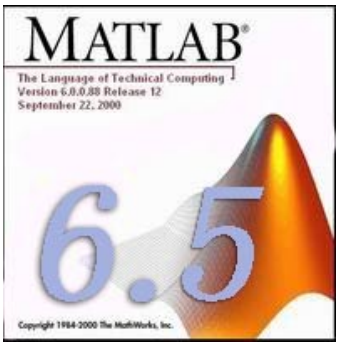
مسلما يكي از قويترين وسايل محاسباتي در رشته هاي مهندسي كامپيوتر ها مي باشند؛ و در انجام محاسبات مهمترين نقش را دارا مي باشند.

حال كه بر حسب نيازمان كامپيوتر را بعنوان ماشين حسابي قدرتمند تعريف كرديم نياز به برنامه اي داريم تا تواناييهايش را بكار ببريم : MATLAB

در اين جا به معرفي اين نرم افزار خواهيم پرداخت, شما را با كاربردها و تواناييهاي آن آشنا خواهيم ساخت به اين اميد كه مورد توجه و عنايت شما عزيزان قرار بگيرد.



MATLAB يک نرم افزار قوي جهت دانشجويان و محققين رشته هاي رياضي و مهندسي است كه اولين نگارشهاي آن در دانشگاه نيومكزيكو و استانفورد در سال ۱۹۷۰ در جهت حل مسائل تئوري ماتريسها, جبر خطي و آناليز عددي بوجود آمد و امروزه صدها هزار كاربر دانشگاهي, آكادميک, صنعتي و ... در زمينه هاي بسيار متنوع مهندسي نظير رياضيات پيشرفته, جبر خطي, مخابرات, مهندسي سيستم و ... با MATLAB بعنوان يكي از اولين محيط هاي محاسباتي و تكنيكي كه قادر به حل مسائل آنهاست, آشنا مي شوند. رياضيات, زبان مشترک بسياري از علوم مهندسي است. ماتريسها, معادلات ديفرانسيل, رشته هاي عددي اطلاعات, ترسيمات و گرافها از لوازم اصلي بكار گرفته در رياضيات و نيز در MATLAB هستند.

MATLAB اكنون يک سيستم مؤثر و زبان برنامه نويسي بسياري از محاسبات علمي و مهندسي است.

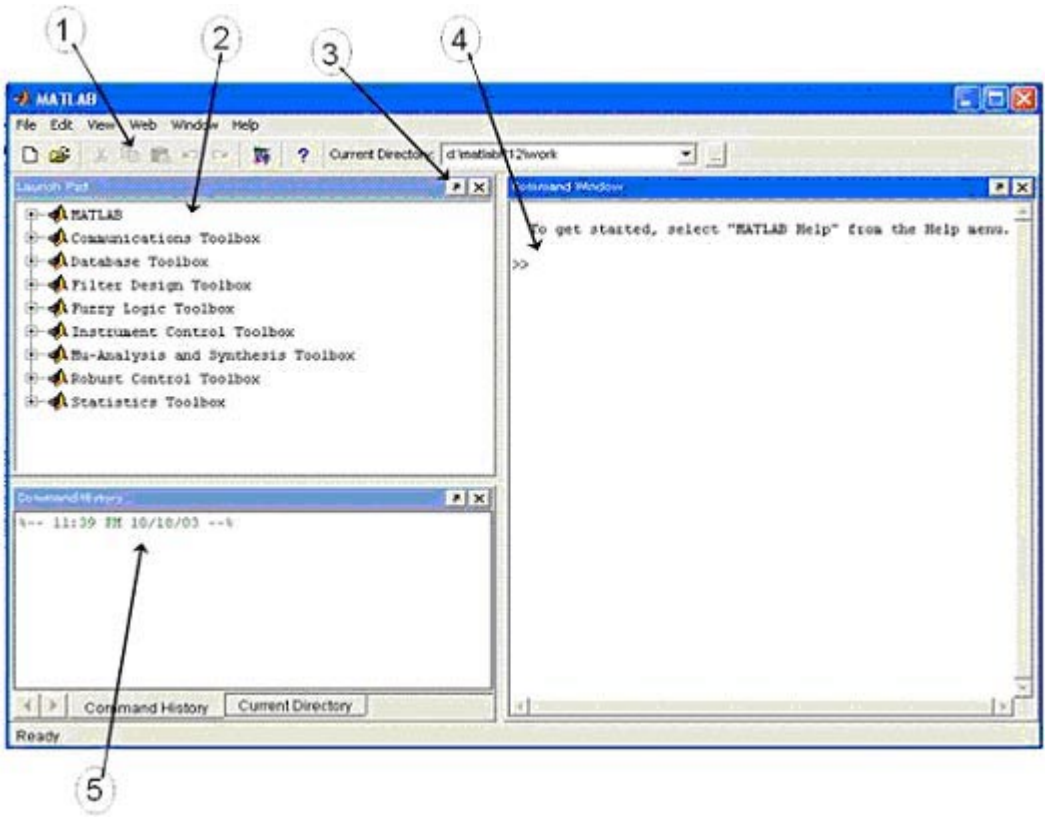
شما مي توانيد بسادگي, توابع و برنامه هاي خاص خودتان را با استفاده از كدها و توابع MATLAB بنويسيد و در صورتيكه تعداد آنها زياد باشد با اختصاص يک زير شاخه براي آنها از مجموعه آنها يک جعبه ابزار درست كنيد.در حقيقت MATLAB يک زبان برنامه نويسي آسان با مشخصات بسيار پيشرفته و ساده تر از زبانهاي كامپيوتري نظير پاسكال و يا C است. اين نرم افزار يک محيط پرقدرت براي تصوير كردن اطلاعات را از طريق قابليتهاي گرافيكي فراهم مي كند.

به دليل وسعت زياد MATLAB فقط به موارد كاربردي تر مي پردازيم .در صورت نياز به اطلاعات بيشتر مي توانيد به help هاي آخر هر قسمت مراجعه كنيد

ramin.samad@yahoo.com

آشنايي با پنجره هاي MATLAB :

هنگامي كه براي اولين بار MATLAB را اجرا مي كنيد پنجره اي مطابق شكل زير مشاهده مي شود:



البته مي توانيد بر اساس نياز خود پنره هاي اضافه را بسته و يا پنجره هايي كه در اين جا ديده نمي شود را باز كرد.

۱) نشان دهنده جعبه ابزار MATLAB مي باشد.كه به ترتيب شامل گزينه هاي New,Open,Cut,Copy,Past مي باشد.

بعد از آن دكمه Help بشكل  قرار دارد و در آخر current directory مشخص كننده Directory جاري MATLAB مي باشد.

۲) شماره ۲ کادر Launch Pad نام دارد. این کادر حاوی مجموعه ای از راهنما, مثالها,ابزارهای جنبی و صفحات Web مرتب شده بصورت موضوعی می باشد. با کلیک بر روی هر کدام از مربعهای کنار اسامی موضوعات به گزینه های آن می توان دسترسی داشت.

۳) روي  كليك كنيد تا كاربرد آن را بفهميد.

٤) محیط کاری اصلی MATLAB می باشد, تمام دستورات و توابع MATLAB در این قسمت درج می شوند

دستورات جلوي علامت<< تايپ مي شود. دستور help باعث باز شدن متن راهنما مي شود. دستور doc همين كار را انجام مي دهد با اين تفاوت كه نتايج را در پنجره help نمايش مي دهد.

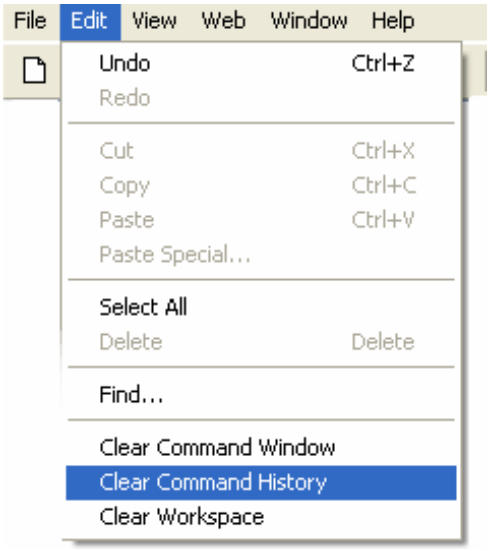
۵) هر بار که شما برنامه MATLAB را باز مي کنيد زمان و تاريخ ورود شما در پنجره command history به رنگ سبز نشان داده مي شود مثلاً: %-- ۱۰/۲۱/۰۳ ۱:۲۷ PM --%

سپس دستوراتي را که به برنامه مي دهيد نيز پشت سر هم در اين پنجره ذخيره مي شوند. جالب اينکه اين دستورات حتي بعد از خارج شدن از برنامه نيز همچنان سر جاي خود باقي مي مانند و حذف نمي شوند. با کليک بر روي اين دستورات ذخيره شده شما مي توانيد آنها را مجدداً اجرا کنيد.

پنجره workspace

يکي ديگر از پنجره هاي MATLAB پنجره workspace مي باشد. در اين پنجره مي توانيم تمام متغيرهايي را که در MATLAB تعريف کرده ايم را ليست وار مشاهده کنيم و حتي در آنها تغييراتي ايجاد کرده و يا آنها را بصورت نمودار مشاهده کنيم.

محتويات تمام اين پنجره ها را مي توان از طريق منوي Edit پاک کرد.



مقدمات كار با MATLAB :

MATLAB اعمال ساده رياضي را به راحتي يك ماشين حساب انجام مي دهد:

```
>> ٢+٤-١
ans =
    ٥
>> ٣+٨/٢
ans =
    ٧
```

راه ديگر انجام محاسبات اين است كه مقادير را در چند متغير ذخيره كرده و روي متغيرها عمليات محاسباتي را انجام دهيم:

```
>> a=٢
a =
    ٢
>> b=٤
b =
    ٤
>> c=١
c =
    ١
>> a+b-c
ans =
    ٥
```

در نام گذاري متغيرها بايد موارد زير را رعايت كرد:

١) MATLAB نسبت به حروف كوچك و بزرگ حساس است.
٢) اسامي متغيرها حداكثر مي تواند ٣١ كركتر باشد.
٣) اسامي متغيرها حتما بايد با حرف شروع شود. (كركتر اول نبايد عدد باشد)
٤) جز كلمات تعريف شده براي MATLAB نباشد. (اين كلمات به رنگ آبي نوشته مي شود، مثل for)
براي شناسايي كلمات كليدي MATLAB ميتوان از دستور iskeyword استفاده كرد.اين تابع در صورتي كه عبارت يك كلمه كليدي باشد مقدار يك و در غير اين صورت مقدار صفر را برمي گرداند.

```
>> iskeyword('for')
ans =
    ١
>> iskeyword('keyword')
ans =
    ٠
>> iskeyword('ans')
ans =
    ٠
>> iskeyword('if')
ans =
    ١
```

٥) تمام دستورات و عبارات كليدي MATLAB با حروف كوچك نوشته مي شود. بنابراين مي توان اسامي آن ها را با حروف بزرگ براي نام گذاري متغيرها به كار برد.

همان طور كه در مثال بالا ديديد MATLAB حاصل a+b-c را در متغير ans ذخيره كرده است. اين متغير به طور پيش فرض براي ذخيره اطلاعات به كار مي رود مگر اين كه كاربر نتايج را در يك متغير ديگر ذخيره كند .

```
>> D=a+b-c
D =
    ٥
```

اگر بخواهيم چند دستور را در يك خط بنويسيم بايد از كاما (،) و سميكالن (؛) استفاده كنيم. سميكالن باعث مي شود محاسبات انجام شود ولي نتايج نمايش داده نشود.

```
>> a=٣ , b=٤ ; c=١;
a =
    ٣
```

براي انجام اعمال ساده رياضي مي توان از عملگرهاي معمول (+ - /يا\ (تقسيم) * (ضرب) ^ (توان)) استفاده كرد.

براي صرفه جويي در وقت با استفاده از كليدهاي up و Down (كليدهاي جهتي) مي توانيد دستوراتي كه قبلا اجرا شده را مرور كنيد.
همچنين ساير كليدهاي ويرايشي(..., Home ,End ,Page Up) وظايف استاندارد خود را دارا مي باشند.

در MATLAB متغيرهاي ويژه اي وجود دارد كه هر يك مقادير خاسي را در خود ذخيره مي كنند:

ans براي ذخيره مقاديري كه كاربر متغيري را براي ذخيره آنها در نظر نگرفته است.
pi مقدار عدد پي (٣,١٤١٦)
eps كوچكترين عدد مثبت بزرگتر از صفر.
inf به عنوان علامت بينهايت
NaN يا nan مقدار غير عددي (نتيجه تمام عملگرها روي NaN ها،NaN است)

بهترين روش براي خواناتر شدن يك برنامه استفاده از جملات توضيحي مي باشد در MATLAB اين جملات بعد از علامت % مي آيد و MATLAB عبارت بعد از % را ويرايش نمي كند.

گاهي اوقات يك فرمان ممكن است آن قدر طولاني باشد كه نتوان آن را در يك خط نمايش داد .براي حل اين مشكل مي توان در آخر خط سه نقطه (. . .) گذاشته و ادامه دستور را در خط بعد تايپ كرد.
براي توقف پردازش برنامه از Ctrl + c استفاده كنيد.

در MATLAB اعداد با فرمت هاي مختلفي به نمايش در مي آيند. از مهمترين آن ها مي توان به Format short اشاره كرد كه فرمت پيش فرض است و اعداد را با دقت ٤ رقم اعشار نمايش مي دهد.همچنين Format bank كه بر اساس سيستم بانكي (دلار و سنت) ايجاد شده اعداد را با دقت ٢ رقم اعشار نمايش مي دهد.

براي گرد كردن اعداد روش هاي مختلفي وجود دارد تمام اين روش ها را مي توان در MATLAB يافت:

fix گرد كردن به طرف صفر
floor گرد كردن به طرف منهي بينهايت
ceil گرد كردن به طرف مثبت بينهايت
round گرد كردن به طرف نزديكترين عدد صحيح

حال به معرفي چند دستور كليدي مي پردازيم:

دستور Date ، تاريخ را نمايش مي دهد

```
>> date
ans =
٠٥-Jul-٢٠٠٤
```

با استفاده از دستور Clear مي توان تمام يا تعدادي از متغيرها را پاك كرد.

```
>> clear a
>> a
??? Undefined function or variable 'a'.

>> b
b =
    ٤
>> clear
>> b
??? Undefined function or variable 'b'.
```

همچنين با استفاده از دستور delete مي توان فايل هاي مورد نظر و موجود در دايركتوري جاري MATLAB را پاك كرد.به عنوان مثال دستور زير تمام p-file هاي موجود در دايركتوري جاري را پاك مي كند.

```
>> delete *.p
```

دستور disp مقادير يك متغير را بدون نمايش نام آن متغير چاپ مي كند.

شايد بخواهيم عملياتي را كه در يك دوره انجام داده ايم ذخيره كرده و از آن پرينت گرفته و يا بعدها از آن استفاده كنيم .براي اين كار از دستور Diary استفاده مي كنيم.
با اجراي دستور diary on ، MATLAB مانند يك دفترچه يادداشت عمل كرده و تمام مطالب موجود در prompt MATLAB در يك فايل ذخيره مي شود تا هنگامي كه diary off اجرا شود.

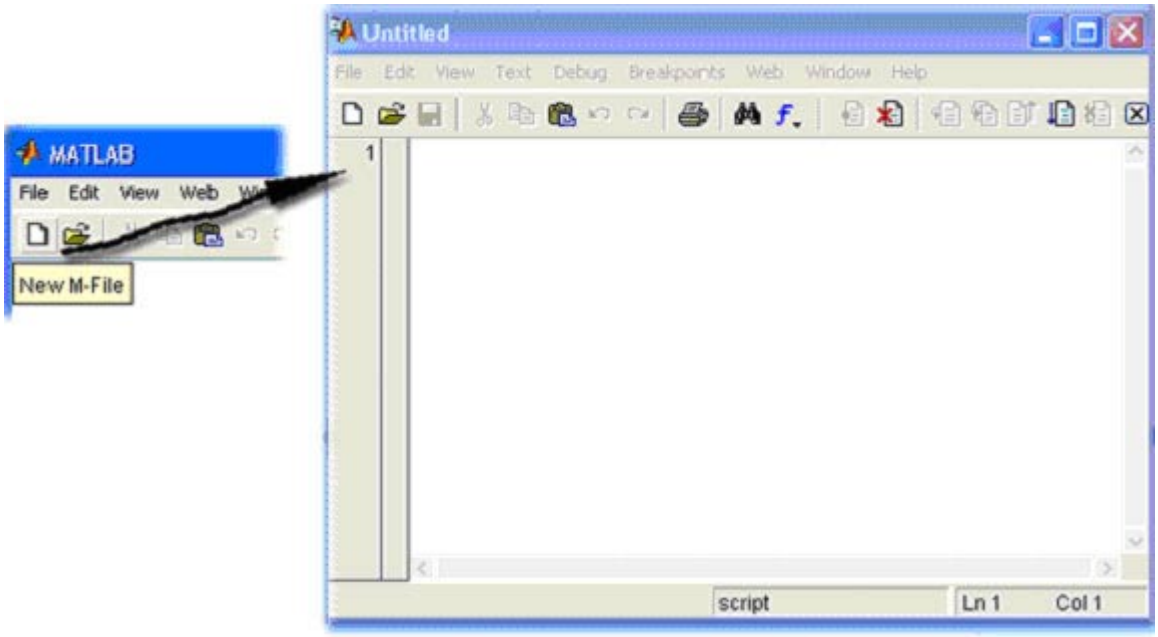
اگر دستور format compact را وارد كرده Enter بزنيد MATLAB خطوطي را كه بصورت خالي بين خروجي قرار مي دهد را حذف مي كند.عكس اين دستور, format loose است كه خطوط خالي حذف شده را بر مي گرداند.

بوسيله دستور format compact مي توانيم داده هاي بيشتري را در پنجره Command window جا دهيم.

همان طور كه قبلا توضيح دادم با استفاده از دستورات Help و Doc مي توان به متن راهنماي يك دستور يا تابع دست پيدا كرد. به اين طريق مي توان با عملكرد آن دستور يا تابع آشنا شده و روش هاي به كاربردن آن را آموخت. همچنين با ساير دستورات مرتبط آشنا مي شو يد.

در هنگام اجراي دستورات تكراري يا هنگام آزمايش كردن مقادير مختلف در يك متغير ممكن است تايپ دستورات خسته كننده باشد.براي حل اين مشكل MATLAB يك راه حل دارد و آن استفاده از M-file مي باشد.
با استفاده از M-fileها مي توانيد دستورات را در يك فايل ذخيره كرده و با باز كردن آن همانند آن كه آن ها را در خط فرمان تايپ كرده باشيد اجرا كنيد.
براي ايجاد يك M-file مي توانيد از گزينه New M-file استفاده كنيد.

ما در اين جا فقط به آموزش موارد مهم و كليدي مي پردازيم و در آخر هر بخش help-file هايي كه مي تواند در تكميل مطلب مفيد باشد معرفي خواهيم كرد.



>>help format
>>doc help
>>doc doc
>>help clear
>>help diary
>>help save
>>help load
>>help date
>>help ops
>>help fix

را يه ها درر MATLAB به سادگي ايجاد مي شوند.ساده ترين و ابتدايي ترين روش تايپ تمام مقادير بين دو [] مي باشد:

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

اما براي توليد آرايه هاي بزرگتر استفاده از روش بالا بسيار وقت گير است.MATLAB چند دستور براي توليد آرايه ها دارد:

linspace(a,b,c)

اين دستور c نقطه با فاصله هاي مساوي در بازه [a,b] را برمي گرداند.

logspace(a,b,c)

اين دستور نيز c نقطه با فواصل لگاريتمي در بازه $10^a,10^b$بر مي گرداند.

a:b:c

مفهوم كلي اين دستور را مي توان به اين صورت بيان كرد؛ از a شروع كن b تا b تا برو جلو تا به c برسي.در اين روش b كه گام حركت نام دارد مي تواند منفي باشد.

با استفاده از دستورات فوق و توابع رياضي مي توان ماتريس هاي متنوعي توليد كرد.

```
>> x=0:.5:2*pi;
>> y=sin(x)
y =
  Columns 1 through 10
         0    0.4794    0.8415    0.9975    0.9093    0.5985    0.1411   -0.3508   -0.7568   -0.9775
  Columns 11 through 13
-0.9589   -0.7055   -0.2794

>> z=tanh(x)
z =
  Columns 1 through 10
         0    0.4621    0.7616    0.9051    0.9640    0.9866    0.9951    0.9982    0.9993    0.9998
  Columns 11 through 13
 0.9999    1.0000    1.0000
```

آرايه ها در MATLAB تنها به آرايه هاي سطري محدود نمي شوند بلكه مي توان آرايه ها ي ستوني نيز معرفي كرد. براي اينكه به MATLAB بفهمانيم كه قصد توليد سطر جديد را داريم از سميكالن(؛) استفاده مي كنيم

```
>> b=[1;3;5;7]
b =
    1
    3
    5
    7
>> b=linspace(1,7,4)'
b =
    1
    3
    5
    7
```

همان طور كه در مثال بالا مشاهده كرديد بااستفاده از عملگر ترانهاده (') يك آرايه سطري را به آرايه ستوني تبديل كرديم.
براي دسترسي به درايه ها از انديس آن استفاده مي شود.به عنوان مثال b(3) سومين درايه آرايه b (يعني 5) را برمي گرداند.
اگر بخواهيم درايه هاي خاصي از يك آرايه را انتخاب كنيم مي توان با استفاده از دو نقطه به آنها دسترسي داشته باشيم.اگر طول آرايه مشخص نباشد مي توان براي دسترسي به آخرين عنصر از end استفاده كرد

```
>> z(end:-3:4)
ans =
   1.0000    0.9998    0.9951    0.9051
```

همچنين مي توان با استفاده از يك آرايه به مقادير يك آرايه ديگر با ترتيب مورد نظر خودمان استفاده كنيم.

```
>> c=[1,5,3,5,1];
>> z(c)
ans =
       0    0.9640    0.7616    0.9640         0
```

همان طور كه مشاهده كرديد با استفاده از آرايه c عناصر اول،پنجم،سوم،پنجم و اول را فراخواني كرديم.

نكته ديگر كه بايد به آن توجه كرد اين است كه انديس يك عدد صحيح مثبت است. اگر كاربر يك عدد غير صحيح و يا منفي را به عنوان انديس وارد كند MATLAB يك پيغام خطا بر مي گرداند.

```
>> b(3.4)
??? Subscript indices must either be real positive integers or logicals.

>> b(-4)
??? Subscript indices must either be real positive integers or logicals.
```

دو آرايه مفروض a و b را مي توان با استفاده از دستورات [a,b] (الحاق سطري)و [a;b] (الحاق ستوني) به يكديگر الحاق كرد و به اين ترتيب آرايه هايي با سطرها و ستون هاي متعدد داشت؛البته در الحاق سطري تعداد سطرها و در الحاق ستوني تعداد ستون هاي دو آرايه بايد برابر باشد.همچنين مي توان تعداد آرايه هاي الحاقي را به طور دلخواه افزايش داد.

```
>> a=[1 2 3];
>> b=[4 5 6];
>> c=[7 8 9];
>> d=[a b],f=[a;b;c]
d =
    1    2    3    4    5    6
f =
    1    2    3
    4    5    6
    7    8    9
```

به اين ترتيب متوجه شديد كا آرايه ها در MATLAB مي توانند داراي سطرها و ستون هاي متعدد باشند.
حال ممكن است اين سوال پيش بيايد كه چگونه مي توان اين آرايه ها را توليد كرد.همان طور كه قبلا ملاحظه نموديد براي معرفي ستون ها از فاصله يا كاما (،) و براي سطر ها از سميكالن (؛) استفاده مي شود.راه ديگر ايجاد سطر اين است كه بعد از معرفي سطر اول با زدن كليد Enter به خط بعد رفته و به معرفي سطر بعد بپردازيد.

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

```
A =
    1    2    3
    6    5    4

>> B= [10 11 12
13 14 15]
B =
   10   11   12
   13   14   15
```

در اين مورد به چند نكته بايد توجه كرد:
تعداد ستون ها در هر سطر بايد برابر باشد در غير اين صورت پيغام خطايي نمايش داده مي شود.
تعداد فواصلي كه براي جدا كردن اعداد به كار مي رود براي MATLAB مهم نيست.يعني MATLAB فضاهاي خالي اضافي را تشخيص داده و حذف مي كند.

محاسبات عددي آرايه ها:

MATLAB عملگرهاي فراواني براي اعمال بر روي آرايه ها دارد.تعدادي از آنها را در مثال هاي زير مشاهده مي كنيد:

```
>> A
A =
    1    2    3
    6    5    4
>> A-2
ans =
   -1    0    1
    4    3    2
>> A*2 + B
ans =
   12   15   18
   25   24   23
>> A + B
ans =
   11   13   15
   19   19   19
```

همان طور كه در بالا ديديد اعمال ساده رياضي را مي توان روي آرايه ها انجام دهيد.A-2 باعث مي شود از تمام درايه هاي آرايه A دو واحد كم شود.همچنين دستور A+B درايه هاي نظير به نظير دو آرايه را جمع مي كند.

ضرب و توان ماتريس ها نيز به سادگي قابل تعريف اند:

```
>> A
A =
    1    2    3
    6    5    4
>> C=[1 2;3 4];
>> C*A
ans =
   13   12   11
   27   26   25
>> C^2
```

```
ans =
    7    10
   15    22
```

حال اگر بخواهيم تك تك درابه ها را به توان n برسانيم يا درايه هاي دو آرايه يا ماتريس را نظير به نظير در هم ضرب يا تقسيم كنيم بايد قبل از اين عملگرها يك نقطه قرار دهيم.به مثال هاي زير توجه كنيد:

```
>> A.*B
ans =
   10    22    36
   78    70    60

>> B.^A
ans =
        10       121      1728
   4826809    537824     50625

>> C.^2
ans =
    1     4
    9    16
```

آرايه هاي استاندارد:

```
>> ones(2,3)
ans =
    1     1     1
    1     1     1
```

ones(n,m) ماتريسي nxm با درايه هاي يك ايجاد مي كند.

```
>> zeros(2,4)
ans =
    0     0     0     0
    0     0     0     0
```

zeros(n,m) ماتريسي nxm با درايه هاي صفر ايجاد مي كند.

```
>> eye(3,4)
ans =
    1     0     0     0
    0     1     0     0
    0     0     1     0
```

اين دستور هم ماتريسيnxm ايجاد مي كند كه درايه هاي روي قطر اصلي آن يك است.اگر اين دستور به صورت eye(n) به كار رود يك ماتريس همانيِ به وجود مي آيد.

```
>> rand(2,3)
ans =
   0.9501    0.6068    0.8913
   0.2311    0.4860    0.7621
```

اين دستور هم يك ماتريس با درايه هاي اتفاقي بين صفر و يك ايجاد مي كند.

```
>> randperm(8)
ans =
    2     4     1     5     8     6     3     7
```

```
>> randperm(10)
ans =
   10    7    1    8    2    5    9    6    4    3
```
اين دستور اعداد 1 تا n را به صورت اتفاقي در يك بردار سطري قرار مي دهد.

```
>> magic(3)
ans =
    8    1    6
    3    5    7
    4    9    2
```
اين دستور ماتريسي كه به ماتريس جادويي معروف است توليد مي كند.ويژگي اين ماتريس اين است مجموع درايه هاي هر سطر، ستون و قطر با هم برابر است.

توجه به اين نكته لازم است كه دو دستور آخر تنها يك ورودي دارند،اما دستورات اول مي توانند داراي دو آرگومان ورودي باشند.در صورتي كه دستورات اول با يك آرگومان به كار بروند يك ماتريس مربعي n x n ايجاد مي شود.

اندازه ماتريس ها:

در MATLAB دستوراتي وجود دارد كه به وسيله آنها مي توان اطلاعاتي در مورد يك ماتريس از قبيل تعداد سطر،ستون و تعداد كل عناصر آن را به دست آورد.

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

>> size(a)
ans =
     2     3
```
اين دستور همان طور كه از اسم آن مشخص است اندازه (تعداد سطر و ستون) ماتريس را برمي گرداند.در دستوراتي مانند دستور فوق كه 2 خروجي يا بيشتر دارند مي توان هر خروجي را در يك متغير ذخيره كرد براي اين كار به صورت زير عمل مي كنيم:

```
>> [s t]= size (a)
s =
    2
t =
    3

>> length(a)
ans =
     3
```
اين دستور بزرگترين مقدار بين سطر و ستون را برمي گرداند.

```
>> numel(a)
ans =
     6
```
اين دستور هم تعداد عناصر(number of elements) ماتريس را برمي گرداند.

فرض كنيد مي خواهيم عناصر قطر اصلي يك ماتريس را به دست آوريم و يا مجموع آنها را به دست آوريم.2 دستور زير به ما كمك خواهد كرد.

```
>> diag(a)
ans =
      1
      2

>> trace(a)
ans =
      3
```

البته دستور diag كاربردهاي ديگري نيز دارد.به عنوان مثال اگر به صورت diag(a,n) به كار رود قطر n ام ماتريس a را مي دهد.n مي تواند مثبت يا منفي باشد.
در صورتي كه به صورت diag(a) به كار رود و a يك بردار باشد ماتريسي ايجاد مي كند كه قطر اصلي آن بردارa و ساير عناصر آن صفر است.

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

>> diag (c)
ans =
      1    0    0
      0    2    0
      0    0    3

>> minfo(ans)
3 rows  3 cols: regular MATLAB matrix
```

دستور آخر هم اطلاعاتي در مورد ماتريس مورد نظر را برمي گرداند.

ايجاد تغيير در ماتريس ها:

MATLAB با اختصاص يك انديس به هر عضو آرايه راه هاي زيادي را براي ايجاد تغيير در درايه هاي ماتريس ها به وجود مي آورد.

```
>> c = ans
c =
   1    0    0
   0    2    0
   0    0    3

>> c(3,3) =8
c =
   1    0    0
   0    2    0
   0    0    8

>> c(9)=4
c =
   1    0    0
   0    2    0
   0    0    4
```

همان طور كه قبلا گفته شد براي دسترسي به عناصر آرايه ها 2 راه وجود دارد.راه اول با 2 آرگومان ورودي كه سطر و ستون را مشخص مي كنند.راه ديگر استفاده از يك آرگومان كه در اين صورت انديس هر درايه به صورت زير (ماتريس s) تعيين مي شود:

```
s=
   1    4    7
   2    5    8
   3    6    9

>> c(4)=5
c =
    1    5    0
    0    2    0
    0    0    4

>> c(:,1)=7
c =
    7    5    0
    7    2    0
    7    0    4
```

مفهوم اين دستور اين است كه تمام سطرها در ستون اول را مساوي 7 قرار بده.

```
>> c([1 end],[1 end])=8
c =
    8    5    8
    7    2    0
    8    0    8
```

اين دستور نيز مثل دستور بالا عمل مي كند يعني در سطر اول و آخر ستون هاي اول و آخر را مساوي 8 قرار مي دهد.

```
>> c(1,:)= []
c =
    7    2    0
    8    0    8
```

اين دستور هم با مساوي قرار دادن سطر اول با آرايه تهي اين سطر را حذف مي كند.

تغييراتي كه در ماتريس ها مي توان ايجاد كرد تنها به تغيير عناصر آن محدود نمي شود بلكه مي توان ابعاد، ترتيب و جاي درايه ها را تغيير داد.

```
>> a=[1 2 3; 4 5 6;7 8 9];
>> flipud(a)
ans =
      7    8    9
      4    5    6
      1    2    3
```

اين دستور ماتريس را از بالا به پايين مي چرخاند . يعني جاي سطرها را نسبت به سطر وسط عوض مي كند.

```
>> b=fliplr(a)
b =
      3    2    1
      6    5    4
```

```
     9     8     7
```
اين دستور هم مانند دستور فوق است با اين تفاوت كه روي ستون ها عمل مي كند.

```
>> rot90(b)
ans =
     1     4     7
     2     5     8
     3     6     9
```
اين دستور ماتريس را 90 درجه در خلاف حركت عقربه هاي ساعت مي چرخاند.البته اگر اين دستور به صورت rot90(a,n) به كار رود n تعداد چرخش را مشخص مي كند.

```
>> triu(ans)
ans =
     1     4     7
     0     5     8
     0     0     9

>> tril(ans)
ans =
     1     0     0
     0     5     0
     0     0     9
```
اين دو دستور هم ماتريس هاي بالا و پايين مثلثي توليد مي كنند.

```
>> c=horzcat(ans,b)
c =
    1    0    0    3    2    1
    0    5    0    6    5    4
    0    0    9    9    8    7

>> c=vertcat(ans,b)
c =
    1    0    0
    0    5    0
    0    0    9
    3    2    1
    6    5    4
    9    8    7
```
اين دو دستور هم الحاق سطري و ستوني را انجام مي دهد.

```
>> reshape(c,2,9)
ans =
     1     0     6     0     0     5     0     9     4
     0     3     9     5     2     8     0     1     7
```
اگر بخواهيم ابعاد يك ماتريس را تغيير بدهيم مي توانيم از اين دستور استفاده كنيم.البته بايد تعداد المنت ها قبل و بعد از تغيير با هم برابر باشد.

```
>> repmat(ans(:,1),2,4)
ans =
     1     1     1     1
     0     0     0     0
     1     1     1     1
     0     0     0     0
```

اين دستور با تكرار يك ماتريس ماتريسي با ابعاد جديد توليد مي كند.به عنوان مثال اگر دستور به اين صورت باشد MATLAB repmat(a,m,n) ماتريس a را به عنوان يك المنت در نظر گرفته و mxn بار تكرار مي كند.

يكي ديگر از دستورات كاربردي MATLAB دستور(cat(n,a,b است.در اين دستور a و b دو ماتريس و n مشخص كننده جهت الحاق مي باشد.مثال هاي زير بهتر مفهوم مورد نظر را مي رساند.مثال سوم نمونه اي از ماتريس هاي 3 بعدي مي باشد.به نحوه نمايش اين ماتريس ها دقت كنيد.

```
>> d=cat(1,a,b)
d =
    1    2    3
    4    5    6
    7    8    9
    3    2    1
    6    5    4
    9    8    7

>> d=cat(2,a,b)
d =
    1    2    3    3    2    1
    4    5    6    6    5    4
    7    8    9    9    8    7

>> d=cat(3,a,b)

d(:,:,1) =
    1    2    3
    4    5    6
    7    8    9
d(:,:,2) =
    3    2    1
    6    5    4
    9    8    7
```

در زير يكي ديگر از دستورات ايجاد تغيير در ماتريس ها را مشاهده مي كنيد.

```
>> a=[1 2;3 4];
>> b=[2 3];
>> kron(a,b)
ans =
     2    3    4    6
     6    9    8   12

>> kron(b,a)
ans =
     2    4    3    6
     6    8    9   12
```

مثال اخير را مي توان به اين صورت نيز نمايش داد:

```
>> [ b(1)*a , b(2)*a ]

ans =
     2    4    3    6
     6    8    9   12
```

مرتب كردن آرايه ها:

يكي از امكانات مفيد MATLAB وجود دستوري براي مرتب كردن آرايه ها مي باشد.در زير ابتدا يك ماتريس با درايه هاي بين 0 و 20 توليد مي كنيم و سپس با استفاده از دستور sort(a,n) كه n مشخص كننده سطر يا ستون مي باشد آن را مرتب مي كنيم.

```
>> a=fix(20*rand(3,4))
a =
     8     8     7     2
    18    17    16     4
    18     1     0     3

>> sort(a,1)
ans =
     8     1     0     2
    18     8     7     3
    18    17    16     4

>> [sor,pos]=sort(a,2)
sor =
     2     7     8     8
     4    16    17    18
     0     1     3    18
pos =
     4     3     1     2
     4     3     2     1
     3     2     4     1
```

همان طور كه مشاهده مي كنيد در صورتي كه از MATLAB دو خروجي بخواهيم آنگاه دو ماتريس برمي گرداند؛ ماتريس اول همان ماتريس مرتب شده و ماتريس دوم انديس مربوط به درايه هاي مرتب شده را نمايش مي دهد.به عبارت ديگر نشان دهنده مكان درايه قبل از مرتب شدن مي باشد.

در اين گونه دستورات در صورتي كه n توسط كاربر مشخص نشود 2 حالت پيش مي آيد.
1) در صورتي كه ماتريس 2 بعدي يا يك بردار ستوني باشد ستون ها مورد بررسي قرار مي گيرند.
2) در صورتي كه ماتريس يك بردار سطري باشد سطرها بررسي مي شوند.

جستجو در آرايه ها:

در MATLAB دستورات زيادي براي جستجو در آرايه ها، پيدا كردن عناصر خاص و ... وجود دارد. در زير به برخي از آنها اشاره مي شود.

```
>> a=[0 1 0;2 0 3;1 3 4];
>> nnz(a)
ans =
     6

>> find(a)
ans =

    2
    3
    4
    6
```

```
    8
    9

>> nonzeros(a)
ans =
    2
    1
    1
    3
    3
    4
```

اين 3 دستور تقريباً با هم در ارتباط اند.دستور اول تعداد درايه هاي غير صفر را برمي گرداند.دستور دوم انديس مربوط به اين درايه ها و دستور سوم خود درايه هاي غير صفر ار برمي گرداند.

```
>> all(a)
ans =
     0     0     0

>> all(a,2)
ans =
     0
     0
     1
```

دستور all(a,n) در صورتي كه تمام درايه هاي سطر يا ستون (بستگي به n دارد) غير صفر باشند مقدار يك و در غير اين صورت صفر را برمي گرداند.

در اين دستور هم چنانچه n مشخص نشود مانند آنچه گفته شد عمل مي شود.

```
>> any(a,2)
ans =
     1
     1
     1
```

اين تابع در صورتي كه يكي از المنت ها غير صفر باشند مقدار يك و در صورتي كه همه صفر باشند مقدار صفر را برمي گرداند.

```
>> isempty(a)
ans =
     0
```

اين تابع در صورتي كه ماتريس يك ماتريس تهي باشد مقدار 1 را بر مي گرداند.

```
>> b=[9 3 0 8];
>> ismember(b,a)
ans =
     0     1     1     0
```

اين دستور در صورتي كه اعضاي ماتريس b (كه مي تواند يك عدد باشد) عضو a نيز باشند مقدار يك و در غير اين صورت مقدار صفر را برمي گرداند.

```
>> unique(a)
ans =
     0
     1
     2
     3
```

4
و اين تابع يك بردار ستوني شامل عناصر غير تكراري a كه به ترتيب صعودي مرتب شده اند را برمي گرداند.

كاربرد در داده هاي آماري:

MATLAB دستورات فراواني براي كار كردن روي داده ها و انجام دادن كارهاي آماري دارد. در زير به مهمترين آنها اشاره مي شود.

```
>> a=fix(20*rand(3,4))
a =
    19     9     9     8
     4    17     0    12
    12    15    16    15
```

يكي از پركاربردترين دستورات 2 دستور زير است كه به ترتيب مينيمم و ماكسيمم هر ستون را به دست مي آورند. در صورتي كه ماتريس يك بردار سطري باشد اين كار را روي سطر انجام مي دهند.

```
>> min(a)
ans =
     4     9     0     8

>> [s,t]=max(a)
s =
    19    17    16    15
t =
     1     2     3     3
```

همان طور كه ملاحظه مي كنيد در صورتي كه مانند دستور sort دو خروجي بخواهيم خروجي دوم مشخص كننده مكان درايه مينيمم يا ماكزيمم خواهد بود.
در صورتي كه اين دستور به صورت max(a,b) (يا min) به كار رود _كه b يك ماتريس با ابعاد ماتريس a يا يك عدد است_ آنگاه خروجي يك ماتريس است با درايه هاي بزرگتر بين a و b.

```
>> b=fix(45*rand(3,4))
b =
    41    18    18    15
    33    42    40    36
     7    41     2     0

>> max(a,b)
ans =
    41    18    18    15
    33    42    40    36
    12    41    16    15
```

همچنين مي توان ميانگين و عضو مياني هر سطر يا ستون را به دست آورد.

```
>> mean(b,2)
ans =
   23.0000
   37.7500
   12.5000

>> median(b,2)
ans =
```

```
   18.0000
   38.0000
    4.5000

>> median(b,1)
ans =
    33    41    18    15
```

به اين نكته دقت كنيد كه تابع median پس از مرتب كردن سطر يا ستون عنصر مياني را برمي گرداند. و در صورتي كه تعداد سطرها يا ستون ها زوج باشد ميانگين 2 عضو وسط را برمي گرداند.

```
>> sum(b)
ans =
    81   101    60    51
```

مجموع هر سطر يا ستون را برمي گرداند.

```
>> cumsum(b)
ans =
    41    18    18    15
    74    60    58    51
    81   101    60    51
```

عملكرد آن مشابه سيگما در رياضي است.يعني مجموع هر درايه با درايه هاي قبل از آن را بر مي گرداند.

```
>> prod(b)
ans =
        9471       30996        1440           0
```

حاصلضرب درايه هاي هر ستون را برمي گرداند.

```
>> cumprod(b)
ans =
          41          18          18          15
        1353         756         720         540
        9471       30996        1440           0
```

اين دستور حاصلضرب هر درايه در درايه هاي ماقبل را بر مي گرداند. از اين دستور مي توان در شبيه سازي فاكتوريل استفاده كرد.

در دستورات مي توان جهت انجام عمليات (سطر يا ستون) را مشخص كرد.در صورتي كه جهت مشخص نشود مانند ديگر دستورها كه در بالا گفته شد عمل مي شود.

يكي ديگر از دستورات كه براي مقايسه دو ماتريس به كار مي رود دستور زير است:

```
>> isequal(a,b)
ans =
     0
```

در صورتي كه دو ماتريس برابر باشند مقدار يك را برمي گرداند.

ماتريس به عنوان ضرايب چند جمله اي:

يك ديگر از كاربردهاي ماتريس ها استفاده از آنها به عنوان ضرايب يك چندجمله اي است.

```
>> x=[2 3 6 10];
>> y=[1 2 3 4];
```

فرض كنيد تعدادي داده آماري داريد و مي خواهيد براي ارتباط دادن آنها با يكديگر تابعي را پيدا كنيد. MATALAB اين كار را به راحتي و به وسيله polyfit(x,y,n) انجام مي دهد.در اينجا x,y داده ها و n مشخص كننده درجه چندجمله اي مورد نظر است.نتيجه اين تابع يك ماتريس است.

```
>> p=polyfit(x,y,3)
p =
    0.0193   -0.3795    2.5298   -2.6964

>> polyval(p,x)
ans =
   1.0000    2.0000    3.0000    4.0000
```

حال فرض كنيد يك چند جمله اي داريم و مي خواهيم مقادير آن را به ازاي مقادير مختلف يك متغير به دست آوريم.براي اين كار از تابع polyval استفاده مي كنيم.

```
>> r=roots(p)
r =
  9.1557 + 4.8026i
  9.1557 - 4.8026i
  1.3040
```

همان طور كه گفته شد مي توان ماتريس ها را به عنوان ضرايب يك چندجمله اي در نظر گرفت.در اين صورت اگرp يك بردار شامل ضرايب چندجمله اي باشد ريشه هاي آن به روش بالا به دست مي آيد.

```
>> p2=poly(r)
p2 =
   1.0000  -19.6154  130.7692 -139.3846
```

اين دستور عكس دستورroots مي باشد.يعني با داشتن ريشه هاي يك چندجمله اي مي توانيد ضرايب آن را به دست آوريد.

در محاسبات رياضي گاهي بايد 2 چندجمله اي را در هم ضرب يا بر هم تقسيم كرد.2 دستور زير به ترتيب اين كارها را انجام مي دهند.

```
>> p1=[2 0 3 -1];
>> p2=[2 4];

>> conv(p1,p2)
ans =
      4     8     6    10    -4

>> deconv(p1,p2)
ans =
      1.0000   -2.0000    5.5000
```

جمع و تفريق چندجمله اي ها نيز به سادگي جمع و تفريق ماتريس ها مي باشد.البته دو ماتريس بايد برابر باشند.

```
>> p3=[3 2 0 1];
>> p1 + p3
ans =
     5     2     3     0
```

ماتريس به عنوان مجموعه:

يك ديگر از كاربردهاي گسترده ماتريس ها در نظر گرفتن آنها به عنوان يك مجموعه مي باشد.

```
>> a=randperm(8)
a =
    4    3    2    6    8    1    5    7

>> b=2:2:8
b =
    2    4    6    8

>> union(a,b)
ans =
    1    2    3    4    5    6    7    8

>> intersect(b,a)
ans =
    2    4    6    8
```

دستور اول اجتماع و دستور دوم اشتراك دو مجموعه را به دست مي آورد.

```
>> setxor(a,b)
ans =
    1    3    5    7
```

اعضايي كه يا فقط در a هستند يا فقط در b.به عبارت ديگر اجتماع منهي اشتراك.(تفاضل متقارن)

```
>> setdiff(a,b)
ans =
    1    3    5    7
```

اعضايي از a كه در b نيستند.

```
>> setdiff(b,a)
ans =
  Empty matrix: 1-by-0
```

اعضايي از b كه در a نيستند.

محاسبات ماتريسي:

در زير به ترتيب معكوس، دترمينان و ترانهاده ماتريس a را محاسبه مي كنيم.

```
>> a=[2 3 -4;0 -4 2;1 -1 5]
a =
    2    3   -4
    0   -4    2
    1   -1    5

>> inv(a)
ans =
    0.3913    0.2391    0.2174
   -0.0435   -0.3043    0.0870
   -0.0870   -0.1087    0.1739

>> det(a)
```

```
ans =
       -46

>> a'
ans =
      2     0     1
      3    -4    -1
     -4     2     5
```

مديريت فايل ها و متغيرها:

MATLAB چند دستور براي آگاهي كاربر از متغيرها و فايل هاي موجود دارد كه در زير به آنها اشاره مي شود.

```
>> what
M-files in the current directory F:\MATLAB6_5\work

bub_dew     cumprod2    exm         mostafa     prod2       size2       sum2
calculate   cumsum2     moadel      name        seri        star
```

اين تابع M-file هاي موجود در دايركتوري جاري را نمايش مي دهد.براي تغيير دايركتوري مي توانيد همانند سيستم عامل dos از دستور cd استفاده كنيد.

```
>> who
Your variables are:
a    ans  b    p    p2   pos  r    sor  x    y

>> whos
  Name      Size                Bytes  Class

  a         3x3                    72  double array
  ans       1x5                    40  double array
  b         1x4                    32  double array
  p         1x4                    32  double array
  p2        1x4                    32  double array
  pos       3x4                    96  double array
  r         3x1                    48  double array (complex)
  sor       3x4                    96  double array
  x         1x4                    32  double array
  y         1x4                    32  double array

Grand total is 61 elements using 512 bytes
```

يكي از دستورات جالب MATLAB كه بيشتر جنبه سرگرمي دارد دستورwhy است.اين دستور را امتحان كنيد.

```
>> why
Some hamster insisted on it.
```

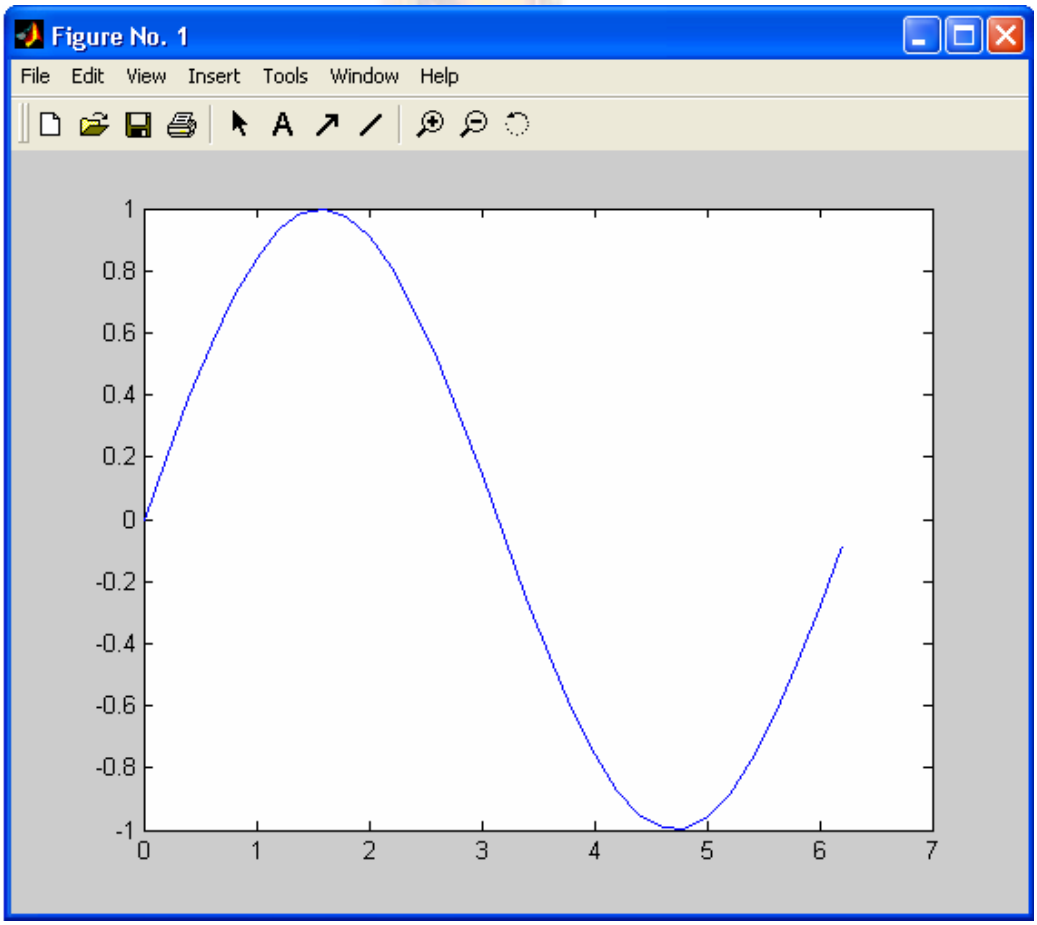
دسته ديگر توابع MATLAB توابع مربوط به رسم نمودار مي باشند.
نمودارها قادر به انتقال اطلاعاتي هستند كه شايد خيلي از جداول و ليست ها قادر به انتقال آن نباشند.به همين دليل اين بخش را به معرفي توابع مربوط به رسم نمودار اختصاص داديم.البته به خاطر گستردگي اين توابع تنها توابع نمودارهاي 2 بعدي را معرفي مي كنيم.شناخت اين توابع كار با ديگر توابع نموداري را راحت مي كند.

تابع plot :

متداول ترين تابع رسم نمودارهاي 2 بعدي اين تابع مي باشد.اين تابع مجموعه اي از آرايه هاي داده ها را بر روي محورهاي مختصات رسم كرده و نقاط تعيين شده را با خطوط مستقيم به هم متصل مي كند.

```
>> x = 0:0.2:2*pi;
>> y = sin(x);
>> plot(x,y)
```

در مثال بالا x محور افقي و y محور عمودي را مي سازند.(در تابع آرگومان اول محور افقي و آرگومان دوم محور عمودي را مشخص مي كند).تابع plot پنجره گرافيكي figure را باز مي كند، سپس اندازه محورهاي مختصات را مطابق داده ها تنظيم مي كند.بعد از رسم نقاط آنها را با خطوط راست به يكديگر متصل مي كند. در زير نتيجه دستورات بالا و پنجره figure را مشاهده مي كنيد.

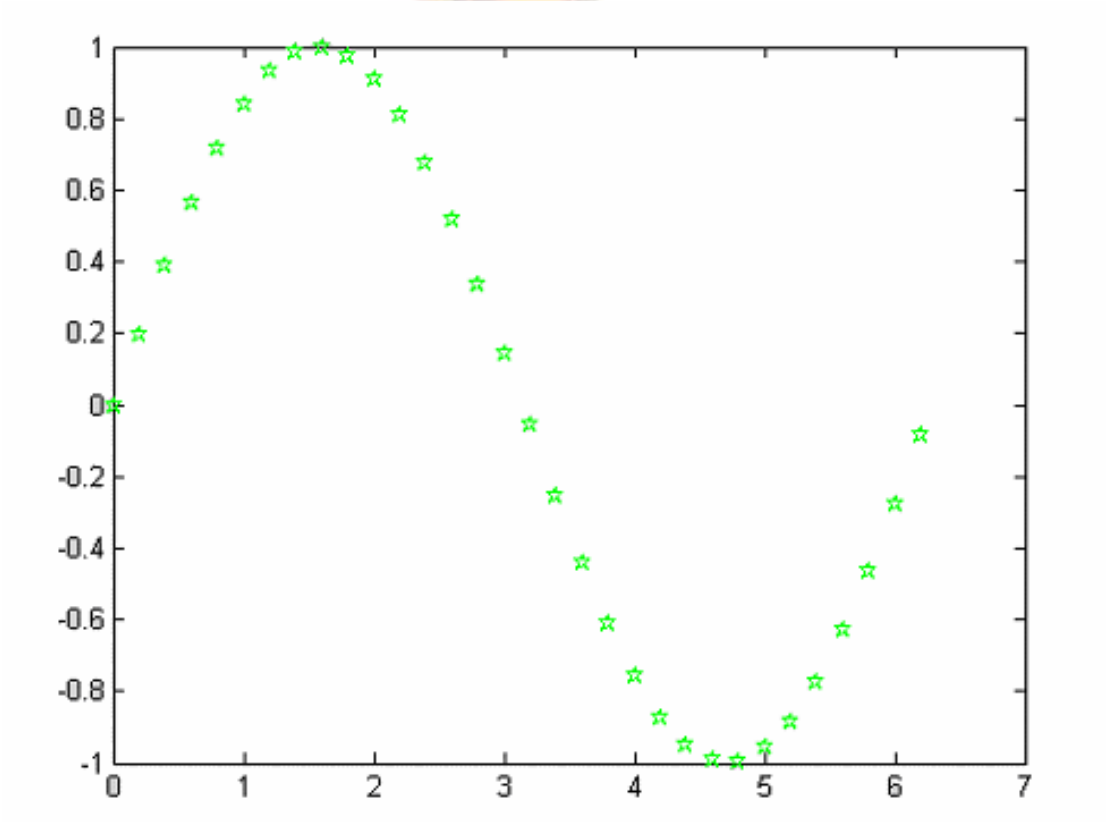


تابع plot را مي توان به همراه آرگومان سومي نيز به كار برد.اين آرگومان كه پس از x و y مي آيد يك رشته كركتري است كه مشخص كننده نوع خطوط و رنگ آنها مي باشد.اين رشته شامل يك يا چند كركتر از جدول زير است.

b	blue	.	point	-	solid
g	green	o	circle	:	dotted
r	red	x	x-mark	-.	dashdot
c	cyan	+	plus	--	dashed
m	magenta	*	star		
y	yellow	s	square		
k	black	d	diamond		
		v	triangle (down)		
		^	triangle (up)		
		<	triangle (left)		
		>	triangle (right)		
		h	hexagram		
		p	pentagram		

به مثال زير توجه كنيد:

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



جدول فوق به سه ستون تقسيم مي شود.ستون اول (از چپ) رنگ ،دوم نقاط و سومين ستون نوع خط را مشخص مي كند.تا هنگامي كه نوع خط را مشخص نكنيد نقاط به هم متصل نمي شوند.در استفاده از كركترهاي اين جدول از هر رديف تنها يك كركتر را انتخاب كنيد در غير اين صورت پيغام خطايي نمايش داده مي شود.به اين صورت رشته مورد نظر حداكثر داراي 3 كركتر است.

```
>> plot(x,y,':-')
```

??? Error using ==> plot
Error in color/linetype argument.

این کرکترها و ترکیب آنها با یکدیگر را امتحان کنید تا با آنها بیشتر آشنا شوید.

در صورتي كه ترتيب آرگومان ها را تغيير دهيد نمودار هم 90 درجه دوران پيدا مي كند.يعني نمدار y برجسب x به نمودار x بر حسب y تبديل مي شود.

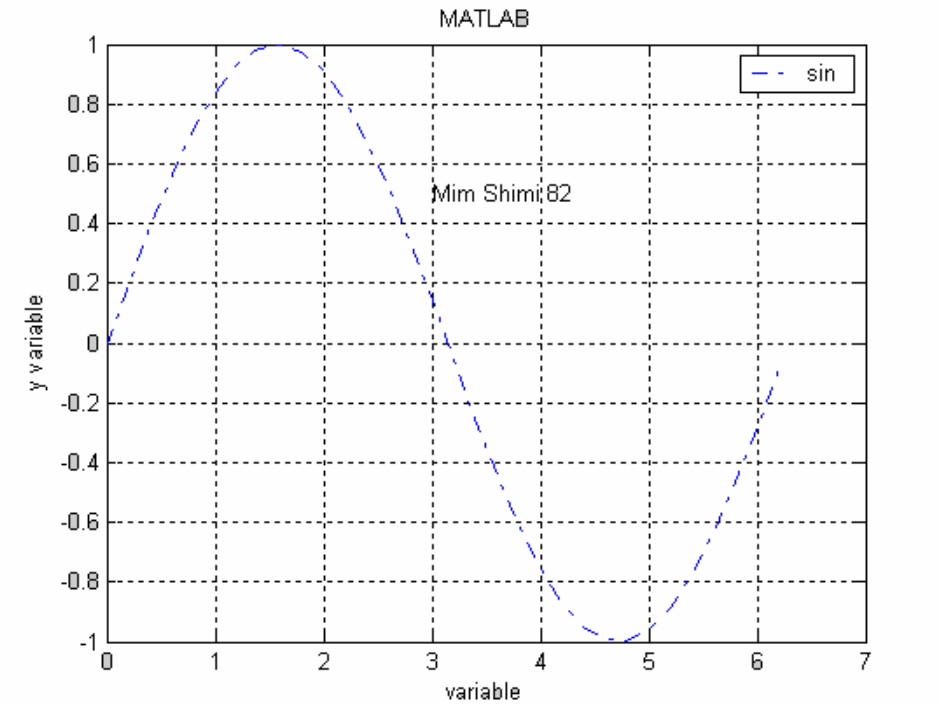
بر چسب ها، تنظيمات و ... :

دستورات xlabel و ylabel برچسب محورها را مشخص مي كنند.دستور title عنوان را بالاي نمودار قرار مي دهد.

راهنماي نمودار نيز در صورت رسم چند نمودار روي يك پنجره مي تواند مفيد باشد.دستور legend اين كار را انجام مي دهد.اين دستور در قسمت بعد بيشتر شرح داده مي شود.
دستور grid on خطوط شبكه اي را روي نمودار فعال مي كند وgrid off آنها را حذف مي كند.
اگر بخواهيم متني را روي نمودار قرار دهيم از تابع text استفاده مي كنيم.
clf پنجره figure را پاك مي كند.clc نيز پنجره command window را پاك مي كند.

حال در مثال زير روش استفاده از اين دستورات را مشاهده مي كنيد.

```
>> clf
>> plot (x,y,'-.')
>> xlabel (' variable ')
>> ylabel (' y variable ')
>> title (' MATLAB ')
>> legend (' sin ')
>> grid
>> text (3,.5,' Mim Shimi 82 ')
```



دستور grid در صورتي كه به تنهايي به كار رود ،در صورتي كه شبكه ها روشن باشد آن را خاموش و در صورت خاموش بودن آن را روشن مي كند.

در دستور text آرگومان اول و دوم مختصات ابتدا متن و آرگومان سوم متن مورد نظر است.در صورتي كه مختصات متن را ندانيم مي توانيم از دستور gtext استفده كنيم. با اجرا اين دستور خطوط متقاطعي روي صفحه نمايش داده مي شود و مكان مورد نظر با كليك ماوس تعيين مي شود.
حالت كلي اين دستور به اين شكل است كه TEXT متن مورد نظر است:

gtext ('TEXT')

همان طور كه مشاهده فرموديد تقريبا در تمام دستورات فوق از رشته هاي كركتري استفاده مي شود. MATLAB علاوه بر رشته هاي معمولي امكاناتي دارد تا بتوان متن هايي شامل كركترهاي ويژه (مثل ∂ ∞ ≠) و در چند خط، همچنين عبارات توان دار و انديس دار را به نمودارها اضافه كرد.

اضافه كردن كركترهاي ويژه به راحتي انجام مي گيرد.با قرار دادن يكي از موارد زير در رشته ميتوان آن را به متن اضافه كرد.

Character Sequence	Symbol	Character Sequence	Symbol	Character Sequence	Symbol
\alpha	α	\upsilon	υ	\sim	~
\beta	β	\phi	φ	\leq	≤
\gamma	γ	\chi	χ	\infty	∞
\delta	δ	\psi	ψ	\clubsuit	♣
\epsilon	ε	\omega	ω	\diamondsuit	♦
\zeta	ζ	\Gamma	Γ	\heartsuit	♥
\eta	η	\Delta	Δ	\spadesuit	♠
\theta	θ	\Theta	Θ	\leftrightarrow	↔
\vartheta	ϑ	\Lambda	Λ	\leftarrow	←
\iota	ι	\Xi	Ξ	\uparrow	↑
\kappa	κ	\Pi	Π	\rightarrow	→
\lambda	λ	\Sigma	Σ	\downarrow	↓
\mu	μ	\Upsilon	Υ	\circ	º
\nu	ν	\Phi	Φ	\pm	±
\xi	ξ	\Psi	Ψ	\geq	≥
\pi	π	\Omega	Ω	\propto	∝
\rho	ρ	\forall	∀	\partial	∂
\sigma	σ	\exists	∃	\bullet	•
\varsigma	ς	\ni	∋	\div	÷

\tau	τ	\cong	≅	\neq	≠
\equiv	≡	\approx	≈	\aleph	ℵ
\Im	ℑ	\Re	ℜ	\wp	℘
\otimes	⊗	\oplus	⊕	\oslash	⊘
\cap	∩	\cup	∪	\supseteq	⊇
\supset	⊃	\subseteq	⊆	\subset	⊂
\int	∫	\in	∈	\o	o
\rfloor	⌋	\lceil	⌈	\nabla	∇
\lfloor	⌊	\cdot	·	\ldots	...
\perp	⊥	\neg	¬	\prime	′
\wedge	∧	\times	x	\0	∅
\rceil	⌉	\surd	√	\mid	|
\vee	∨	\varpi	ϖ	\copyright	©
\langle	⟨	\rangle	⟩		

براي ايجاد متن هاي چند خطي مي توانيد از آرايه هاي رشته اي به صورت زير استفاده كنيد.

text ({'LINE1' , 'LINE2' })

براي قرار دادن توان بر روي يك عبارت از علامت توان بعد از عبارت استفاده مي شود.در صورتي كه عبارتي كه در توان قرار مي گيرد بيش از يك كركتر باشد آن را بين دو { } قرار مي دهيم. و براي ايجاد انديس از كركتر '_' استفاده مي كنيم.

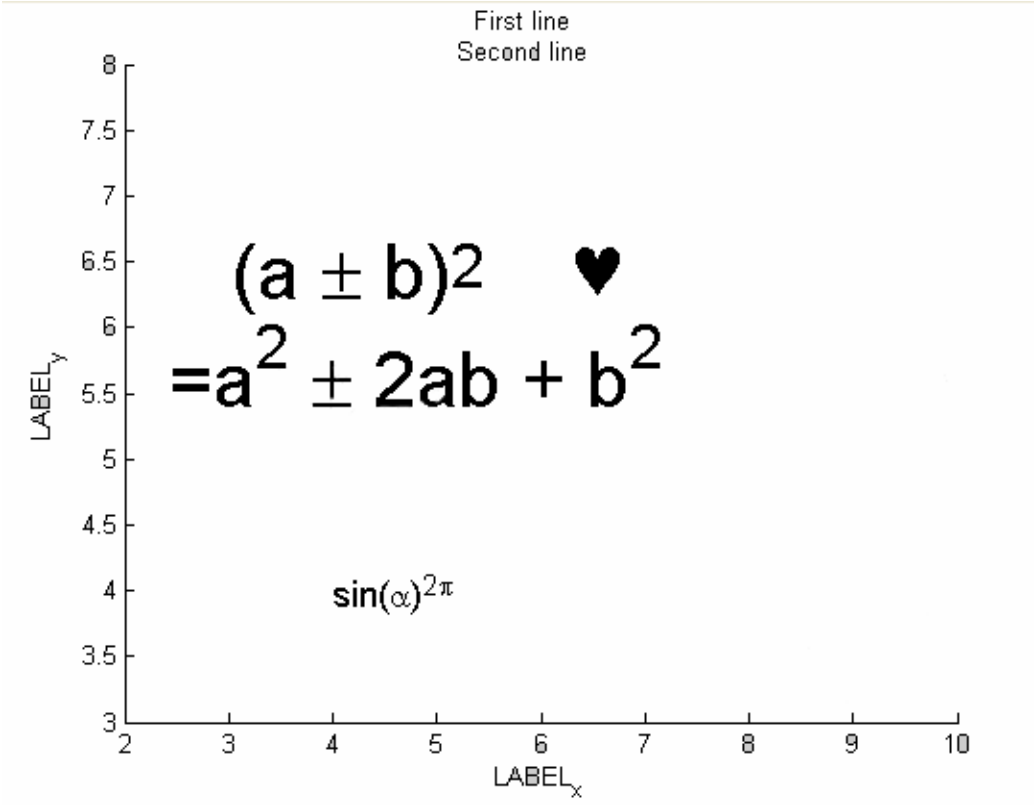
حتي با استفاده از دستور \fontsize مي توان اندازه متن را نيز مشخص كرد.

همچنين ممكن است تنها نمايش قسمتي از نمودار براي ما مهم باشد. دستور axis با مشخص كردن حدود محورها اين كار را انجام مي دهد.همان طور كه در زير مي بينيد آرگومان ورودي دستور شامل يك بردار كه مشخص كننده حدود محورها است مي باشد.

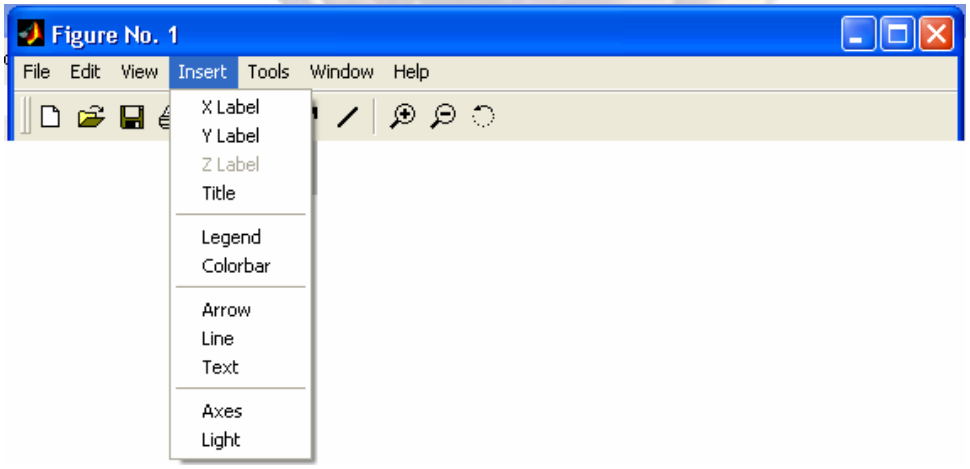
axis ([XMIN XMAX YMIN YMAX])

در مثال زير با روش استفاده از دستورات اخير آشنا مي شويد.

```
>> axis ([2 10 3 8])
>> xlabel ('LABEL_x')
>> ylabel ('LABEL_y')
>> title ({'First line','Second line'})
>> text (6,6.5,'\fontsize{34} \heartsuit')
>> gtext ({'\fontsize{30}   (a \pm b)^2','=a^2 \pm 2ab + b^2'})
>> text (4,4,'\fontsize{14}sin(\alpha)^{2\pi}')
```



بسياري از دستوراتي كه در بالا توضيح داده شد بدون تايپ در پنجره command و از طريق منوي ensert پنجره figure قابل دسترسي هستند.اگر احتياجي به ياد گرفتن دستورات بالا نمي بينيد مي توانيد به اين طريق عمل كنيد.



تا به حال نمودارهايي را رسم كرديم كه محورهاي مختصات آنها به صورت خطي تقسيم بندي شده بود؛ ولي در برخي از مواقع لازم است كه يك يا هر دو محور را با تقسيمات لگاريتمي نمايش دهيم .براي اين كار نيز دستوراتي وجود دارد.

از تابع semilogx براي نموداري كه محور x آن برحسب مقدار لگاريتمي تقسيم بندي شده و از تابع semilogy براي نموداري با محور y لگاريتمي استفاده كنيد.همچنين تابع loglog نموداري رسم مي كند كه هر دو محور آن لگاريتمي است. آرگومان هاي ورودي اين توابع مانند تابع plot ميباشد.

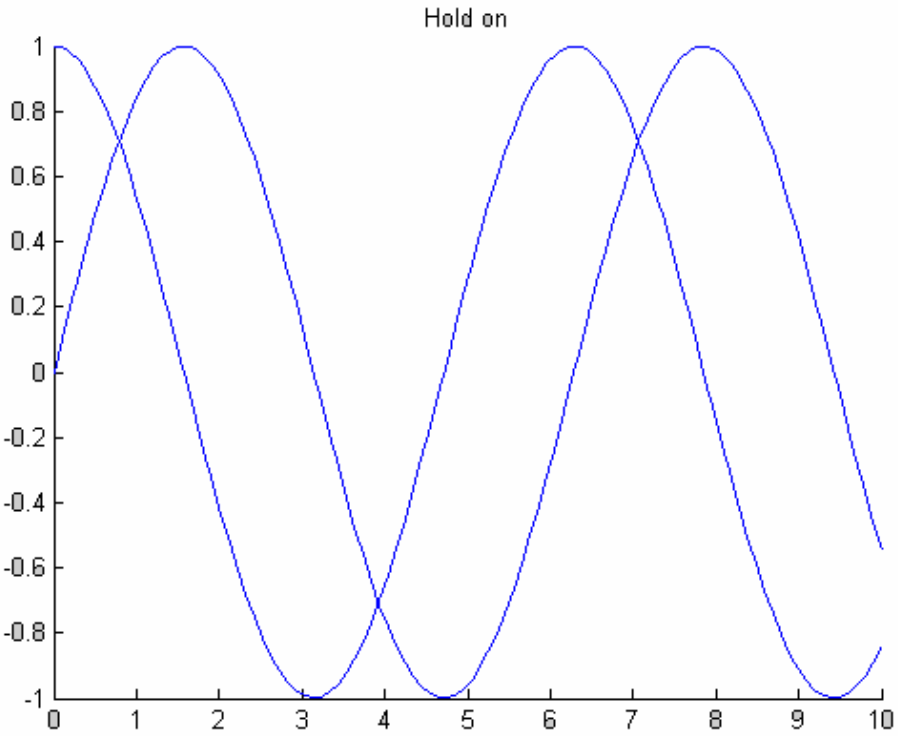
نمودارهاي متعدد:

تا اينجا دستورات متنوعي براي رسم نمودار آموختيم.ولي امكان دارد كه بخواهيم چند نمودار را همزمان بخواهيم و اين خواسته با توجه با اين كه پنجره figure با رسم نمودار جديد پاك مي شود و نمودار جديد جايگزين قبلي مي شود به روش معمولي امكان پذير نيست.

در زير چند روش را براي اين كار بيان مي كنيم.

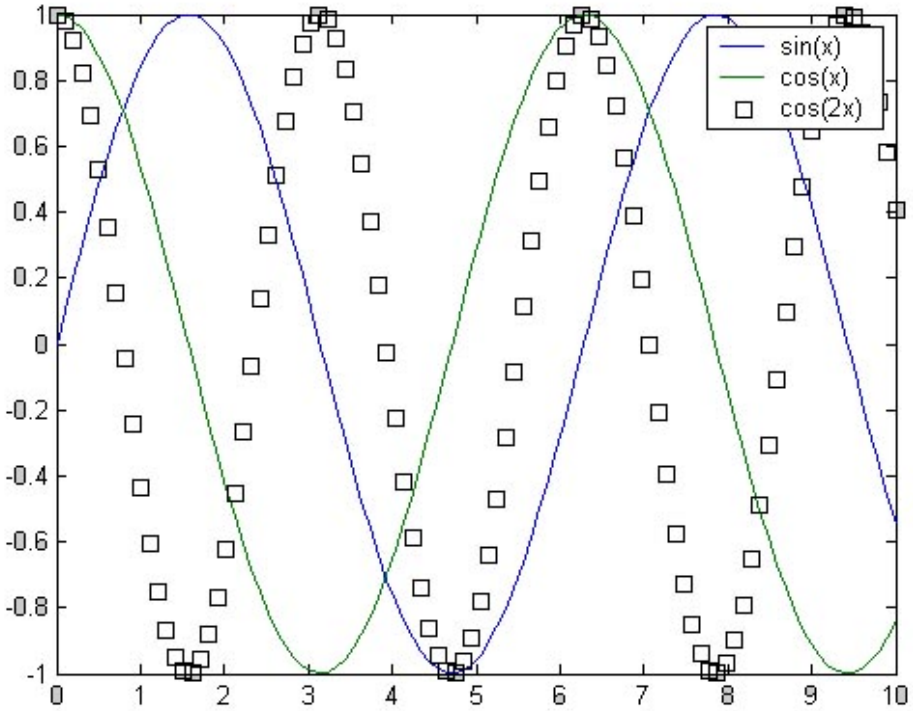
» روش اول:
در اين روش از دستور hold استفاده مي شود.اين دستور محتويات پنجره figure را نگه داشته و نمودار جديد را روي نمودار قبلي رسم مي كند.در اين روش با توجه به اين كه نمودارها به يك رنگ رسم مي شوند بهتر است رنگ و نوع خط نمودار توسط كاربر مشخص شود.

```
>> x=linspace(0,10);
>> y=sin(x);z=tan(x);s=cos(x);t=x.^2;
>> hold on
>> plot(x,y)
>> plot(x,s)
>> title('Hold on')
>> hold off
```



» روش دوم:
در اين روش از تابع plot استفاده مي شود.در اين تابع مي توان بعد از جفت آرگومان اول ،جفت آرگومان مربوط به نمودار بعدي را به عنوان آرگومان هاي بعدي وارد كرد.به اين ترتيب اين تابع مي تواند بيشمار آرگومان ورودي داشته باشد.MATLAB اين نمودارها را با رنگ هاي مختلف رسم مي كند.
در صورتي كه بخواهيم نوع خط و ... را مشخص كنيم بايد بعد از هر جفت آرگومان اين كار را انجام دهيد.

```
>> plot (x,y,x,s,x,cos(2*x),'sk')
>> legend('sin(x)','cos(x)','cos(2x)')
```



» روش سوم:
در اين روش دو نمودار با محور x مشترك و محور y مختص به خود كه تقسيم بندي متفاوتي دارند رسم مي شود. اين كار توسط تابع plotyy انجام مي گيرد.اين تابع حداكثر دو نمودار را رسم مي كند، به اين ترتيب داراي دو جفت آرگومان ورودي است.حالت كلي آن را در زير مي بينيد:

plotyy(x1,y1,x2,y2,' fun1' ,' fun2')

دو آرگومان آخر مشخص كننده نوع محورهاي مختصات براي نمودار اول و دوم مي باشد؛ و مي تواند يكي از موارد زير باشد.

semilogx, semilogy, plot, loglog, stem

در اين تابع نمي توان به سادگي تابع plot نوع خطوط و ... را مشخص كرد.براي اين كار بايد از اشاره گرها يا روش هاي ديگر استفاده كرد. به دليل گستردگي اين بحث تنها به يك مثال قناعت مي كنيم.

» روش چهارم:
در اين روش از پنجره هاي متعدد استفاده مي شود.به اين طريق كه قبل از هر تابع رسم نمودار از دستور figure(n) استفاده مي كنيم كه n مشخص كننده شماره پنجره است كه براي فراخواني پنجره از آن استفاده مي شود.اين دستور پنجره figure جديدي را باز كرده و نمودار را در اين پنجره رسم مي كند.
دستورات زير را تايپ كنيد و نتيجه آن را مشاهده كنيد.

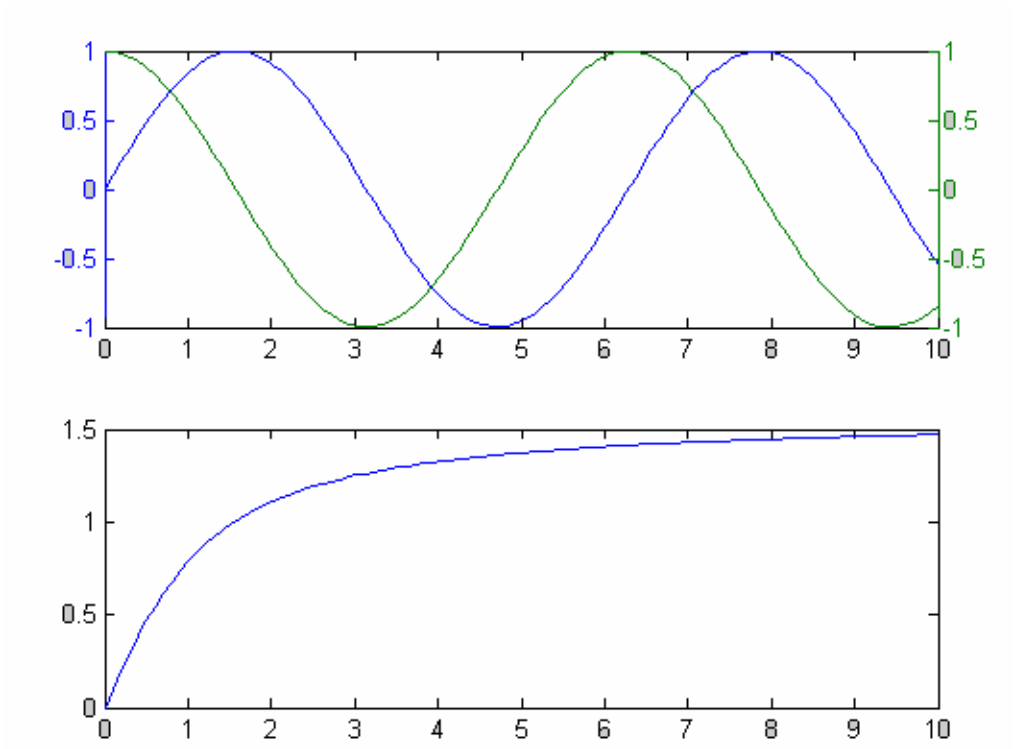
```
>> figure(1)
>> x=linspace(0,10);
>> y=sin(x);s=cos(x);
>> plot(x,y)
>> figure(2)
>> plot(x,s)
```

» روش پنجم:
در اين روش پنجره را به چند قسمت تقسيم كرده و هر نمودار را در يكي از اين قسمت ها رسم مي كينم.
اين تقسيم توسط دستور subplot انجام مي شود.حالت كلي اين دستور به صورت زير است:

subplot (m,n,p) or subplot(mnp)

اين دستور پنجره figure را به يك ماتريس mxn تقسيم مي كند و p امين خانه آن را انتخاب مي كند.شماره هر خانه به صورت رديفي تعيين مي شود.

```
>> subplot (2,1,1)
>> plotyy (x,y,x,s)
>> subplot (2,1,2)
>> plot (x,atan(x))
```



» روش ششم:
در اين روش نمودار جديد با محورها جديد و مقياس متفاوت روي نمودار قبلي قرار مي گيرد. تابع مورد استفاده در روش تابع axes مي باشد.حالت كلي آن به صورت زير است:

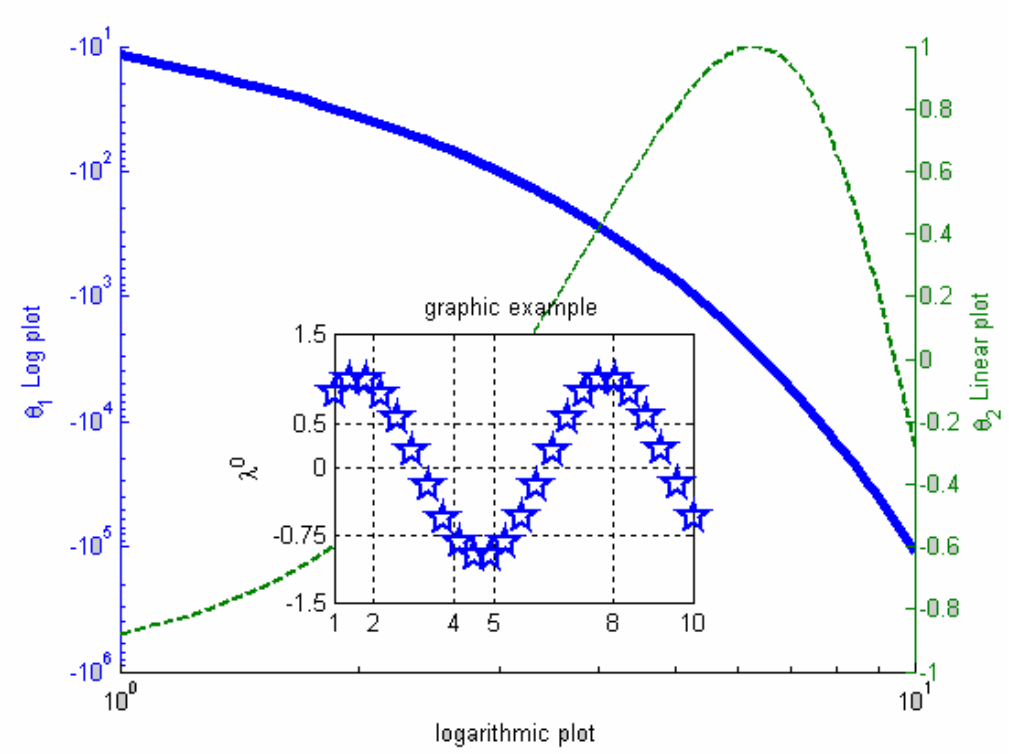
axes (' position' , [left, bottom, width, height])

اين دستور داراي دو آرگومان ورودي است ؛آرگومان اول يك رشته كركتري به صورت بالا و آرگومان بعدي يك بردار است. دو عنصر اول بردار مشخص كننده مكان نمودار جديد و دو عنصر بعدي مشخص كننده اندازه آن است.
اين دستور مختصات (0،0) را براي گوشه پايين سمت چپ و (1،1) را براي گوشه بالا سمت راست در نظر مي گيرد.

مثال زير نمونه اي از يك M-file است كه تقريبا در بر گيرنده تمام دستوراتي است كه در اين بخش توضيح داده شد.به تنظيماتي كه مي توان روي نمودارها اعمال كرد دقت كنيد و سعي كنيد دستور مربوط به هر يك را پيدا كنيد.

```
x=1:.1:10;
y=-10*sinh(x);
z=-cos(x/2);
[AX,H1,H2]=plotyy (x,y,x,z,'loglog','semilogx')
set(get(AX(1),'Ylabel'),'String','\theta_1 Log plot');
set(get(AX(2),'Ylabel'),'String','\theta_2 Linear plot');
set(get(AX(2),'Xlabel'),'String','logarithmic plot');
set(H1,'LineWidth',4);
set(H2,'LineStyle','--','LineWidth',2);


axes('Position',[.34 .2 .35 .35]);
x1=linspace(1,10,24);
y1=sin(x1);
f=plot(x1,y1,'p');
axis([1 10 -1.5 1.5]);
set(f,'MarkerSize',12,'LineWidth',2);
ylabel('\lambda^0');
title('graphic example');
set(gca,'XTick',[1 2 4 5 8 10],'YTick',[-1.5 -.75 0 .5 1.5]);
grid on;
```



در اين بخش دستوراتي را معرفي مي كنيم كه در نوشتن برنامه هاي مختلف به ما كمك مي كند.اين دستورات در اغلب زبان هاي برنامه نويسي مشترك است و تنها گرامر آن ها متفاوت است.

ساخت function file:

تا به حال تنها از توابعي استفاده مي كرديم كه قبلا براي MATLAB تعريف شده بود؛ ولي ممكن است اين توابع نتوانند نيازهاي ما را پاسخ دهند، يا بخواهيم توابعي با كاربري خاص بنويسيم.

يك تابع (function file) مانند يك M-file است با اين تفاوت كه خط اول آن به صورت زير است:

```
function [outputs]= name(inputs)
```

اين خط مشخص مي كند كه اين M-file يك تابع است. همچنين تعداد ورودي ها و خروجي ها را مشخص كرده و هر يك را در يك متغير قرار مي دهد.در صورتي كه تنها يك ورودي داشته باشيم نيازي به كلوشه ([]) نيست. name نيز نام تابع را مشخص مي كند.

بهتر است براي خواناتر شدن برنامه از عبارات توضيحي استفاده كنيم.اين عبارات بايد ورودي ها و خروجي ها را مشخص كند.همچنين مي توان نام برنامه نويس و تاريخ نوشتن آن را نيز مشخص كرد. اين خطوط با اجرا دستور help name>> به نمايش درمي آيند.
به عنوان مثال M-file زير دستور prod را شبيه سازي مي كند.

```
function p=prod۲ (x)
% function p=PROD۲ (x)
% shabih sazi farman PROD
% a:radif   b:sotoon
[a,b]=size۲(x);
p(۱,:)=x(۱,:);
for i=۲:a,
%   satr aval p dar satr haye x zarb shode
%   va dar satr aval p zakhir mishavad
   p(۱,:)=p(۱,:).*x(i,:);
end
% if x is rowvector
while a==۱;
   p=۱;
   for i=۱:b,
      p=p*x(i);
   end
   a=۰;
end
```

حلقه هاي تكرار:

اين دستورات در اغلب زبان هاي برنامه نويسي به خصوص c وجود دارند.

» حلقه For
اين حلقه اين امكان را به وجود مي آورد كه تعدادي از دستورات به تعداد دفعات از قبل تعيين شده تكرار شوند.شكل كلي آن به صورت زير است:

```
for variable = a
    statement ۱
    statement ۲
   ...
end
```

كه a يك ماتريس است.در هر بار تكرار حلقه يك ستون ماتريس a در variable قرار مي گيرد.به اين ترتيب حلقه به تعداد ستون هاي a تكرار مي شود.
اين حلقه را مي توان به صورت تو در تو استفاده كرد. مثال زير با استفاده از حلقه هاي تو در تو جدول ضرب ايجاد مي كند.

```
for i=١:٥
   for j=١:٥
      s(i,j)=i*j;
   end
end
```

» حلقه While

اين حلقه چند دستور را به تعداد دفعات نامحدود تكرار مي كند. از اين دستور هنگامي استفاده مي شود كه تعداد دفعات تكرار مشخص نباشد. شكل كلي اين دستور به صورت زير است:

```
while expression
        statements
end
```

expression يك عبارت شرطي است و تا هنگامي كه درست باشد، حلقه تكرار مي شود. (عبارات شرطي در قسمت بعد شرح داده مي شود)

عملگرهاي رابطه اي:

اين عملگرها شامل موارد زير مي باشد:

عملگرهاي رابطه اي	شرح
<	كوچكتر از
<=	كوچكتر يا مساوي
>	بزرگتر
>=	بزرگتر يا مساوي
==	مساوي با
~=	مخالف با (نامساوي)

عملگرهاي منطقي:

اين عملگرها را در جدول زير مشاهده مي كنيدك

عملگر منطقي	شرح
&	AND
|	OR
~	NOT
xor(x,y)	OR انحصاري (در صورتي كه تنها يكي (x يا y) مقدار درستي داشته باشند مقدار True برمي گرداند)

ساختارهاي تصميم:

» شرط If - Else – End

حتما با عملكرد اين دستور در زبان هاي برنامه نويسي ديگر آشنا شده ايد.شكل كلي اين دستور را در زير مي بينيد.

```
 if expression ١
    statements ١
 elseif expression ٢
    statements ٢
...
elseif expression n
    statements n
else
    statements
end
```

همان طور كه مشاهده مي كنيد در حالت كلي مي توان از يك If ،بيشمار Elseif و يك Else و End استفاده كرد.استفاده از Else و Elseif اختياري است.

اگر شرط مقابل If درست باشد دستورات شماره ١ اجرا مي شوند، در غير اين صورت شرط ٢ (مقابل Elseif) بررسي مي شود در صورتي كه درست باشد دستورات ٢ ودر غير اين صورت شرط ٣ بررسي مي شود ... در صورتي كه n شرط بررسي شد و درست نبود دستورات قسمت Else اجرا مي شود.

» شرط Switch-Case

از اين ساختار براي تصميم گيري چندگانه بر اساس مقادير مختلف يك عبارت استفاده مي شود.به طور كلي در تمام تصميم گيري ها كه بيش از ٣ انتخاب وجود داشته باشد از اين دستور استفاده مي شود.

به عنوان مثال فرض كنيد متغيري مثل x مقادير ١، ٢، ٣ را اختيار مي كند و مي خواهيد بر اساس مقادير مختلف x تصميم گيري مختلفي را انجام دهيد.اگر برابر ١ بود دستورات ١، اگر برابر ٢ بود دستورات ٢ و اگر برابر ٣ بود دستورات ٣ اجرا شوند و در صورتي كه هيچ كدام از اين ها نبود دستورات ٤ (otherwise) اجرا شوند.حالت كلي اين دستور را مشاهده مي كنيد:

```
switch switch_expr
case case_expr,
       statement, ..., statement
case {case_expr١, case_expr٢, case_expr٣,...}
       statement, ..., statement
  ...
otherwise,
       statement, ..., statement
end
```

به چند نكته در اين مورد بايد دقت كرد:
١) پس از اجراي هر يك از دستورات روند اجرا برنامه به بعد از End منتقل مي شود و ساير Case ها كنترل نمي شوند.
٢) در بالا در مورد Case دوم در صورتي كه عبارت مورد نظر با هر يك ٣ عبارت داخل كلوشه ({}) برابر باشد دستورات اجرا مي شوند.
٣) استفاده از Otherwise نيز اختياري است.

بلوك Try-Catch:

شكل كلي اين دستور به اين صورت مي باشد:

```
try
   commands
catch
    commands
end
```

عملكرد اين دستور به اين صورت است كه دستورات زير Try اجرا مي شوند؛ در صورتي كه خطايي رخ دهد كنترل برنامه به Catch منتقل شده و دستورات موجود در اين قسمت اجرا مي شود. اين خاصيت باعث مي شود از آن براي خطايابي برنامه ها استفاده شود.

توقف روند اجراي برنامه:

Break «

هنگامي كه اين دستور اجرا مي شود MATLAB به اولين دستور كه بعد از حلقه For قرار دارد مي رود.در صورتي كه اين دستور در حلقه هاي تو درتو (For يا While) به كار رود MATLAB فقط از حلقه جاري خارج مي شود.

Error «

اين دستور باعث توقف اجرا برنامه شده و مي تواند يك رشته كركتري را برگرداند.

```
error (' STATEMENT ')
```

Return «

هر گاه روند اجرا برنامه به اين دستور برسد مقدار مورد نظر را برمي گرداند (در Command window نمايش مي دهد)؛ و ادامه اجراي برنامه متوقف مي شود.

از اين دستور براي نمايش زود هنگام مقادير يعني قبل از به پايان رسيدن كامل برنامه استفاده مي شود. به اين ترتيب هرگاه جواب مورد نظر به دست آمد روند اجراي برنامه نيز متوقف مي شود و مقدار مورد نظر را برمي گرداند.