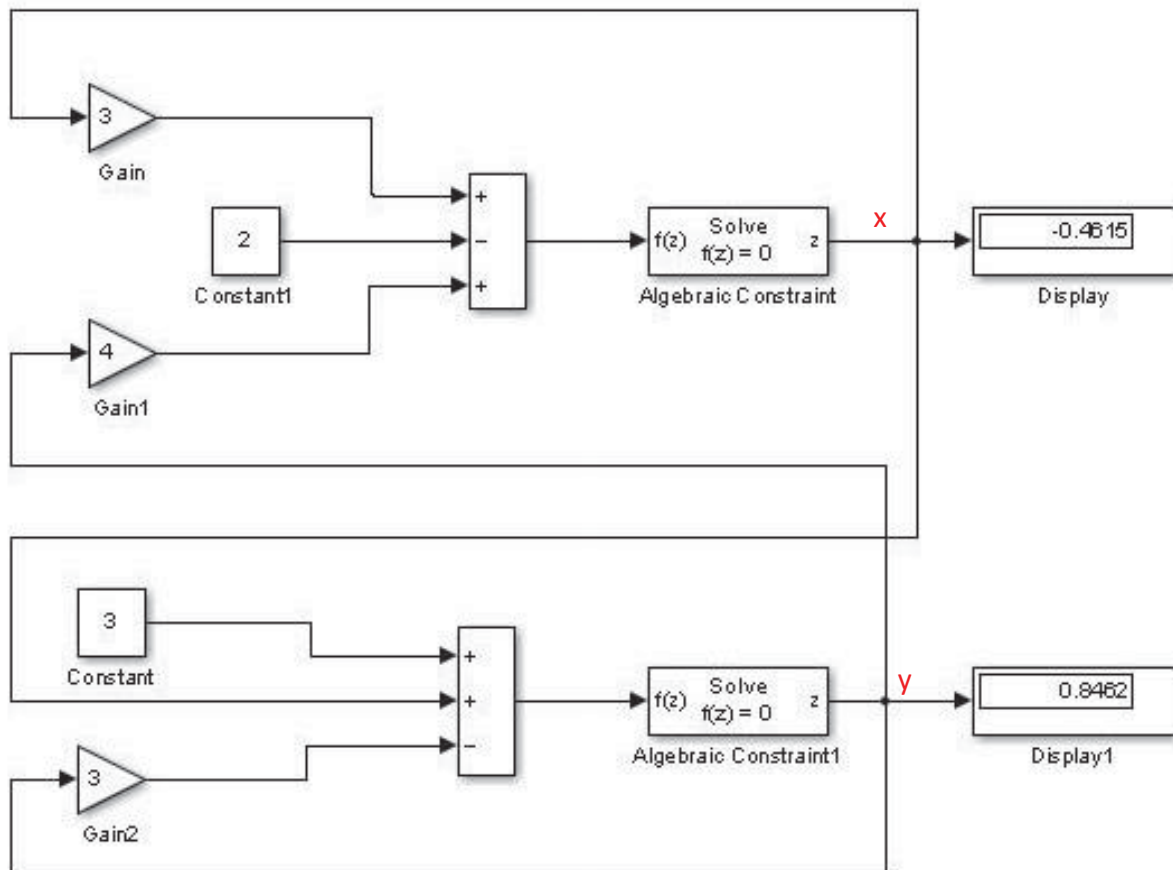
Simulink/ Sinks/ Display

برای Algebraic Constraint یک حدس اولیه وارد کرده و آن را اجرا می کنیم.

initial guess: 5 \rightarrow x= 1.165

مثال:

$$\begin{cases} 3x + 4y - 2 = 0 \\ x - 3y + 3 = 0 \end{cases}$$



Simulink/ Sources/ Constant

Simulink/ Math Operations/ Algebraic Constraint

Simulink/ Math Operations/ Gain

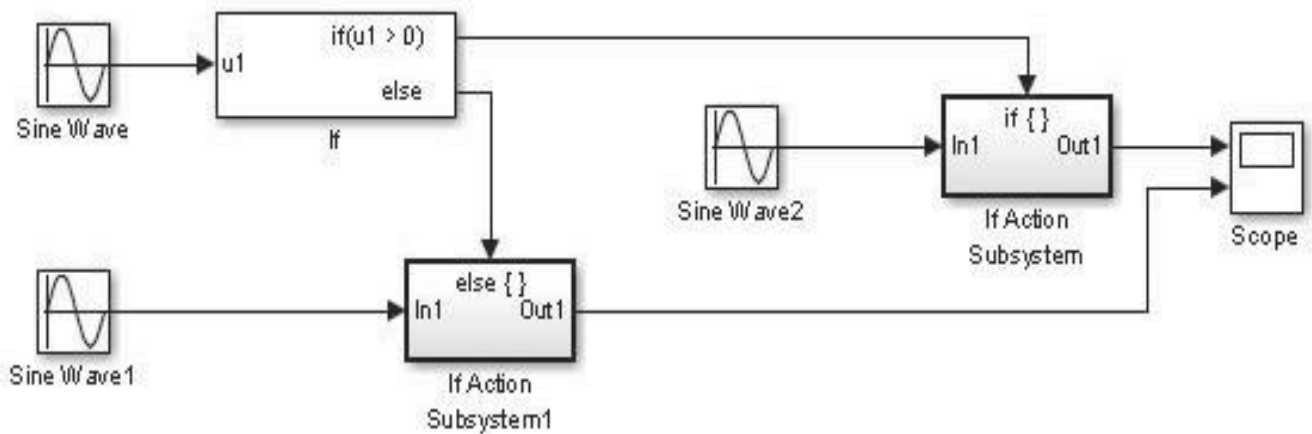
Simulink/ Math Operations/ Sum

Simulink/ Sinks/ Display

در هر دو بلوک Algebraic constraint مقدار initial guess را 5- وارد کرده و مدل را اجرا کردیم. جواب ها به صورت زیر بدست آمدند.

$x = -0.4615$ $y = 0.8462$

مثال:



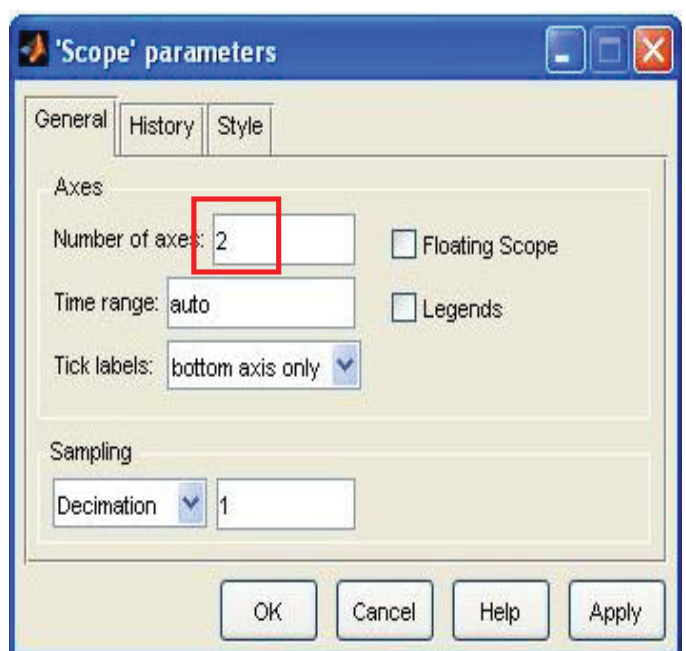
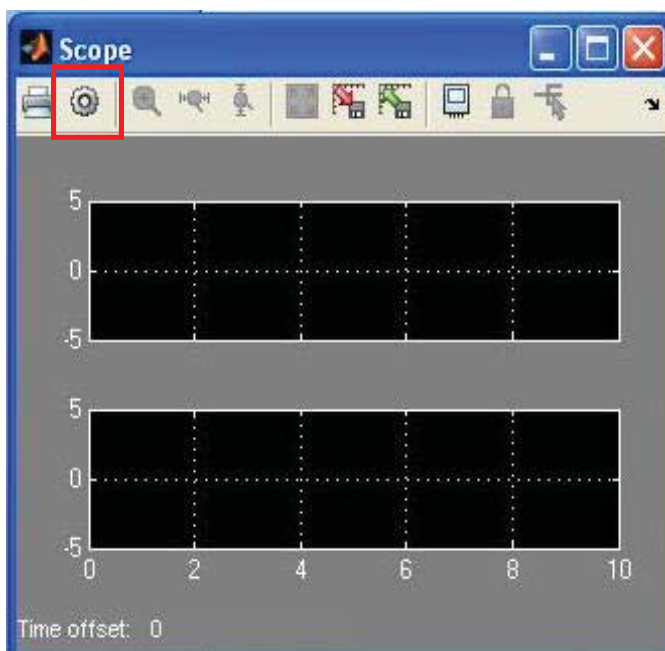
Simulink/ Sources/ Sine Wave

Simulink/ Sinks/ Scope

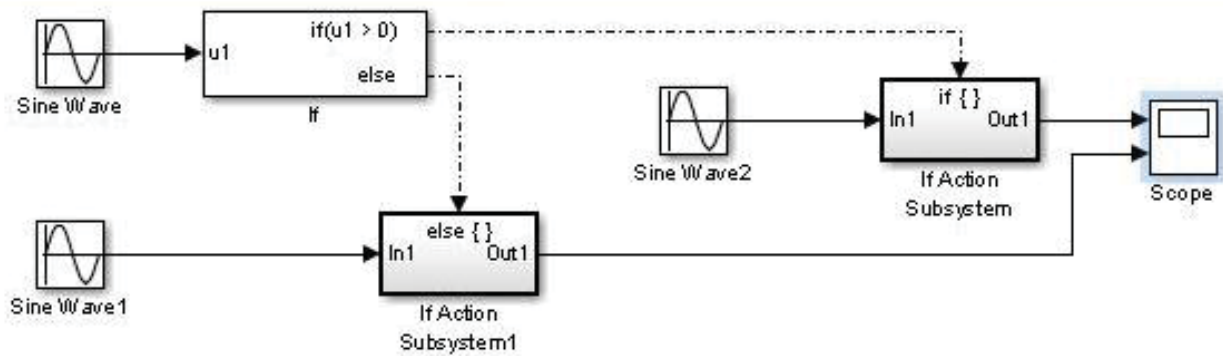
Simulink/ Ports & Subsystems/ If

Simulink/ Ports & Subsystems/ If Action Subsystem

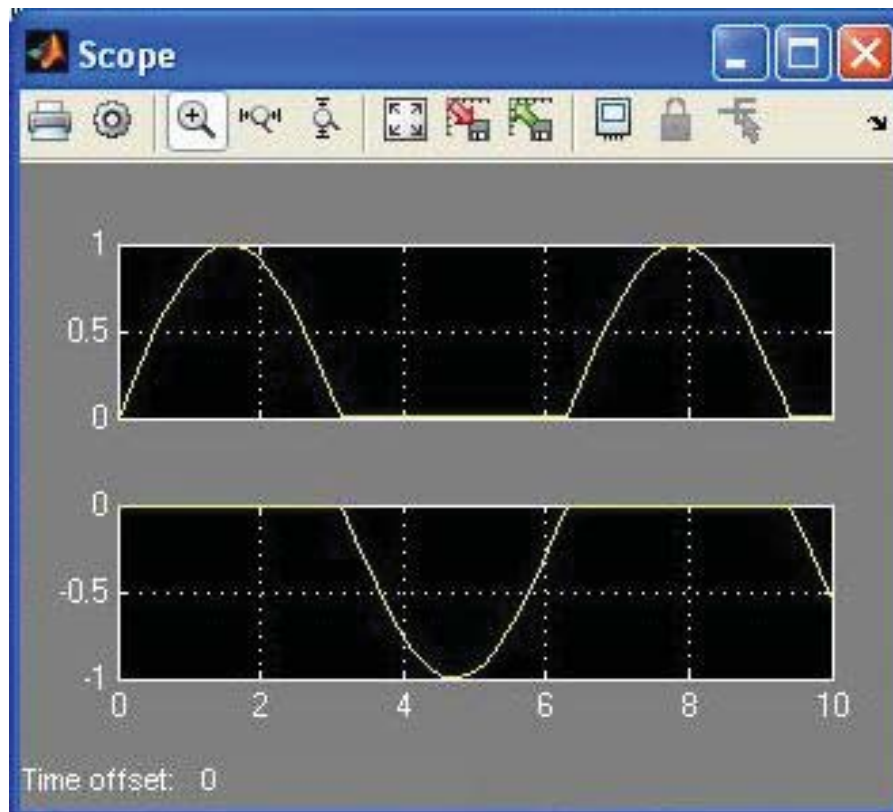
- روی scope دابل کلیک کرده و سپس روی آیکن parameters کلیک کرده و مقدار number of axes را از 1 به 2 تغییر می دهیم.



حال بر روی Run کلیک می کنیم.



حال می توانیم نمودار حاصل را در صفحه اسکوپ مشاهده کنیم.



سلول گیلبرت:

شماتيك سلول گیلبرت به صورت زیر می باشد. همانطور که مشاهده می شود، دارای دو ورودی V_{id1} و V_{id2} می باشد. بسته به اینکه سیگنال کوچک باشند یا سیگنال بزرگ، کارایی سلول گیلبرت فرق خواهد کرد بنابراین برای سلول گیلبرت کاربردهای زیر متصور می باشد:

۱. در صورتی که هر دو ورودی فوق سیگنال کوچک باشد ($V_T < 2V_T$) در اینصورت مدار مذکور بصورت یک ضرب کننده آنالوگ عمل خواهد کرد.

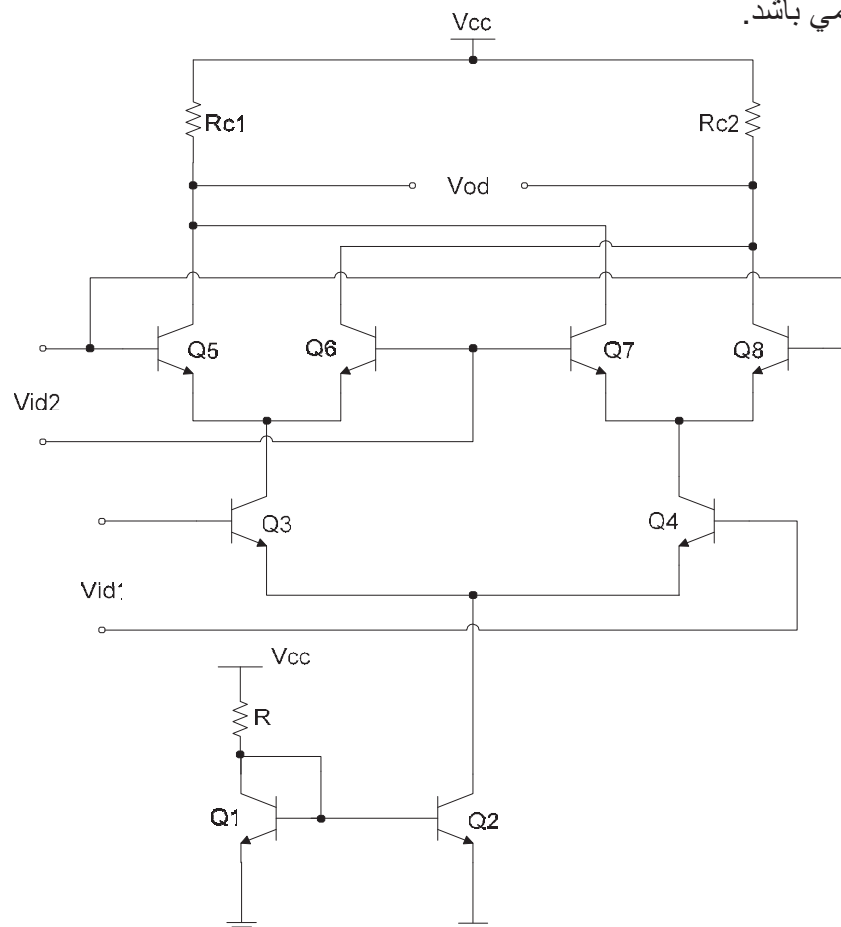
۲. در صورتی که یکی از آنها به صورت سیگنال کوچک و دیگری بصورت سیگنال بزرگ باشد، مدار بصورت یک مدولاتور عمل خواهد کرد.

۳. در صورتیکه هر دو سیگنال بصورت سیگنال بزرگ باشد، می توان از سلول گیلبرت به همراه یک فیلتر پائین گذر به عنوان آشکار ساز فاز استفاده کرد.

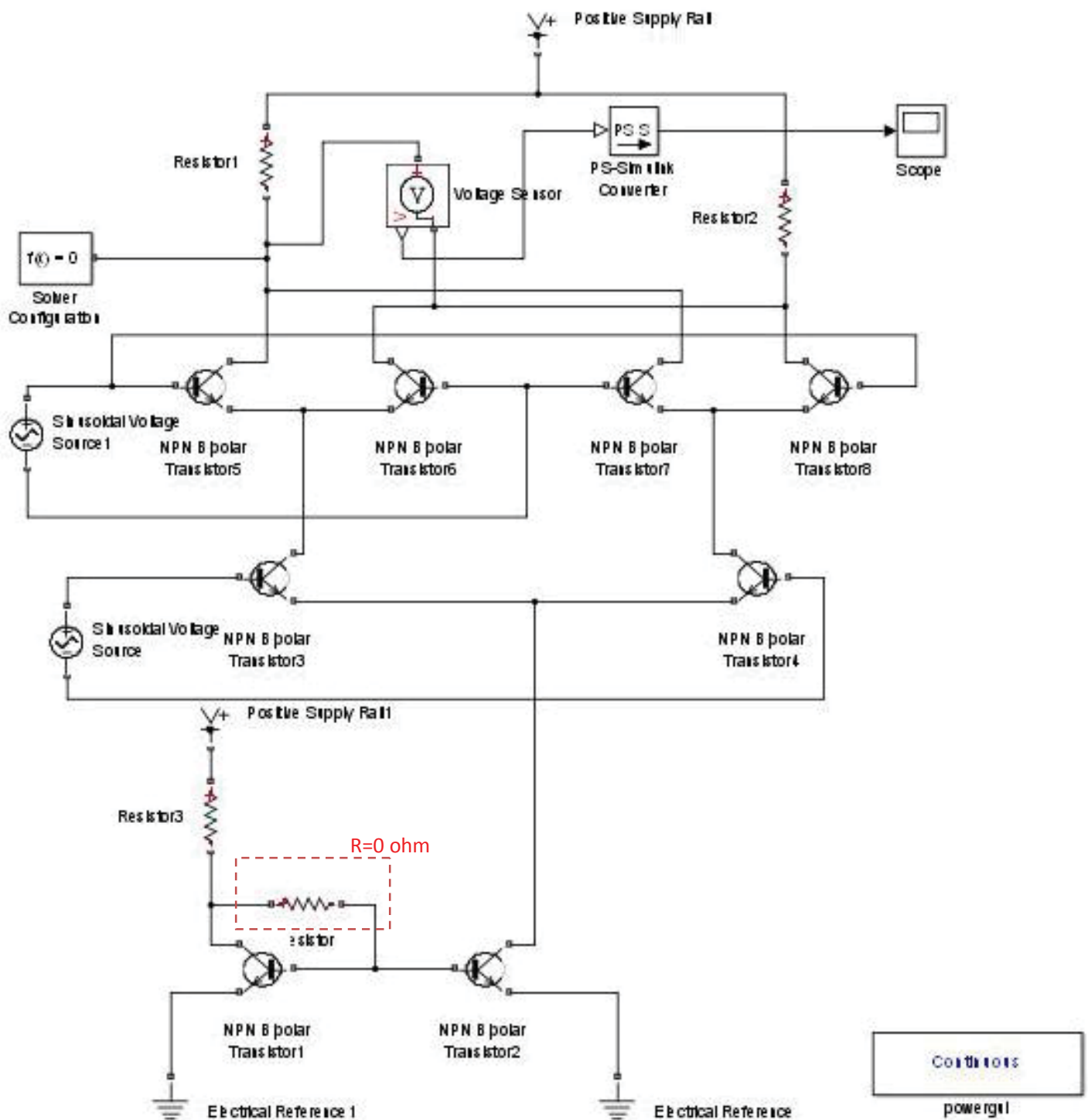
دستور کار:

تشریح کار مدار در حالت کلی و تعمیم آن برای کاربردهای ذکر شده

شبیه سازی هر یک از کاربردهای ذکر شده. برای شبیه سازی از هر نرم افزاری با ذکر نام آن در گزارش کار می توانید استفاده کنید. لازم به ذکر است که هر کاربردی شبیه سازی خاصی ممکن است لازم داشته باشد، به عنوان مثال برای کاربرد مدولاتور لازم است که طیف فرکانسی سیگنال خروجی کشیده شود تا معلوم شود در خروجی مدار چه فرکانسهایی موجود می باشد.



این مدل را به صورت زیر شبیه سازی می کنیم.



بلوک های مورد نیاز را از مسیرهای زیر بیابید.

Simscape /SimPowerSystems /powergui

Simscape /Utilities /Solver Configuration

Simscape /Utilities /PS-Simulink Converter

Simscape /Foundation Library /Electrical /Electrical Sensors /Voltage Sensor

Simscape /SimElectronics /Semiconductor Devices /NPN Bipolar Transistor

Simscape /Foundation Library /Electrical /Electrical Elements /Resistor

Simscape /SimElectronics /Sources /Positive Supply Rail

Simscape /Foundation Library /Electrical /Electrical Elements /Electrical Reference

Simulink /Sinks /Scope

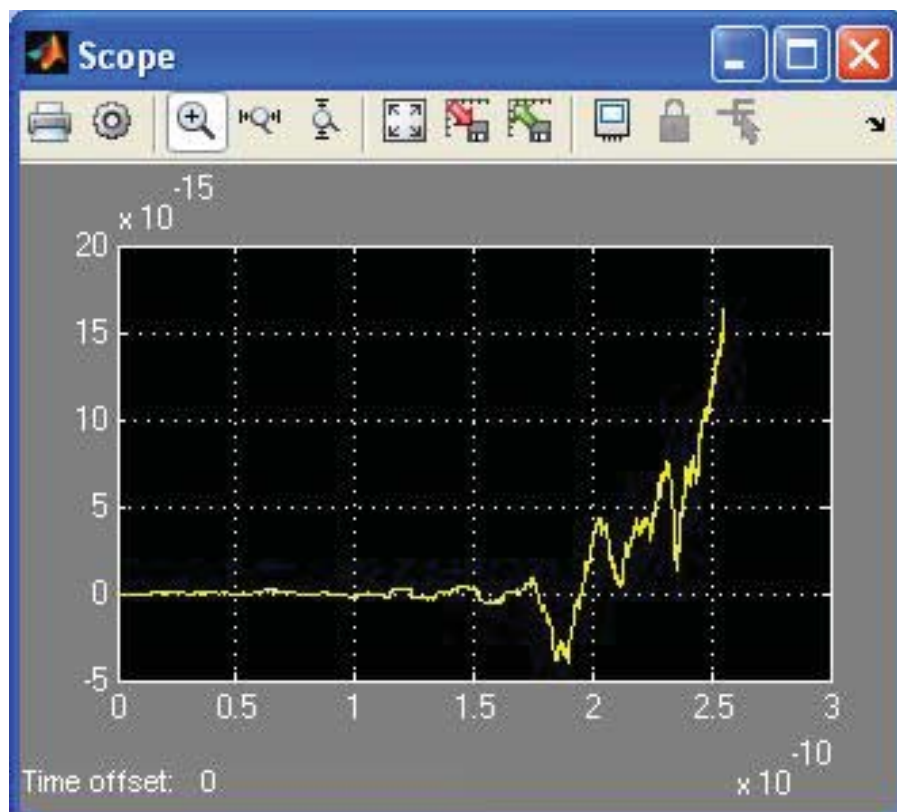
- برای برگرداندن ترانزیستور از $Ctrl + R$ استفاده نکنید. برای این کار روی ترانزیستور کلیک راست کرده و به مسیر زیر بروید.

Rotate& Flip /Flip Block /Left-Right

تنظیمات مورد نیاز:

- $R1=R2=1Kohm$ $R=3kohm$
- دامنه و فرکانس $Vid2$ را به ترتیب روی $2V$ و $5Hz$ تنظیم کنید.
- دامنه و فرکانس $Vid1$ را به ترتیب روی $20V$ و $50Hz$ تنظیم کنید.
- Vcc ها را روی $8V$ تنظیم کنید.

حال Run کرده و کمی صبر کنید. سپس روی آیکن  (Stop) کلیک کنید و نمودار حاصل را از scop مشاهده کنید.



مباحث فازی (Fuzzy):

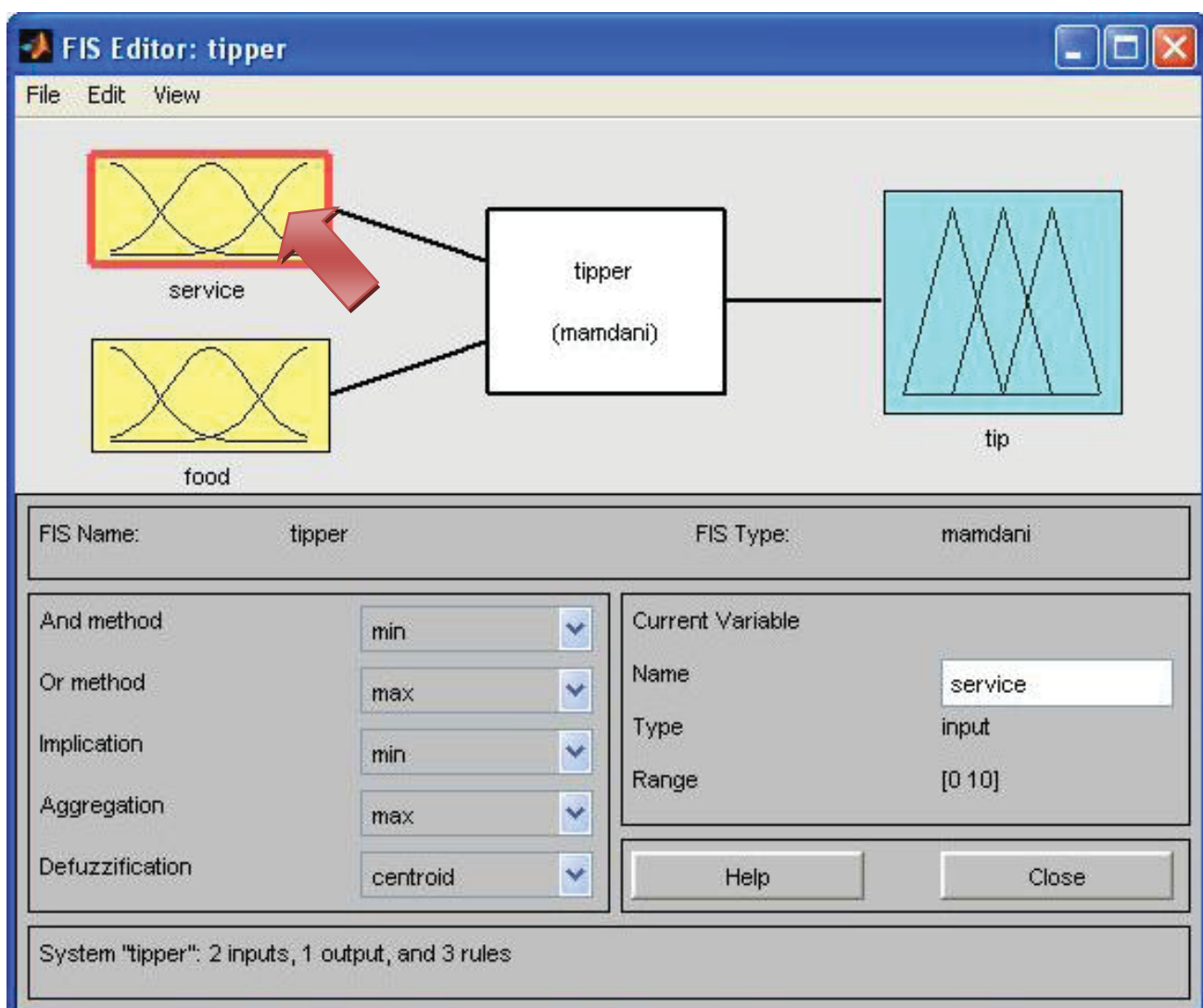
این سیستم در سال ۱۹۸۵ توسط sugeno معرفی شد. این روش شبیه به روش ممدانی (۱۹۸۰) است. فرآیند فازی سازی و اپراتورهای به کار گرفته شده دقیقاً با روش ممدانی یکی بوده و تفاوت این روش خطی یا ثابت بودن تابع عضویت متغیر خروجی است.

مثال حل شده:

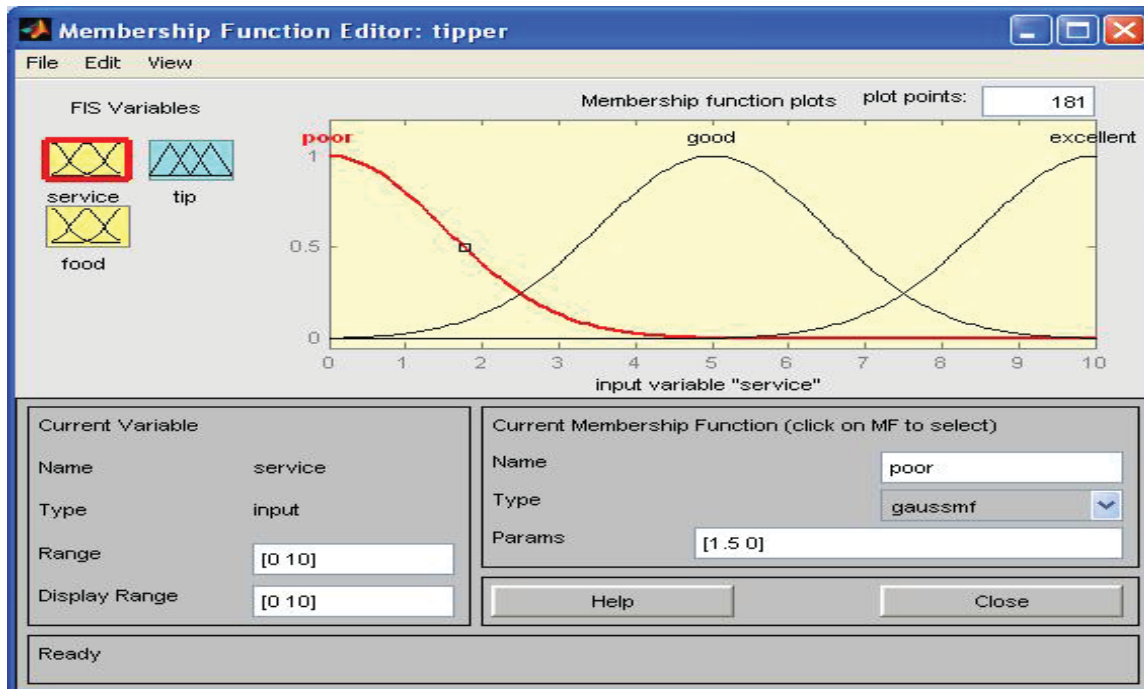
این مثال بر آن است که قواعدی را وضع کند که بر اساس آنها میزان انعام تعیین گردد.

>> fuzzy tipper \longrightarrow انعام دهنده

با اجرای این دستور شکل زیر ظاهر می شود.



برای مشاهده نمودارهای تابع وضعیت روی یکی از نمودارهای food ، service یا tip در شکل بالا دابل کلیک کنید.

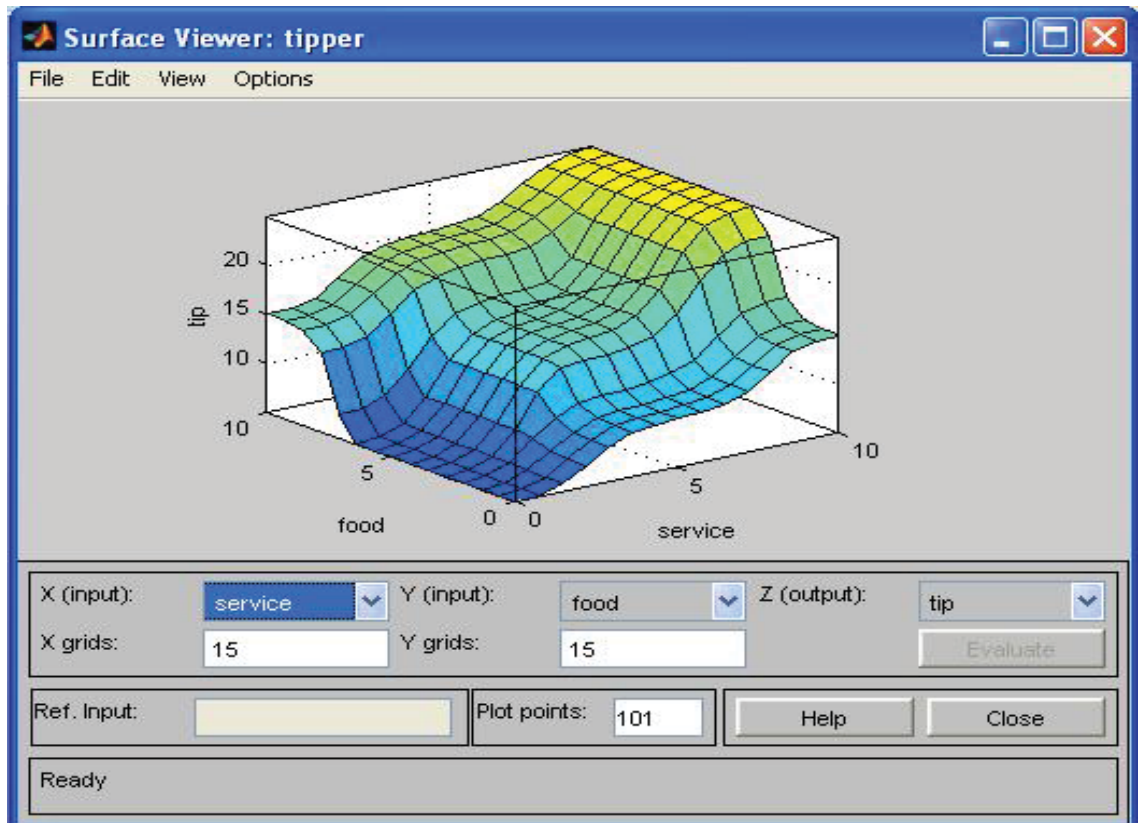


با مسیر زیر می توانید قوانین انعام را مشاهده نموده، آنها را تغییر داده، حذف کرده و یا قانون جدیدی اضافه نمایید.

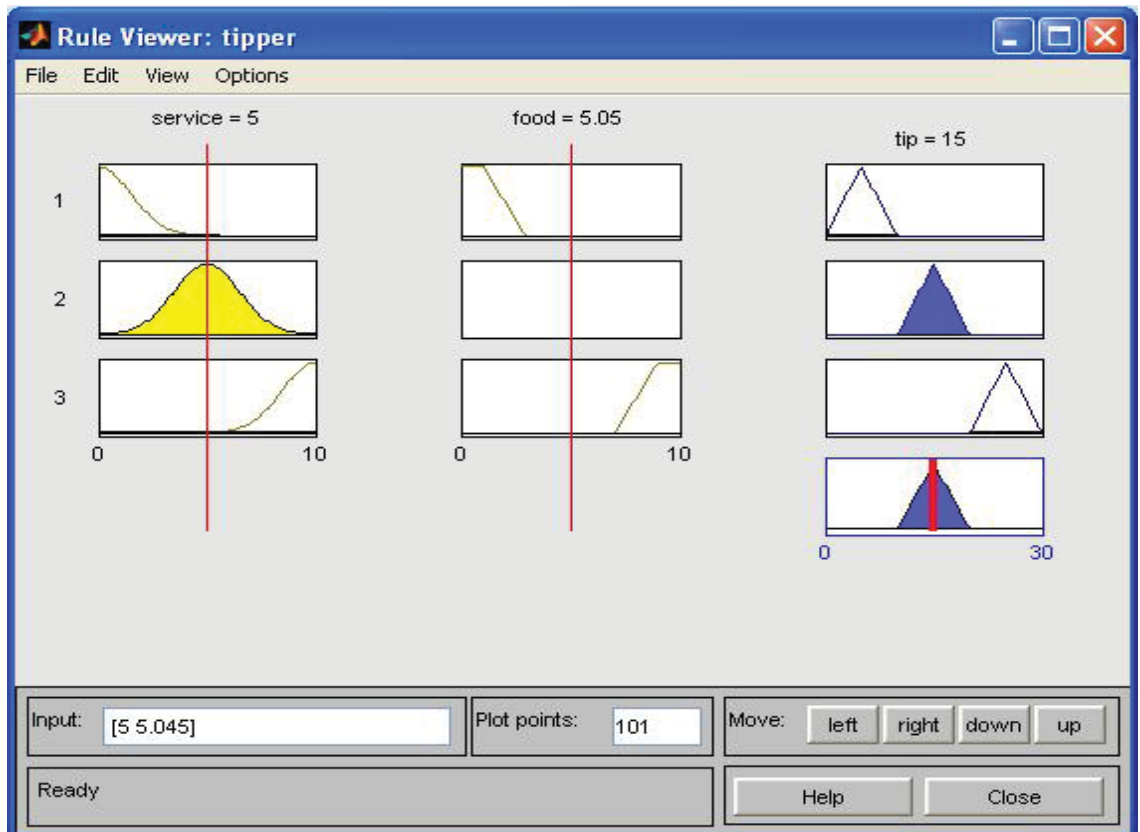
Edit/ Rules

از مسیرهای زیر می توان نمودار Rules و surface را مشاهده کرد.

View/ surface

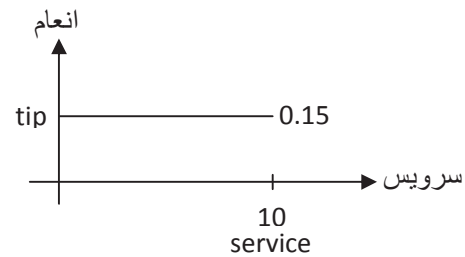


View/ Rules



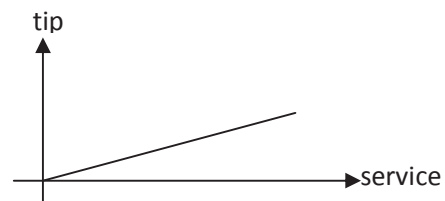
15% سرویس \rightarrow $tip=0.15$

$$0 < service < 10$$



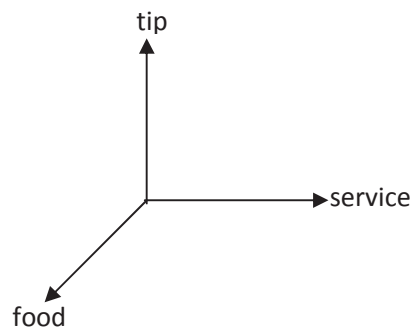
Range \rightarrow 5% ~ 25%

$$tip = \frac{0.2}{10} * service + 0.05$$



$$tip = \frac{0.2}{20} (service + food) + 0.05$$

اگر کیفیت غذاها را یکی از فاکتورها بدانیم

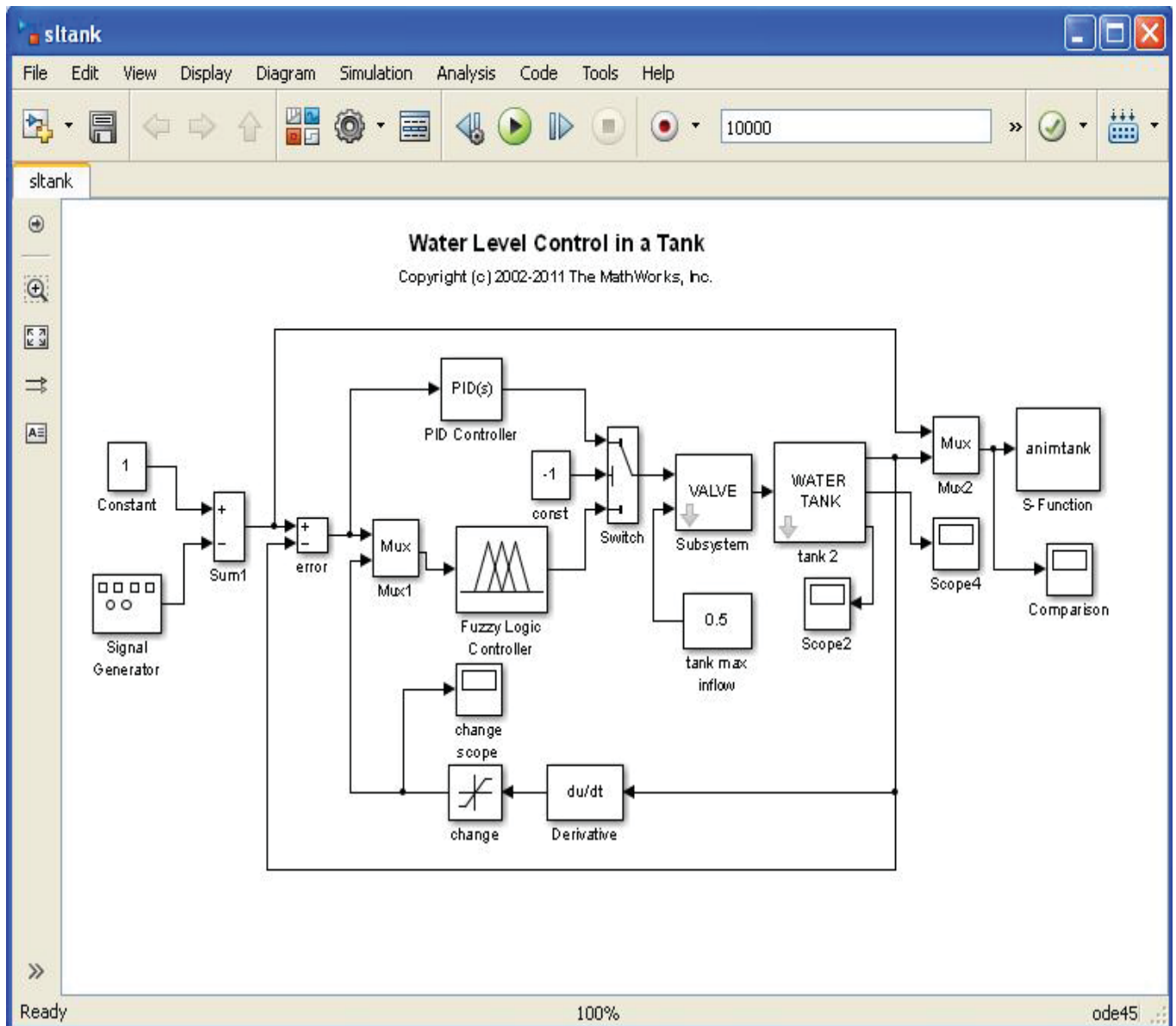


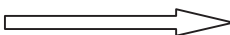
$$\left\{ \begin{array}{l} \text{کیفیت غذا} \quad 80\% \\ \text{سرویس} \quad 20\% \end{array} \right. \quad tip = (0.8) \left(\left(\frac{0.2}{10} * service + food \right) + 0.05 \right) + \dots$$

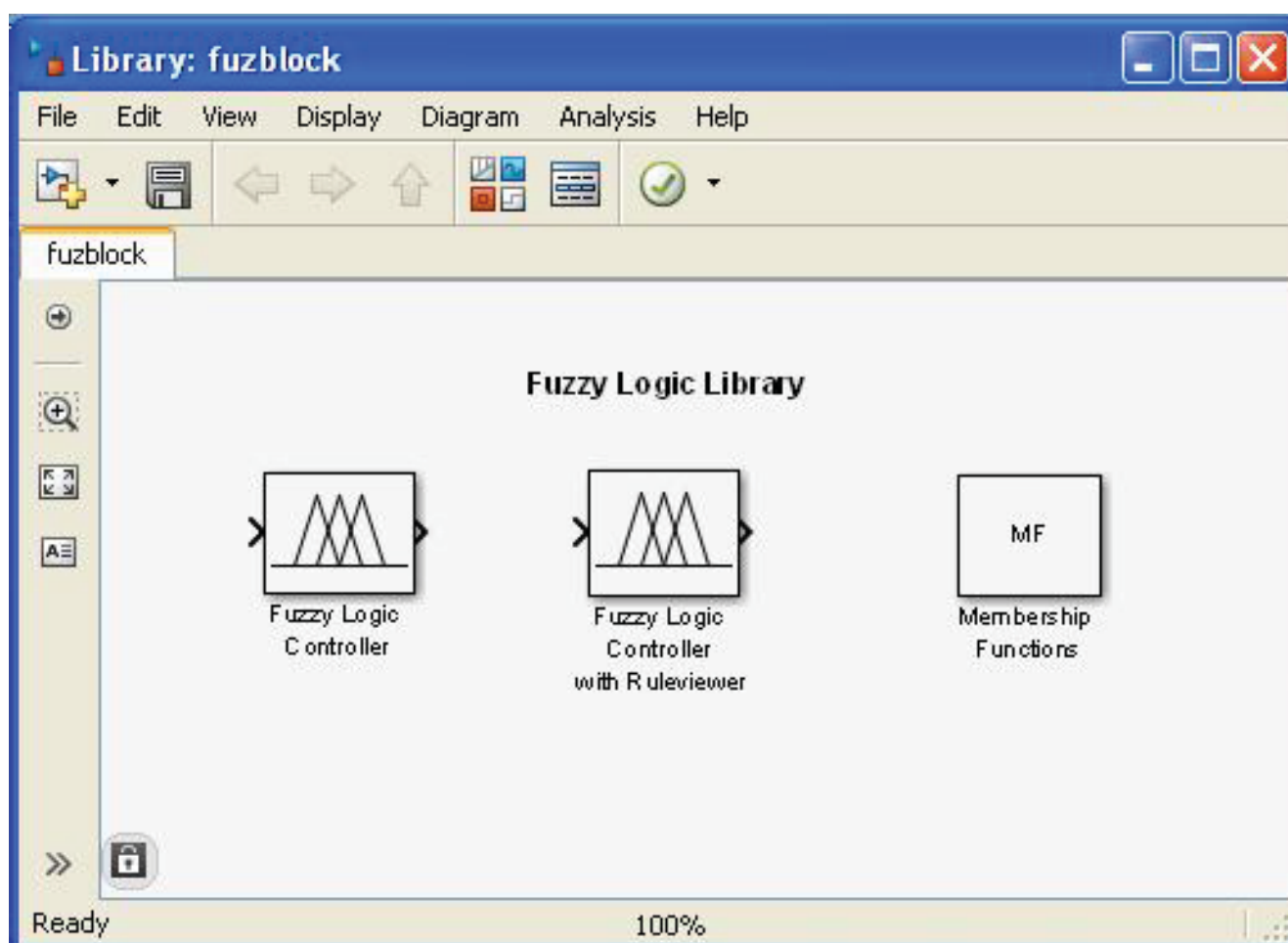
$$(0.2) \left(\left(\frac{0.2}{10} * service + food \right) + 0.05 \right)$$

- پسوند mf (membership function) به معنی تابع عضویت است. مانند gaussmf (تابع گاوس)

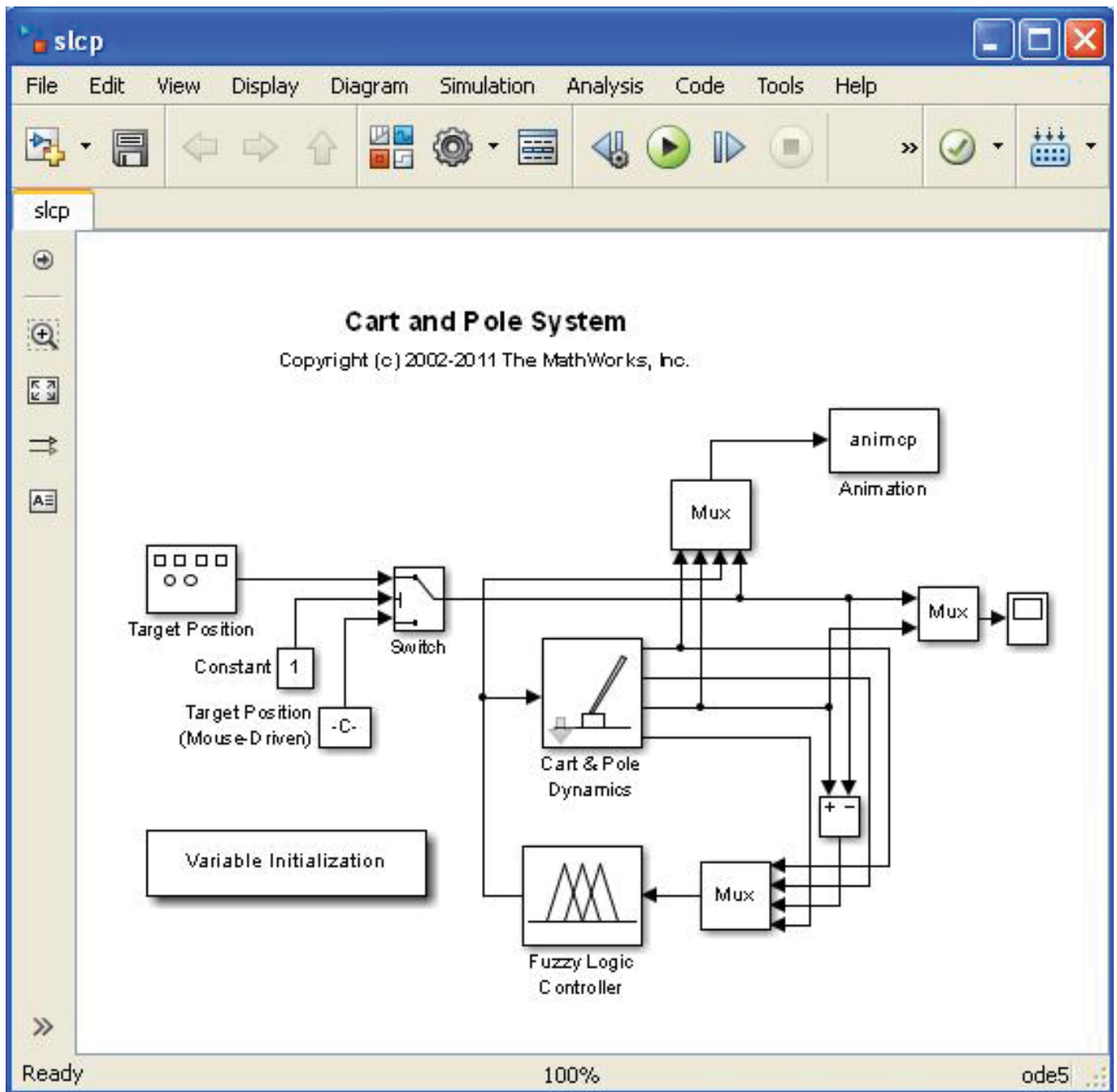
>> sltank  سیستم دینامیکی سطح یک مایع



>> fuzblock  سه قاعده فازی



>> slcp → کنترل پاندول معکوس با استفاده از شبکه عصبی

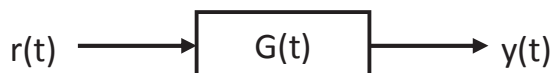


سیستم های کنترل (خطی یا غیر خطی):

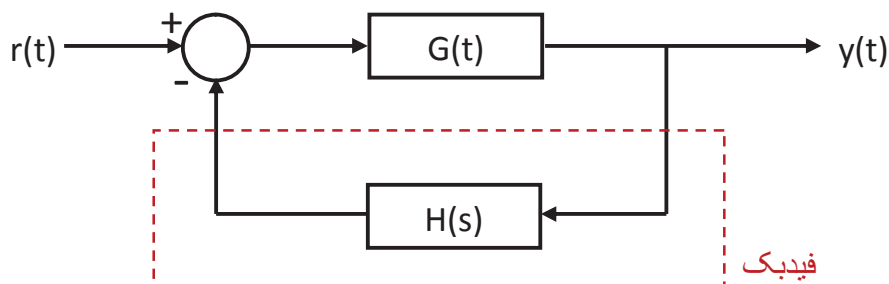
هدف: رسیدن به خروجی مطلوب از ورودی



- سیستم کنترل حلقه باز: روشی ارزان، ساده، نویز و اغتشاش



- سیستم کنترل حلقه بسته: روشی گران تر، پیچیده تر، دقت بیشتر، نویز و اغتشاش ← مقاومت



تابع تبدیل حلقه بسته:

$$M(s) = \frac{G(s)}{1 + G(s)H(s)}$$

توجه: مراجعه کنید به کتاب لوگاتو- بنجامین کو

تابع تبدیل Transfer function $\xrightarrow{\text{Matlab}}$ tf

$$tf = \frac{\text{تبدیل لاپلاس خروجی}}{\text{تبدیل لاپلاس ورودی}} = \frac{y(s)}{r(s)}$$

$$tf = \frac{1}{s + 2}$$

توجه: چون می خواهیم از قید زمان خلاص شده و وارد حوزه فرکانس شویم از تبدیل لاپلاس استفاده می کنیم.

هدف: ۱- کاهش خطا

$$E(s) = R(s) - y(s) H(s)$$

حالت ماندگار خطای

۲- دنبال کردن ورودی از خروجی

نمایش صفر و قطب در صفحه مختلط:

$$m = \frac{s + 1}{s^2 + 2s + 5}$$

```
>> m=tf([1 1],[1 2 5])
```

```
m =
```

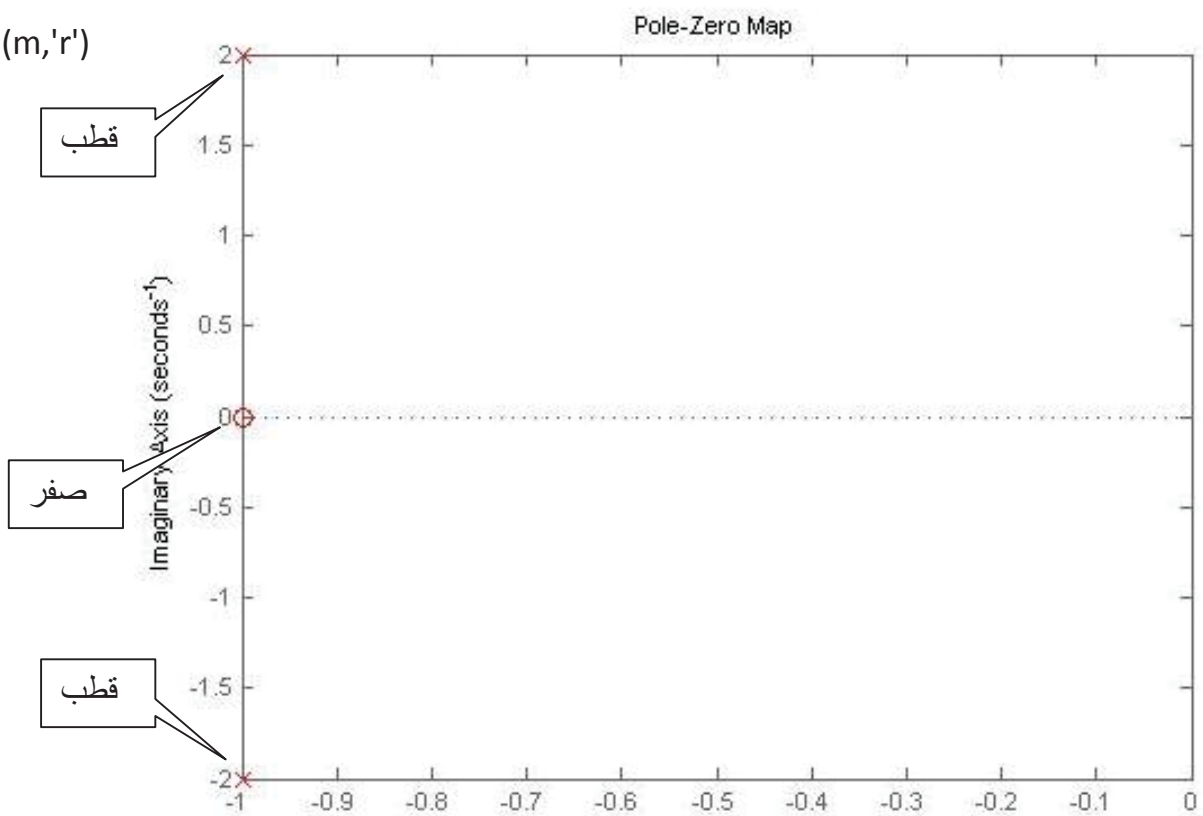
```
    s + 1
```

```
-----
```

```
 s^2 + 2 s + 5
```

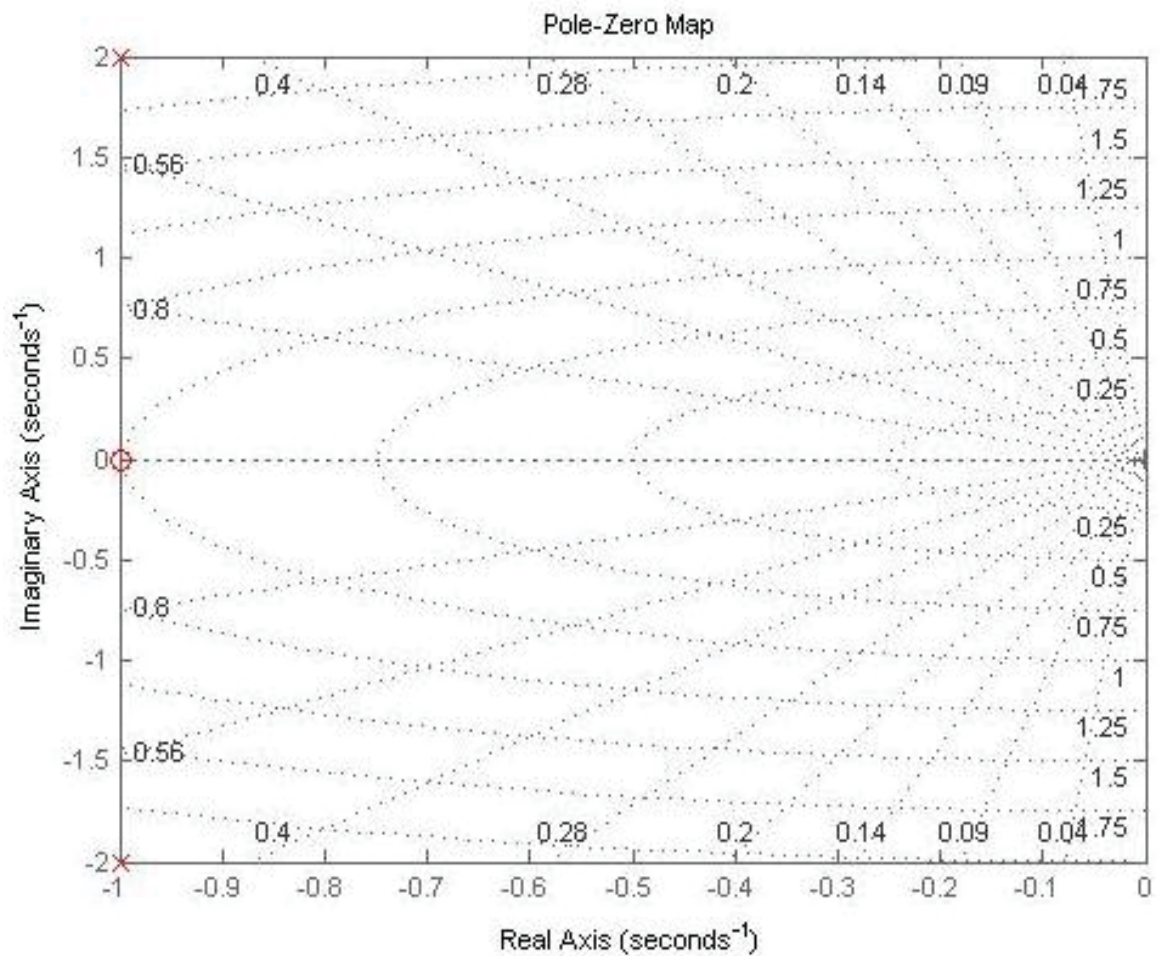
Continuous-time transfer function.

```
>> pzmap(m,'r')
```



نکته: قطب ها با ضربدر و صفرها با دایره نشان داده می شود.

```
>> grid on
```



$$m = \frac{1}{(s+1)(s+2)}$$

$$\text{قطب ها} \begin{cases} S=-1 \\ S=-2 \end{cases}$$

```
>> m=tf([1],conv([1 1],[1 2]))
```

```
m =
```

```
1
```

```
-----
```

```
s^2 + 3 s + 2
```

دستور zpk :

>> b=zpk([-1 -2],[-3 -5 -7],[10]) \longrightarrow بهره k: قطب p: صفر z:

b =
 $10 (s+1) (s+2)$

 $(s+3) (s+5) (s+7)$

Continuous-time zero/pole/gain model.

قواعد ساده سازی:

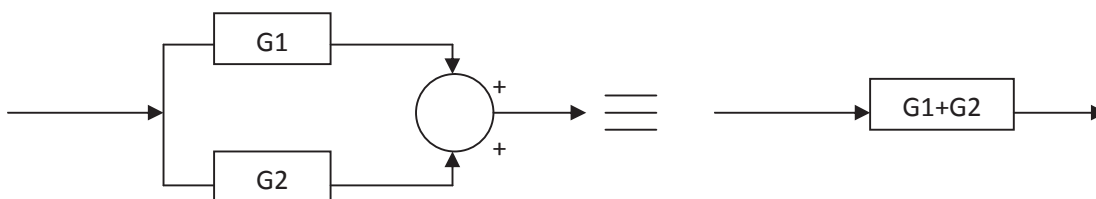
- بلوک های سری

>> series



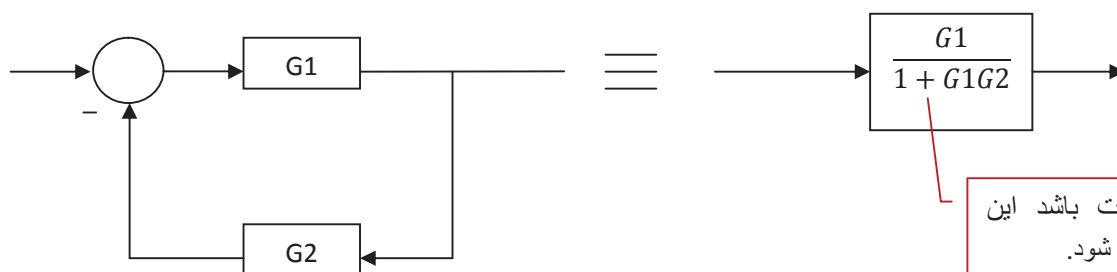
- بلوک های موازی

>> parallel



- بلوک های فیدبک منفی

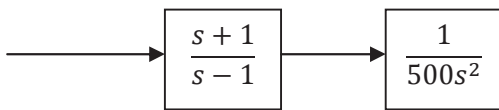
>> feedback



اگر فیدبک مثبت باشد این علامت منفی می شود.

حالت خاص \leftarrow cloop

دستور : series



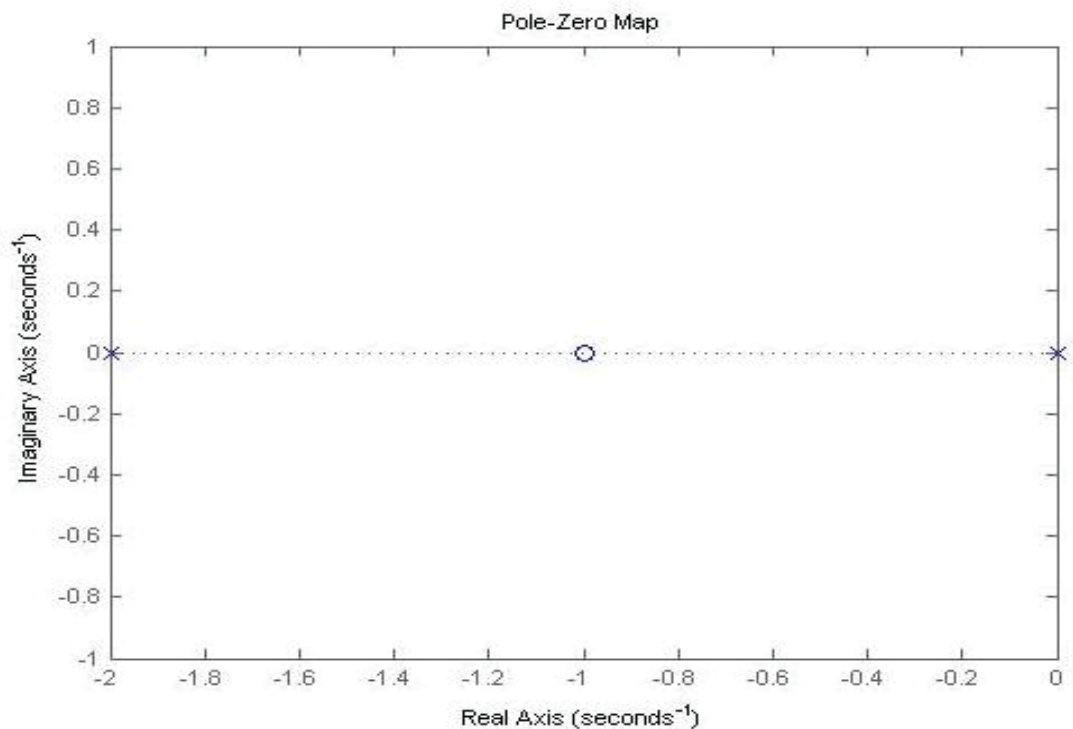
1. num1=[1 1]; \Rightarrow مخفف numerator به معنی صورت است
2. den1=[1 2]; \Rightarrow مخفف denominator به معنی مخرج است
3. num2=[1];
4. den2=[500 0 0];
5. [num,den]=series(num1,den1,num2,den2);
6. printsys(num,den)

num/den =

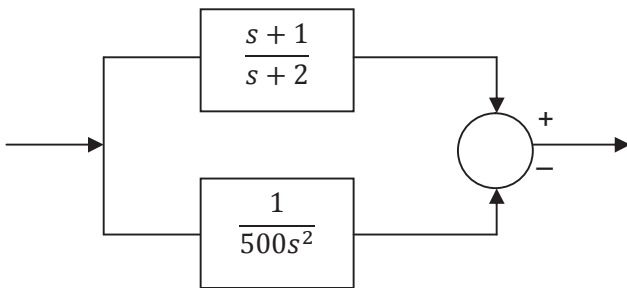
$$\frac{s + 1}{500 s^3 + 1000 s^2}$$

transfer function : **tf** دستور

```
>> a=tf([1 1],[500 1000 0 0])
>> pzmap(a)
```



دستور parallel :



برنامه این شکل همانند برنامه قبل است با این تفاوت که به جای دستور series از دستور parallel استفاده می شود.

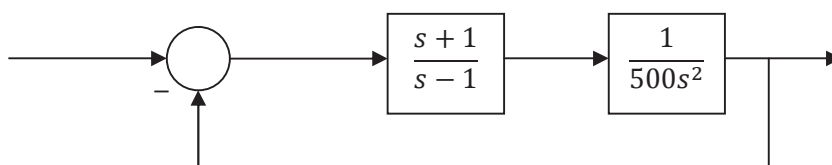
```
1.num1=[1 1];
2.den1=[1 2];
3.num2=[1];
4.den2=[500 0 0];
5.[num,den]=parallel(num1,den1,num2,den2);
6.printsys(num,den)
```

num/den =

$$500 s^3 + 500 s^2 + s + 2$$

$$500 s^3 + 1000 s^2$$

دستور cloop :



حالت خاص فیدبک (فیدبک واحد)

```
1.num1=[1 1];
2.den1=[1 2];
3.num2=[1];
4.den2=[500 0 0];
5.[num,den]=series(num1,den1,num2,den2);
6.[num,den]=cloop(num,den,-1);
7.printsys(num,den)
```

به جای این دستور می توان از دستور feedback استفاده کرد

فیدبک منفی است

- در اینجا از دستور cloop استفاده شده زیرا فیدبک، واحد است.

num/den =

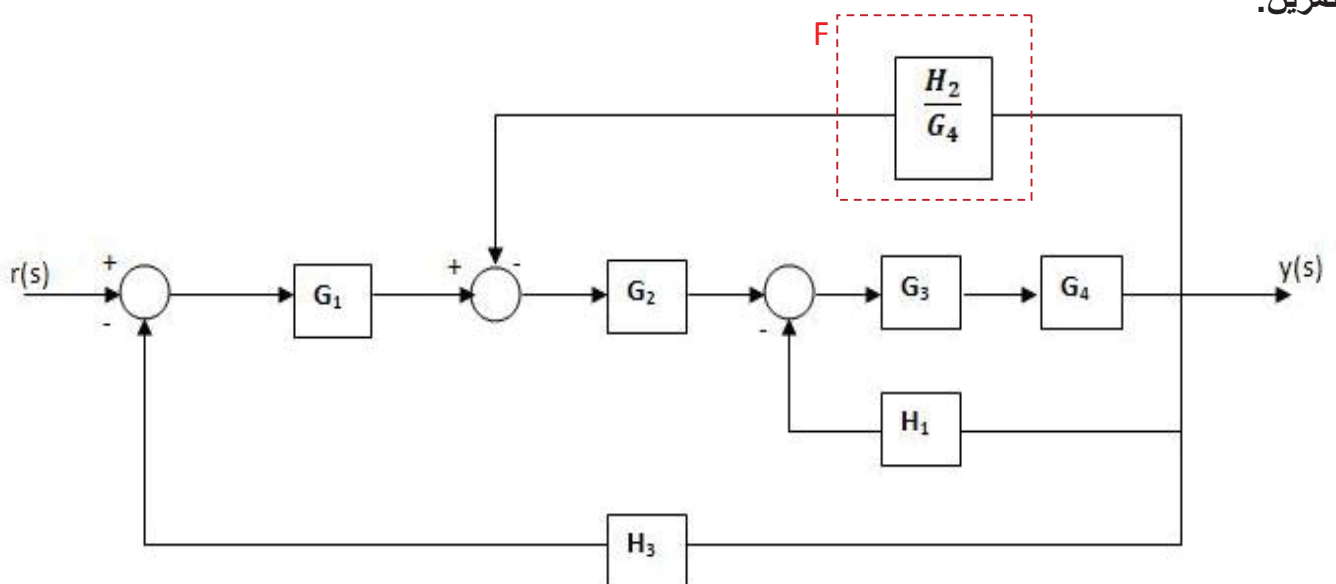
$$s + 1$$

$$500 s^3 + 1000 s^2 + s + 1$$

- می توان خط ششم برنامه را به صورت زیر نوشت:

```
[num,den]=feedback(num,den,1,1,-1);
```

تمرین:



$$G_1(s) = \frac{1}{s+10}$$

$$G_2(s) = \frac{1}{s+1}$$

$$G_3(s) = \frac{s^2+1}{s^2+4s+1}$$

$$G_4(s) = \frac{s+1}{s+4}$$

$$H_1(s) = \frac{s+1}{s+2}$$

$$H_2(s) = 2$$

$$H_3(s) = 1$$

$$F = \frac{H_2}{G_4} = \frac{2s + 8}{s + 1}$$

```

1. numG1=[1];
2. denG1=[1 10];
3. numG2=[1];
4. denG2=[1 1];
5. numG3=[1 0 1];
6. denG3=[1 4 1];
7. numG4=[1 1];
8. denG4=[1 4];
9. numH1=[1 1];
10. denH1=[1 2];
11. numH3=[1];
12. denH3=[1];
13. numF=[2 8];
14. denF=[1 1];
15. [num,den]=series(numG3,denG3,numG4,denG4);
16. [num,den]=feedback(num,den,numH1,denH1);
17. [num,den]=series(num,den,numG2,denG2);
18. [num,den]=feedback(num,den,numF,denF);
19. [num,den]=series(num,den,numG1,denG1);
20. [num,den]=feedback(num,den,numH3,denH3);
21. printsys(num,den)

```

num/den =

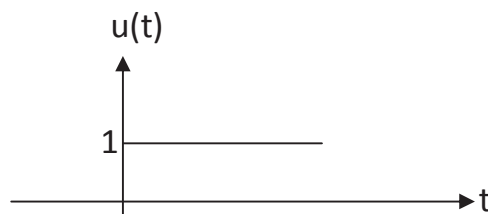
$$s^5 + 4 s^4 + 6 s^3 + 6 s^2 + 5 s + 2$$

$$2 s^7 + 38 s^6 + 256 s^5 + 906 s^4 + 1680 s^3 + 1632 s^2 + 890 s + 252$$

انواع ورودی:

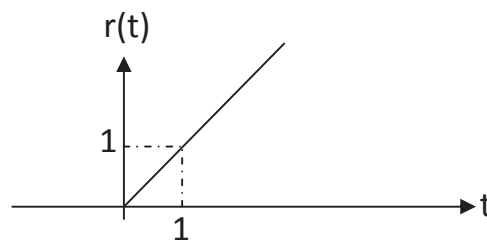
ورودی پله: step

$$u(t) = \begin{cases} 1 & t \geq 0 \\ 0 & t < 0 \end{cases}$$



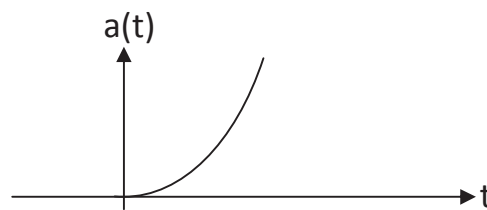
ورودی شیب: ramp

$$r(t) = \begin{cases} t & t \geq 0 \\ 0 & t < 0 \end{cases}$$



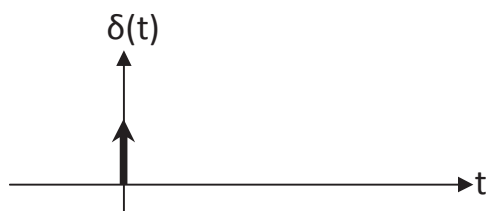
ورودی سهمی:

$$a(t) = \begin{cases} t^2 & t \geq 0 \\ 0 & t < 0 \end{cases}$$



ورودی ضربه (دلتای دیراک یا پالس):

$$\delta(t) = \frac{du(t)}{dt}$$



مفهوم خطای حالت ماندگار:

$$\mu(s) = \frac{Y(s)}{R(s)} = \frac{\text{قسمت لاپلاس خروجی}}{\text{قسمت لاپلاس ورودی}}$$

مثال:

$$G_s = \frac{1}{(s + 2)(s + 3)}$$

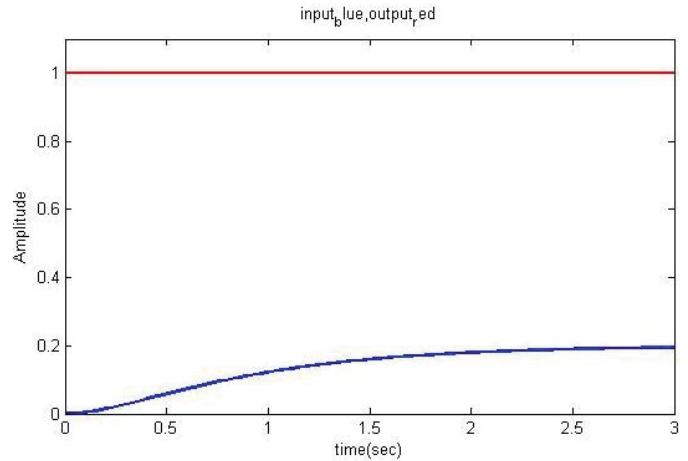
num=1;

den=conv([1 2],[1 3]);

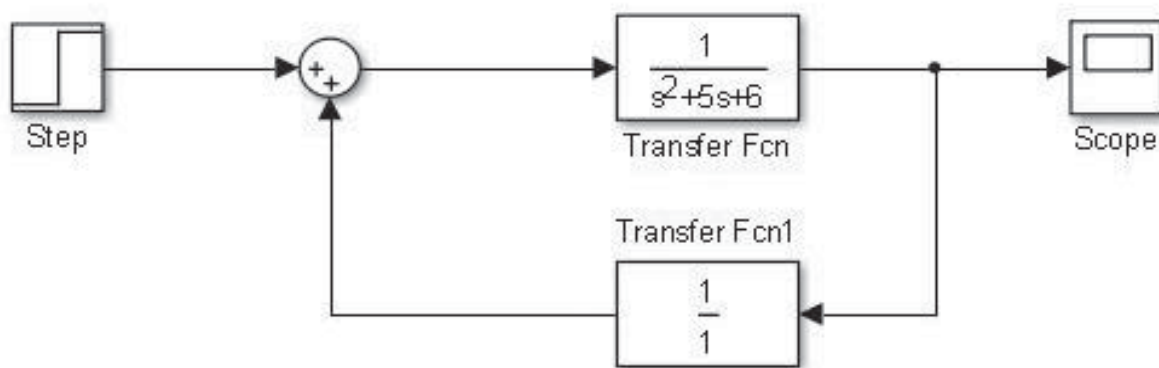
sys=tf(num,den);

sys_cl=feedback(sys,-1);


```
[y,t]=step(sys_c1);
u=ones(size(t));
plot(t,y,'b',t,u,'r')
axis([0 3 0 1.1]);
xlabel('time(sec)');
ylabel('Amplitude');
title('input_blue,output_red')
```



توجه: می توان این سیستم کنترل را در سیمولینک شبیه سازی کرده و پاسخ پله آن را مشاهده نمود.



می توانید بلوک های مورد نیاز را از مسیر های زیر انتخاب کنید.

Simulink /Sources /Step

Simulink /Math Operations /Sum

Simulink /Continuous /Transfer Fcn

Simulink /Sinks /Scope

تنظیمات مورد نیاز:

- روی Transfer Fcn دابل کلیک کرده و تنظیمات آن را به صورت زیر تغییر دهید.

Parameters

Numerator coefficients:

Denominator coefficients:

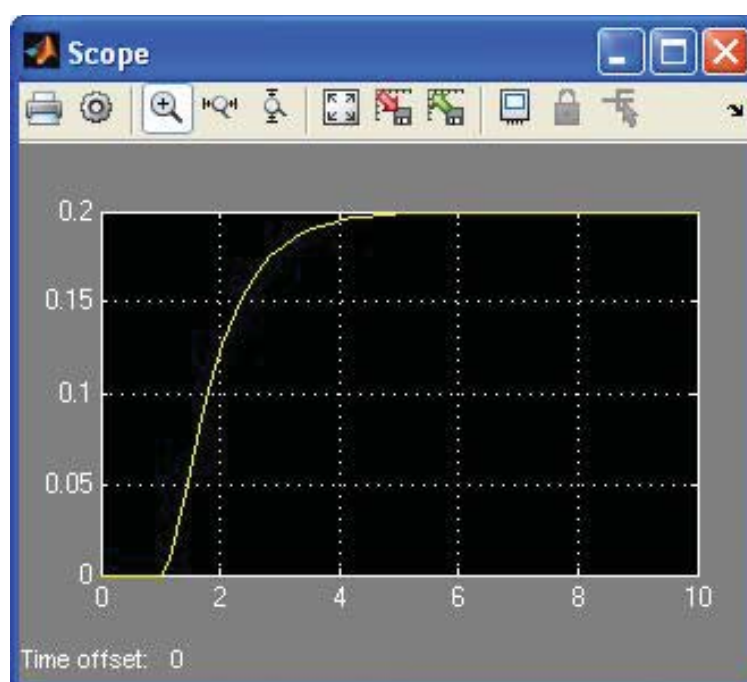
- روی Transfer Fcn1 دابل کلیک کرده و تنظیمات آن را به صورت زیر تغییر دهید.

Parameters

Numerator coefficients:

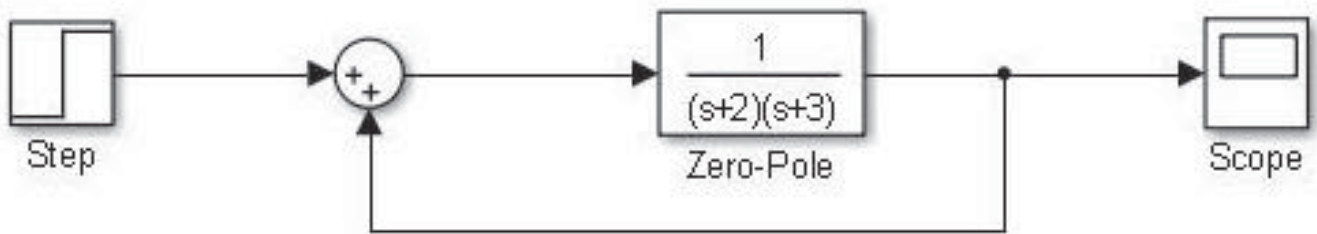
Denominator coefficients:

حال سیستم را Run کرده و پاسخ پله را از اسکوپ مشاهده کنید.



می توانید بجای بلوک Transfer Fcn از بلوک Zero-Pole استفاده کنید. این بلوک را از مسیر زیر بیابید.

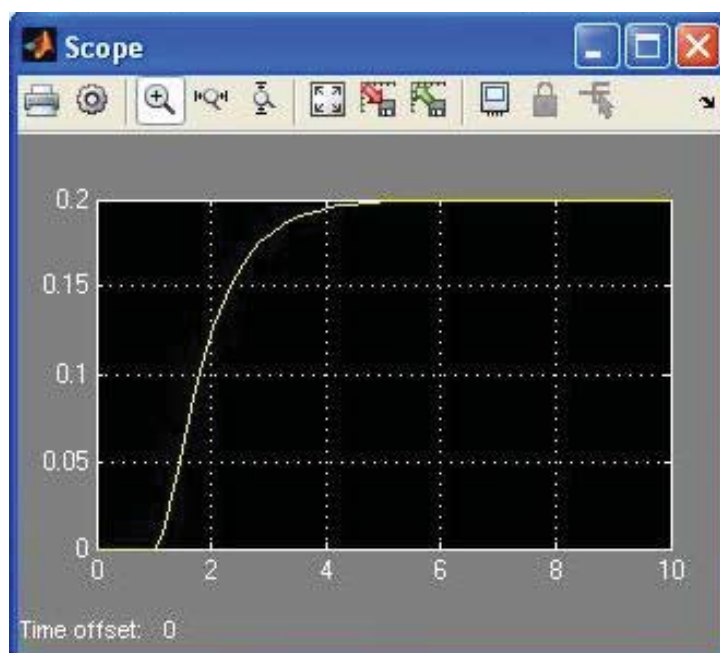
Simulink /Continuous /Zero-Pole



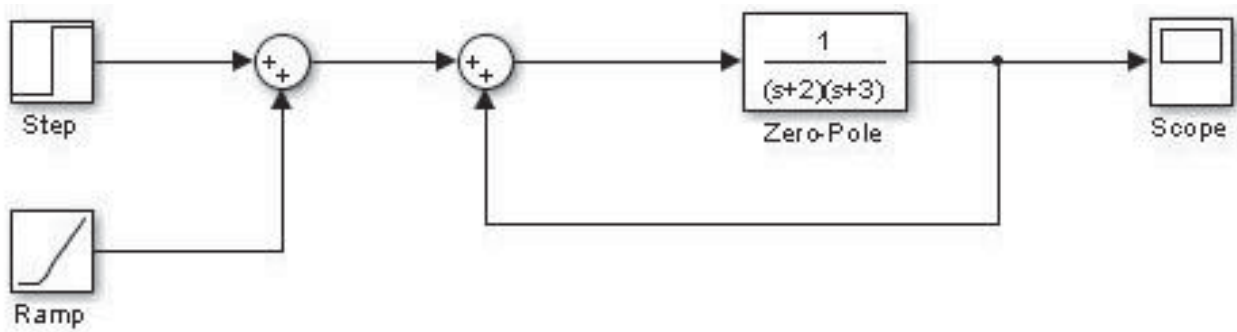
روی بلوک Zero-Pole دابل کلیک کرده و تنظیمات آن را به صورت زیر تغییر می دهیم.

Parameters	
Zeros:	[]
Poles:	[-2 -3]
Gain:	[1]

حالا سیستم را Run کرده و پاسخ پله را مشاهده می کنیم.

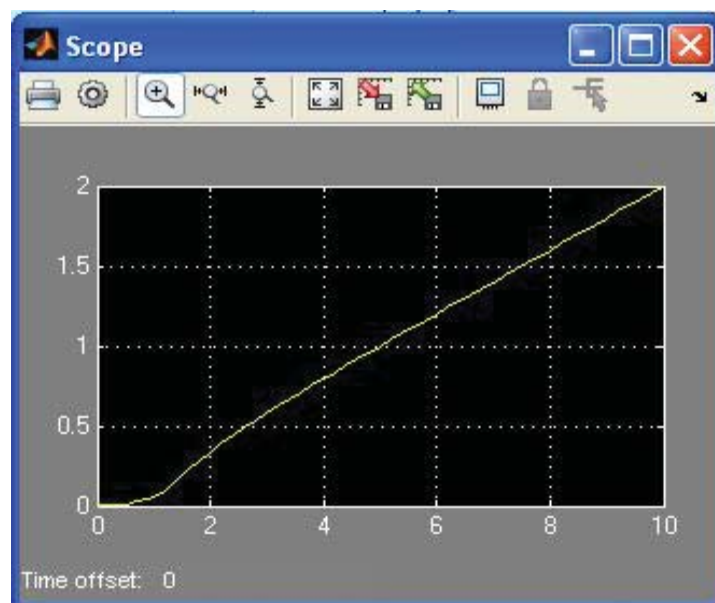


اگر بخواهیم پاسخ این سیستم را با دو ورودی شیب و پله مشاهده کنیم به صورت زیر عمل می‌کنیم.



بلوک ورودی شیب را از مسیر زیر بیابید.

Simulink /Sources /Ramp



مکان هندسی ریشه‌ها:

```
>> a=zpk([-1],[-2 -3],[10])
```

a =

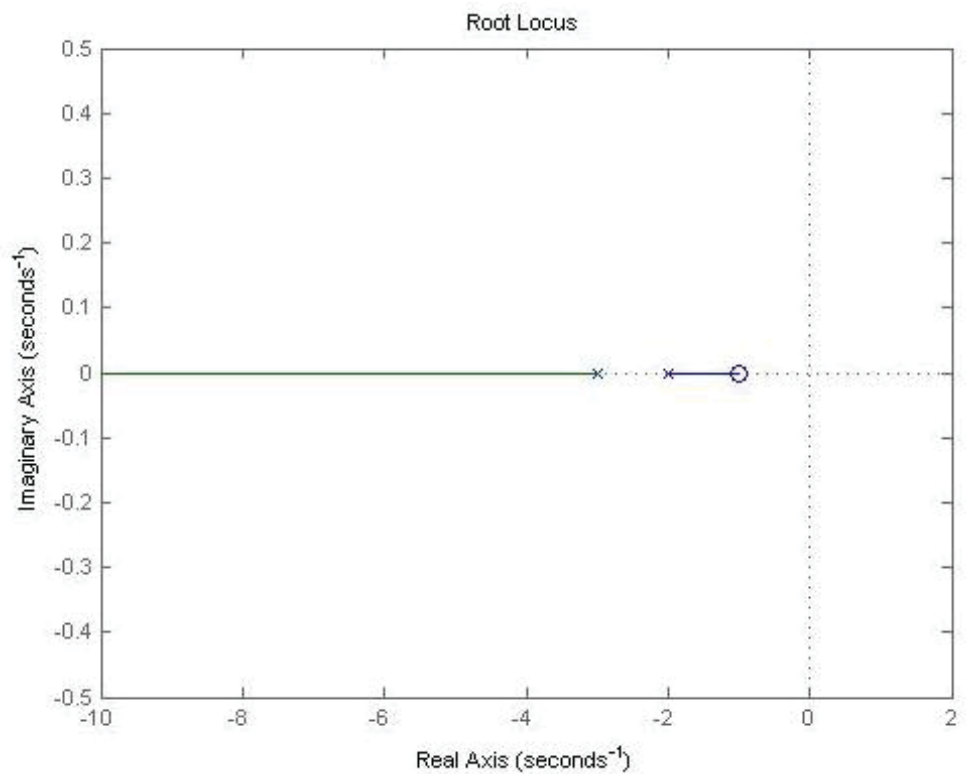
10 (s+1)

(s+2) (s+3)

Continuous-time zero/pole/gain model.

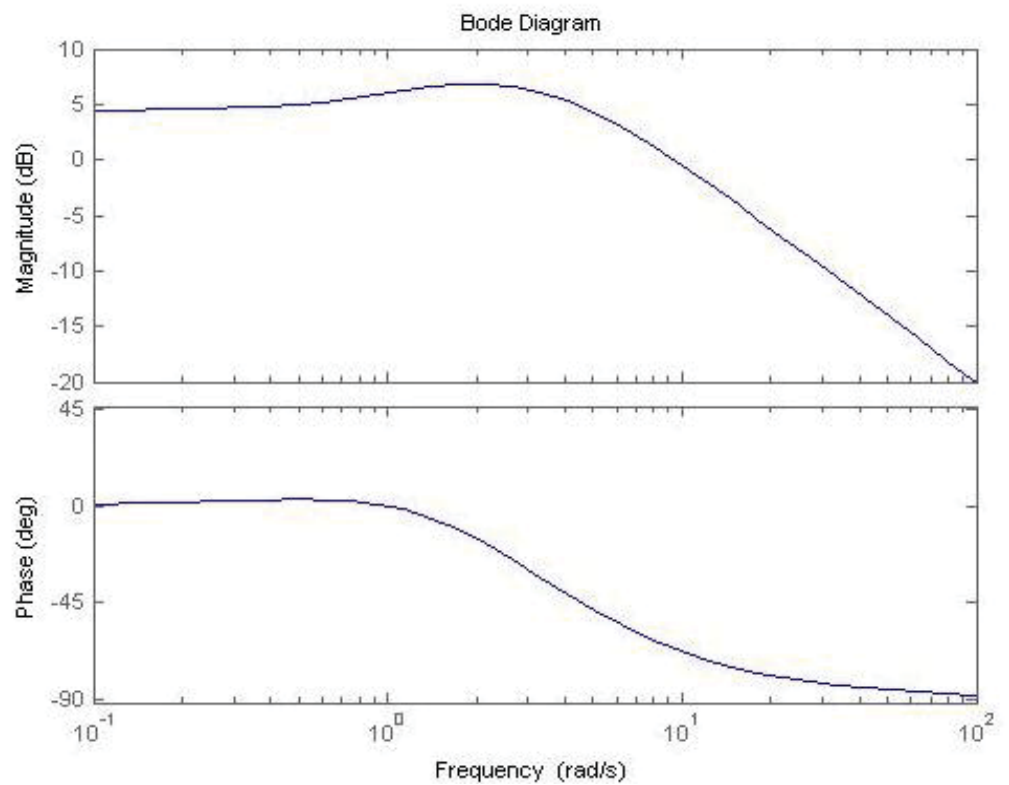
دستور rlocus :

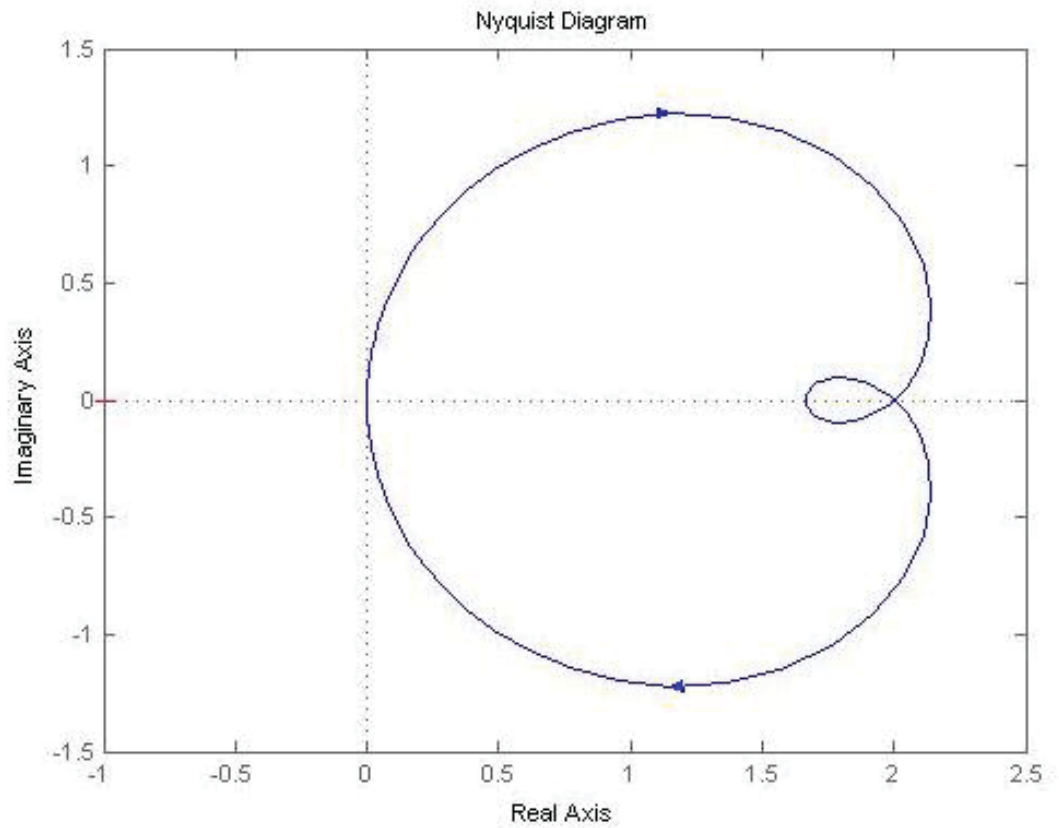
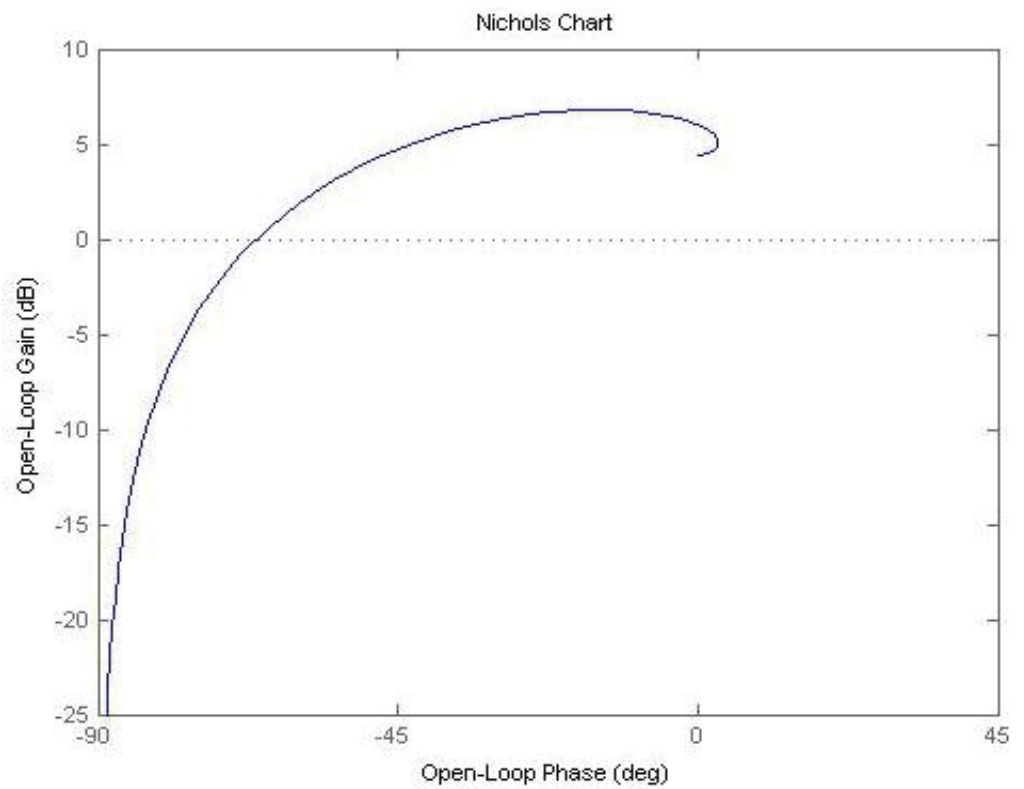
>> rlocus(a)



دستور bode :

>> bode(a)



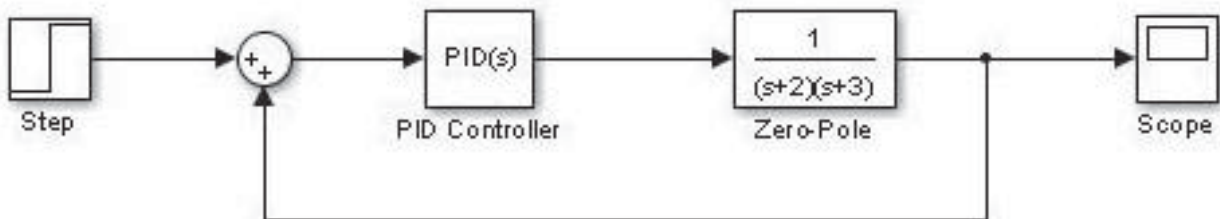
دستور `nyquist(a)` :`>> nyquist(a)`دستور `nichols` :`>> nichols(a)`

جبران ساز: PID

P : عدد ثابت

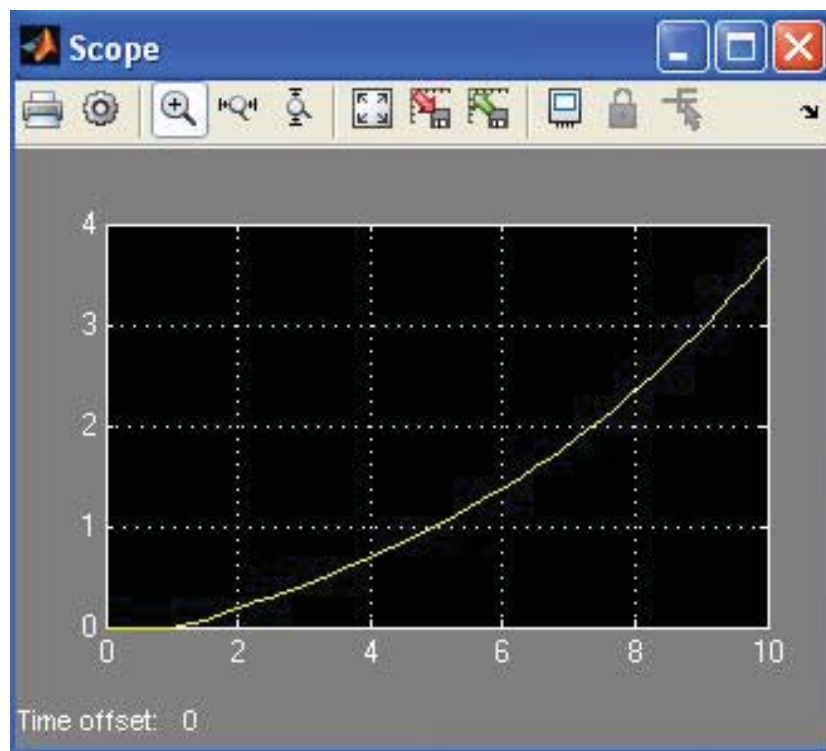
I : انتگرال گیر

D : مشتق گیر



مسیر بلوک PID :

Simulink /Continuous /PID Controller



واسط گرافیکی (GUI): Graphical User Interface

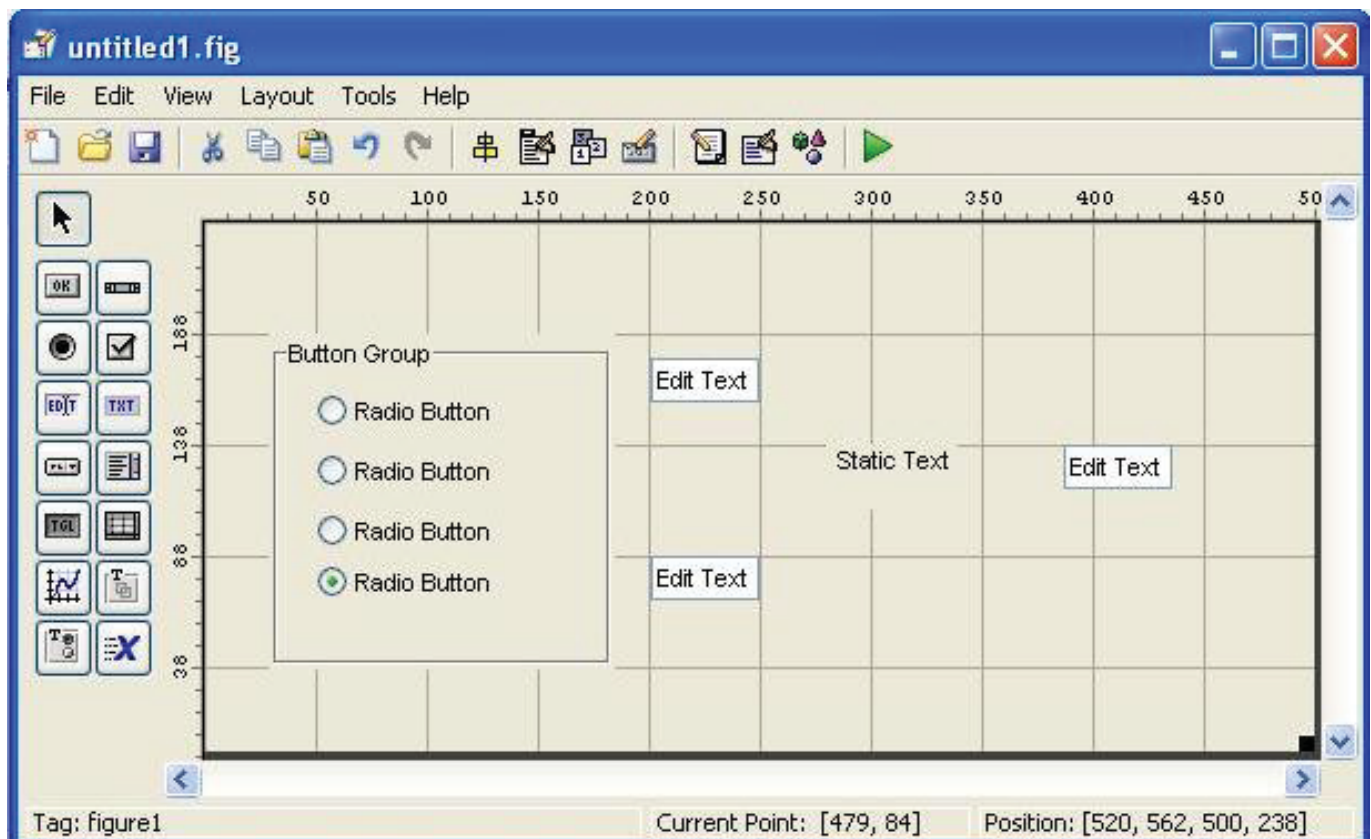
file / new / GUI / blank GUI / ok

- در یک GUI ، کنترلر گرافیکی slider برای مقادیر پیوسته است و با دادن دو حد بالا و پایین می تواند به طور پیوسته تغییر کند.
- در یک GUI ، المان check box همانند Radio button است.
- در یک GUI ، برای تغییر نام Axes قسمت Tag آن را تغییر می دهیم.
- در یک GUI ، در قسمت جعبه گفتگو questdlg برای سوال پرسیدن از کاربر استفاده می شود.
- در جعبه گفتگوی GUI برای ذخیره فایل از دستور uiputfile استفاده می شود.
- برای فراخوانی جعبه گفتگو و خطا و هشدار ، دستور errordlg به کار می رود.
- برای ایجاد یک جعبه گفتگوی کلی در GUI از dialog استفاده می شود.
- یک GUI با پسوند fig در متلب ذخیره می شود.
- برای فراخوانی پیام هشدار در GUI از dialog استفاده می شود.
- یک GUI با پسوند fig در متلب ذخیره می شود.
- برای فراخوانی پیام هشدار در GUI از دستورات errordlg و dialog استفاده می شود.
- برای تغییر فونت یک GUI به قسمت preference از منوی آن مراجعه می کنیم.
- برای نمایش نمودارها در GUI از المان axes استفاده می شود.
- در GUI ، تابع مورد نیاز برای ایجاد یک شیء را با uicontrol فراخوانی می کنیم.
- در یک GUI ، برای ذخیره داده کاربردی در یک شکل از دستور guidata استفاده می شود.
- در GUI ، به کدی که برنامه ای در پاسخ به یک رویداد ایجاد می کند call back می گویند.
- در GUI ، المان Edit text برای مقدار متغیر و statictext برای مقدار ثابت می باشد.

مثال: نمونه ای از یک ماشین حساب ساده با ۴ عمل اصلی (-, +, *, /):
ابتدا از مسیر زیر یک صفحه خالی برای طراحی GUI ایجاد می کنیم.

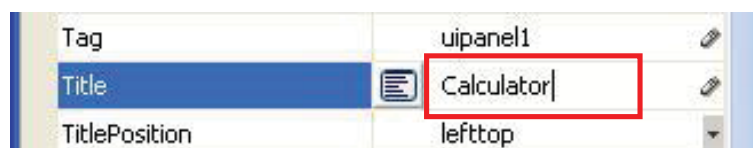
file / new / GUI / blank GUI / ok

ابتدا با استفاده از منوی ابزار سمت چپ ، طرح اولیه ماشین حساب را به طور دلخواه بر روی صفحه خالی پیاده می کنیم.

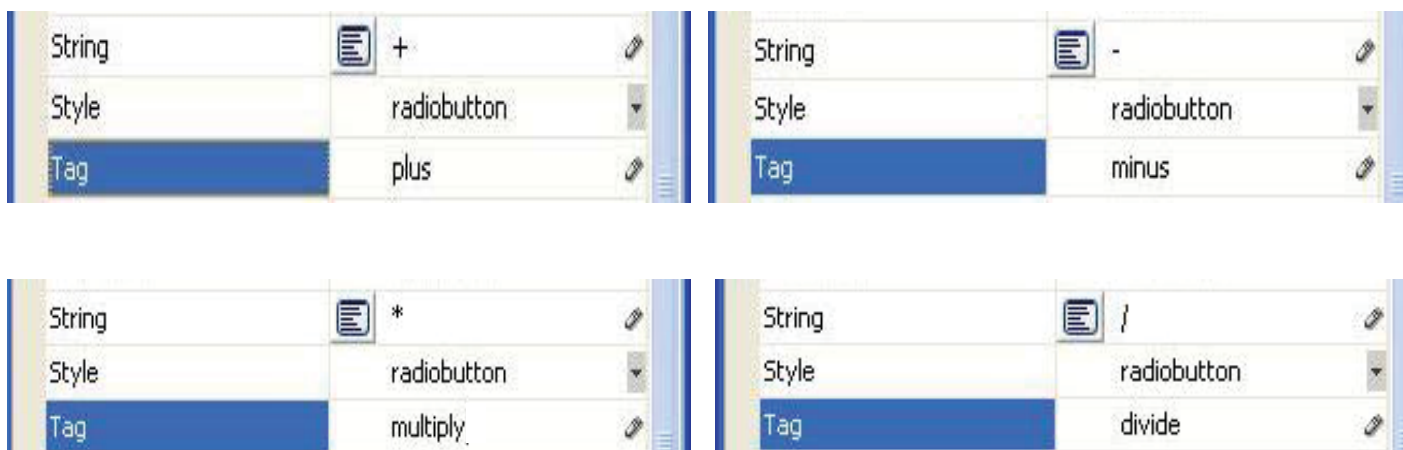


حال روی هر کدام از این المان ها دابل کلیک می کنیم تا صفحه ی Property Inspector مربوط به آنها باز شود. سپس برخی از تنظیمات پیش فرض را در این صفحه به صورت زیر تغییر می دهیم.

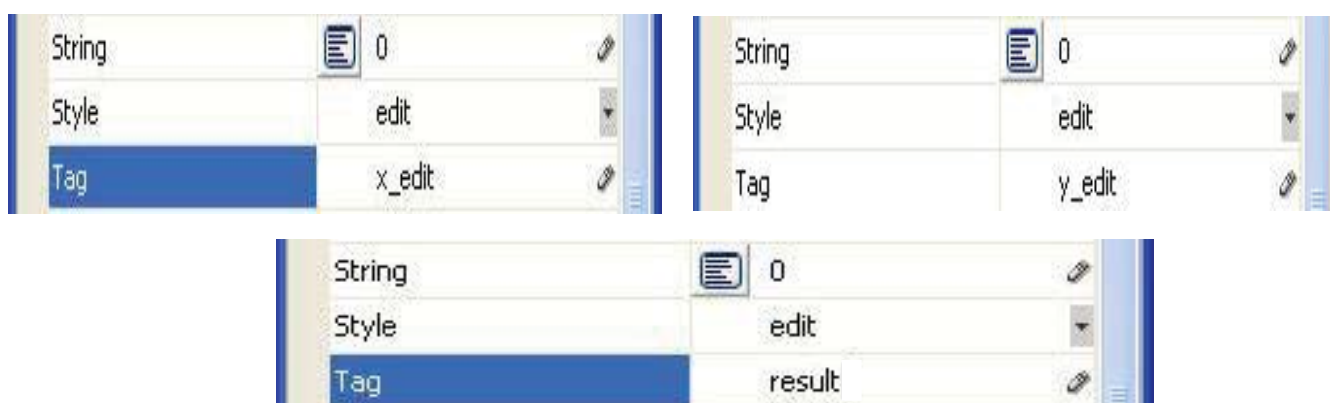
- روی button group دابل کلیک کرده و از قسمت title نام آنرا به Calculator تغییر می دهیم.



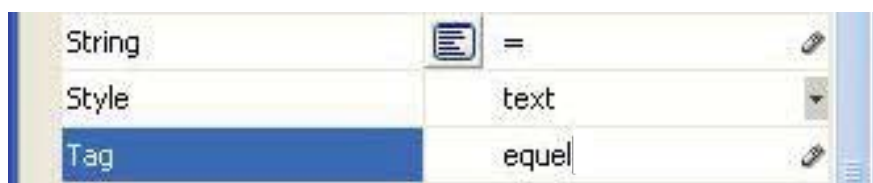
- روی هر کدام از Radio button ها دابل کلیک کرده و قسمت String و Tag آنها را به صورت زیر تغییر می دهیم.



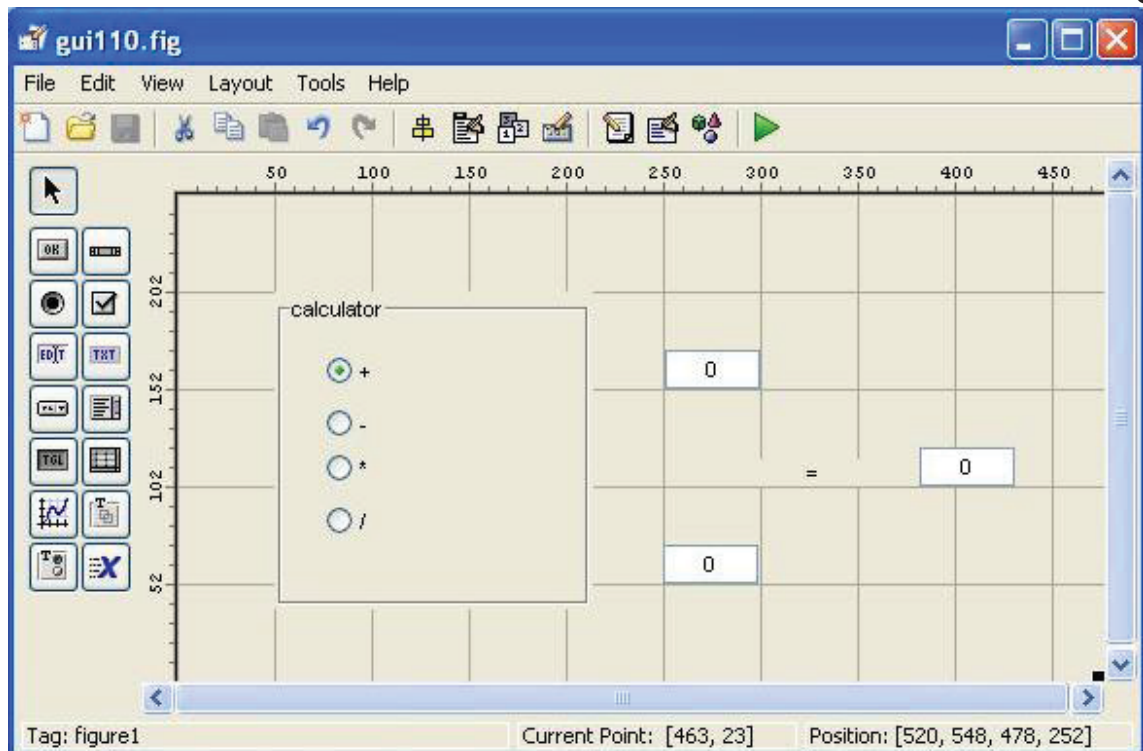
- در اینجا از سه المان edit text استفاده شده، که دو المان اول برای وارد کردن اعداد و المان سوم برای نمایش جواب در نظر گرفته شده است.



- روی المان static text دابل کلیک کرده و تنظیمات آن را به صورت زیر تغییر می دهیم.



طرح اولیه ماشین حساب به صورت زیر مشاهده می شود.



حال برنامه را save می کنیم تا یک editor باز شود. در این editor تمام برنامه های استفاده شده نشان داده می شود.

صفحه ی ادیتور را می بندیم و روی group button کلیک راست کرده و به مسیر زیر می رویم.

view callbacks /SelectionChangeFcn

ادیتور دیگری باز می شود که در انتهای آن دستوری به شکل زیر مشاهده می شود.

```
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
```

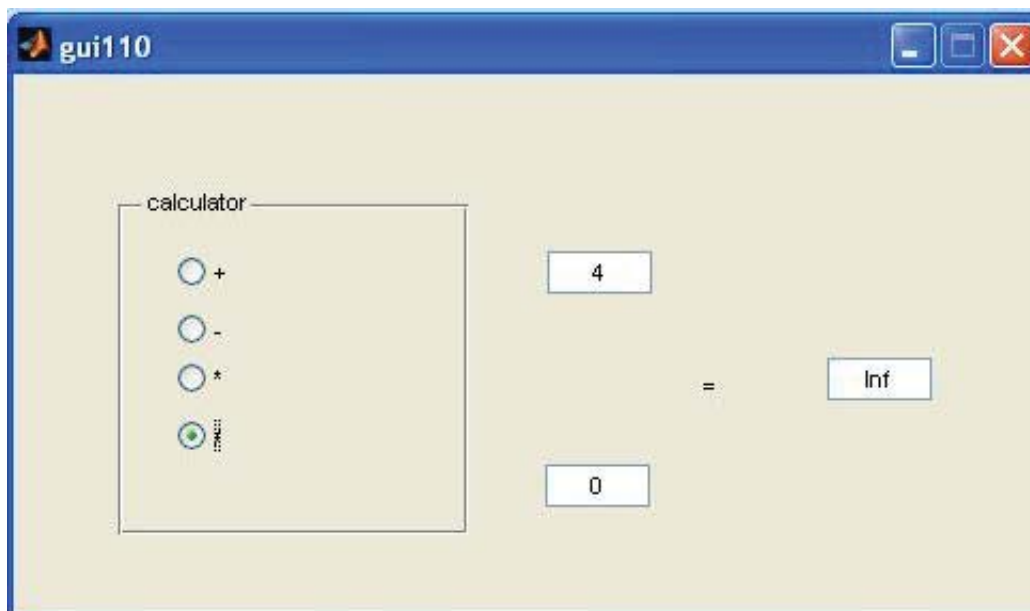
در پایین این دستور برنامه ی مورد نظر خود را می نویسیم.

```
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
```

```
t1=get(handles.plus, 'value');
t2=get(handles.minus, 'value');
t3=get(handles.multiply, 'value');
t4=get(handles.divide, 'value');
if t1==1
    x=str2num(get(handles.x_edit, 'string'));
    y=str2num(get(handles.y_edit, 'string'));
    set(handles.result, 'string', num2str(x+y));
```

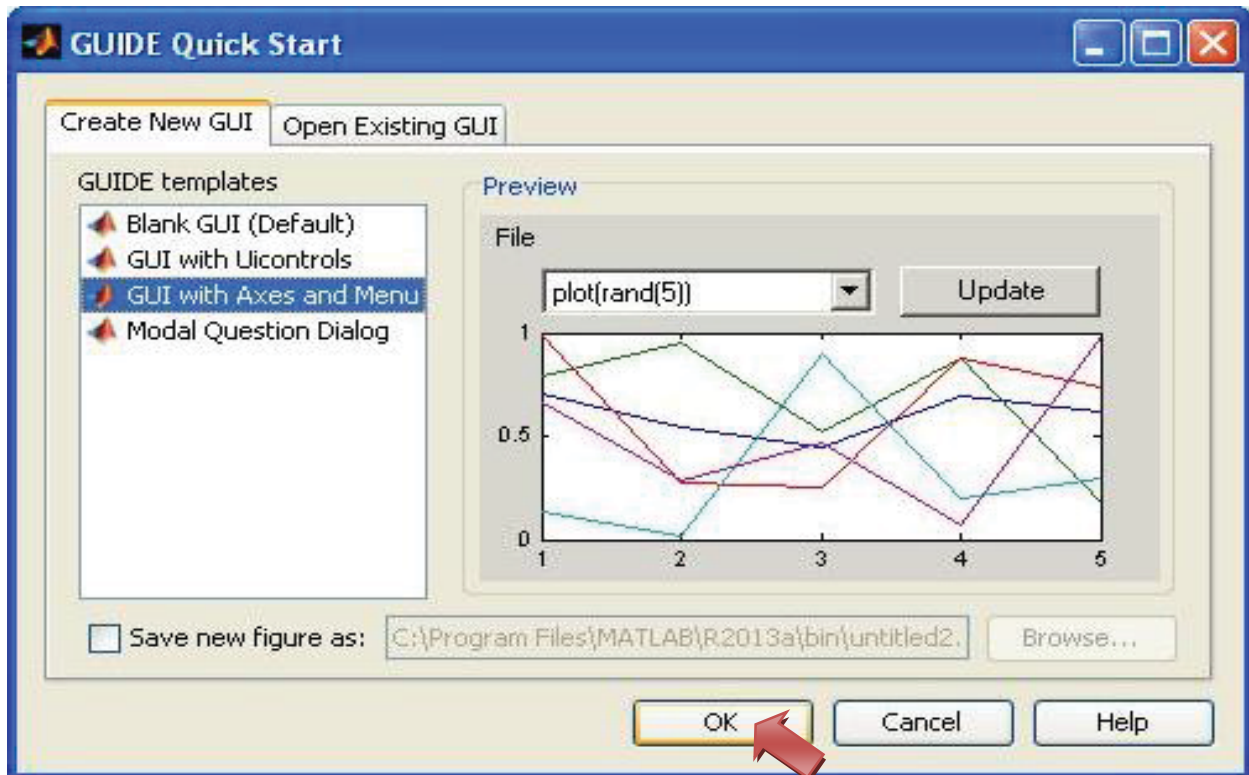
```
elseif t2==1
    x=str2num(get(handles.x_edit,'string'));
    y=str2num(get(handles.y_edit,'string'));
    set(handles.result,'string',num2str(x-y));
elseif t3==1
    x=str2num(get(handles.x_edit,'string'));
    y=str2num(get(handles.y_edit,'string'));
    set(handles.result,'string',num2str(x*y));
else t4==1
    x=str2num(get(handles.x_edit,'string'));
    y=str2num(get(handles.y_edit,'string'));
    set(handles.result,'string',num2str(x/y));
end;
```

سپس برنامه را save کرده و اجرا می کنیم.

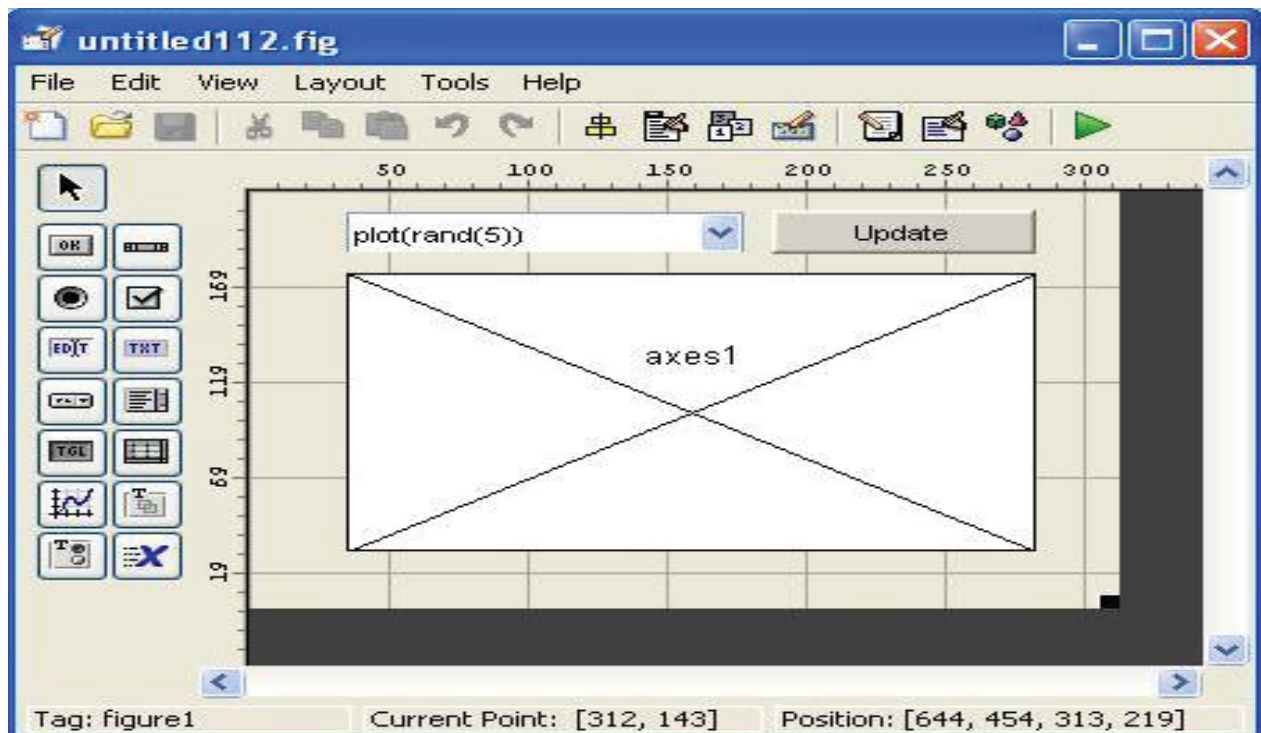


مثال: برای مشاهده یک مثال آماده به مسیر زیر بروید.

file / new / GUI / GUI With Axes and Menu / ok



بر روی ok کلیک می کنیم تا صفحه زیر باز شود



همانطور که مشاهده می شود برای طراحی این GUI از یک push button ، pop-up menu و Axes استفاده شده است. اگر روی هر کدام از این المان ها دابل کلیک کنید می توانید تنظیمات مربوط به آنها را نیز ببینید. حال GUI را save می کنیم تا صفحه ادیتور باز شود. در این ادیتور کل برنامه های مربوط به این GUI قابل مشاهده است. تابع مرتبط با pushbutton به صورت زیر نوشته شده است.

```
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see
GUIDATA)
axes(handles.axes1);
cla;

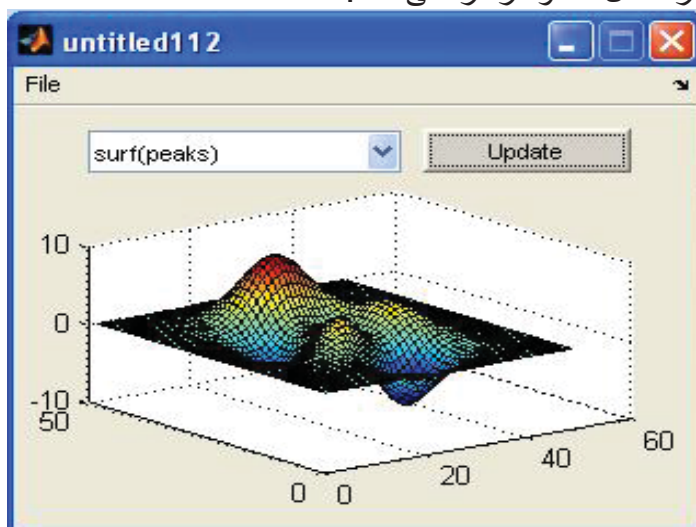
popup_sel_index = get(handles.popupmenu1, 'Value');
switch popup_sel_index
    case 1
        plot(rand(5));
    case 2
        plot(sin(1:0.01:25.99));
    case 3
        bar(1:.5:10);
    case 4
        plot(membrane);
    case 5
        surf(peaks);
end
```

حالت های مختلف منوی pop-up

• دستور switch/case مشابه دستور if می باشد.

• دستور peaks به طور پیش فرض یک نمودار شامل قله و دره را می دهد.

حال برنامه را اجرا می کنیم.



پردازش تصویر

نوشتن فایل های گرافیکی im write

فراخوانی تصویر:

```
>> a=imread('trees.tif');
```

نمایش تصویر:

```
>> imshow(a)
```



نمایش اطلاعات تصویر:

```
>> info=imfinfo('trees.tif')
```

info =

7x1 struct array with fields:

Filename

FileModDate

FileSize

Format

FormatVersion

Width

Height

BitDepth

ColorType
FormatSignature
ByteOrder
NewSubFileType
BitsPerSample
Compression
PhotometricInterpretation
StripOffsets
SamplesPerPixel
RowsPerStrip
StripByteCounts
XResolution
YResolution
ResolutionUnit
Colormap
PlanarConfiguration
TileWidth
TileLength
TileOffsets
TileByteCounts
Orientation
FillOrder
GrayResponseUnit
MaxSampleValue
MinSampleValue

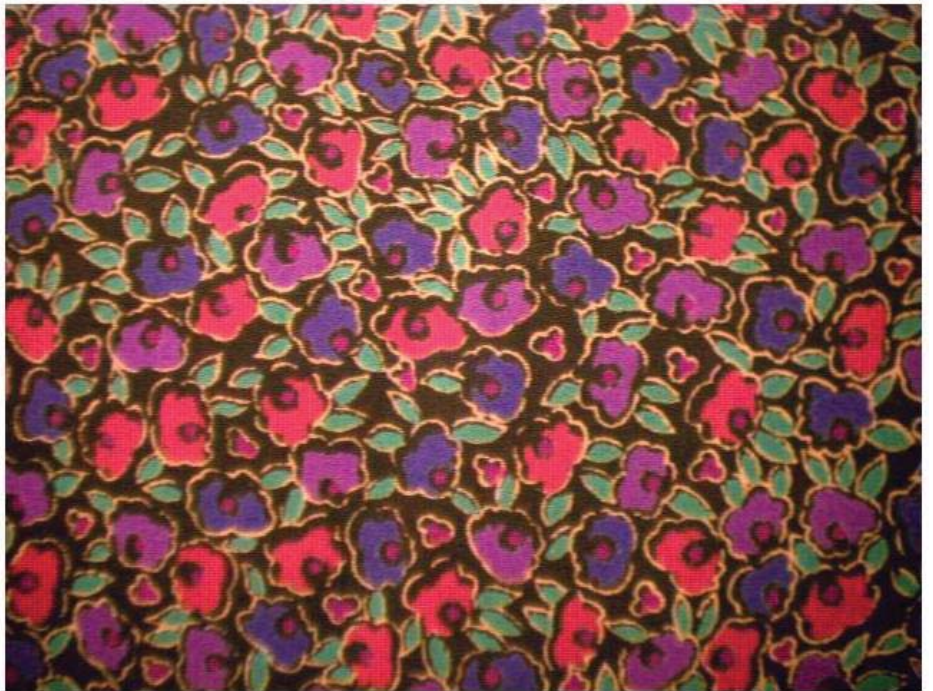
Thresholding

Offset

• gif ، jpg ، tif پسوند تصویر در متلب

```
>> a=imread('fabric.png');
```

```
>> imshow(a)
```

دستور `im2bw` : عکس را باینری (0,1) نشان می دهد.

```
>> load trees
```

```
>> bw=im2bw(X,map,0.4);
```

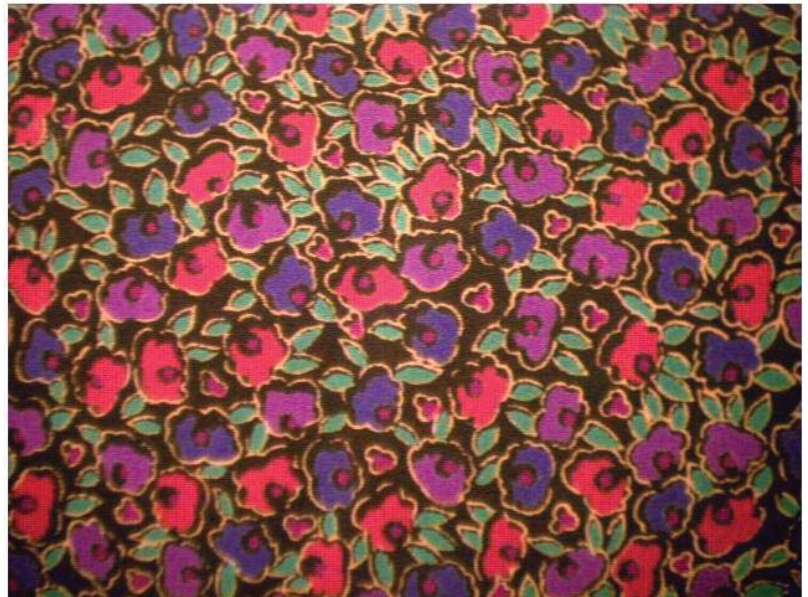
```
>> imshow(bw)
```

درجه سیاه و سفیدی



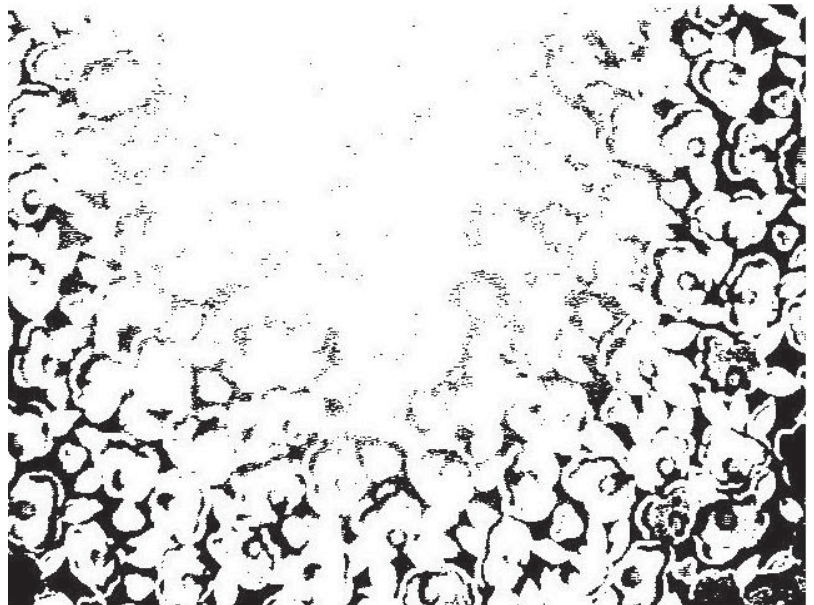
```
>> a=imread('fabric.png');
```

```
>> imshow(a)
```



```
>> z=im2bw(a,0.2);
```

```
>> figure,imshow(z)
```



تبدیل عکس RGB به Gray :

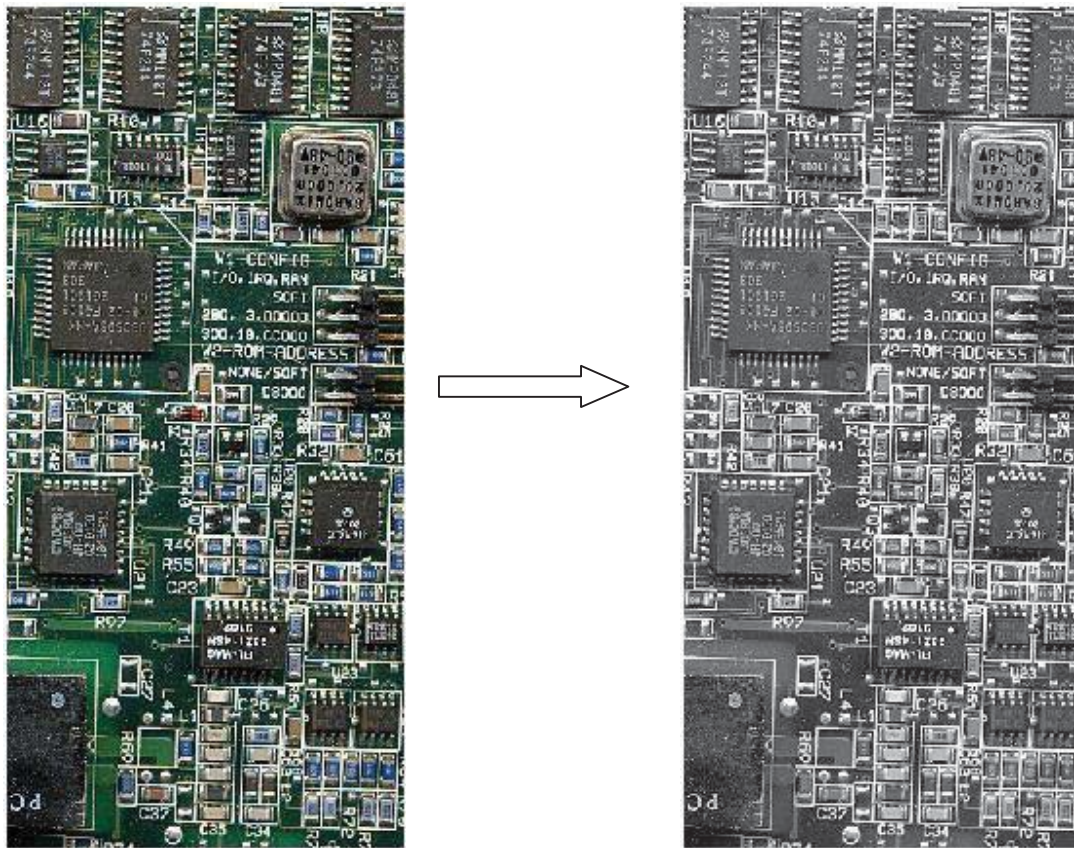
```
>> n=rgb2gray
```

مثال:

```
>> I = imread('board.tif');
```

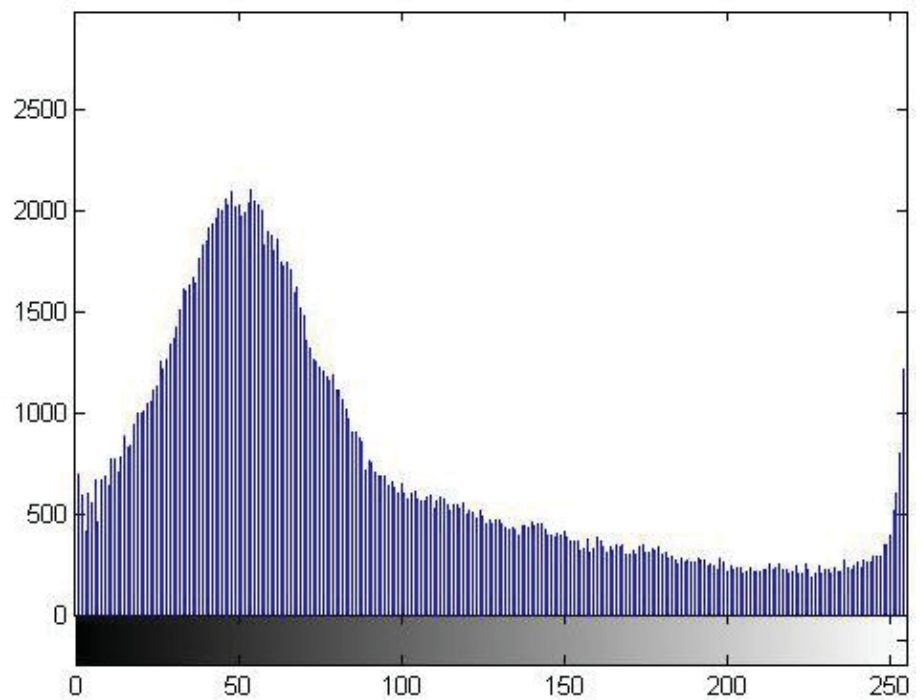
```
>> J = rgb2gray(I);
```

```
>> figure, imshow(I), figure, imshow(J);
```



هیستوگرام:

```
>> imhist(J)
```



تبدیل عکس Gray به index :

```
>> t=gray2ind(J);
```

```
>> imshow(t)
```



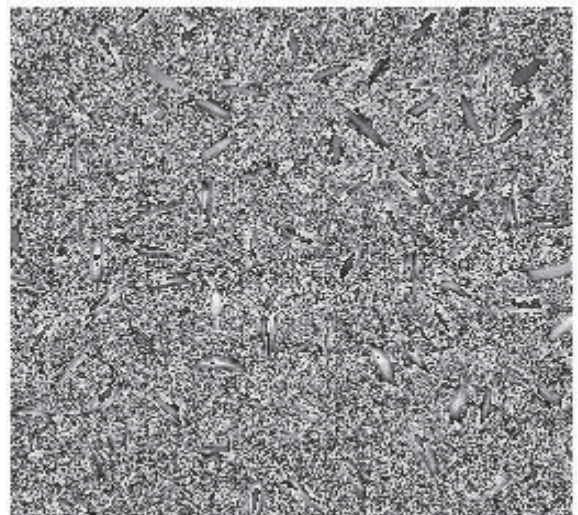
مثال:

```
>> a=imread('rice.png');
```

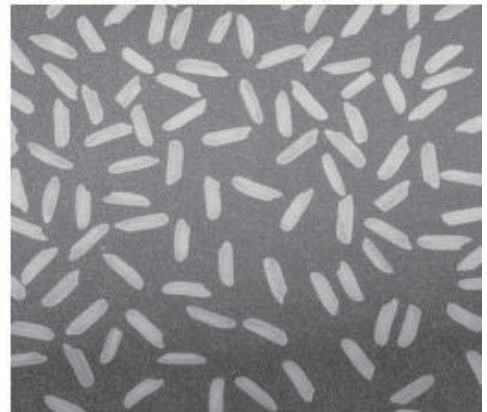
```
>> b=bitshift(a,4);
```

```
>> imshow(b)
```

تغییر درجه خاکستری بودن تصویر



```
>> figure , imshow('rice.png')
```



```
>> c=bitshift(a,-2);
```

```
>> figure , imshow(c)
```

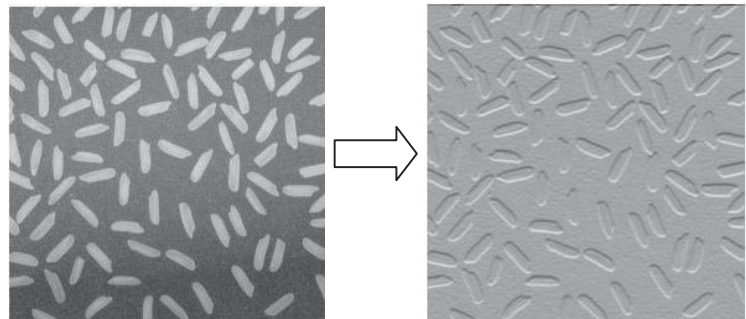


```
>> I=imread('rice.png');
```

```
>> J=filter2(fspecial('sobel'),I);
```

```
>> K=mat2gray(J);
```

```
>> imshow(I),figure , imshow(K)
```

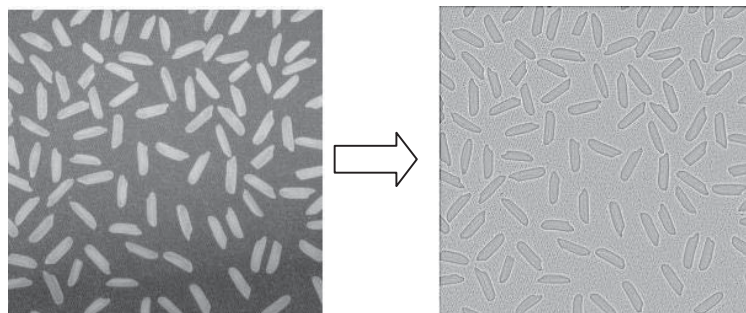


دستور فیلتر و دستور تبدیل ماتریس به گری: `filter` و `mat2gray`

```
>> J=filter2(fspecial('log'),I);
```

```
>> K=mat2gray(J);
```

```
>> imshow(I),figure,imshow(K)
```



مثال: با این مثال می خواهیم ترکیب رنگ ها را بررسی کنیم. توجه شود که همان قرص سفید نیوتن بدست می آید.
تصویر RGB یک ماتریس $M*N*3$ است که رنگ های آن قرمز، سبز و آبی می باشد.
معادله دایره ی توپر به صورت زیر است:

$$(x - x_0)^2 + (y - y_0)^2 = R^2$$

مرکز دایره قرمز را روی $(-0.4, 0.4)$ می گذاریم با شعاع ۱

```
>> [x,y]=meshgrid(linspace(-2,2,200));
```

```
>> R=1;
```

```
>> r=zeros(size(x));
```

```
>> rind=find((x+0.4).^2+(y+0.4).^2<R^2);
```

```
>> r(rind)=1;
```

برای دایره های سبز و آبی نیز همین کار را انجام می دهیم و سپس فصل مشترک سه دایره پیدا می شود.

```
>> g=zeros(size(x));
```

```
>> gind=find((x-0.4).^2+(y+0.4).^2<R^2);
```

```
>> g(gind)=1;
```

```
>> b=zeros(size(x));
```

```
>> bind=find((x+0.4).^2+(y-0.4).^2<R^2);
```

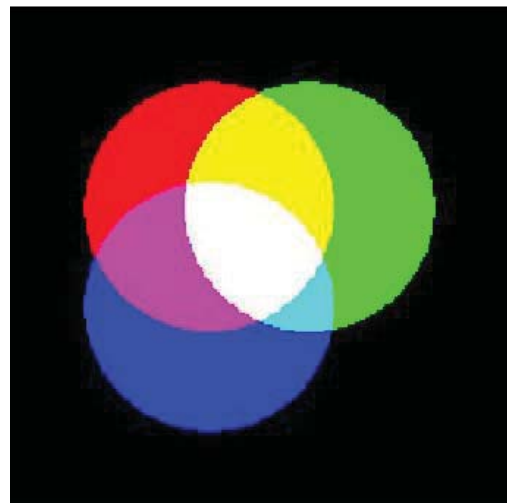
```
>> b(bind)=1;
```

```
>> rgb=cat(3,r,g,b);
```

```
>> imagesc(rgb)
```

```
>> axis equal off
```

برای ادغام سه ماتریس دستور concatenate بکار می رود.



مثال: فرض کنید می خواهیم یک تصویر را بصورت لایه لایه بررسی کنیم. با مثالی از عکس مغز انسان بررسی می کنیم.

```
>> load mri
```

```
>> D = squeeze(D);
```

→ برداری برای ایجاد رنگ

```
>> figure('Colormap',map)
```

```
>> image_num = 8;
```

```
>> image(D(:,:,image_num))
```

```
>> axis image
```

```
>> x = xlim;
```

```
>> y = ylim;
```

```
>> contourslice(D,[],[],image_num)
```

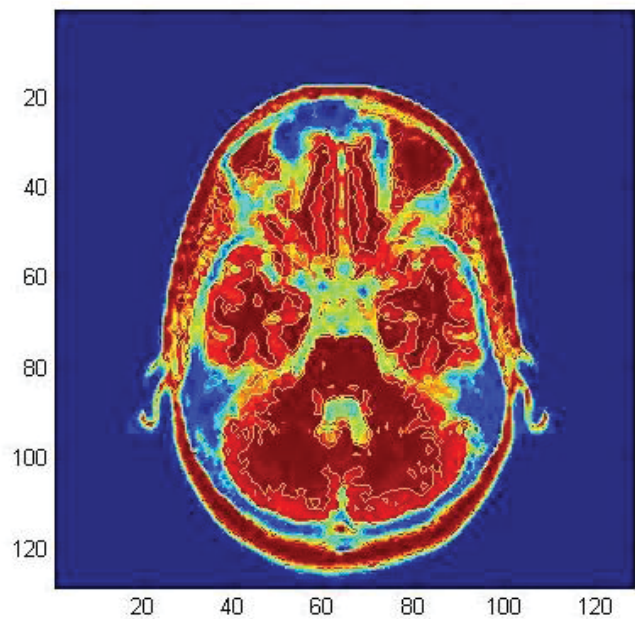
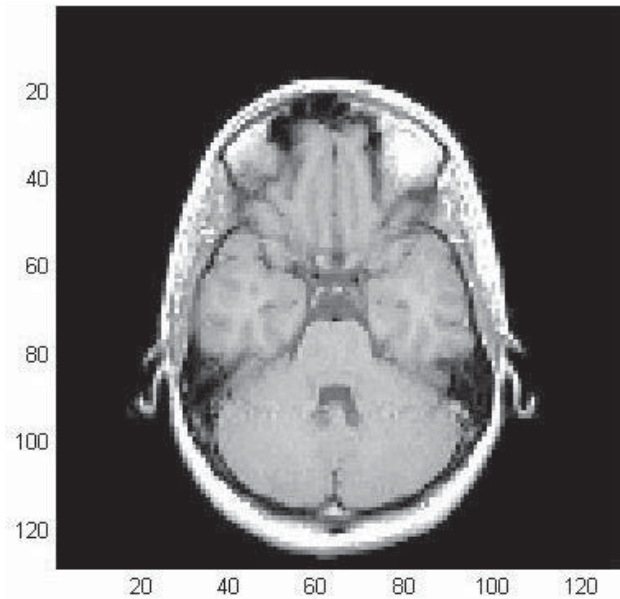
```
>> axis ij
```

```
>> xlim(x)
```

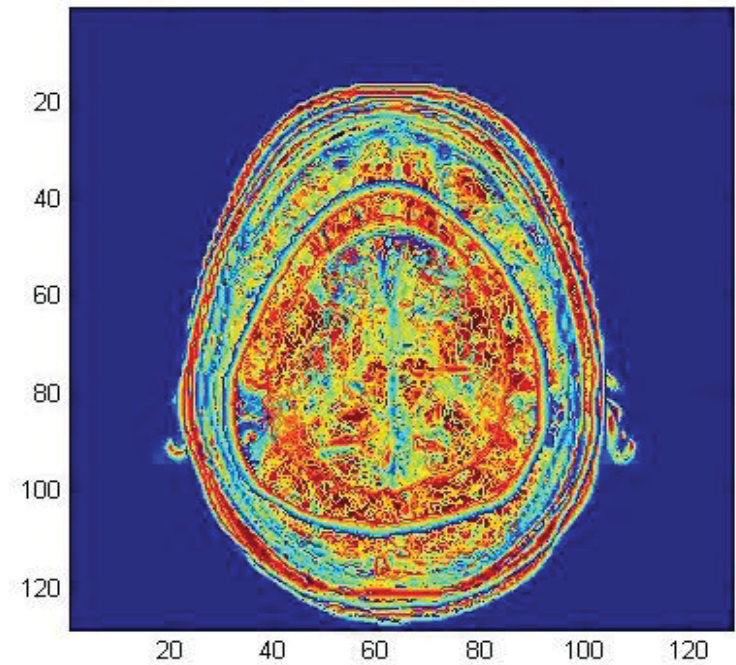
```
>> ylim(y)
```

```
>> daspect([1,1,1])
```

```
>> colormap('default')
```



```
>> phandles=contourslice(D,[],[],[1,12,19,27],8);
```

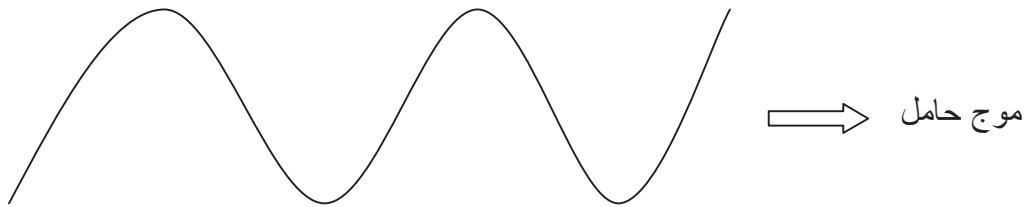


```
>> view(3); axis tight
```

```
>> set(phandles,'linewidth',2)
```

جدول رنگ ها:

red	green	blue	
0	0	0	Black
1	1	1	White
1	0	0	Red
0	1	0	Green
0	0	1	Blue
1	1	0	Yellow
0.5	0.5	0.5	Gray



$$x(t) = A \sin(\omega t + \varphi)$$

A : دامنه

$\omega = 2\pi f$ \implies f بر حسب هرتز است

φ : فاز زاویه

۱- انتقال دامنه AM \leftarrow در متلب ASK

`y=qammod(x,m)`

`z=qamdemod(x,m,ini-phase)`

موج

فاز اولیه

• ساخت مدولاسیون دامنه به صورت پیش فرض:

```
>> h=modem.pskmod
```

h =

Type: 'PSK Modulator'

M: 2

PhaseOffset: 0

Constellation: [1.0000 + 0.0000i -1.0000 + 0.0000i]

SymbolOrder: 'Binary'

SymbolMapping: [0 1]

InputType: 'Integer'

مثال:

```
>> X=randint(10,1,8)
```

```
X =  
      یک ماتریس 10*1  
      با ورودی تصادفی 0-7  
6  
7  
1  
7  
5  
0  
2  
4  
7  
7
```

```
>> h=modem.qammod(8)
```

```
h =  
      Type: 'QAM Modulator'  
      M: 8  
      PhaseOffset: 0  
      Constellation: [1x8 double]  
      SymbolOrder: 'Binary'  
      SymbolMapping: [0 1 2 3 4 5 6 7]  
      InputType: 'Integer'
```

```
>> y=modulate(h,X)            مدولاسیون
```

```
y =  
3.0000 + 1.0000i  
3.0000 - 1.0000i
```

```
-3.0000 - 1.0000i
 3.0000 - 1.0000i
 1.0000 - 1.0000i
-3.0000 + 1.0000i
-1.0000 + 1.0000i
 1.0000 + 1.0000i
 3.0000 - 1.0000i
 3.0000 - 1.0000i
```

```
>> g=modem.qamdemod(h)
```

```
g =
```

```
    Type: 'QAM Demodulator'
```

```
    M: 8
```

```
    PhaseOffset: 0
```

```
    Constellation: [1x8 double]
```

```
    SymbolOrder: 'Binary'
```

```
    SymbolMapping: [0 1 2 3 4 5 6 7]
```

```
    OutputType: 'Integer'
```

```
    DecisionType: 'Hard decision'
```

```
>> z=demodulate(g,y)
```

```
z =
```

```
 6
```

```
 7
```

```
 1
```

```
 7
```

```
 5
```

```
 0
```

```
 2
```

4
7
7

۲- انتقال فرکانس FM ← FSK در متلب

```
y=fskmod(x,m,freq-sep,nsamp,fs)
```

```
z=fskdemod(x,m,freq-sep,nsamp,fs)
```

مثال: برنامه ای بنویسید برای مدولاسیون شیفیت فرکانسی بر روی سیگنال تصادفی و تابع چگالی طیف توان آن.

```
>> m=4;
```

```
>> freqsep=8;
```

```
>> nsamp=8;
```

```
>> fs=32;
```

```
>> x=randint(1000,1,m);
```

```
>> y=fskmod(x,m,freqsep,nsamp,fs);
```

```
>> ly=length(y);
```

```
>> freq=[-fs/2:fs/ly:fs/2-fs/ly]; → طول فرکانس
```

```
>> syy=10*log10(fftshift(abs(fft(y))));
```

شیفت فازی

تبدیل فوریه

```
>> plot(freq,syy)
```

```
>> xlabel('Frequency in Hertz')
```

```
>> ylabel('Fsk modulation')
```

```
>> grid on
```

۳- انتقال فازی PM ← PSK در متلب

```
y=pskmod('x',m,ini-phase)
```

```
0<x<m-1
```

```
z=pskdemod('y',m,ini-phase)
```

مثال: برنامه ای بنویسید که حساسیت مدولاسیون های PSK و QAM را نسبت به نویز فاز با یکدیگر مقایسه کند بطوریکه از مقایسه دیاگرام سمبل ها نتیجه گیری شود که مدولاسیون فاز به علت دایره ای بودن موقعیت سمبل ها نسبت به تغییرات فاز حساس تر از مدولاسیون دارند که موقعیت سمبل های آن خطی است، می باشد.

```
>> len = 10000; M = 16;
```

```
طول رشته عدد
```

```
>>msg = randi([0 M-1],len,1);
```

```
>>txpsk = pskmod(msg,M);
```

```
>>txpam = pammod(msg,M);
```

```
مدولاسیون دامنه و فاز
```

```
>>phasenoise = randn(len,1)*.015;
```

```
فرکانس
```

```
ایجاد سیگنال با نویز
```

```
>>rxpsk = txpsk.*exp(j*2*pi*phasenoise);
```

```
>>rxpam = txpam.*exp(j*2*pi*phasenoise);
```

```
فراخوانی مدولاسیون دامنه و فاز
```

```
>>scatterplot(rxpsk); title('Noisy PSK Scatter Plot');grid
```

```
>>scatterplot(rxpam); title('Noisy PAM Scatter Plot');grid
```

```
ایجاد نمودار
```

```
مخابراتی
```

```
>>recovpsk = pskdemod(rxpsk,M);
```

```
>>recovpam = pamdemod(rxpam,M);
```

```
آشکارسازی مدولاسیون
```

```
>>numerrs_psk = symerr(msg,recovpsk)
```

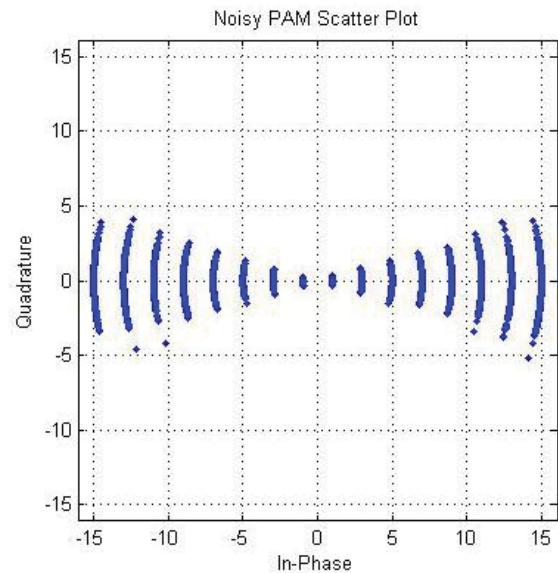
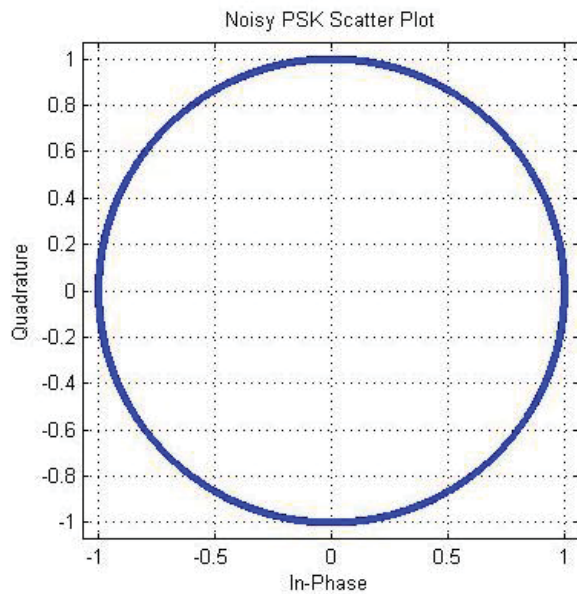
```
>>numerrs_pam = symerr(msg,recovpam)
```

```
دادن خطای سیستم (نویز)
```

```
numerrs_psk = 376
```

```
numerrs_pam = 0
```

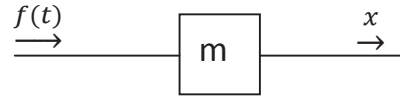
❖ خطای نویز تغییر پذیر است، چون تابع نویز random (تصادفی) می باشد.



این سیستم بهتری می باشد چون خطای آن کمتر است.

مدل ریاضی سیستم فیزیکی:

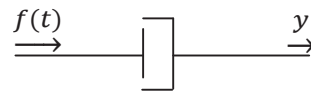
$$f = ma = m \frac{d^2x}{dt^2} \xrightarrow{L} F(s) = ms^2x(s)$$



$$f = k(x - y) \xrightarrow{L} F(s) = k(x(s) - y(s))$$



$$f = B(x' - y') \xrightarrow{L} F(s) = Bs(X(s) - Y(s))$$



$$\sum F = \sum ma$$

$$m \frac{d^2x}{dt^2} = B \left(\frac{df}{dt} - \frac{dy}{dt} \right) + k(F - y)$$

$$m \frac{d^2x}{dt^2} + B \frac{dy}{dt} + kF \xrightarrow{\text{لاپلاس}}$$

$$m[s^2y(s) - sy(0) - y(0)] + B[sy(s) - y(0)] + ky(s) = b[sF(s) - F(0)] + kF(s)$$

$$\text{تابع تبدیل سیستم } G(s) = \frac{y(s)}{F(s)} = \frac{B(s)+k}{ms^2+Bs+k}$$

>> tf([B k],[m B k])

فضای حالت (state space) :

هر معادله دیفرانسیل مرتبه m با یک متغیر را می توان به n معادله دیفرانسیل درجه یک با n متغیر تبدیل کرد.

$$\begin{cases} \dot{x}_1 = a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n + b_{11}u_1 + \dots + b_{1r}u_r \\ \dot{x}_2 = a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n + b_{21}u_1 + \dots + b_{2r}u_r \\ \vdots \\ \dot{x}_n = a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n + b_{n1}u_1 + \dots + b_{nr}u_r \end{cases}$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_{11} \\ b_{21} \\ \vdots \\ b_{n1} \end{bmatrix} \begin{bmatrix} \dots \\ \vdots \\ \dots \end{bmatrix} \begin{bmatrix} b_{1r} \\ \vdots \\ b_{nr} \end{bmatrix} \begin{bmatrix} u_1 \\ \vdots \\ u_r \end{bmatrix}$$

$$\begin{cases} \dot{X} = AX + Bu \\ y = CX + Du \end{cases}$$

مثال:

$$y'' + 5y' + 6y = 7u$$

$$\begin{cases} x_1 = y \\ x_2 = y' \end{cases} \implies \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -6x_1 - 5x_2 + 7u \end{cases} \implies y = x_1$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -6 & -5 \end{bmatrix}}_A \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 7 \end{bmatrix}}_B u$$

$$y = \underbrace{[1 \ 0]}_C x + \underbrace{[0]}_D u$$

دستور **ss2tf** : دستور تبدیل فضای حالت به تابع تبدیل

```
>> [num,den]=ss2tf(A,B,C,D)
```

دستور **tf2ss** : دستور تبدیل تابع تبدیل به فضای حالت

```
>>[A,B,c,D]=tf2ss(num,den)
```

```
>> [num,den]=ss2tf([0 1;-6 -5],[0;7],[1 0],[0])
```

```
num = 0 0 7
```

```
den = 1.0000 5.0000 6.0000
```

$$\Longrightarrow G(s) = \frac{7}{s^2 + 5s + 6}$$

مثال:

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

```
Num=[1 0 5] den=[1 3 1 1 6]
```

```
>> [A,B,C,D]=tf2ss([1 0 5],[1 3 1 1 6])
```

```
A =
```

```
-3 -1 -1 -6
```

```
1 0 0 0
```

```
0 1 0 0
```

```
0 0 1 0
```

```
B =
```

```
1
```

```
0
```

```
0
```

```
0
```

```
C =
```

```
0 1 0 5
```

```
D =
```

```
0
```

تکنیک های بهینه سازی با optimization tools:

- در خود نرم افزار روی منوی start کلیک کرده و از آنجا به Tool box بعد more و گزینه optimization را انتخاب کنید.
- اگر از MATLAB 2013 استفاده می کنید از نوار بالای نرم افزار وارد منوی APPS شده و گزینه optimization را انتخاب کنید.

مثال:

$$\text{Min } I_x = x_1^2 + x_2^2$$

$$X_1 \geq 0.5 \rightarrow \text{boundry} \quad \text{محدوده}$$

$$\text{حدس اولیه} \rightarrow X_1=3 \quad \text{و} \quad X_2=1$$

$$-x_1^2 - x_2^2 + 1 \leq 0 \rightarrow \text{linear inequality} \rightarrow \text{شرایط خطی مسئله:}$$

$$-x_1^2 - x_2^2 + 1 \leq 0$$

$$-9x_1^2 - x_2^2 + 9 \leq 0$$

$$-x_1^2 + x_2 \leq 0$$

$$-x_2^2 + x_1 \leq 0$$

شرایط غیر خطی مسئله: \rightarrow nonlinear inequality

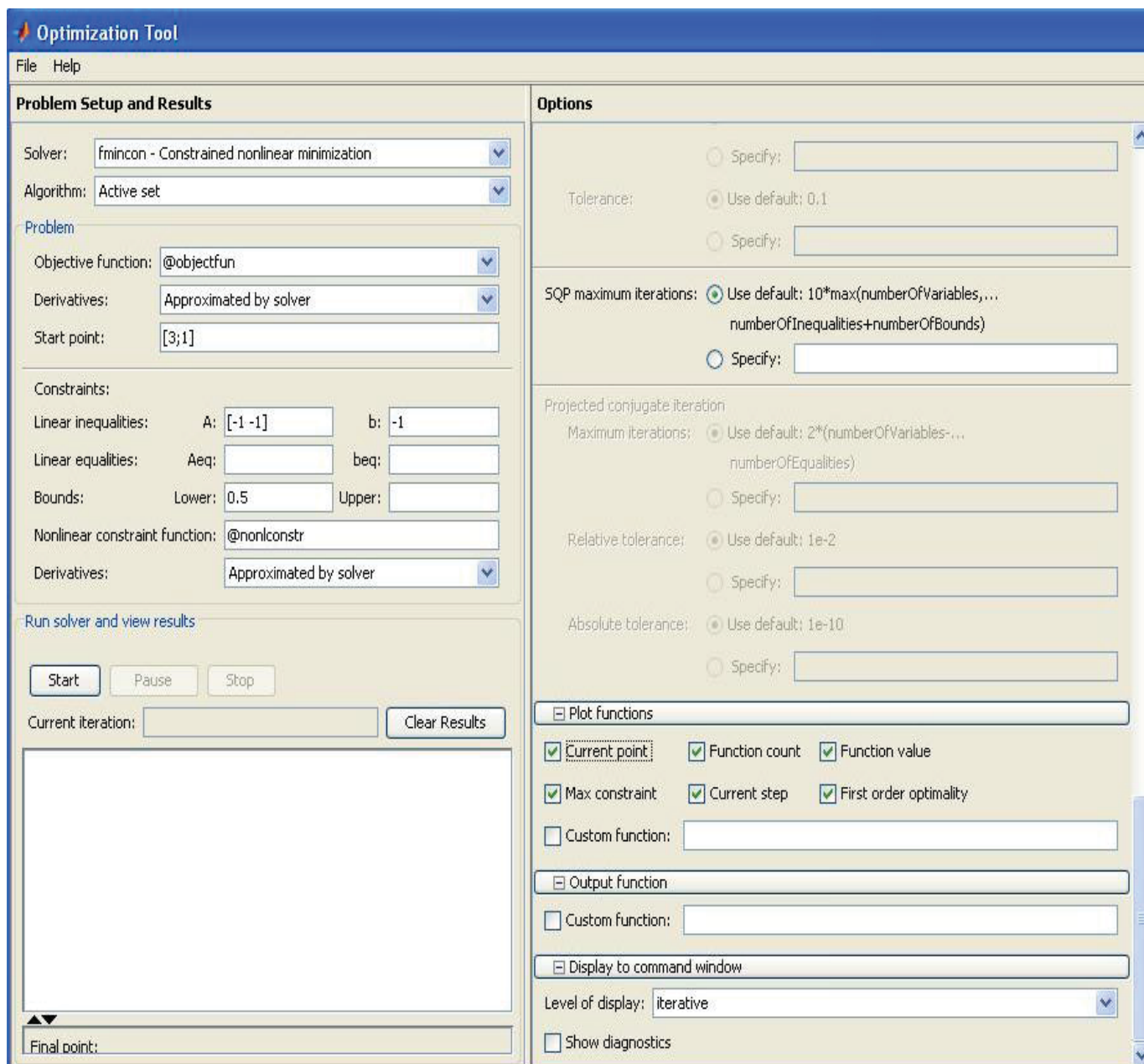
ابتدا تابع زیر را در یک m-file نوشته و ذخیره میکنیم:

```
function f=objectfun(x)
f=x(1)^2+x(2)^2;
```

در یک m-file دیگر برنامه زیر را می نویسیم:

```
function [c,ceq]=nonlconstr(x)
c=[-x(1)^2-x(2)^2+1;-9*x(1)^2-x(2)^2+9;...
-x(1)^2+x(2);-x(2)^2+x(1)];
ceq=[];
```

در پنجره optimization تنظیمات زیر را انجام می دهیم.



همانطور که در شکل بالا مشاهده می کنید تنظیمات به صورت زیر است:

Solver: fmincon

Algorithm:Active set

Object function: @objectfun

Derivatives: Approximated by solver

Start point: [3;1]

A:[-1 -1]

B:-1

Lower: 0.5

Nonlinear constraint function:@nonlconstr

در قسمت Display to command window مقدار level of display را روی iterative (تکرار) قرار دهید.

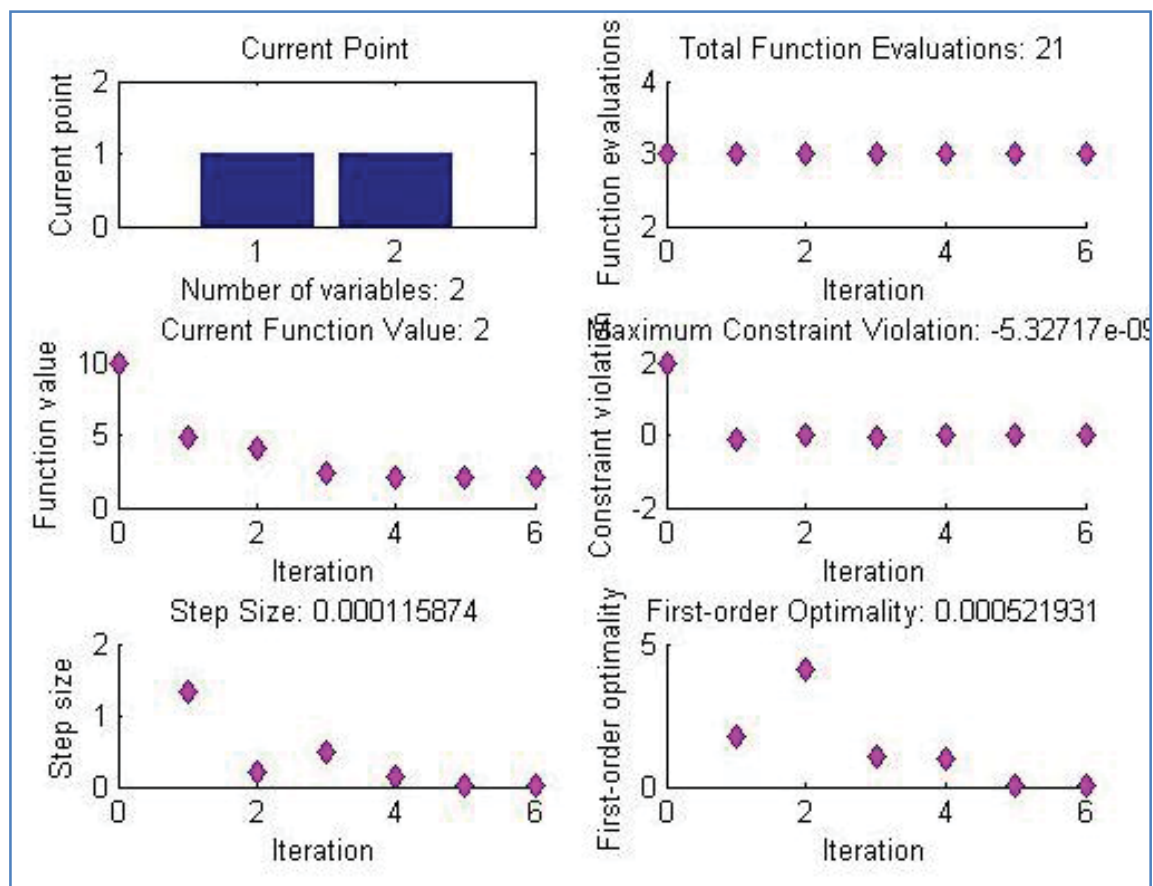
در قسمت plot functions تیک همه ی گزینه ها را فعال کنید.

حال روی گزینه start کلیک میکنیم. نتایج زیر بدست می آید.

Current iteration: 7

Index 1 value 1 نقاط بهینه : 3 , 4

2 1



مثال: مسئله مقابل را با روش حداقل مربعات (least square) حل کنید.

$$\min f=x_1^2+x_2^2+x_3^2$$

$$x_1+2*x_2+4*x_3=7$$

تنظیمات:

Solver: lsqin- constrained linear least squares

Algorithm: large scale

C: eye(3) d:zeros(3)

Aeq:[1 2 4] beq:7

Start point : specify point

Index	value
1	0.333
2	0.667
3	1.333

در ادامه مسائل بهینه سازی optimization می خواهیم مینیمم پیدا کنیم.

برای تابع humps در محدوده 0.3~1 می خواهیم min پیدا کنیم.

```
>> x=fminbnd(@humps,0.3,1)
```

```
x = 0.6370
```

برای دریافت اطلاعات بیشتر:

```
>> x=fminbnd(@humps,0.3,1,optimset('Display','iter'))
```

Func-count	x	f(x)	Procedure
1	0.567376	12.9098	initial
2	0.732624	13.7746	golden
3	0.465248	25.1714	golden
4	0.644416	11.2693	parabolic
5	0.6413	11.2583	parabolic
6	0.637618	11.2529	parabolic

7	0.636985	11.2528	parabolic
8	0.637019	11.2528	parabolic
9	0.637052	11.2528	parabolic

Optimization terminated:

the current x satisfies the termination criteria using OPTIONS.ToIX of 1.000000e-04

x = 0.6370

نکته: اگر ماکزیمم خواستیم کافی است تابع را در یک منفی ضرب کنیم و سپس برای آن min پیدا کنیم.

مثلا برای $\tan(\cos(x))$ در نزدیکی $x=5$ خواهیم داشت:

```
>> [x,fval]=fminbnd(@(x)-tan(cos(x)),3,8)
```

x = 6.2832

fval = -1.5574

توجه کنید که برای min کردن تابع بصورت $y(x)=-f(x)$ عمل کردیم.

استفاده از تابع **fzero** :

```
>> options=optimset('Display','iter');
```

```
>> a=fzero(@humps,-0.2,options)
```

Search for an interval around -0.2 containing a sign change:

Func-count	a	f(a)	b	f(b)	Procedure
1	-0.2	-1.35385	-0.2	-1.35385	initial interval
3	-0.194343	-1.26077	-0.205657	-1.44411	search
5	-0.192	-1.22137	-0.208	-1.4807	search
7	-0.188686	-1.16477	-0.211314	-1.53167	search
9	-0.184	-1.08293	-0.216	-1.60224	search

```

11  -0.177373  -0.963455  -0.222627  -1.69911 search
13   -0.168   -0.786636  -0.232   -1.83055 search
15  -0.154745  -0.51962   -0.245255 -2.00602 search
17   -0.136   -0.104165  -0.264   -2.23521 search
18  -0.10949   0.572246   -0.264   -2.23521 search

```

Search for a zero in the interval [-0.10949, -0.264]:

Func-count	x	f(x)	Procedure
18	-0.10949	0.572246	initial
19	-0.140984	-0.219277	interpolation
20	-0.132259	-0.0154224	interpolation
21	-0.131617	3.40729e-05	interpolation
22	-0.131618	-6.79505e-08	interpolation
23	-0.131618	-2.98428e-13	interpolation
24	-0.131618	8.88178e-16	interpolation
25	-0.131618	8.88178e-16	interpolation

Zero found in the interval [-0.10949, -0.264]

a = -0.1316

در نزدیکی نقطه 0.2- انجام می دهد.

توجه: برای شناخت بیشتر تابع humps یا هر تابع دیگر مثل Hilbert می توان به روش زیر عمل کرد تا source آن بدست آید.

```
>> type humps
```

```
>> type Hilbert
```

Minimize کردن یک تابع همراه با ترسیم توسط نوشتن تابع و استفاده از خط فرمان:

ابتدا تابع مقابل را در یک script ذخیره کنید.

```
1. function f=onehumps(x)
2. r=x(1)^2+x(2)^2;
3. s=exp(-r);
4. f=x(1)*s+r/20
```

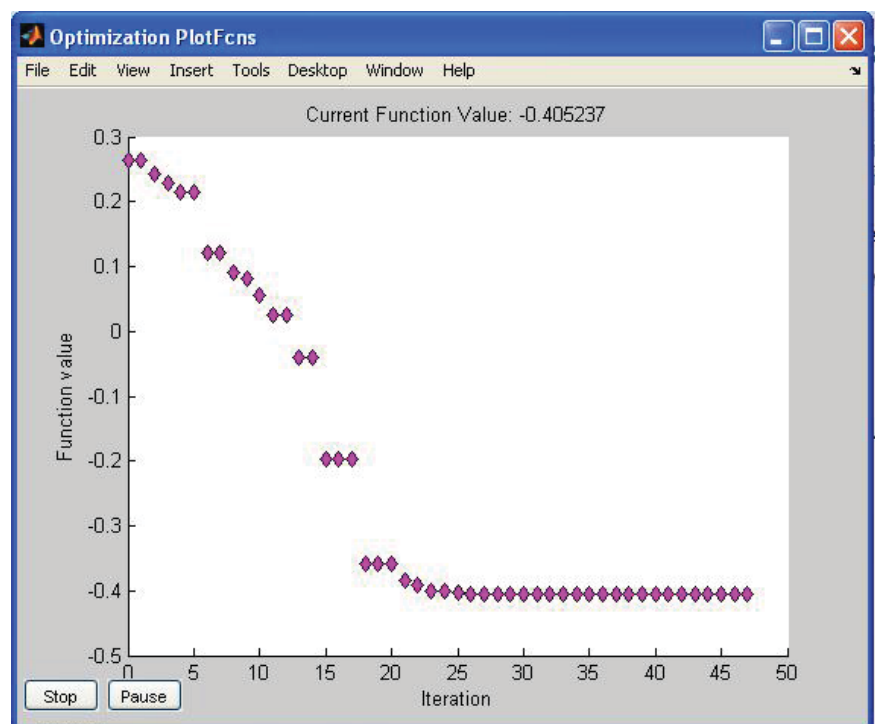
حال در خط فرمان:

```
>> options=optimset('plotFcns',@optimplotfval);
```

```
>> [x ffinal]=fminsearch(@onehump,[2 1],options)
```

```
x =
-0.6691  0.0000

ffinal =
-0.4052
```



حال برای یافتن min در حوالی [-1 1] :

$$\min f(x) = e^{x_1}(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1)$$

```
>> options=optimset('plotFcns',@optimplotfval);
```

```
>> hold on
```

```
>> objfun=@(x)exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)...
```

```
+2*x(2)+1);
```



```
>> [x ffinal]=fminsearch(objfun,[-1 1],options)
```

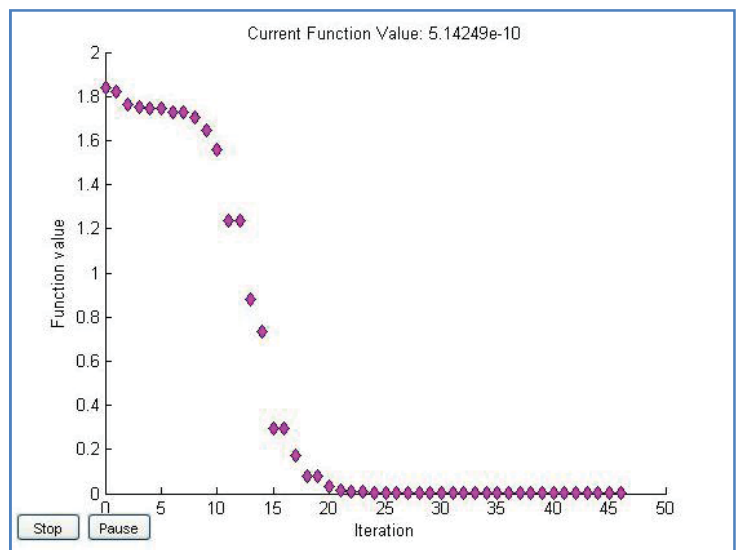
```
x =
```

```
0.5000 -1.0000
```

```
ffinal =
```

```
5.1425e-10
```

```
>> hold off
```



بهینه سازی برای چند تابع همزمان: Multi objective

ابتدا تابع زیر را در یک script نوشته و ذخیره کنید. سپس در tools بهینه سازی تنظیمات زیر را انجام دهید.

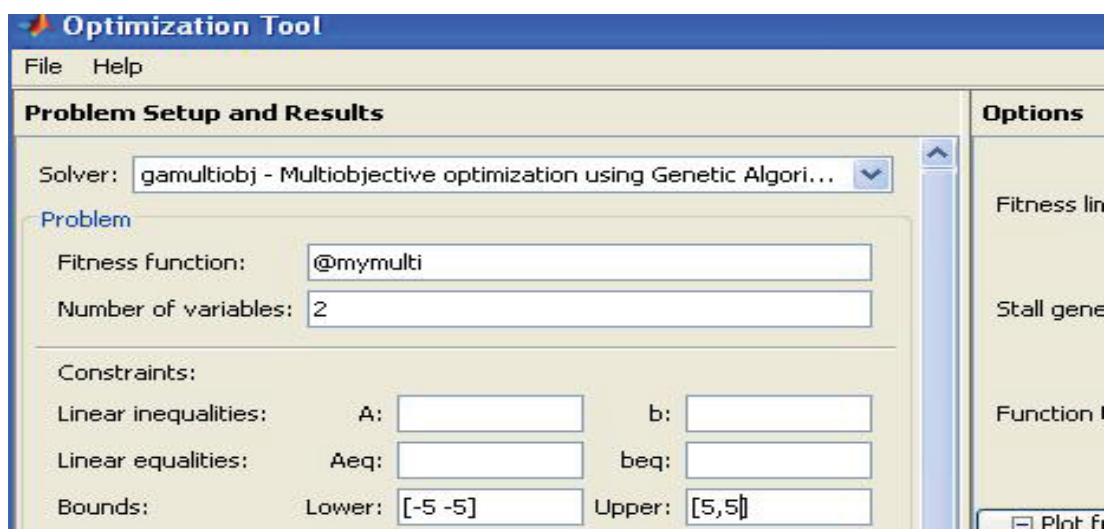
1. `function f=mymulti(x)`
2. `f(1)=x(1)^4-10*x(1)^2+x(1)*x(2)+x(2)^4-(x(1)^2)*(x(2)^2);`
3. `f(2)=x(2)^4-(x(1)^2)*(x(2)^2)+x(1)^4+x(1)*x(2);`

solver: gamultiobj-Multiobjective optimization using Genetic Algorithm

fitness function: @mymulti

number of value: 2

Bounds: Lower: [-5,-5] upper: [5,5]



توجه: روشی که مسئله را حل می کند نوعی الگوریتم ژنتیک است که در ادامه به آن می پردازیم.
برای بهتر دیدن حل مسئله در قسمت راست tools تنظیمات زیر را انجام دهید.

Population → population type : Double vector

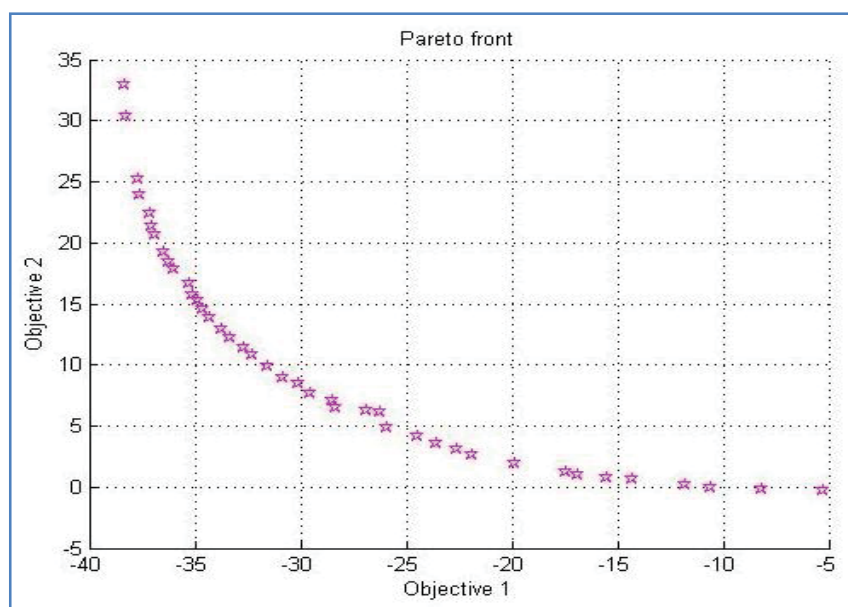
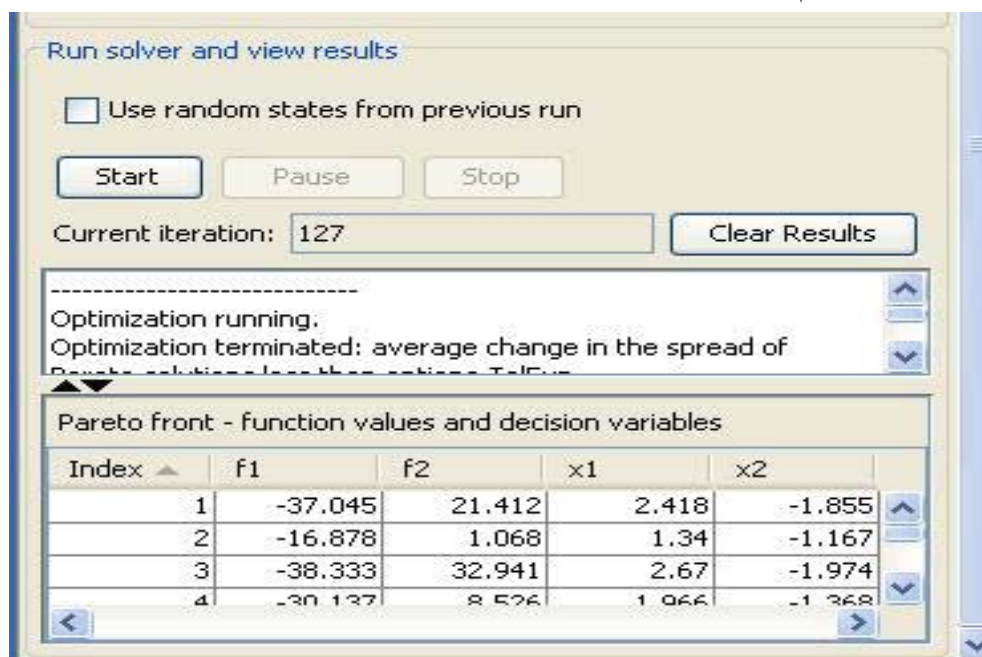
Specify: 60

Distance measure function : use default

Pareto front population fraction: specify : 0.7

Plot function: pareto front

حال روی start کلیک می کنیم.



الگوریتم ژنتیک Genetic algorithm

تابع زیر را می خواهیم min کنیم:

$$\begin{cases} x_1 + x_2 \leq 2 \\ -x_1 + 2x_2 \leq 2 \\ 2x_1 + x_2 \leq 3 \end{cases} \quad x_1, x_2 \geq 0$$

با شرایط محدود کننده بالا ابتدا تابع زیر را که می خواهیم بهینه کنیم در یک script نوشته و ذخیره می کنیم.

1. `function y=lincontest6(x)`
2. `p1=0.5;`
3. `p2=6.0;`
4. `y=p1*x(1)^2+x(2)^2-x(1)*x(2)-2*x(1)-p2*x(2);`

توجه: دقت کنید که تابع lincontest6 از توابع متلب بوده و فقط جهت مشاهده تابع آن را در بالا برایتان نوشتیم.

در خط فرمان داریم:

```
>> A = [1 1; -1 2; 2 1];
```

```
b = [2; 2; 3];
```

```
lb = zeros(2,1);
```

```
[x,fval,exitflag] = ga(@lincontest6,2,A,b,[],[],lb)
```

Optimization terminated: average change in the fitness value less than options.TolFun.

```
x = 0.6681 1.3329
```

```
fval = -8.2244
```

```
exitflag = 1
```

اگر همین مسئله را در tools بخواهیم حل کنیم داریم:

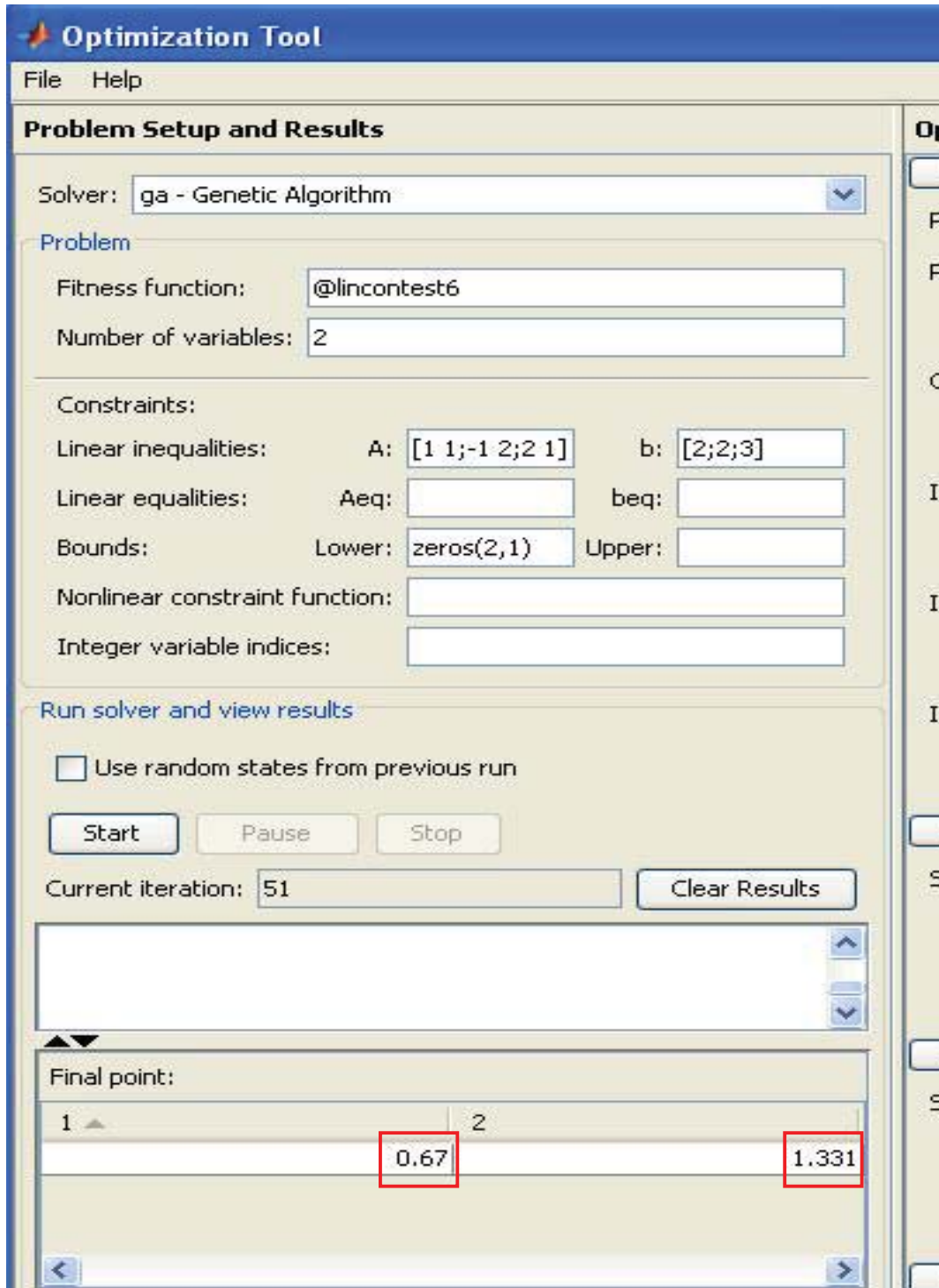
Solver: ga

Fitness fun :@lincontest6

Num of var: 2

Linear inequality: $A=[1 \ 1; -1 \ 2; 2 \ 1]$ $b=[2; 2; 3]$

Bound: lower: $\text{zeros}(2,1)$



The screenshot shows the MATLAB Optimization Tool window. The "Problem Setup and Results" section is active. The solver is set to "ga - Genetic Algorithm". The fitness function is "@lincontest6" and the number of variables is 2. The constraints are defined as follows:

- Linear inequalities: $A = [1 \ 1; -1 \ 2; 2 \ 1]$ and $b = [2; 2; 3]$
- Linear equalities: Aeq: (empty) and beq: (empty)
- Bounds: Lower: $\text{zeros}(2,1)$ and Upper: (empty)
- Nonlinear constraint function: (empty)
- Integer variable indices: (empty)

The "Run solver and view results" section shows the "Use random states from previous run" checkbox is unchecked. The "Start" button is highlighted. The current iteration is 51. The "Clear Results" button is visible. The "Final point:" section shows the following values:

1	2
0.67	1.331

مثال: مثال دیگر برای مینیمم کردن یک تابع با tools

$$\text{Min } Z = 40x_1 + 36x_2$$

$$\left\{ \begin{array}{l} x_1 \leq 8 \\ x_2 \leq 10 \\ 5x_1 + 3x_2 \geq 45 \\ x_1 \geq 0, x_2 \geq 0 \end{array} \right. \implies x_1 + x_2 \leq 18$$

solver: fminimax-Minimax optimization

Derivative:

Start point: [6 1]

constraint:

linear inequalities A:[5 3] b: 45

linear equalities Aeq : [1 1] beq : 15

Bound lower: 8 upper: 10

یادآوری از مطالب قبل:

مثال: تابع زیر را با مجانب هایش رسم کنید.

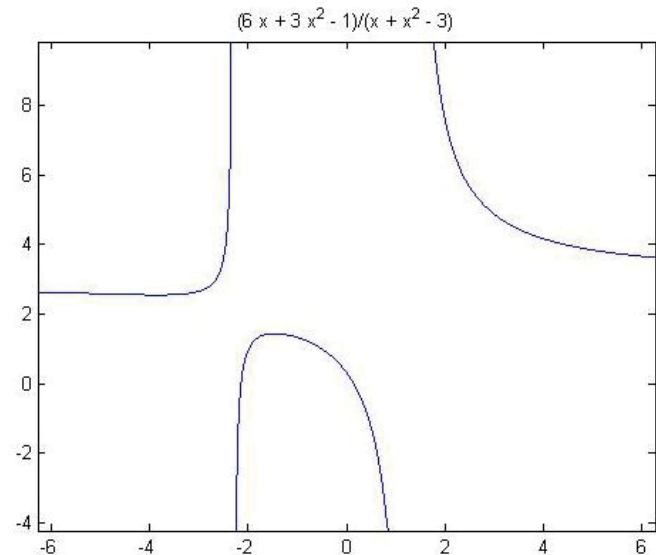
$$f(x) = \frac{3x^2 + 6x - 1}{x^2 + x - 3}$$

```
>> syms x
```

```
>> f=(3*x^2+6*x-1)/(x^2+x-3);
```

```
>> ezplot(f)
```

```
>> hold on
```



```
>> roots=solve('x^2+x-3');
```

```
>> vpa(roots,3)
```

```
ans =
```

```
1.3
```

```
-2.3
```

```
>> x1=1.3;x2=-2.3;
```

```
>> y1=-10:.001:10;
```

```
>> plot(x1,y1,'r',x2,y1,'r')
```

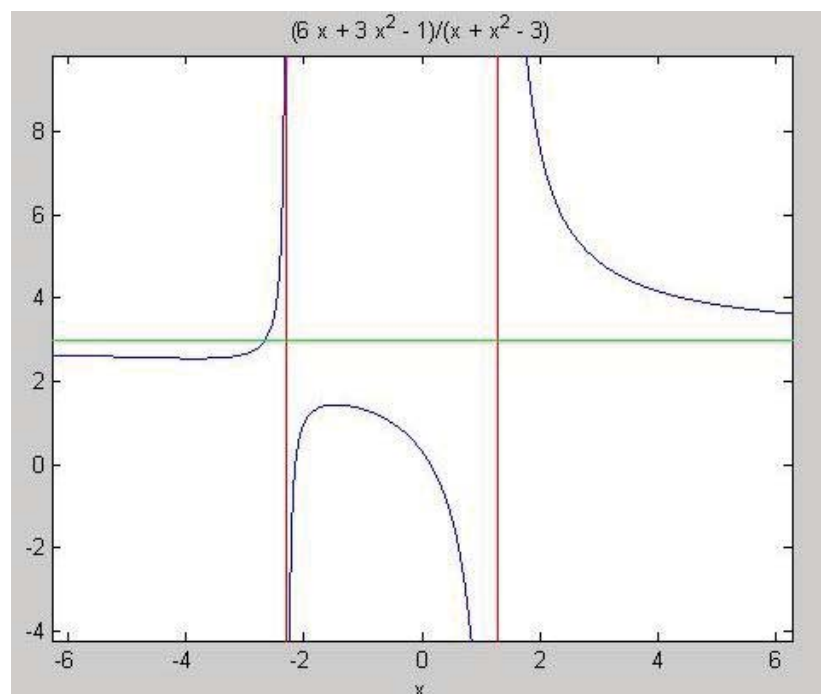
```
>> hold off
```

```
>> x=-10:.001:10;
```

```
>> y=3;
```

```
>> hold on
```

```
>> plot(x,y,'g')
```



مثال: برای داده های زیر میانگین، واریانس، انحراف معیار و کواریانس را بیابید.

```
x=1,3,7,9,15,19,27
```

```
y=5,4,3,2,1,-1,-3
```

```
>> x=[1,3,7,9,15,19,27];
```

```
>> y=[5,4,3,2,1,-1,-3];
```

```
>> mean(x)
```

```
ans = 11.5714
```

```
>> mean(y)
```

```
ans = 1.5714
```

```
>> var(x)
```

```
ans = 86.2857
```

```
>> var(y)
```

```
ans = 7.9524
```

```
>> std(x)
```

```
ans = 9.2890
```

```
>> std(y)
```

```
ans = 2.8200
```

```
>> cov(x,y)
```

```
ans =
```

```
86.2857 -26.0476
```

```
-26.0476 7.9524
```

دستور میانگین

دستور واریانس

دستور انحراف معیار

دستور کواریانس (نسبت دو مجموعه)

برخی توابع آماری:

- تابع توزیع تجمعی: cdf
- تابع چگالی احتمال: pdf
- تابع توزیع دو جمله ای: binomial
- تابع توزیع پواسون: poisson
- تابع توزیع یکنواخت: uniform
- تابع توزیع هندسی: geometric

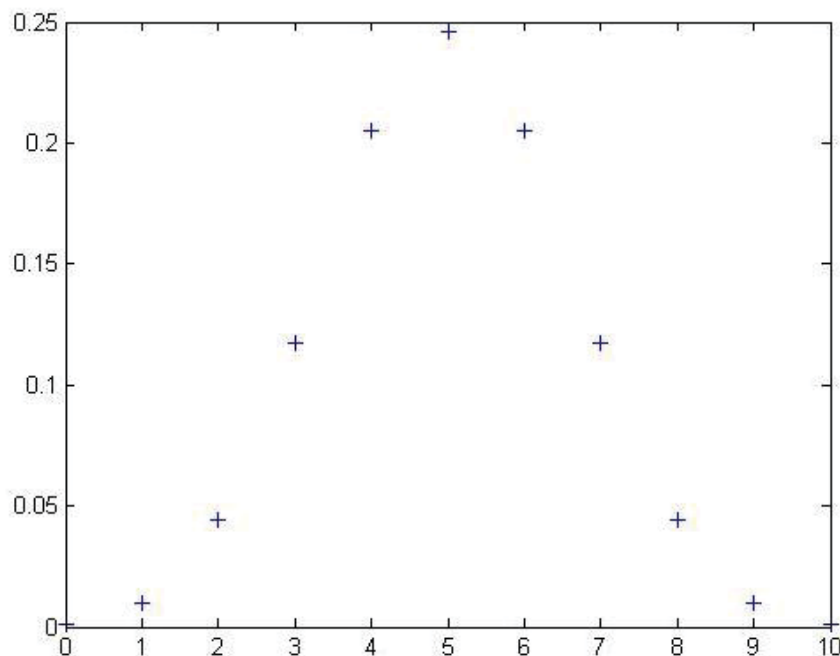
مثال: تابع توزیع دو جمله ای $f\left(\frac{k}{n}, p\right) = \binom{n}{k} p^k (1-p)^{n-k}$

n=10 p=1/2

```
>> x = 0:10;
```

```
>> y = binopdf(x,10,0.5);
```

```
>> plot(x,y,'+')
```



تابع توزیع هندسی Geometric Distribution :

$$f(x|p)=p.q^x \quad x=0,1,\dots$$

```
>> geopdf
```

```
>>geocdf
```

مثال: اگر احتمال خرابی باتری با 5 سال عمر 3/100 باشد، احتمال خراب شدن در 25 روز چقدر است؟

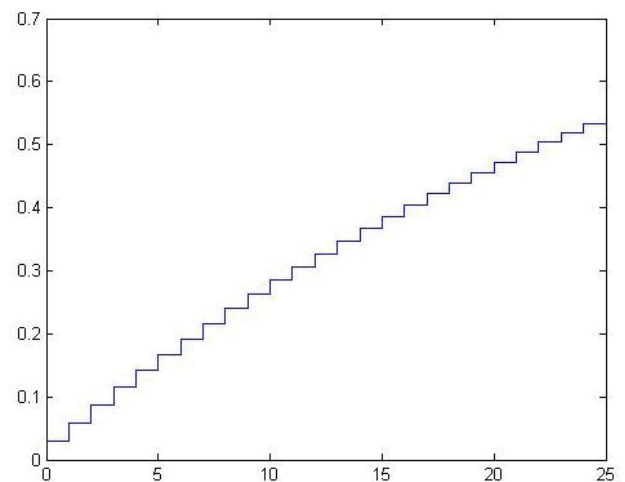
```
>> geocdf(25,0.03)
```

```
ans = 0.5470
```

```
>> x=0:25;
```

```
>> y=geocdf(x,0.03);
```

```
>> stairs(x,y)
```



تابع تولید تصادفی دو جمله ای:

```
>> binornd(n,p)
```

n: تعداد

p: احتمال تولید

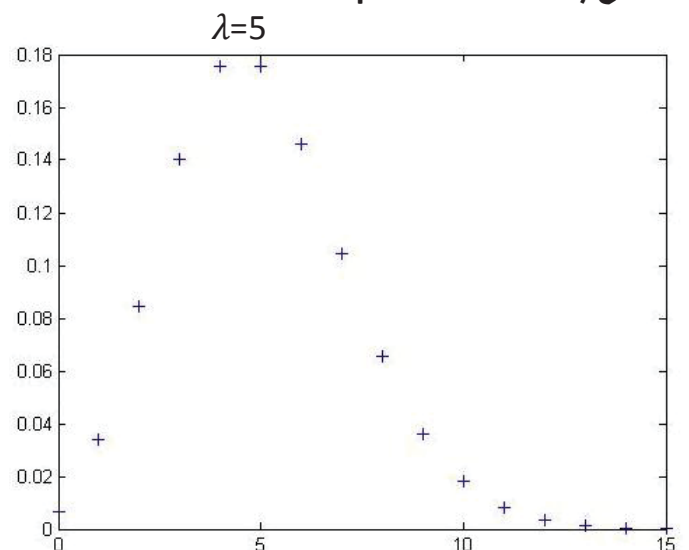
$$y = \frac{\lambda^x - e^{-\lambda}}{x!} \quad x = 0, 1, \dots$$

```
>> x = 0:15;
```

```
>> y = poisspdf(x,5);
```

```
>> plot(x,y,'+')
```

توزیع پواسون: poisson



تبدیل جیبشلف:

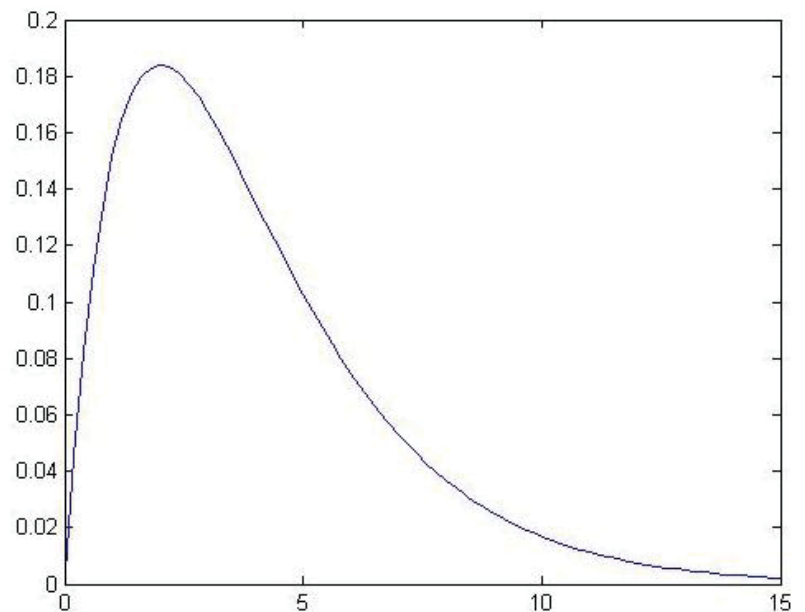
Chi-square $\rightarrow x^2$

$$f(x|v) = \frac{x^{(v-2)/2} e^{-x/2}}{2^{v/2} \Gamma(v/2)}$$

>> x = 0:0.2:15;

>> y = chi2pdf(x,4);

>> plot(x,y)



مثال هایی از pdf و cdf به صورت جدولی:

>> p1=pdf('Normal',-2:2,0,1)

p1 = 0.0540 0.2420 0.3989 0.2420 0.0540

>> p2=pdf('poisson',0:4,1:5)

p2 = 0.3679 0.2707 0.2240 0.1954 0.1755

>> p3=cdf('Normal',-2:2,0,1)

p3 = 0.0228 0.1587 0.5000 0.8413 0.9772

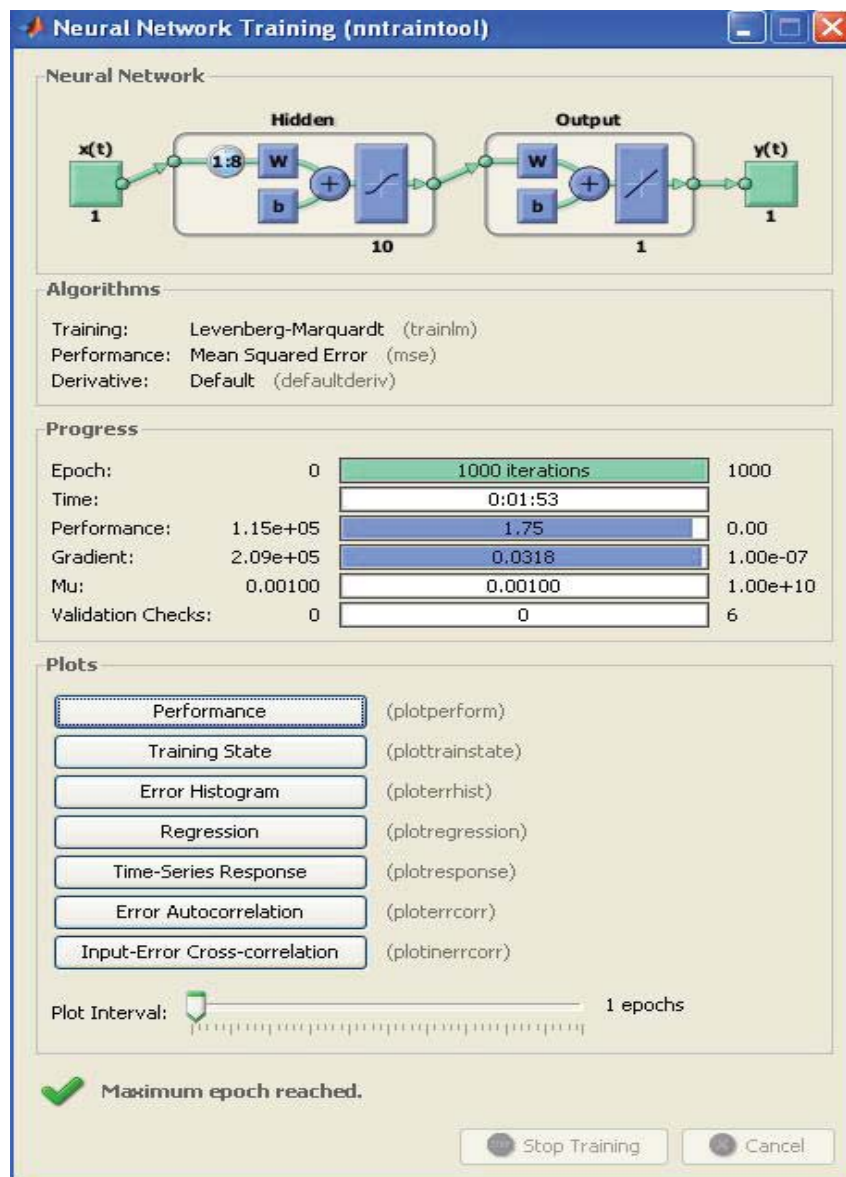
>> p4=cdf('poisson',0:4,1:5)

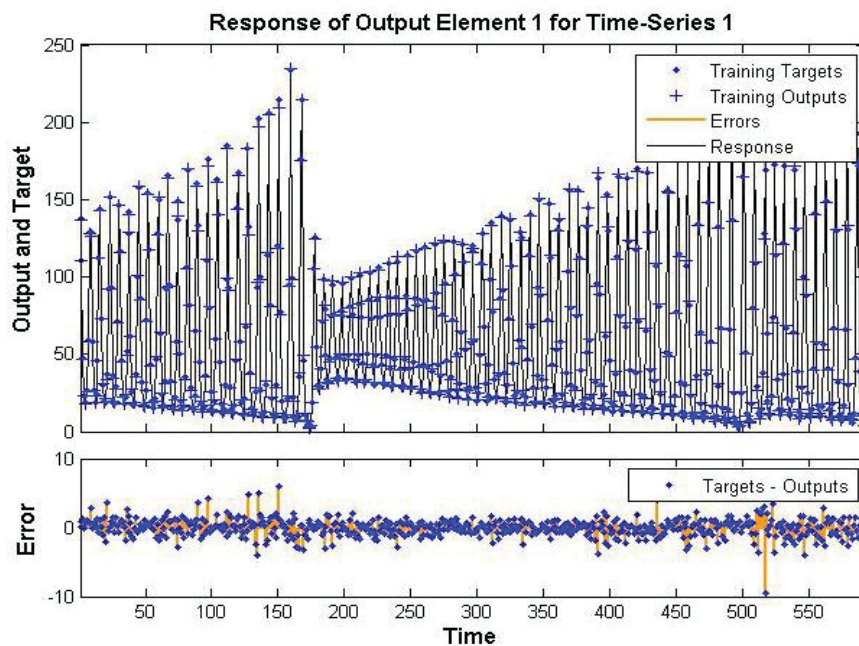
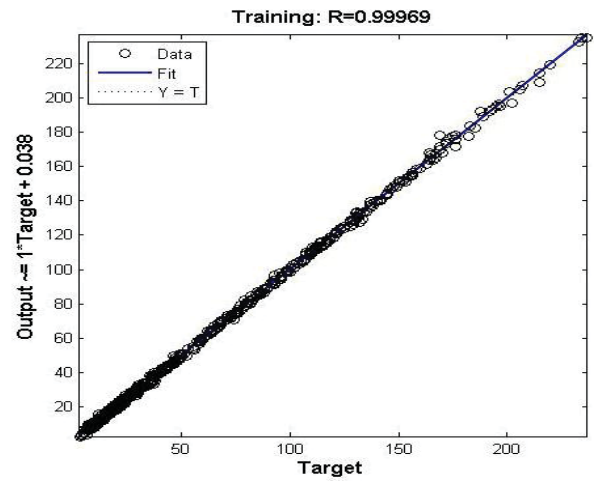
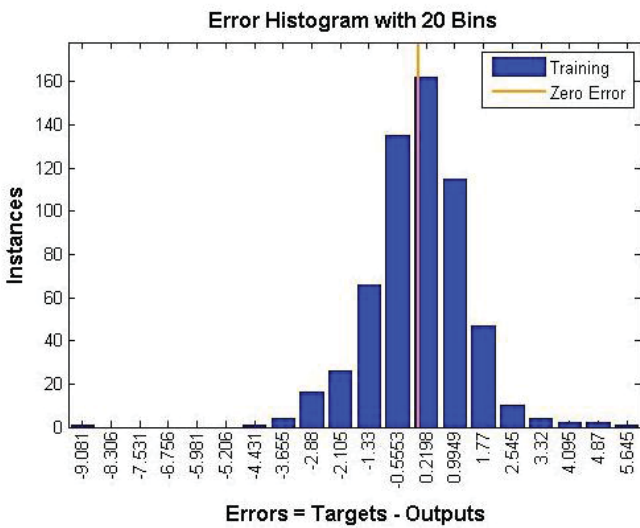
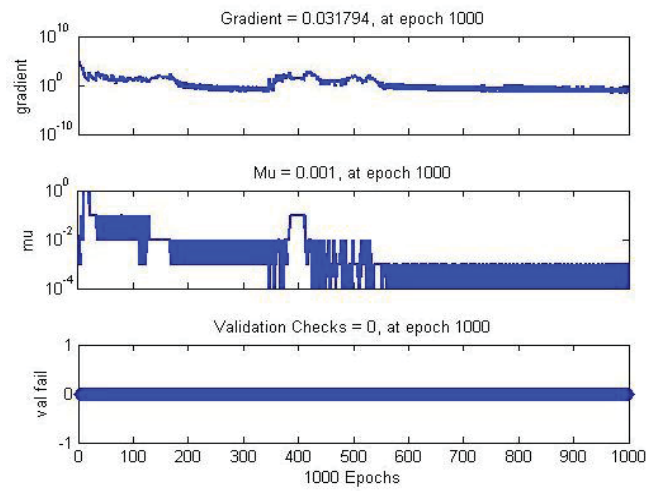
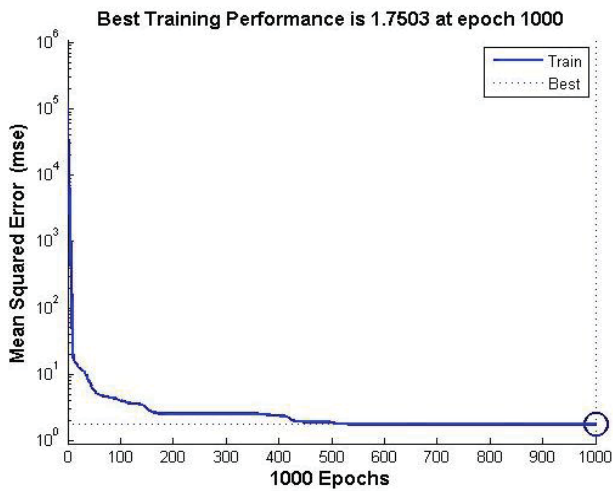
p4 = 0.3679 0.4060 0.4232 0.4335 0.4405

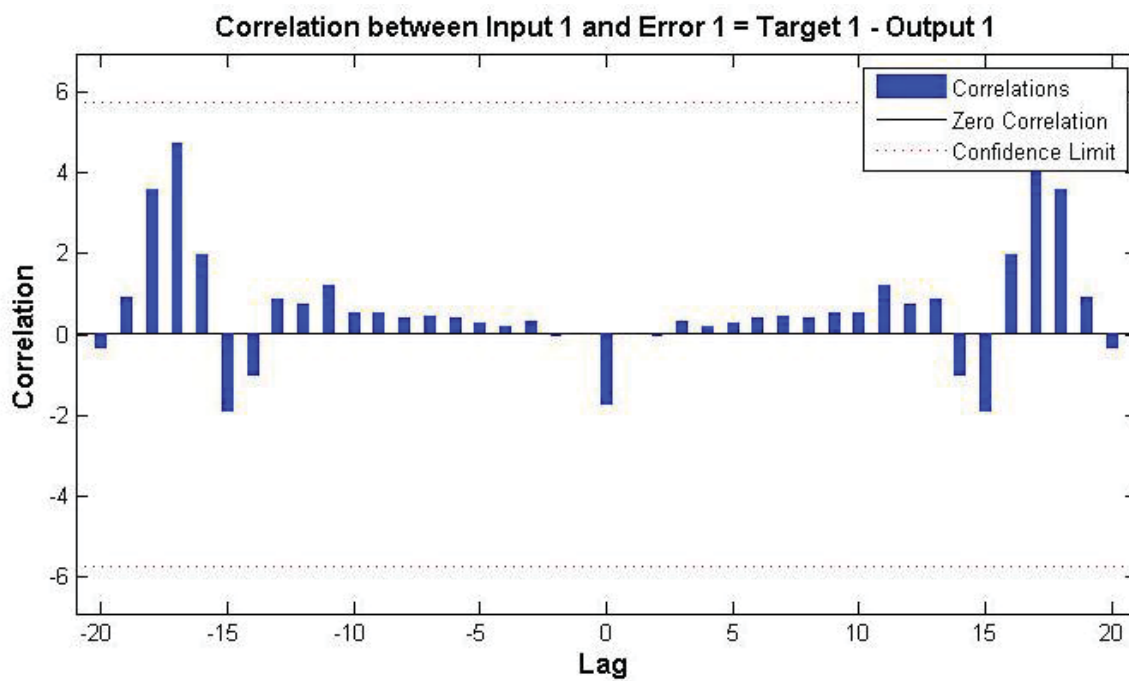
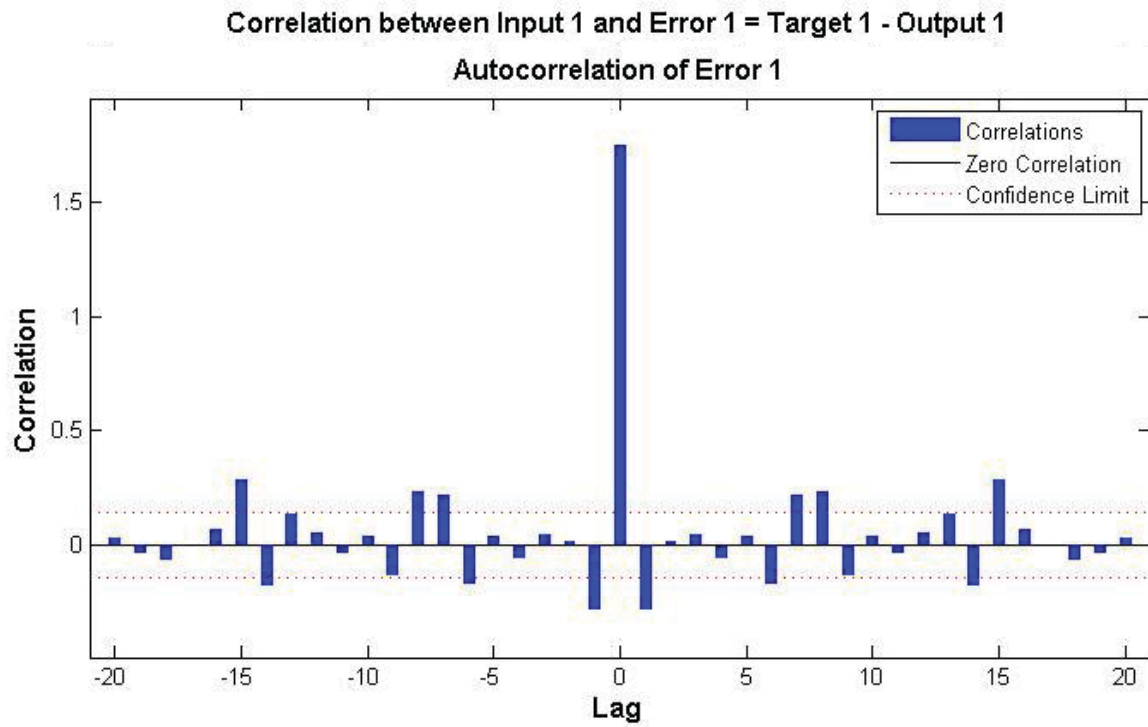


```

>> y=laser_dataset;           => طیف
>> y=y(1:600);
>> ffdnn_net=timedelaynet([1:8],10);   => ایجاد تاخیر
>> ffdnn_net.divideFcn="";           => دوره تناوب
>> p=y(9:end);                   => ورودی
>> t=y(9:end);                   => هدف target
>> r=y(1:8);                     => متغیر ۸ تایی
>> ffdnn_net=train(ffdnn_net,p,t,r); => آموزش شبکه با دو نورون
    
```







```

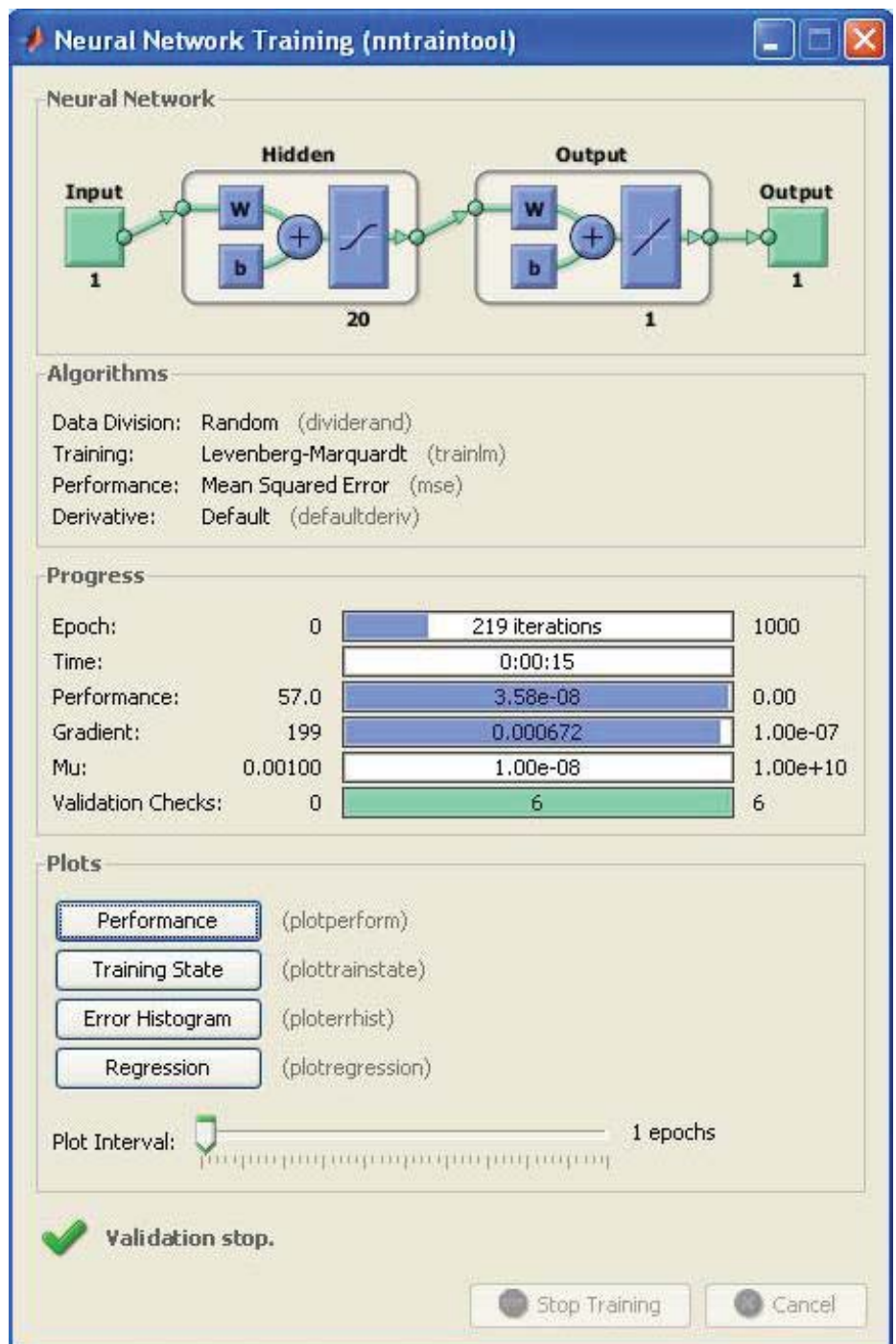
>> nntraintool
>> [x,t]=simplefit_dataset;
>> net=feedforwardnet(20);
>> Net=train(net,x,t);
>> y = net(x);
>> [r,m,b] = regression(t,y)
>> plotregression(t,y)

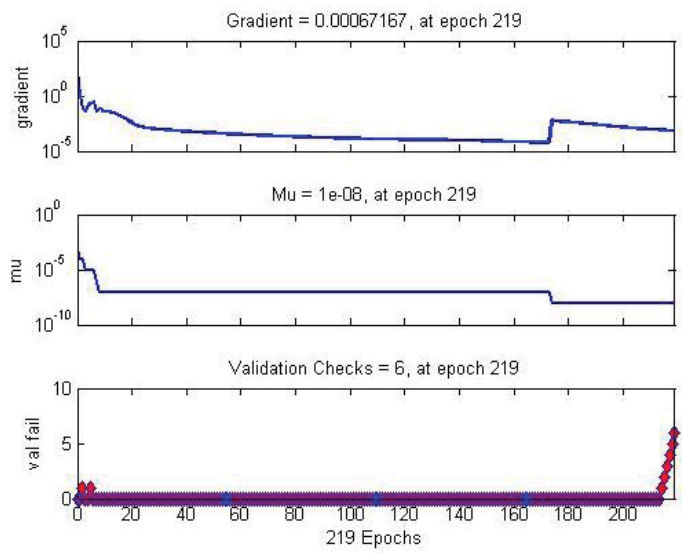
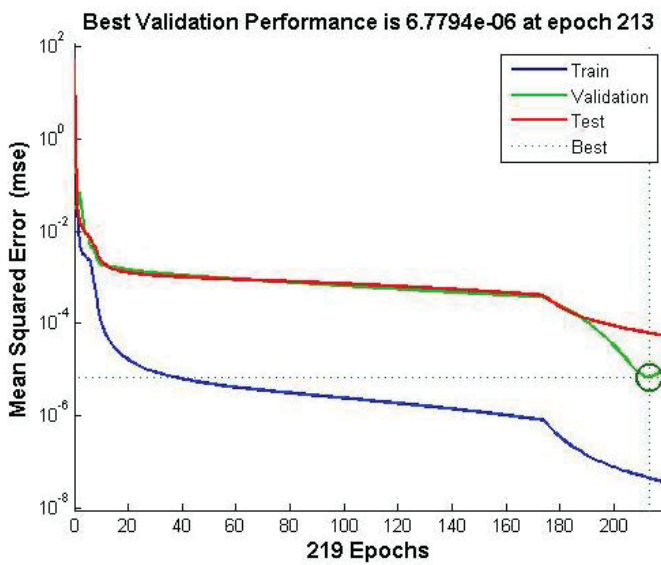
```

r = 0.9990

m = 0.9868

b = 0.1046





Error Histogram with 20 Bins

