تابستان ۱۳۹۵



Matlab Training

دوره آموزشی MATLAB

مقدماتی و پیشرفته

مرجع تخصصی کدنویسی متلب | تهیه و تنظیم : لعیا افضلی پور

هوالحق

سر فصل های مقدماتی :

- آشنایی با نرم افزار ریاضی – مهندسی MATLAB , معرفی محیط کار نرم افزار
- آشنایی با دستورات ابتدایی و مفاهیم اولیه
- رسم در MATLAB
- توابع تعریف شده
- کنترل (برنامه نویسی) در MATLAB

سر فصل های پیشرفته :

- **ساختمان داده ها**
- **رشته ها و مهندسی فایل ها**
- **ریاضیات کاربردی**
  - **دستگاه معادلات**
  - **سیستم های بهینه سازی**
  - **معادلات دیفرانسیل**
  - **آمار**

lafzalipour@hotmail.com

l.afzalipour@gmail.com

a_davari@sci.ui.ac.ir

تنظیم: لعیا افضلی پور ؛ علی داوری

تنظیم: لیلا افضلی پور و مهدی داوری

فصل اول

آشنایی با محیط MATLAB

آغاز راه

Matlab مخفف Matrix Laboratry

- ابداع: اواخر ۱۹۷۰ توسط cleve Molder رییس بخش علوم کامپیوتر دانشگاه نیومکزیکو
- برپایه زبان برنامه نویسی فرترن
- مورد استقبال ریاضیات کاربردی
- اشنایی Molder و jack little در سال ۱۹۸۳

توسعه

- کشف پتانسیل تجاری توسط jack little
- آغاز همکاری و باز نویسی به زبان C
- واگذاری ادامه راه بهMathworks در سال ۱۹۸۴
- اولین توسعه Matlab درحوزه مهندسی کنترل توسط Little
- رشد سریع Matlab در سایر زمینه ها

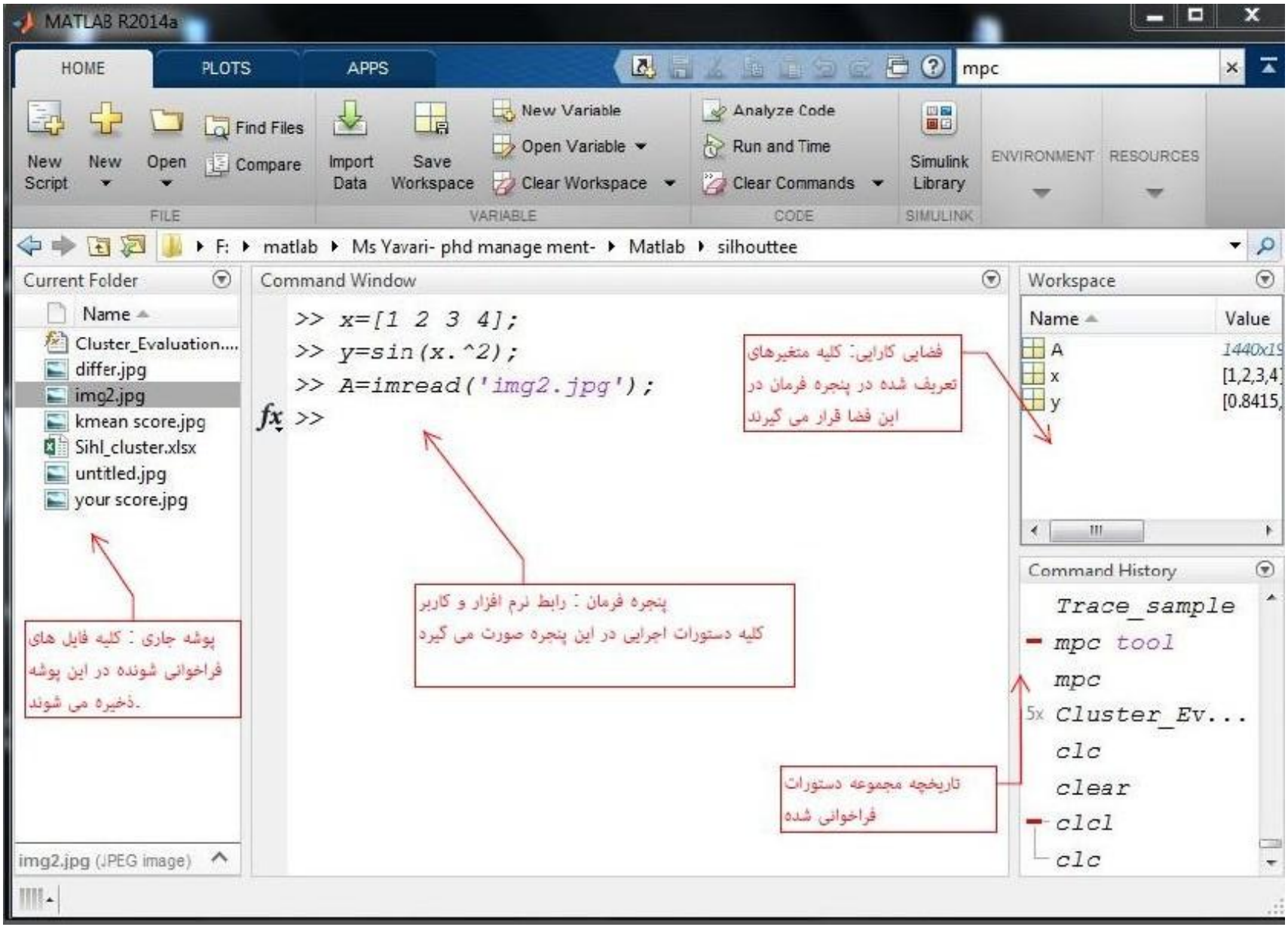
تنظیم: لعیا افضلی پور ؛ علی داوری

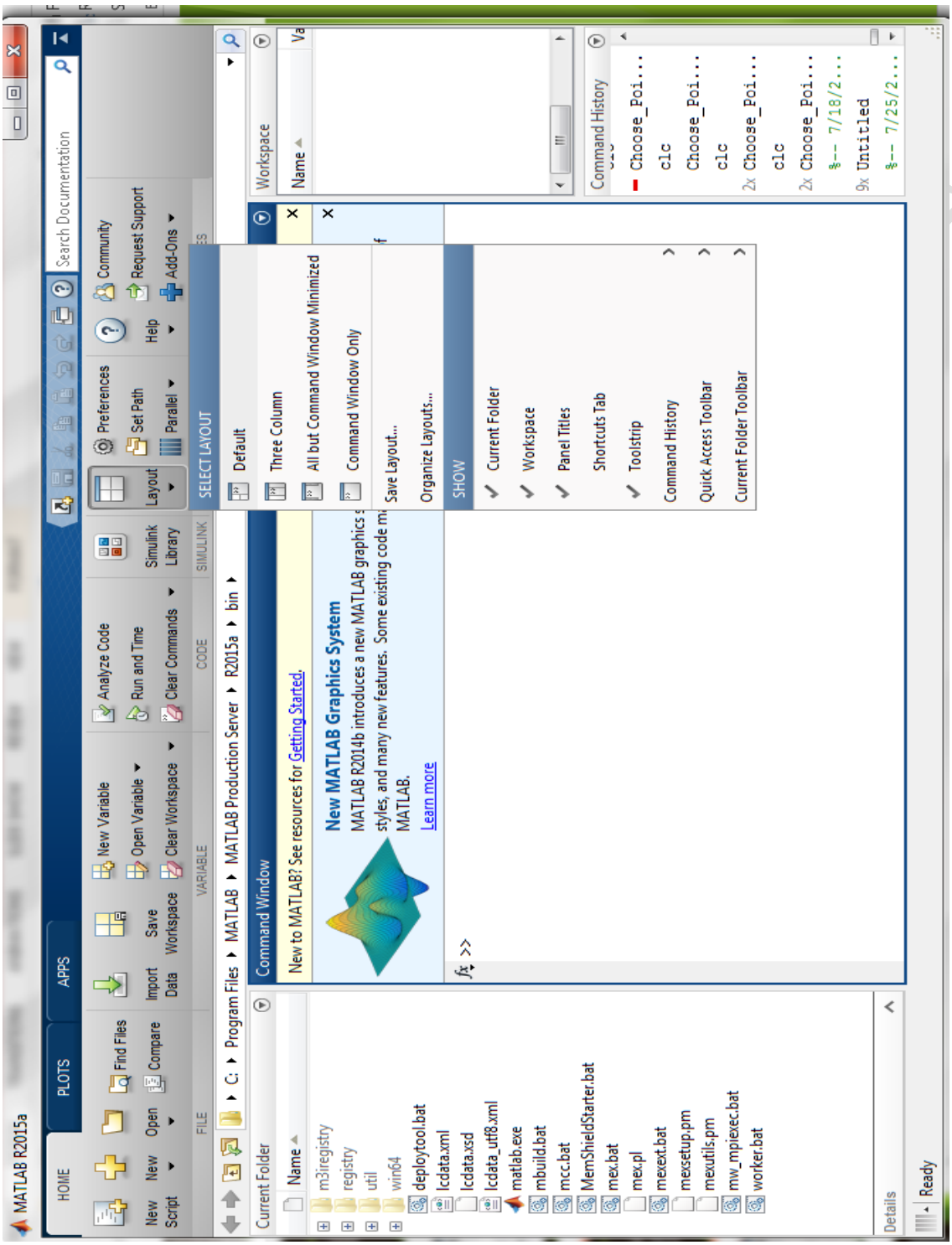
- از دیر باز تا کنون این نرم افزار در جبر خطی و آنالیز عددی و پردازش تصویر و.... کاربرد فراوان دارد

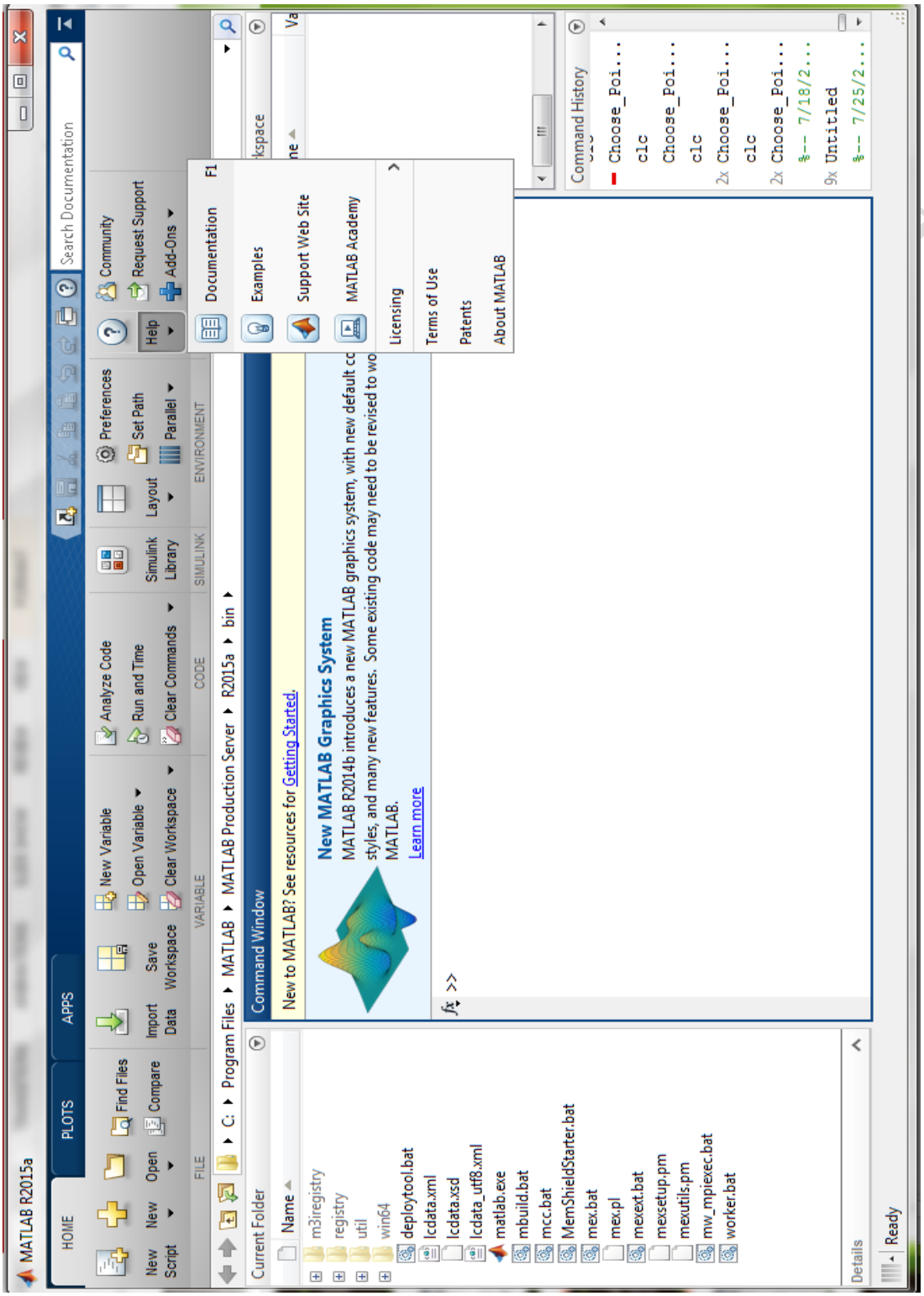
اجزای محیط

- پوشه جاری Current Folder
- پنجره فرمان Command Window
- فضای کاری Workspace
- تاریخچه فرمان Command History

تنظیم: نعیما فضلی پور؛ علی داوری


MATLAB R2014a
HOME
PLOTS
APPS
mpc
New Script
New
Open
Find Files
Compare
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspace
Analyze Code
Run and Time
Clear Commands
Simulink Library
ENVIRONMENT
RESOURCES
FILE
VARIABLE
CODE
SIMULINK
F: ▸ matlab ▸ Ms Yavari- phd manage ment- ▸ Matlab ▸ silhouttee
Current Folder
Name
Cluster_Evaluation....
differ.jpg
img2.jpg
kmean score.jpg
Sihl_cluster.xlsx
untitled.jpg
your score.jpg
Command Window
>> x=[1 2 3 4];
>> y=sin(x.^2);
>> A=imread('img2.jpg');
>>
Workspace
Name
Value
A
1440x19
x
[1,2,3,4]
y
[0.8415,
Command History
Trace_sample
mpc tool
mpc
5x Cluster_Ev...
clc
clear
clcl
clc
فضایی کارایی: کلیه متغیرهای تعریف شده در پنجره فرمان در این فضا قرار می گیرند
پنجره فرمان : رابط نرم افزار و کاربر کلیه دستورات اجرایی در این پنجره صورت می گیرد
پوشه جاری : کلیه فایل های فراخوانی شونده در این پوشه ذخیره می شوند.
تاریخچه مجموعه دستورات فراخوانی شده
img2.jpg (JPEG image)


MATLAB R2015a
HOME
PLOTS
APPS
Search Documentation
New Script
New
Open
Find Files
Compare
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspace
Analyze Code
Run and Time
Clear Commands
Simulink Library
Layout
Preferences
Set Path
Parallel
Help
Community
Request Support
Add-Ons
FILE
VARIABLE
CODE
SIMULINK
RESOURCES
SELECT LAYOUT
Default
Three Column
All but Command Window Minimized
Command Window Only
Save Layout...
Organize Layouts...
SHOW
Current Folder
Workspace
Panel Titles
Shortcuts Tab
Toolstrip
Command History
Quick Access Toolbar
Current Folder Toolbar
C: ▸ Program Files ▸ MATLAB ▸ MATLAB Production Server ▸ R2015a ▸ bin ▸
Current Folder
Name
m3iregistry
registry
util
win64
deploytool.bat
lcdata.xml
lcdata.xsd
lcdata_utf8.xml
matlab.exe
mbuild.bat
mcc.bat
MemShieldStarter.bat
mex.bat
mex.pl
mexext.bat
mexsetup.pm
mexutils.pm
mw_mpiexec.bat
worker.bat
Command Window
New to MATLAB? See resources for Getting Started.
New MATLAB Graphics System
MATLAB R2014b introduces a new MATLAB graphics s
styles, and many new features. Some existing code m
MATLAB.
Learn more
>>
Workspace
Name
Command History
clc
Choose_Poi...
Untitled
Details
Ready


MATLAB R2015a
HOME
PLOTS
APPS
Search Documentation
New Script
New
Open
Find Files
Compare
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspace
Analyze Code
Run and Time
Clear Commands
Simulink Library
Layout
Preferences
Set Path
Parallel
Help
Community
Request Support
Add-Ons
FILE
VARIABLE
CODE
SIMULINK
ENVIRONMENT
C: ▸ Program Files ▸ MATLAB ▸ MATLAB Production Server ▸ R2015a ▸ bin ▸
Current Folder
Name
m3iregistry
registry
util
win64
deploytool.bat
lcdata.xml
lcdata.xsd
lcdata_utf8.xml
matlab.exe
mbuild.bat
mcc.bat
MemShieldStarter.bat
mex.bat
mex.pl
mexext.bat
mexsetup.pm
mexutils.pm
mw_mpiexec.bat
worker.bat
Details
Command Window
New to MATLAB? See resources for Getting Started.
New MATLAB Graphics System
MATLAB R2014b introduces a new MATLAB graphics system, with new default co
styles, and many new features. Some existing code may need to be revised to wo
MATLAB.
Learn more
fx >>
Documentation F1
Examples
Support Web Site
MATLAB Academy
Licensing
Terms of Use
Patents
About MATLAB
Command History
Choose_Poi...
clc
Choose_Poi...
clc
2x Choose_Poi...
clc
2x Choose_Poi...
%-- 7/18/2...
9x Untitled
%-- 7/25/2...
Ready

```
8/7/11 10:24 AM  MATLAB Command Window      1 of 1

>> a=ones(3)

a =

     1     1     1
     1     1     1
     1     1     1

>> z=inv(a)
Warning: Matrix is singular to working precision.

z =

   Inf   Inf   Inf
   Inf   Inf   Inf
   Inf   Inf   Inf

>> b=3*eye(3,4)

b =

     3     0     0     0
     0     3     0     0
     0     0     3     0

>> det(b)
??? Error using ==> det
Matrix must be square.

>>
```

پیغام هشدار در متلب

پیغام خطا در متلب

سر فصل های اساسی

❑ عمومی

- آشنایی با دستورات ابتدایی و مفاهیم اولیه
- ترسیم
- توابع تعریف شده
- کنترل (برنامه نویسی)
- ساختمان داده ها
- رشته ها و سسیتم فایل

❑ تخصصی

- جبر خطی – دستگاه خطی
- بهینه سازی – غیر خطی
- آمار
- معادلات دیفرانسیل
- معادلات با مشتقات جزیی
- سیمولینک

تنظیم: لیلا افضلی پور ؛ علی داوری

- پردازش سیگنال
- پردازش صوت
- پردازش تصویر
- طراحی مدل های کنترلی
- انیمیشن سازی
- بیولوژی
- داده کاوی
- شبکه عصبی
- پایگاه داده
- دسترسی به هسته نرم افزار های دیگر
- کامپایل و فایل اجرایی
- برازش منحنی
- موجک ها
- جعبه ابزار مالی
- جعبه ابزار اقتصاد
- منطق فازی

تنظیم: لعیا افضلی پور ؛ علی داوری

تنظیم: لیلا افضلی پور؛ علی داوری

فصل دوم

دستورات ابتدایی و مفاهیم اولیه

متغیر ها و آرایه ها:

متغیر ها و ارایه ها ابزاری برای ذخیره و بازیابی اطلاعات است. متلب یک زبان برنامه نویسی است که ساختارها و دستورات آن بر اساس عملیات ماتریسی و گسترش آنها اجرا می شود. حوزه های متفاوت ذخیره سازی در متلب عبارتند از حوزه اعداد صحیح (integer)، حوزه اعداد ممیز شناور (double) ، حوزه اعداد مختلط (complex) و رشته ها (string).

```
a=2
b=3.2
s=' this is a test'
z=a+i*b
```

در نام گذاری متغیر ها به اصول زیر توجه کنید :

- در نامگذاری ها از space استفاده نشود.
- از اعداد تنها برای نامگذاری فایل ها استفاده نشود. مجموعه‌ای از حروف و اعداد مناسب است .
- متلب به کوچک و بزرگ بودن حروف حساس است.
- توصیه می شود نامگذاری متغیر ها و توابع هدفمند باشد بدان معنا که نام های متناسب با موقعیت استفاده شود.
- از عبارات کلیدی و مقادیر از پیش تعریف شده و کاراکترهای خاص در متلب برای نامگذاری ها استفاده نشود. لیستی از مقادیر مذکور در جداول ۱و۲و۳ضمیمه است.

Operators and Special Characters	
+	Plus; addition operator.
-	Minus; subtraction operator.
*	Scalar and matrix multiplication operator.
.*	Array multiplication operator.
^	Scalar and matrix exponentiation operator.
.^	Array exponentiation operator.
\	Left-division operator.
/	Right-division operator.
.\	Array left-division operator.
./	Array right-division operator.
:	Colon; generates regularly spaced elements and represents an entire row or column.
( )	Parentheses; encloses function arguments and array indices; overrides precedence.
[ ]	Brackets; enclosures array elements.
.	Decimal point.
...	Ellipsis; line-continuation operator.
,	Comma; separates statements and elements in a row.
;	Semicolon; separates columns and suppresses display.
%	Percent sign; designates a comment and specifies formatting.
_	Quote sign and transpose operator.
._	Nonconjugated transpose operator.
=	Assignment (replacement) operator.

جدول ۱ : عملگر ها و کاراکتر های خاص متلب

Commands for Managing a Session	
clc	Clears Command window.
clear	Removes variables from memory
exist	Checks for existence of file or va
global	Declares variables to be global.
help	Searches for a help topic.
lookfor	Searches help entries for a keywc
quit	Stops MATLAB.
who	Lists current variables.
whos	Lists current variables (long displ

جدول۲ : توابع خاص در متلب

Special Variables and Constants	
ans	Most recent answer.
eps	Accuracy of floating-point precision.
i,j	The imaginary unit $\sqrt{-1}$.
Inf	Infinity.
NaN	Undefined numerical result (not a numbe
pi	The number π .

جدول ۳ : مقادیر پیش فرض تعریف در متلب

مقدار دهی :

۱. وارد کردن لیست صریحی از داده ها:

```
array1=[1 2 3 5 10];
```

۲. مقدار دهی با توابع کتابخانه ای :

```
z1=sin(array1);
```

۳. خواندن از صفحه کلید :

```
z2=input('!* Please enter favor vector ')
```

۴. خواندن از فایل های خارجی

مثال های بیشتر در m– فایل chapter_1 موجود است. لیستی از تابع کتابخانه ای که در مقدار دهی و محاسبات مورد استفاده قرار می گیرند در جدول ۴ ضمیمه است .

جدول ۴: تابع کتابخانه ای مهم و کاربردی

Exponential and Logarithmic Functions	
exp(x)	Exponential; e^x.
log(x)	Natural logarithm; ln(x).
log10(x)	Common (base 10) logarithm; log(x)= $\log_{10}(x)$.
sqrt(x)	Square root; $\sqrt{x}$.

Complex Functions	
`abs(x)`	Absolute value; |x|.
`angle(x)`	Angle of a complex number x.
`conj(x)`	Complex conjugate of x.
`imag(x)`	Imaginary part of a complex number x
`real(x)`	Real part of a complex number x.

Trigonometric Functions	
`acos(x)`	Inverse cosine; arcos x = $\cos^{-1}(x)$.
`acot(x)`	Inverse cotangent; arccot x = $\cot^{-1}(x)$
`acsc(x)`	Inverse cosecant; arcs x = $\csc^{-1}(x)$.
`asec(x)`	Inverse secant; arcsec x = $\sec^{-1}(x)$.
`asin(x)`	Inverse sine; arcsin x = $\sin^{-1}(x)$.
`atan(x)`	Inverse tangent; arctan x = $\tan^{-1}(x)$.
`atan2(y,x)`	Four-quadrant inverse tangent.
`cos(x)`	Cosine; cos(x).
`cot(x)`	Cotangent; cot(x).
`csc(x)`	Cosecant; csc(x).
`sec(x)`	Secant; sec(x).
`sin(x)`	Sine; sin(x).
`tan(x)`	Tangent; tan(x).

Random Number Functions	
`rand`	Generates uniformly distributed random numbers between 0 and 1
`randn`	Generates normally distributed random numbers.

Special Matrices	
`eye`	Creates an identity matrix.
`ones`	Creates an array of ones.
`zeros`	Creates an array of zeros.

Hyperbolic Functions	
`acosh(x)`	Inverse hyperbolic cosine; $\cosh^{-1}(x)$.
`acoth(x)`	Inverse hyperbolic cotangent; $\coth^{-1}(x)$.
`acsch(x)`	Inverse hyperbolic cosecant; $\mathrm{csch}^{-1}(x)$.
`asech(x)`	Inverse hyperbolic secant; $\mathrm{sech}^{-1}(x)$.
`asinh(x)`	Inverse hyperbolic sine; $\sinh^{-1}(x)$.
`atanh(x)`	Inverse hyperbolic tangent; $\tanh^{-1}(x)$.
`cosh(x)`	Hyperbolic cosine; cosh(x).
`coth(x)`	Hyperbolic cotangent; cosh(x)/sinh(x).
`csch(x)`	Hyperbolic cosecant; 1/sinh(x).
`sech(x)`	Hyperbolic secant; 1/cosh(x).
`sinh(x)`	Hyperbolic sine; sinh(x).
`tanh(x)`	Hyperbolic tangent; sinh(x)/cosh(x).

Numeric Functions	
ceil	Rounds to the nearest integer toward ∞.
fix	Rounds to the nearest integer toward zero.
floor	Rounds to the nearest integer toward - ∞.
round	Rounds towards the nearest integer.
sign	Signum function.

به عنوان مثال نحوه فراخوانی برخی از این تابع در m – فایل با نام های setting_data و most_important_lib_func آورده شده است.

❖ توجه کنید که متغیر x می تواند تک مقدار ، بردار و یا ماتریس باشد، به نحوه تاثیر این توابع بر بردار ها و ماتریس ها توجه کنید.

ذخیره و بازیابی داده ها:

برای ذخیره لیست داده ها از دستور save استفاده می شود . این دستور سه فرم کلی دارد :

```
save
save  نام فایل
save  لیست داده ها نام فایل
```

در فرم اول ، پس از فراخوانی دستور save ، کلیه داده های موجود در فضای work space در فایلی به نام matlab.mat ذخیره می شود.

در فرم دوم پس از فراخوانی دستور save ، کلیه داده های موجود در فضای work space در فایلی به نام ذکر شده ذخیره می شود. این فایل نیز در پوشه جاری قرار خواهد گرفت .

در فرم سوم پس از فراخوانی دستور save ، صرفا داده های ذکر شده در لیست در فایلی به نام ذکر شده ذخیره می شود.

در هر سه مورد فایل مذکوربه صورت خودکار در پوشه جاری قرار خواهد گرفت.

مثال : فرض کنید لیستی از متغیر ها در فضای کار موجود باشند ،

```
%storing data
save
save myfile
save myfile2 A A2 z1 z2
```

دستورات فوق نحوه فراخوانیها را نشان می دهد.

برای بازیابی دادها از دستور load استفاده می شود. متناظر با سه فرم دستور save در اینجا نیز سه نحوه فراخوانی از دستور load وجود دارد :

```
load
load myfile
load myfile2 A2
```

❖ توجه کنید که فایل مورد نظر حتما در پوشه جاری قرار گرفته باشد، در غیر این صورت فراخوانی دستور load با پیغام خطا مواجه خواهد شد.

دسترسی به عناصر آرایه ای :

همانطورکه تاکنون دیده شد، آرایه های متلب می تواند یک بعد (بردار)، دو بعد (ماتریس) یا تعداد بیشتری بعد داشته باشد. با اختصاص یک اندیس به هر بعد می توان آرایه هارا کنترل کرد. تعداد المانهای یک آرایه چند بعدی با حاصلضرب ماکزیمم مقادیر اندیس ها برابر خواهد بود.

با استفاده از دستور size(a) می توان ابعاد ماتریس a را بدست آورد.

```
a=[1 2 3 4 5;6 7 8 9 10;11 12 13 14 15]
[m,n]=size(a)
```

پس از اجرا دستور فوق مقادیر m,n به ترتیب ۵ و۳ خواهد بود که متناظر تعداد ستون وتعداد سطر ها در ماتریس a است.

```
ii=2;
jj=3;

i2=3;
j2=5;


z=a(ii,:)% iith row
z2=a(:,jj)%jjth column
z3=a(ii,jj)% (ii,jj)th element

z4=a(ii:i2,jj:j2)% block ii to i2(row ),jj to j2(column)
z4=a([ii,i2],[jj,j2])% cross of ii & i2(row )by jj & j2(column)


% inserting data in a row or column
a(ii,:)=0%
a(:,jj)=0
a(ii:i2,jj:j2)=20% inserting in block ii to i2(row ),jj to j2(column)
a([ii,i2],[jj,j2])=0% inserting in cross of ii & i2(row )by jj & j2(column)

% deleting a row  or a column
a(ii,:)=[]
a(:,jj)=[]
```

همان طور که در دستورات فوق مشهود است، دراین نحوه دسترسی هر دو قابلیت بازیابی و ذخیره داده ها ممکن است.

آرایه های چند بعدی :

فرض کنید تابعی بر حسب سه پارامتر تعریف شده باشد مثلا قرار است دما را در موقعیت مختلف یک شبکه دو بعدی و در ۴ فاصله زمانی ثبت شود. برای ذخیره این داده ها باید مختصات مکانی به انضمام اطلاعات مربوط به فاصله زمانی به عنوان اندیس و دما به عنوان مقدار المان ذخیره شود.

```
m=3;n=2;k=3
c=zeros(m,n,k);
c(:,:,1)=[1 2;3 4;5 6]
c(:,:,2)=2*ones(m,n)
c(:,:,2)=2*ones(m,n)
```

دستور c=zeros(m,n,k) متناظر با سایز مورد نظر به ارایه c فضا اختصاص می دهد ، دستورات متوالی مقادیر مورد نظر را به ارایه اختصاص می دهد.

کنترل آرایه های چند بعدی با استفاده از یک اندیس :

یک آرایه دو بعدی با m سطر و n ستون شامل $m \times n$ عنصر است. این عناصر $m \times n$ موقعیت را در حافظه اشغال می کند. متلب عناصر آرایه را به ترتیب شماره ستون ها ذخیره و بازیابی می کند، این بدان معنا است که مقادیر ستون اول را به حافظه تخصیص می دهد سپس ستون دوم و بعد ستون سوم و الی آخر. این نحوه تخصیص دهی در مورد آرایه ها با ابعاد بزرگتر نیز

کاربرد دارد، اولین اندیس آرایه با سرعت بیشتر و دومین اندیس با سرعت کمتری افزایش می یابد و همینطور در سومین اندیس، افزایش در آخرین اندیس بسیار کندتر است. در یک آرایه $2\times2\times2$ عناصر به ترتیب زیر تخصیص داده می شوند:

$$(1,1,1),(2,1,1),(1,2,1),(2,2,1),(1,1,2),(2,1,2),(1,2,2),(2,2,2)$$

فرض کنید ماتریس مورد نظر از مرتبه $m\times n$ باشد در این صورت به جای استفاده از اندیس دوتایی (i,j) از تک اندیس $k=(j-1)m+i$ استفاده کرد. برای مراتب بالاتر هم می‌توان به طریق مشابه عمل کرد.

تولید بردارها:

- **بردار ها با فاصله خطی :**

۱. استفاده از عملگر :

۲. استفاده از دستور linspace

```
 % linear distance
a=1
b=3
n=10
step=.02
x=a:step:b;
y=linspace(a,b,n)
```

- بردار با فاصله لگاریتمی

۱. با استفاده از دستور logspace

```
    % logarithm space
z=logspace(a,b,n)
```

تنظیم: نعیما افضلی پور ؛ علی داوری

خواندن و چاپ داده :

برای خواندن داده ها از صفحه کلید از دستور input استفاده می شود. این دستور به دو صورت قابل فراخوانی است.

```
 % getting gata
a=input('stater message :')
s=input(stater message :','s')
```

در فرم اول، داده وارد شده به همان صورت که کاربر وارد می‌کند ذخیره می‌شود. فرم دوم داده وارد شده را به صورت رشته ذخیره می‌کند.

برای چاپ داده‌ها در متلب می‌توان از دستورات disp ، fprintf و عدم استفاده از ; در انتهای فراخوانی نام متغیر بهره برد.

```
% printing data
disp(a)
fprintf('\n those parameters are %.3f : \n',a)
format long
fprintf('\n\t those parameters are %.6d : ',a)
```

Input/Output Commands	
disp	Displays contents of an array or string.
fscanf	Read formatted data from a file.
format	Controls screen-display format.
fprintf	Performs formatted writes to screen or file
input	Displays prompts and waits for input.
;	Suppresses screen printing.

جدول ۵ : دستورا ت ورودی و خروجی در متلب

Format Codes for fprintf and fscanf	
%s	Format as a string.
%d	Format as an integer.
%f	Format as a floating point value.
%e	Format as a floating point value in scientific notation
%g	Format in the most compact form: %f or %e.
\n	Insert a new line in the output string.
\t	Insert a tab in the output string.

جدول ۶ : کاراکتر های تبدیل در دستور fprintf

اعمال رو ی ماتریس ها :

اعمال متناظر با آرایه ها به چند گروه عمده تقسیم بندی می شوند. دو گروه آن عملگر های حسابی و عملگر های آرایه (نقطه‌ای) هستند، لیست این عملگر ها در جدول ۷ ضمیمه است. دسته دیگر عملگر های مقایسه ای و منطقی اند که در جدول ۸ آورده شده اند. به کمک این عملگر ها می توان آرایه ماسک را برای پردازش سریع داده‌ها به کار برد.

Logical and Relational Operators	
==	Relational operator: equal to.
~=	Relational operator: not equal to.
<	Relational operator: less than.
<=	Relational operator: less than or equal to.
>	Relational operator: greater than.
>=	Relational operator: greater than or equal tc
&	Logical operator: AND.
|	Logical operator: OR.
~	Logical operator: NOT.
xor	Logical operator: EXCLUSIVE OR.

جدول ۷: عملگر های منطقی و مقایسه ای

علاوه بر عملگر های فوق دسته ای از توابع منطقی نیز وجود دارد که نتیجه تاثیر آن ها بر آرایه ها هم ارزش منطقی دارد.

Logical Functions	
any	True if any elements are nonzero.
all	True if all elements are nonzero.
find	Finds indices of nonzero elements
finite	True if elements are finite.
isnan	True if elements are undefined.
isinf	True if elements are infinite.
isempty	True if matrix is empty.
isreal	True if all elements are real.

جدول ۸: تابع منطقی

خروجی دستورات فوق را روی بردارها و ماتریس ها امتحان کنید به تفاوت های آن دقت کنید. دسته آخر توابع متناظر با مفاهیم خاص ریاضی است که در حل دستگاه معادلات خطی کاربرد دارد.

Matrix Commands for Solving Linear Equations	
det	Computes determinant of an array.
inv	Computes inverse of a matrix.
pinv	Computes pseudoinverse of a matrix.
rank	Computes rank of a matrix.
rref	Computes reduced row echelon form.

special Matrix Commends	
flipdim	**Flip array along specified dimension**
fliplr	**Flip elements from left to right**
flipud	**Flip elements from up to down**
rot90	**Rotate matrix 90 degrees**
diag	**Return the diagonal elements**
tril	**Return corresponding lower triangular matrix**
triu	**Return corresponding upper triangular matrix**
magic	**Return the Magic matrix**
permute	**Rearrange dimensions of N-D array**

توابع مذکور در آخرین جدول ، دستورات چرخشی در متلب هستند. دستورات مثلثی و دستور diag امکان دسترسی به عناصر قطری را فراهم می‌کنند.

علی پور ؛ علی داوری

اهداف و تمرین های فصل :

۱. قطعه برنامه ای بنویسید که ماتریس ضرایب و بردار معلوم یک دستگاه معادلات خطی را از ورودی بخواند و جواب سیستم را در خروجی چاپ کند. (هدف این تمرین کار با ابزار ورودی خروجی متلب و استفاده از اعمال مربوط به ماتریس هاست .)

۲. دستوری بنویسید که ماتریس زیر را تولید کند.(کاربرد ماتریس های خاص در متلب)

$$A=\begin{bmatrix}-10 & 3 & 3 & 3 & 3\\ 3 & -10 & 3 & 3 & 3\\ 3 & 3 & -10 & 3 & 3\end{bmatrix}$$

۳. دستوری بنویسید که قطر اصلی ماتریس زیر را همگی برابر ۱ قرار دهد.(کاربرد Matrix commonds)

$$A=\begin{bmatrix}-1 & 4 & 0 & 0 & 2\\ 5 & 1 & -2 & 0 & 1\\ 0 & 4 & 7 & -7 & 2\\ 1 & -1 & 0 & 4 & 7\end{bmatrix}$$

۴. در خصوص ماتریس فوق مجموعه از دستورات بنویسید که شماره سطر و ستون دلخواه از کاربر را بگیرد و سطر وستون مربوطه را حذف کند.(تمرین کنترل آرایه ها)

۵. فرض کنید دو بردار x, y مفروض اند. هرگاه $(x_i,y_i)\neq(0,0)$ در این صورت $z_i=\frac{x_i*y_i}{x_i^{\,2}+y_i^{\,2}}$ در غیر این صورت $z_i=0$. قطعه برنامه ای بنویسید که بردار z را تولید کند.(کاربرد آرایه ماسک)

lafzalipour@hotmail.com

l.afzalipour@gmail.com

تمرین های پیشرفته و کاربردی

۱. داده های جدولی زیر مفروض اند.

x_i	-1	0	1	2	3
f_i	-4	1	6	10	20

ضرایب a_i را چنان تعیین کنید که

$$p(x_i) = f_i \quad i = 0,...n \qquad , p(x) = \sum_{i=0}^{n} a_i x^i \quad .$$

۲. برای هر جفت (u, v) که $u=1,...,m-1$ و $v=u+1,...,m$، w_j (مجهول) وزن یا سهم ویژگی‌های مشتریان در تفکیک گروه‌ها از یکدیگر، $j=1,...,q$ متغیر آزاد در علامت است . x_{ij} (معلوم) مقدار ویژگی jام برای مشتری iام و $d_i>=0$ **(مجهول)** برای $i \in Gu$ و $j \in Gv$.

۳.

$$\sum_{j=1}^{q} w_j (x_{ij} - med_j(u)) + d_i \geq 0, \quad \forall i \in G_u$$

$$\sum_{j=1}^{q} w_j (x_{ij} - med_j(v)) - d_i \leq 0, \quad \forall i \in G_v$$

$$\sum_{j=1}^{q} w_j (med_j(u) - med_j(k)) \geq 1,$$

- معادل سیستم خطی $AX \leq b$ برای روابط زیر بازنویسی کنید.
- کد Matlab برای ساختن ماتریس های A ,b کدامند .

تنظیم: لعیا افضلی پور؛ علی داوری

۴. پیرو مسئله قبل معادلات زیر مفروض اند :

$$S_{iuv} + d_{iuv} \geq C_{uv} \text{ , for } u = 1,...,m-1,\ \ v = u+1,...,m,\ \ i \in G_u$$
$$S_{iuv} - d_{iuv} \leq C_{uv} \text{ , for } u = 1,...,m-1,\ \ v = u+1,...,m,\ \ i \in G_v$$

که C_{uv} مجهول و آزاد در علامتند و همچنین $d_{iuv} \geq 0$. و (S_i) از رابطه زیر به دست می‌آید:

$$S_i = \sum_{j=1}^{q} w_j x_{ij} \text{ , for } i \in G_k$$

- معادل سیستم خطی $AX \leq b$ برای روابط زیر بازنویسی کنید.
- کد Matlab برای ساختن ماتریس های A ,b کدامند .

تنظیم: لعیا افضلی پور ؛ علی داوری

تنظیم: لعیا لطفی پور و علی داوری

فصل سوم

ترسیم در MATLAB

در این فصل به معرفی ابزار ترسیم دوبعدی، قطبی، سه بعدی (اعم از خم های فضایی ورویه ها و خم های تراز) پرداخته می شود.

ترسیم دوبعدی

- رسم توابع $y = f(x)$:

ابتدا یک افراز مناسب برای x (متغیر مستقل) تعریف کنید، سپس با استفاده از عملیات آرایه ای y (متغیر وابسته) را برحسب x تعریف کنید. اکنون تابع plot رسم را انجام می‌دهد. در لیست آرگومان های ورودی تابع plot ، ابتدا نام متغیر مستقل و سپس نام متغیر وابسته ذکر می شود.

```
% creating mesh (partition )
x=0:1:10;
y=x.^2-10.*x+15;

% commond ofploting
figure(1)
plot(x,y)

title('plot of y=x^2-10*x+15')
xlabel('X')
ylabel('Y')

% commond network
grid on
```

فرم کلی‌تر دستور plot به صورت

```
plot(x,y1,' نماد رسم کاراکتر رنگ' )
```

لیست کاراکترها ونماد های مجاز در جدول ۱ ضمیمه است.

Colors, Symbols and Line Types					
Color		**Symbol**		**Line**	
y	yellow	.	point	-	solid
m	magenta	o	circle	:	dotted
c	cyan	x	x-mark	-.	dash dotted
r	red	+	plus	--	dashed
g	green	*	star		
b	blue	d	diamond		
w	white	v	triangle (down)		
k	black	^	triangle (up)		
		<	triangle (left)		
		>	triangle (right)		
		p	pentagram		
		h	hexagram		

جدول ۱: لیست کاراکترهاو نمادهای مجاز در دستور plot

- ترسیم قطبی $r = f(\theta)$:

برای رسم توابع قطبی نیز هم، ابتدا یک افراز مناسب برای θ (متغیر مستقل) تعریف کنید، سپس با استفاده از عملیات آرایه ای r (متغیر وابسته) را برحسب θ تعریف کنید. اکنون تابع polar رسم را انجام می‌دهد. در لیست آرگومان های ورودی تابع plot ، ابتدا نام متغیر مستقل و سپس نام متغیر وابسته ذکر می شود.

```
figure(4)
t=0:.1:2*pi;
r=1+cos(t);
r2=1+sin(5*t);
polar(t,r,'g-*')
hold on
polar(t,r2,'m-h')
legend('1+cos(t)','1+sin(5t)',4)
hold on
polar(t,-r,'b-p')
```

رسم همزمان چند تابع:

- استفاده از دستور **hold on** : پس از اتمام مراحل رسم اولین نمودار کافی است دستور hold on فراخوانی شود و سپس دستورات رسم نمودار بعدی اجرا می‌شود. برای رسم نمودارهای بعدی این روند تکرار می‌شود.
- دستور **plot** توام : در این نحوه فراخوانی فرم دستور plot چنین است :

```
 x=0:pi/100:2*pi;
y1=sin(2*x);
y2=2*cos(2*x);
plot(x,y1,'gd',x,y2,'m-p')
```

در این نحوه فراخوانی اول لیست متغیر مستقل و وابسته و رنگ ونوع خط برای نمودار اول مشخص می‌شود و سپس همین لیست برای نمودارهای بعدی ذکر می‌شود.

- استفاده از **subplot**: دستور subplot هر پنجره رسم (Figure) را به چند زیر بخش رسم (subplot) تقسیم می‌کند، سپس هر نمودار را در یک زیر بخش رسم می‌کند.

```
 x=0:pi/100:2*pi;
y1=sin(2*x);
```

```
y2=2*cos(2*x);
figure(4)
subplot(2,3,1)
plot(x,y1,'yo')
title('y=sin(2*x)')
subplot(2,3,2)
plot(x,y2,'m',x,2*sin(2*x),'g-.')
title('y=2*cos(2*x)& y=2*sin(2*x)')
legend('2cos(2x)','2sin(2x)',-1)
```

دستورات جانبی در رسم :

دوتا از دستورات جانبی در مثال فوق مشهود اند. به کمک دستور title(‘string’) یک عنوان به نمودار رسم شده اختصاص داده می‌شود. با استفاده از دستور legend(‘string1’,’string2’,…,position) می‌توان جزئیات مربوط به نمودارها را مصور کرد. مقادیر مجاز پارامتر position در جدول ۲ ضمیمه است. فرم دیگر این دستور legend(‘string1’,’string2’,…,’Location’,’Lstring’) است. مقادیر مجاز Lstring در جدول ۳ ضمیمه است.

دستور xlable(‘string’) عمل برچسب گذاری بر محور x و ylable(‘string’) عمل برچسب گذاری بر محور y را انجام می‌دهد. دستور text(x,y,'string’) بر نقطه (x,y) از نمودار ترسیم شده، محتوای رشته string را چاپ می‌کند. به نحوه کاربرد این دستورات در اجرا m– فایل با نام ploting_test دقت کنید.

Obsolete Specifier	Location in Axes
-1	Outside axes on right side
0	Inside axes
1	Upper right corner of axes
2	Upper left corner of axes
3	Lower left corner of axes
4	Lower right corner of axes

جدول ۲ : مقادیر مجاز position در دستور legend

تنظیم : لیلا افضلی پور ؛ علی داوری

Specifier	Location in Axes
North	Inside plot box near top
South	Inside bottom
East	Inside right
West	Inside left
NorthEast	Inside top right (default for 2-D plots)
NorthWest	Inside top left
SouthEast	Inside bottom right
SouthWest	Inside bottom left
NorthOutside	Outside plot box near top
SouthOutside	Outside bottom
EastOutside	Outside right
WestOutside	Outside left
NorthEastOutside	Outside top right (default for 3-D plots)
NorthWestOutside	Outside top left
SouthEastOutside	Outside bottom right
SouthWestOutside	Outside bottom left
Best	Least conflict with data in plot
BestOutside	Least unused space outside plot

جدول ۳: مقادیر مجاز Location در دستور legend

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

جدول ۴ : علایم ریاضی و یونانی در کنترل رشته های متن

نحوه استفاده موردی این علایم در m– فایل ploting_test در خط ۲۷ ارائه شده است.

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

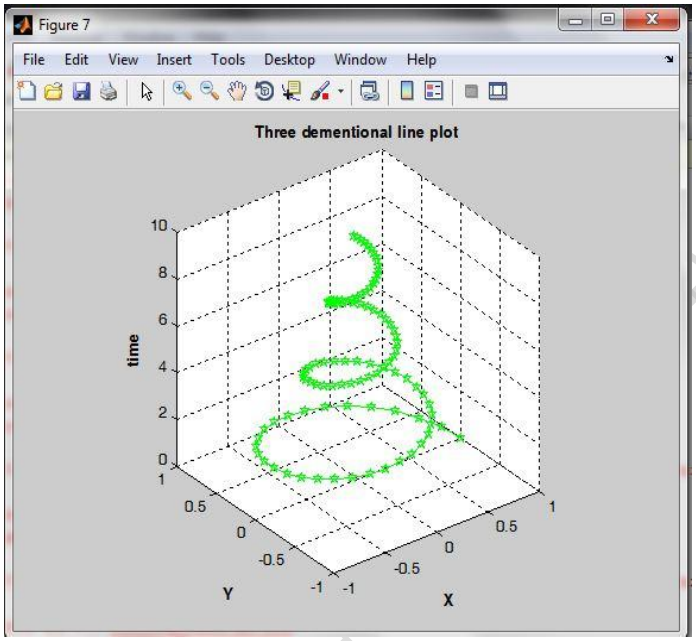
ادامه جدول ۴

ترسیم سه بعدی:

- خم سه بعدی :

```
% 3D curve
figure(7)
t=0:.1:10;
x=exp(-.2*t).*cos(2*t);
y=exp(-.2*t).*sin(2*t);
plot3(x,y,t,'g-p')
title('\bfThree dementional line plot')
xlabel('\bfX')
ylabel('\bfY')
```

```
zlabel('\bftime')
axis square
grid on
```



شکل۱: یک خم سه بعدی

- رویه ها: رویه‌ها را به دوصورت surface یا mesh رسم کرد. فرض‌کنید $z(x,y)=e^{-.5(x^2+.5(x-y)^2)}$, $-4\le x\le 4, -4\le y\le 4$ برای ایجاد یک منحنی mesh از این تابع ریاضی، دستورات زیر اجرا می‌شود.

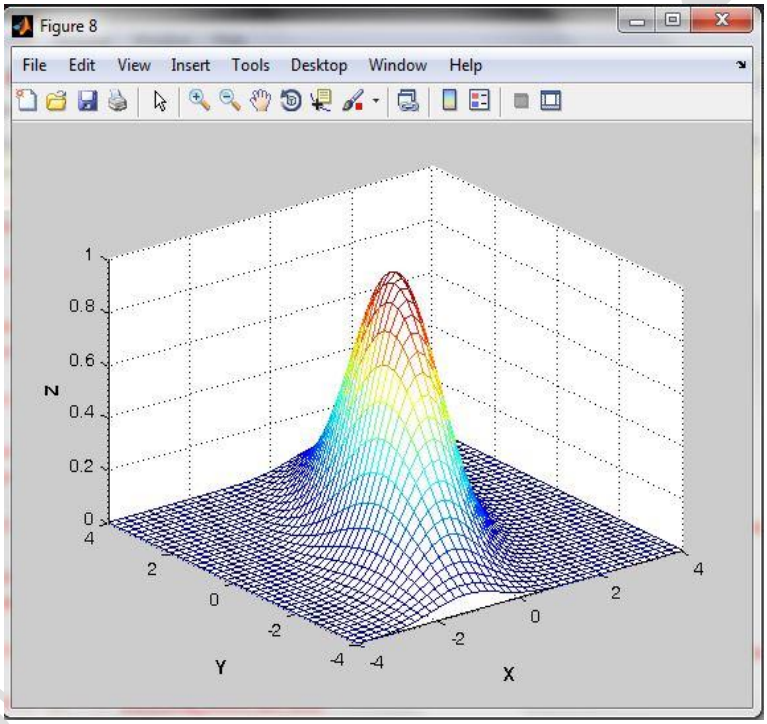
```
% Mesh ploting
figure(8)
xstart=-4;
xend=4;
xinc=.2;

ystart=-4;
yend=4;
yinc=.2;

[x,y]=meshgrid(xstart:xinc:xend,ystart:yinc:yend);
```

```
z=exp(-.5*(x.^2+.5*(x-y).^2));
mesh(x,y,z)
xlabel('\bfX')
ylabel('\bfY')
zlabel('\bfZ')
```

ابتدا با دستور meshgrid یک شبکه برای متغیرهای x, y ایجاد می شود. سپس با استفاده از عملیات آرایه ای z را برحسب x, y تعریف می گردد. درانتها با فراخوانی دستور(mesh(x,y,z روند رسم تکمیل می شود.

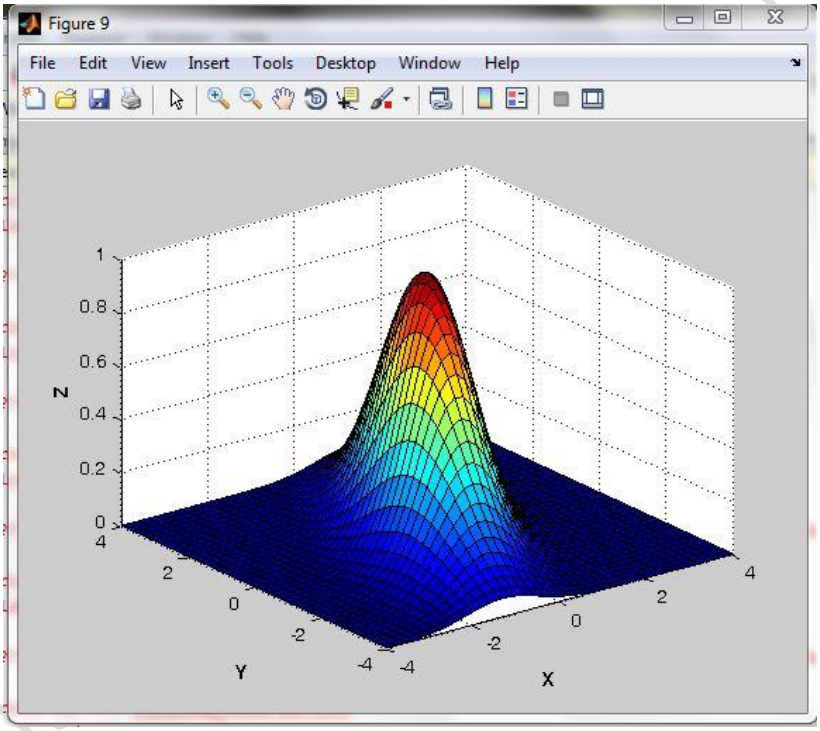


شکل۲ : یک mesh

نحوه کاربری دستور surf برای رسم رویه‌ها مشابه دستور mesh است.

```
% Surface ploting
figure(9)
[x,y]=meshgrid(xstart:xinc:xend,ystart:yinc:yend);
z=exp(-.5*(x.^2+.5*(x-y).^2));

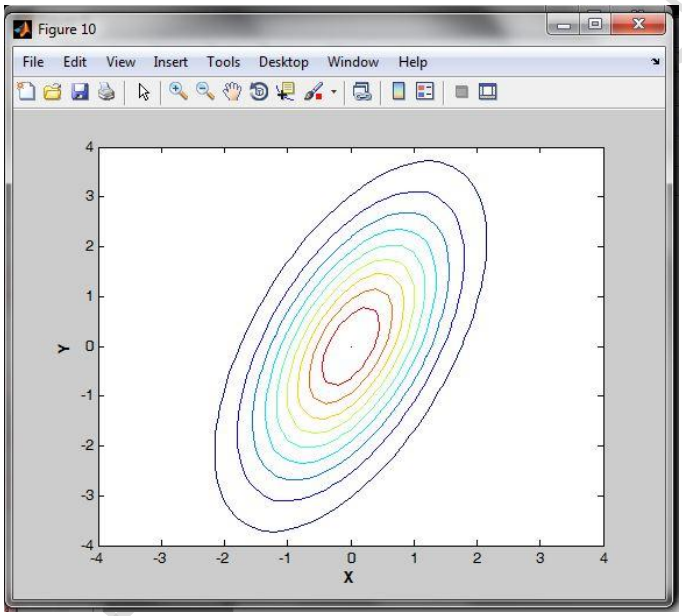
surf(x,y,z)
xlabel('\bfX')
ylabel('\bfY')
zlabel('\bfZ')
```



شکل۳: یک surface

دستور contour برای ترسم سطح مقطع منحنی‌ها استفاده می‌شود.

```
% countour ploting
figure(10)
[x,y]=meshgrid(xstart:xinc:xend,ystart:yinc:yend);
z=exp(-.5*(x.^2+.5*(x-y).^2));
contour(x,y,z)
xlabel('\bfX')
ylabel('\bfY')
zlabel('\bfZ')
```



شکل ۴: یک contour

رشته ها :

رشته در Matlab یک آرایه از نوع char است. هر کاراکتر در دو بایت از حافظه ذخیره می‌شود. لیستی از پرکاربردترین ابزارهای پردازش رشته‌ها در جدول ۵ ضمیمه است.

Commands for Managing Strings	
$ischar(a)$	Returns l if a is a character array and 0 otherwise
$double(s)$	Returns an array the same size as str containing 1 where the elements of s are letters of the alphabet and 0 where they are not
$char(X)$	converts array X of nonnegative integer codes into a character array
$strcat(s_1, s_2)$	Output can be combinations of single s_1, s_2
$strvcat(s_1, s_2)$	It is same as strcat, but in vertical direction
$strcmp(s_1, s_2)$	compares the strings s_1 and s_1 and returns 1 if they are identical, and returns 0 otherwise
$strcmpi(s_1, s_2)$	It is same as strcmp, but not case sensetive
$strncmp(s_1, s_2, n)$	compares the first n characters of strings s_1 and s_2 and returns 1if they are identical, and returns 0 otherwise
$strncmpi(s_1, s_2, n)$	It is same as strncmp, but not case sensetive
$isletter(s)$	returns an array the same size as str containing 1 where the elements of str are letters of the alphabet and 0 where they are not
$isspace(s)$	returns an array the same size as 'str' containing 1 where the elements of s are whitespace characters, and 0 where they are not
$findstr(s, pat)$	searches the s for occurrences of a shorter string, pat, and returns the starting index of each such occurrence in the double array
$strrep(s, pat1, pat2)$	replaces all occurrences of the string pat1 within string s with the string pat2
$[tok, re] = strtok(s)$	parses input string s from left to right, returning part or all of that string in token. Using the white-space character as a delimiter,

upper(*s*)	converts any lowercase characters in the string str to the corresponding uppercase characters to upper case
lower(*s*)	returns the string formed by converting any uppercase characters to lowercase characters
*num*2*str*(*x*, *n*)	converts numbers to their string representations
int2*str*(*x*)	converts integers to their string representations
*mat*2*str*(*A*)	converts matrix A into a string
eval(*s*)	executes s
blanks(*n*)	is a string of n blanks
deblank(*s*)	removes all trailing whitespace and null characters from the end of character string s

جدول ۵ : ابزار پردازش رشته‌ها

فاضلی پور ؛ علی داوری

تمرین‌ها و اهداف فصل :

۱. توابع زیر را رسم کنید.(هدف تمرین: تشخیص دستور مناسب رسم، استفاده از تکنینک مناسب رسم همزمان نمودار ها وکنترل نوع خط و رنگ در رسم)

رنگ	ضابطه	
قرمز	e^x	Y=
سیاه	e^{x^2}	Y=
آبی	$\cos(2t)$	r =
زرد	$\sin(t^2)$	r =
ارغوانی	$x\sin(\frac{1}{x})$	Y=

۲. موارد خواسته شده را رسم کنید.

نوع	$y \in [c,d]$	$x \in [a,b]$	ضابطه
رویه	[-10,10]	[-10,10]	$z = -xye^{-x^2-y^2}$
خم تراز	[-20,10]	[-8,20]	$z = \frac{-3y}{x^2+y^2+1}$
هردو	[-π,π]	[-π,π]	$z = \frac{\sin x \sin y}{xy}$

تنظیم: لعیا افضلی پور و علی داوری

lafzalipour@hotmail.com

l.afzalipour@gmail.com

تنظیم: علی داودی

فصل چهارم

تعریف

توابع دست‌نویس(handled function)

و

توابع ریاضی نمادین(math function)

(symbolic

توابع دست‌نویس(handled function) :

منظور از تابع دست‌نویس، تابعی است که کاربران برحسب نیاز در موقعیت‌های متفاوت تعریف می‌کنند. هر تابع دست‌نویس ذاتاً یک m– فایل است، با این تفاوت که اولین سطر از این m– فایل با کلمه function شروع می‌شود و حتماً نامی که برای ذخیره کردن m– فایل استفاده می‌شود باید با نامی که به تابع اختصاص داده شده است یکسان باشد.

❖ به این نکته در هنگام تغییر نام توابعی که به صورت m– فایل از پیش ذخیره شده‌اند دقت کنید.

در حالت کلی اولین سطر m– فایل متناظر هر تابع با دستوری مشابه

```
function ['لیست آرگومان‌های خروجی']=نام_تابع('لیست پارامترهای ورودی')
```

شروع می‌شود. این در حالی است که لیست پارامترهای ورودی ویا لیست آرگومان‌های خروجی می‌توانند تهی باشد. به نظر می‌رسد که تعریف مجموعه ای از دستورات در قالب یک تابع که نه آرگومان خروجی و نه پارامتر ورودی داشته باشد چندان ضروری نیست، اما باید متذکر شد که این کار، خوانایی و خطایابی روال دستورات را بهبود می‌دهد و دستورات را در قالبی ساخت یافته ارائه می‌کند، لذا تعریف توابع بدون آرگومان خروجی و پارامتر ورودی کارآمد خواهد بود.

تنظیم: نعیم افضلی پور؛ علی داوری

مثال۱: تابعی بنویسید که مقدار رادیان را به عنوان پارامتر ورودی بگیرد و مقدار درجه متناظر رابرگرداند.

```
function z=Rad2digit(x)
% pi radian is equal 180(digitd)
% x radian is equal x*180/pi

z=x*180/pi;
```

در مثال فوق z آرگومان خروجی و x پارامتر ورودی و Rad2digit نام تابع است. برای اجرا این تابع، m- فایل متناظر آن با نام Rad2digit در پوشه جاری ذخیره می‌شود.

مثال ۲: تابعی بنویسید که وزن یک جسم را بر حسب کیلو ازطریق پارامتر ورودی بخواند و وزن را برحسب واحد پوند و تن برگرداند.

```
function [P,T]=Weight_Converter(x)
% 1 kilogrammes is equal 2.204 ponds
% 1 ton is equal 1000 kilogrammes
pounds=2.204;
ton=10^-3;
P=x*pounds;
T=x*ton
```

نحوه فراخوانی توابع دست‌نویس:

همان‌طور که در قسمت قبل ذکر شد m– فایل متناظر با تابع باید در پوشه جاری ذخیره شده باشد.

برای فرخوانی (چه در پنجره فرمان باشد یا در روال تابع و m– فایل دیگر) کافی است

```
['لیست آرگومان‌های خروجی']=نام_تابع('لیست پارامترهای ورودی')
```

فراخوانی شود.

❖ چنانچه تعداد ظرف‌های اختصاص داده شده به لیست آرگومان‌های خروجی کمتراز تعداد اصلی آرگومان‌های خروجی در تعریف تابع باشد، اطلاعات در ظرف‌ها موجود کپی می‌شود و بقیه اطلاعات که برای ذخیره شدن آن‌ها ظرفی در نظر گرفته نشده از بین می‌روند.

```
x=2.34;
[P,T]=Weight_Converter(x)
```

به خروجی اجرا m– فایل Weight_convertor دقت کنید.

تعریف تابع‌های جامع تر را بعد ارائه دستورات کنترلی در MATLAB طرح می‌شود.

متغیرها در حوزه توابع :

هرمتغیر که در یک تابع تعریف می‌شود صرفاً در حوزه ی آن تابع قابل دسترسی و بازیابی است. هنگامی که اجرای تابع به پایان می‌رسد فضای کاری ایجاد شده توسط تابع از بین می‌رود. چنانچه لازم باشدحجم زیادی از دادها در اکثر توابع قابل دسترسی باشد کافی است این متغیر را به صورت عمومی (global) تعریف کرد. شکل عبارت global به صورت زیر است:

```
global var1 var2 ....
```

هر متغیر عمومی، باید قبل از استفاده در تابع به صورت عمومی بیان شود. این اشتباه است که بعد از یک متغیر موضعی در فضای کاری متغیری را به‌صورت متغیر عمومی بیان کرد. برای اجتناب از این خطا، باید متغیر عمومی بلافاصله بعد از توضیح اولیه وقبل از اولین عبارت قابل اجرا بیان گردد.

نگهداری داده‌ها بین فراخوانی‌های یک تابع:

گاهی لازم است تا قسمتی از اطلاعات محلی یک تابع بین فراخوانی‌های تابع حفظ شود. برای مثال از یک شمارنده برای شمارش دفعات فراخوانی تابع استفاده شود. حافظه نگهدارنده (persistent memory) نوعی حافظه است که تنها بوسیله توابع مورد استفاده قرار می‌گیرد. این مقادیر در بین فراخوانی‌های ثابت می‌ماند. یک متغیر نگهدارنده با عبارت persistent بیان می‌شود. فرم این عبارت به صورت زیر است:

```
persistent var1 var2 ....
```

زیر تابع :

هرگاه روال تعریف تابعی در انتهای تابع دیگری قرار گیرد به تابع دوم زیر تابع گویند. در این حالت زیر تابع تعریف شده، صرفاً برای تابع اصلی یا سایر زیر توابع از تابع اصلی قابل فراخوانی و دسترسی است. چنانچه لازم است حوزه دسترسی به تابعی محدود شود بهتر است تابع مذکور به صورت زیر تابع تعریف شود.

❖ توجه داشته باشید که در حالت عادی هر تابع در موقعیت پوشه جاری قرار می‌گیرد.

توابع ریاضی نمادین(symbolic math function):

یک تابع ریاضی برحسب نمادها (سمبول) را می‌توان به‌کمک رشته‌ها یا عبارات نمادین تعریف کرد. هنگامی که یک تابع ریاضی به صورت رشته یا عبارت نمادین تعریف می‌شود نیازی به استفاده از عملگرهای آرایه‌ای نیست.

- رشته‌ها:

برای تعریف یک تابع ریاضی بااستفاده از رشته‌ها، باید عبارت ریاضی را در یک جفت گوتیشن قرار داد. تابع ریاضی ممکن است برحسب یک یا چند متغیر باشد.

```
f='x+sin(x^2)-1';
f2='sin(x+y)';
```

گاهی اوقات لازم است که تابع ریاضی در حین اجرا پروسه از ورودی خوانده شود، دراین صورت می توان از دستور input کمک گرفت. به نحوه فراخوانی دستور input در این خصوص دقت کنید.

```
g=input('! Please enter favour Math_function :','s');
```

- عبارات نمادین :

در این حالت ابتدا متغیرهای مسئله به صورت نماد معرفی می‌شوند، پس از آن با انتساب عبارت ریاضی به یک ظرف تابع تعریف می‌گردد.

```
function Math_function_sympolic

% symbolic experation
syms x y
f=x+sin(x^2)-1;
```

```
f2=sin(x+y);
g=input('! Please enter favour Math_function :');
```

به تفاوت فراخوانی دستور input برای خواندن تابع ریاضی از ورودی دقت کنید.

Commands for Managing Mathematical Function(string & symbolic)	
subs(f,a)	Replaces the default symbolic variable in f with a
diff(f)	Calculates differences elements of f
diff(f,n)	Calculates n-th differences elements of f
int(f)	$F = \int f dx$
int(f,a,b)	$\int_a^b f dx = F(b) - F(a) \qquad , \frac{\partial F}{\partial x} = f$
eval(z)	Return numerical value of z string
ezplot(f)	Plot the graph of f
simplify(f)	Simplifying f in a more readable form by entering
Solve(f)	Solving equation of f=0
limit(f, x, x0, 'left')	$\lim_{x \to x0^-} f$
limit(f, x, x0, 'right')	$\lim_{x \to x0^+} f$
limit(f, x, x0)	$\lim_{x \to x0} f$
symsum(f,a,b)	Compute symbolic summations

جدول ۱: توابع کاربردی در کنترل توابع ریاضی که به صورت نمادین یا رشته تعریف شده‌اند

نمونه‌ای از توابع فوق در m –فایل Math_function_symbolic ارائه شده است.

توابع ریاضی را به صورت تابع دست‌نویس نیز می‌توان تعریف کرد. در این حالت m– فایل متناظر با تابع صرفاً شامل ضابطه عبارت ریاضی است.

```
function y=f(x)
y=x.^2+3*sin(x);
```

❖ هنگامی‌که تابع ریاضی به صورت تابعی دست‌نویس تعریف می‌شود، ضابطه تابع ریاضی باید برحسب عملگرهای آرایه‌ای نوشته شود.

Commands for Managing Mathematical Function(handled function)	
f(a)	Replaces the default symbolic variable in f with a
$quad(fun,a,b)$	$\int_a^b fdx$, Use adaptive Simpson quadrature
$quadgk(fun,a,b)$	$\int_a^b fdx$, Use adaptive Gauss-Kronrod quadrature
dblquad(fun, xmin, xmax, ymin, ymax)	Numerically evaluate double integral
Numerically evaluate triple integral	triplequad(fun,xmin,xmax,ymin,ymax,zmin,zmax)
fsolve(fun, x0)	Solving equation of f=0
fzero(fun, x0)	Solving equation of f=0
fminbnd(fun,x1,x2)	Finding the minimum of a function on $[x_1, x_2]$ interval
fminsearch(fun,x0)	Finding the minimum of a function which is closed to x0

جدول ۲: توابع کاربردی در کنترل توابع ریاضی که به صورت تابع دست‌نویس تعریف شده‌اند

مثال۳: حجم ناحیه واقع در یک‌هشتم اول که به صفحات مختصات واز بالا به استوانه $x^2+z=1$ و از راست به سهمیوار $y=x^2+z^2$ محدود است، بدست آورید.

حل : جواب مسئله جواب انتگرال سه‌گانه $\int_0^1 \int_0^{\sqrt{1-z}} \int_0^{x^2+z^2} dydxdz$ است. برای محاسبه انتگرال مذکور دستورات زیر را در نظر بگیرید.

```
function integration_test

syms x y z

q=int(int(int(1,0,x^2+z^2),0,sqrt(1-z)),0,1)
```

تنظیم: لعیا افضلی پور؛ علی داوری

- **یک کلاس خاص از توابع ریاضی(چند جمله‌ها) :**

عبارت $p(x)=a_n x^n+a_{n-1}x^{n-1}+...+a_1x+a_0$ یک چند جمله‌ای بر حسب x است. به ترتیب به ضرایب a_n و a_0 ضریب پیشرو جمله ثابت چندجمله‌ای گفته می‌شود. یک چندجمله‌ای در Matlab بوسیله یک بردار نمایش داده می‌شود. این بردار به ترتیب شامل ضرایب a_n تا a_0 است.

Commands for Managing Polynomials	
$conv(p,q)$	Convolution and polynomial multiplication
$deconv(p_1,p_2)$	Deconvolution and polynomial division
$poly(R)$	Polynomial with specified roots
$polyder(p)$	Polynomial derivative
$poly\ \mathrm{int}(p)$	Integrate polynomial analytically
$polyval(p,k)$	Polynomial evaluation
$roots(p)$	Polynomial roots
$polyvalm(p,k)$	Matrix polynomial evaluation
$\mathrm{polyeig}(A_0,A_1,...A_p)$	Solves the polynomial eigenvalue problem of degree p $(A_0+\lambda A_1+...+\lambda^p A_p)x=0$

جدول۳ : توابع کاربردی در کنترل چندجمله‌ای‌ها

تمرین‌ها و اهداف فصل :

۱. قطعه برنامه‌ای بنویسید که یک تابع ریاضی را از ورودی خوانده، ریشه‌های آن را محاسبه نماید. رفتار تابع را حول ریشه های ان از طریق رسم بررسی کنید.

۲. شرایط تمرین۱را در نظر بگیرید، ماکزیمم تابع را بدست آورید.(راهمنایی: ریشه‌های معادله تابع مشتق، اکسترمم‌های تابع مورد نظر هستند.)

۳. حجم ناحیه T محدود به : داخل پارچه‌های مخروط $x^2+y^2-z^2=0$ و بین کرات $x^2+y^2+z^2=16$, $x^2+y^2+z^2=1$ قرار دارد. (جواب مسئله معادل انتگرال سه‌گانه $\int_0^{2\pi}\int_0^{\frac{\pi}{4}}\int_1^4 \rho^2 \sin\varphi \, d\rho d\varphi d\theta$)

۴. مقدار حدود زیر را محاسبه کنید:

$$\lim_{x\to 0}\frac{\sqrt[3]{x+a^3}-a}{x} \quad (a\neq 0) \qquad \lim_{x\to 2}\frac{\sin(x^2-4)}{x-2}$$

lafzalipour@hotmail.com

l.afzalipour@gmail.com

تنظیم: لعیا افضلی پور ؛ علی داوری

مباحث تکمیلی بر فصل سوم و چهارم

- رسم توابع ضمنی: بهترین راه برای رسم توابع که به صورت ضمنی تعریف می‌شوند استفاده از دستور ezplot است. البته در خصوص رسم سه‌بعدی باید از مجموعه دستورات تعمیم یافته زیر استفاده کرد.

```
function h = ezimplot3(varargin)
% EZIMPLOT3    Easy to use 3D implicit plotter.
%   EZIMPLOT3(FUN) plots the function FUN(X,Y,Z) = 0 (vectorized or not)
%   over the default domain:
%   -2*PI < X < 2*PI, -2*PI < Y < 2*PI, -2*PI < Z < 2*PI.
%   FUN can be a string, an anonymous function handle, a .M-file handle, an
%   inline function or a symbolic function (see examples below)
%
%   EZIMPLOT3(FUN,DOMAIN)plots FUN over the specified DOMAIN instead of the
%   default domain. DOMAIN can be vector [XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX] or
%   vector [A,B] (to plot over A < X < B, A < Y < B, A < Z < B).
%
%   EZIMPLOT3(..,N) plots FUN using an N-by-N grid. The default value for
%   N is 60.
%   EZIMPLOT3(..,'color') plots FUN with color 'color'. The default value
%   for 'color' is 'red'. 'color' must be a valid Matlab color identifier.
%
%   EZIMPLOT3(axes_handle,..) plots into the axes with handle axes_handle
%   instead of into current axes (gca).
%
%   H = EZIMPLOT3(...) returns the handle to the patch object this function
%   creates.
%
% Example:
% Plot x^3+exp(y)-cosh(z)=4, between -5 and 5 for x,y and z
%
%   via a string:
% f = 'x^3+exp(y)-cosh(z)-4'
% ezimplot3(f,[-5 5])
%
%   via a anonymous function handle:
% f = @(x,y,z) x^3+exp(y)-cosh(z)-4
% ezimplot3(f,[-5 5])
%
%   via a function .m file:
%------------------------------%
```

```
% function out = myfun(x,y,z)
% out = x^3+exp(y)-cosh(z)-4;
%------------------------------%
% ezimplot3(@myfun,[-5 5])   or  ezimplot('myfun',[-5 5])
%
%   via a inline function:
% f = inline('x^3+exp(y)-cosh(z)-4')
% ezimplot3(f,[-5 5])
%
%   via a symbolic expression:
% syms x y z
% f = x^3+exp(y)-cosh(z)-4
% ezimplot3(f,[-5 5])
% Note: this function do not use the "ezgraph3" standard, like ezsurf,
% ezmesh, etc, does. Because of this, ezimplot3 only tries to imitate that
% interface. A future work must be to modify "ezgraph3" to include a
% routine for implicit surfaces based on this file
%
%   Inspired by works of:   Artur Jutan   UWO 02-02-98 ajutan@julian.uwo.ca
%   Made by:            Gustavo Morales   UC  04-12-09 gmorales@uc.edu.ve
%
%%% Checking & Parsing input arguments:
if ishandle(varargin{1})
   cax = varargin{1}; % User selected axes handle for graphics
   axes(cax);
   args{:} = varargin{2:end}; %ensuring args be a cell array
else
   args = varargin;
end
[fun domain n color] = argcheck(args{:});
%%% Generating the volumetric domain data:
xm = linspace(domain(1),domain(2),n);
ym = linspace(domain(3),domain(4),n);
zm = linspace(domain(5),domain(6),n);
[x,y,z] = meshgrid(xm,ym,zm);
%%% Formatting "fun"
[f_handle f_text] = fix_fun(fun); % f_handle is the anonymous f-handle for
"fun"
                                  % f_text is "fun" ready to be a title
%%% Evaluating "f_handle" in domain:
try
  fvalues = f_handle(x,y,z);        % fvalues: volume data
catch ME
  error('Ezimplot3:Functions', 'FUN must have no more than 3 arguments');
end
%%% Making the 3D graph of the 0-level surface of the 4D function "fun":
h = patch(isosurface(x,y,z,fvalues,0)); % "patch" handles the structure...
                                        % sent by "isosurface"
isonormals(x,y,z,fvalues,h)% Recalculating the isosurface normals based...
```

```
                            % on the volume data
set(h,'FaceColor',color,'EdgeColor','none');
%%% Aditional graphic details:
xlabel('x');ylabel('y');zlabel('z');% naming the axis
alpha(0.7) % adjusting for some transparency
grid on; view([1,1,1]); axis equal; camlight; lighting gouraud
%%% Showing title:
title([f_text,' = 0']);
%-------------------------------------------Sub-functions HERE---
function [f dom n color] = argcheck(varargin)
%ARGCHECK(arg) parses "args" to the variables "f"(function),"dom"(domain)
%,"n"(grid size) and "c"(color)and TRIES to check its validity
switch nargin
       case 0
            error('Ezimplot3:Arguments',...
            'At least "fun" argument must be given');
       case 1
            f = varargin{1};
            dom = [-2*pi, 2*pi]; % default domain: -2*pi < xi < 2*pi
            n = 60;              % default grid size
            color = 'red';       % default graph color
       case 2
            f = varargin{1};
            if isa(varargin{2},'double') && length(varargin{2})>1
               dom = varargin{2};
               n = 60;
               color = 'red';
            elseif isa(varargin{2},'double') && length(varargin{2})==1
               n = varargin{2};
               dom = [-2*pi, 2*pi];
               color = 'red';
            elseif isa(varargin{2},'char')
               dom = [-2*pi, 2*pi];
               n = 60;
               color = varargin{2};
            end
       case 3                   % If more than 2 arguments are given, it's
            f = varargin{1};    % assumed they are in the correct order
            dom = varargin{2};
            n = varargin{3};
            color = 'red';      % default color
       case 4                   % If more than 2 arguments are given, it's
            f = varargin{1};    % assumed they are in the correct order
            dom = varargin{2};
            n = varargin{3};
            color = varargin{4};
       otherwise
            warning('Ezimplot3:Arguments', ...
            'Attempt will be made only with the 4 first arguments');
```

```
            f = varargin{1};
            dom = varargin{2};
            n = varargin{3};
            color = varargin{4};
end
if  length(dom) == 2
    dom = repmat(dom,1,3);        %domain repeated in all variables
elseif length(dom) ~= 6
    error('Ezimplot3:Arguments',...
    'Input argument "domain" must be a row vector of size 2 or size 6');
end
%-------------------------------------------
function [f_hand f_text] = fix_fun(fun)
% FIX_FUN(fun) Converts "fun" into an anonymous function of 3 variables
(x,y,z)
% with handle "f_hand" and a string "f_text" to use it as title
types = {'char','sym','function_handle','inline'}; % cell array of 'types'
type = ''; %Identifing FUN object class
for i=1:size(types,2)
    if isa(fun,types{i})
       type = types{i};
       break;
    end
end
switch type
       case 'char' % Formatting FUN if it is char type. There's 2
possibilities:
       % A string with the name of the .m file
            if exist([fun,'.m'],'file')
               syms x y z;
               if nargin(str2func(fun)) == 3
                  f_sym = eval([fun,'(x,y,z)']); % evaluating FUN at the sym
point (x,y,z)
               else
                  error('Ezimplot3:Arguments',...
                  '%s must be a function of 3 arguments or unknown
function',fun);
               end
               f_text = strrep(char(f_sym),' ',''); % converting to char and
eliminating spaces
               f_hand = eval(['@(x,y,z)',vectorize(f_text),';']); %
converting string to anonymous f_handle
            else
       % A string with the function's expression
               f_hand = eval(['@(x,y,z)',vectorize(fun),';']); % converting
string to anonymous f_handle
               f_text = strrep(fun,'.',''); f_text = strrep(f_text,' ',''); %
removing vectorization & spaces
            end
```

```
        case 'sym' % Formatting FUN if it is a symbolic object
             f_hand = eval(['@(x,y,z)',vectorize(fun),';']); % converting
string to anonymous f_handle
             f_text = strrep(char(fun),' ',''); % removing spaces
        case {'function_handle', 'inline'} % Formatting FUN if it is a
function_handle or an inline object
             syms x y z;
             if nargin(fun) == 3 && numel(symvar(char(fun))) == 3 %
Determining if # variables == 3
                f_sym = fun(x,y,z);   % evaluating FUN at the sym point
(x,y,z)
             else
                error('Ezimplot3:Arguments',...
                '%s must be function of 3 arguments or unknown
function',char(fun));
             end
             f_text = strrep(char(f_sym),' ','');  % converting into string to
removing spaces
             f_hand = eval(['@(x,y,z)',vectorize(f_text),';']); % converting
string to anonymous f_handle
    otherwise
             error('First argument "fun" must be of type character, simbolic,
function handle or inline');
end
```

مثال : به نحوه استفاده دستورات ezplot و ezimplot3 توجه کنید.

```
function ezplot_2d_3d

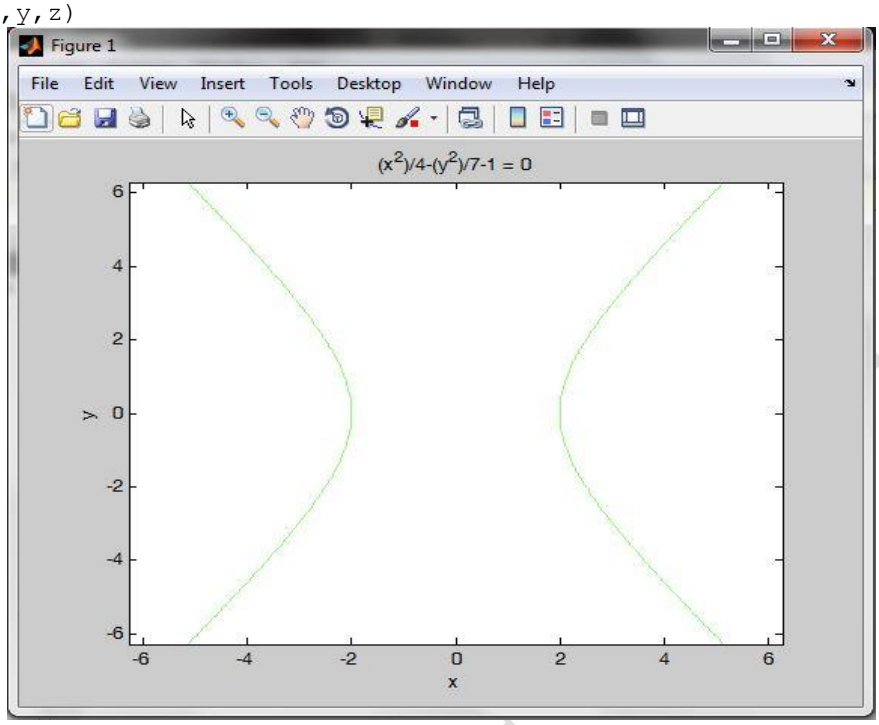
%implicit ploting 2D
f='(x^2)/4-(y^2)/7-1';
ezplot(f)

%implicit ploting 3D
figure(2)
f2='-(x^2/9)-(y^2/16)+(z^2/25)-1';
 ezimplot3(f2,'c')

 % parametric ploting
 figure(3)
 theta=linspace(0,2*pi,200);
 phi=linspace(0,pi,200)';
 roph=9;
 x=roph*sin(phi)*cos(theta);
```

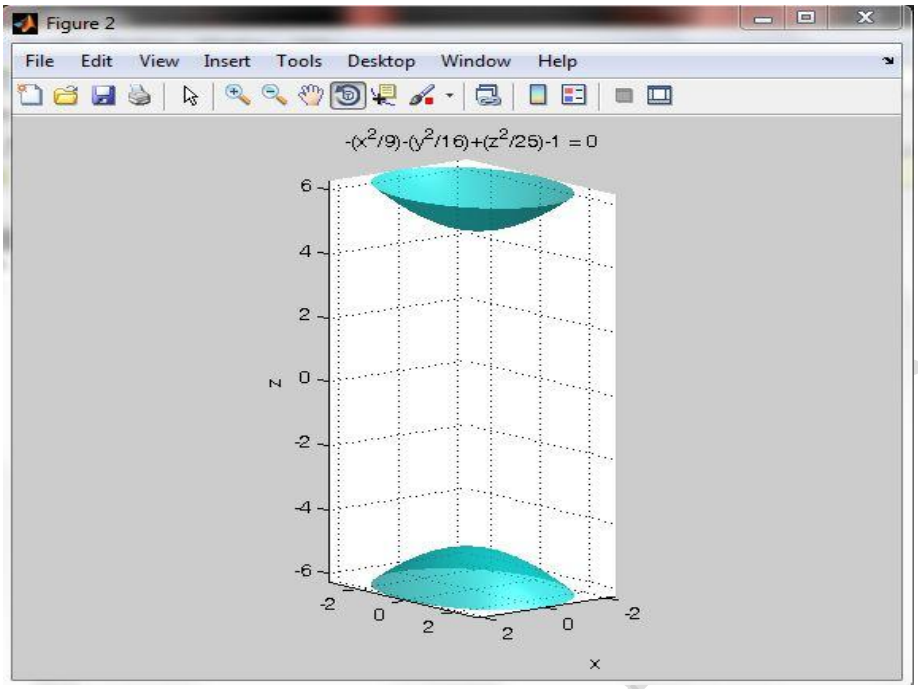
تنظیم: لعیا افضلی پور و علی داودی

```
y=roph*sin(phi)*sin(theta);
z=roph*cos(phi)*ones(size(theta));
surf(x,y,z)
```

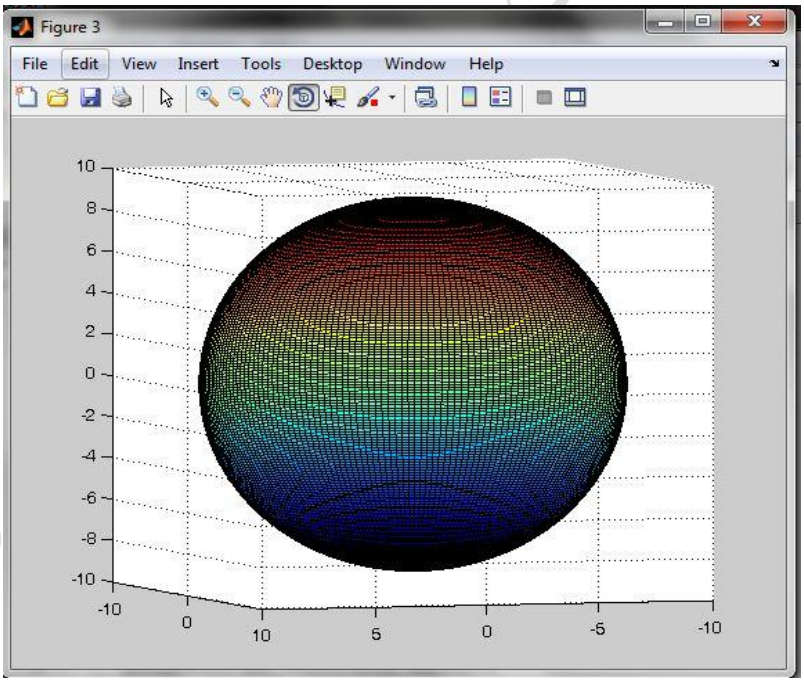


شکل ۱: ترسیم دوبعدی یک تابع ضمنی

تنظیم: پویا علی داوری



شکل ۲: ترسیم سه بعدی یک تابع ضمنی



تنظیم: لعیا افضلی پور؛ علی داوری

lafzalipour@hotmail.com

l.afzalipour@gmail.com

تنظیم: لعیا افضلی پور؛ علی داوری

فصل پنجم

کنترل (برنامه نویسی) در Matlab

ابزار برنامه در Matlab حلقه‌های تکرار و ساختارهای شرطی و دستورات انشعابی هستند.

حلقه‌های تکرار : حلقه‌های تکرار ساختارهایی هستندکه امکان بیش از یک بار اجرا شدن مجموعه‌ای ازعبارات را ممکن می‌کنند. حلقه‌ها دارای دو فرم اصلی‌اند: حلقه‌های while و حلقه‌های for. تفاوت اصلی بین این دو نوع حلقه در نحوه کنترل حلقه است. حلقه while تا زمان برقراری شرایط تعریف شده توسط کاربر اجرا می‌شود، در حالی که حلقه for به تعداد دفعات تعریف شده در ابتدای حلقه اجرا می‌شود.

- **حلقه while**:

حلقه while یک بلوک از عباراتی می‌باشد که تا زمان برقرار بودن شرط، تکرار می‌شود. فرم کلی حلقه به صورت زیر است:

```
while   expression
  …
  …   }code block
  …
end
```

تا زمانیکه expression ، ۱ (درست) باشد عبارات در بلوک اجرا می‌شود. هنگامی که کنترل به عبارت while باز می‌گردد و expression صفر شود برنامه اولین دستور بعد از end را اجرا خواهد کرد.

- **حلقه for**:

حلقه for حلقه‌ای است که یک بلوک از عبارات را به تعداد مشخصی انجام می‌دهد و فرم آن به صورت زیر است:

```
for   index = expr : step : last
  …
  …   } code block
  …
end
```

مثال: تابعی بنویسید که فاکتوریل یک عدد صحیح مثبت را محاسبه کند.

```
function z=myfactorial(n)
% this function compute n!
z=1;
for ii=1:n
    z=z*ii;
end
```

❖ همانند سایر زبان‌های برنامه نویسی در Matlab نیز می‌توان برای حل مسائل پیشرو از توابع بازگشتی استفاده کرد.

```
 function z=revers_fac(n)
% this function compute n! by a recursive function
if n==1
    z=1;
    return;
else
    z=n*revers_fac(n-1);
end
```

تنظیم: لعیا افضلی پور؛ علی داودی

- می‌توان یک حلقه را بطور کامل در حلقه دیگری جای داد. اگر یک حلقه در داخل حلقه دیگری قرار گیرد، حلقه‌ها را تو در تو گفته می‌شوند.

دستورات انشعابی:

انشعابات، عبارت‌هایی هستند که به برنامه این امکان را می‌دهند تا قسمتی از آن اجرا شود، در حالیکه از سایر بخش‌های برنامه چشم‌پوشی می‌شود. انواع مختلف آن در ساختار if، ساختار switch ، و ساختار try/catch وجود دارد.

- ساختار **if** : if دارای ساختار زیر است:

```
if   control_expr_1
...
...} Block 1
...
elseif   control_expr_2
...
...} Block 2
...
else
...
...} Block 3
...
end
```

اگر control_expr_1 مخالف صفر باشد، برنامه عبارت داخل Block 1 را اجرا می‌کند و به اولین عبارت قابل اجرا بعد از end می‌رود. در غیراین‌صورت برنامه وضعیت control_expr_2 را چک می‌کند. اگر control_expr_2 مخالف صفر باشد، آنگاه

برنامه عبارت‌های داخل Block 2 را اجرا می‌کند و به اولین عبارت قابل اجرا بعد از end می‌رود. اگر تمامی عبارت‌های کنترل برابر صفر باشد، عبارت موجود دربلوک شرط else اجرا می‌شود. در ساختار if می‌توان از چندین عبارت شرطی elseif استفاده کرد، ولی تنها می‌توان از یک else استفاده کرد. عبارت کنترل در هر شرط، تنها زمانی امتحان می‌شود که عبارات کنترل در تمامی شرط‌های بالا دارای مقدار صفر باشد. اگر هیچ شرط else وجود نداشته باشد، دستورات بعد از عبارت end ادامه می‌یابد.

در اکثر مواقع، عبارت‌های کنترل ترکیبی از عملگرهای منطقی و رابطه‌ای خواهند بود.

بعلاوه ساختار if می‌تواند به‌صورت تودرتو باشد. این ساختار را if تودرتو گویند. این ساختار زمانی که شرط‌ها بر هم تقدم داشته باشند مورد استفاده قرار می‌گیرد.

مثال : برنامه ای بنویسید که ریشه‌های یک معادله درجه ۲ را محاسبه و چاپ کند.

```
function [z1,z2]=root_2_poly(a,b,c)
% this function find roots of a ax^2+bx+c=0

delta=b^2-4*a*c;
if delta>0
    z1=(-b+sqrt(delta))/(2*a);
    z2=(-b-sqrt(delta))/(2*a);
elseif delta<0
    z1=(-b+1i*sqrt(-delta))/(2*a);
    z2=(-b-1i*sqrt(-delta))/(2*a);
else
    z1=-b/(2*a);
    z2=z1;
end
```

مثال۲: برنامه‌ای بنویسید که تابع $f(x,y)$ را برای هر مقدار x,y که توسط کاربر مشخص می‌شود، محاسبه کند. تابع به صورت زیر تعریف می‌شود:

$$f(x,y)=\begin{cases} x+y & x\geq 0 \;\; y\geq 0 \\ x+y^2 & x\geq 0 \;\; y<0 \\ x^2+y & x<0 \;\; y\geq 0 \\ x^2+y^2 & x<0 \;\; y>0 \end{cases}$$

حل:

```
function z=multi_rule(x,y)

if x>=0 && y>=0
    z=x+y;
elseif x>=0 && y<0
    z=x+y^2;
elseif x<0 && y>=0
    z=x^2+y;
else
    z=x^2+y^2;
end

disp('The value of the function is :')
```

- ساختار **switch**:

ساختار switch یکی دیگر از شکل‌های انشعاب است که درآن براساس مقدار یک کاراکتر، عدد ویا عبارت منطقی، بلوکی از برنامه اجرا می‌شود. فرم کلی دستور به‌صورت زیر است:

تنظیم: لعیا افضلی پور؛ علی داوری

```
switch  switch_expr
case  case_expr_1
   …
   …}  Block 1
   …
case  case_expr_2
   …
   …}  Block 2
   …
⋮
otherwise
   …
   …}  Block n
   …
end
```

اگر مقدار switch_expr برابر با case_expr_i باشد، بلوکi ام برنامه اجرا می‌شود و برنامه به اولین عبارت بعد از end در ساختار switch می‌رود.

اگر تعدادی از مقادیر عبارت switch_expr باعث شود که یک قسمت برنامه اجرا شود، تمامی آن مقادیر می‌توانند در یک بلوک و در داخل دو آکولاد قرار گیرد.

تنظیم: لعیا افضلی پور: علی داوری

```
switch  switch_expr
case  {case_expr_1, case_expr_2, case_expr_3}
   …
   …}  Block 1
   …
case  case_expr_4
   …
   …}  Block 2
   …
⋮
otherwise
   …
   …}  Block n
   …
end
```

- ساختار **try/catch** :

ساختار try/catch یک فرم ویژه ساختار انشعابی است که برای به دام انداختن خطاها طراحی شده است. هنگامی‌که خطایی در عبارت موجود در بلوک try اتفاق بیفتد، به‌جای نادیده گرفتن آن، برنامه موجود در بلوک catch اجرا شده و برنامه در حالت اجرا باقی می‌ماند. این به برنامه‌نویس اجازه می‌دهد که بدون توقف برنامه خطاهای برنامه را اداره کند. فرم کلی ساختار try/catch به‌صورت زیر است:

تنظیم: لعیا افضلی پور و علی داوری

```
try
   …
   …}  Try Block
   …
catch
   …
   …}  Catch Block
   …
end
```

هنگامی‌که برامه به ساختار try/catch می‌رسد. عبارت داخل بلوک try اجرا می‌شود. اگر هیچ خطایی اتفاق نیفتد، عبارت داخل بلوک catch نادیده گرفته شده و ادامه برنامه از اولین عبارت بعد از end دنبال می‌شود. هنگامی‌که خطا در بلوک try اتفاق بیفتد، برنامه اجرای بلوک try را متوقف می‌کند و بلافاصله عبارت موجود در بلوک catch اجرا می‌شود.

```
function try_catch_example
clc

%initialize array
a=[1 -2 3 5];
try
    % try to display an element
    index=input('enter subscript of element to display:');
    fprintf('\n a(%d)=%f\n',index,a(index));
catch
       % if got here an error occured
    fprintf('\n illegal subscript : %d \n',index);
    error('illegal subscript')

end
```

تنظیم: نعیم افضلی پور و علی داوری

- عبارت‌های **break** و **continue** و **return**:

از دو عبارت break و continue می‌توان برای کنترل عملکرد حلقه‌های while و for استفاده کرد. عبارت break اجرای یک حلقه را متوقف کرده و اجرای برنامه را به عبارت بعد از end حلقه تکرار منتقل می‌کند. عبارت continue عبور از حلقه را به پایان رسانده و اجرای برنامه را به بالای حلقه باز می گرداند.

به محض فراخوانی دستور return اجرای یک تابع متوقف می‌شود و آرگومان‌های خروجی باز گردانده می‌شود.

❖ دستور return فقط در توابع قابل اجرا است، در غیر این‌صورت فراخوانی دستور return با خطا مواجه خواهد شد. همچنین دستورات break و continue صرفاً در حلقه‌های تکرار امکان پذیر است.

مثال ۳: تابع زیر عددی را از طریق آگومان ورودی خوانده و در مبنای ۲ نمایش می‌دهد.

```
function [R]=binary(x)
% show x in base 2
count=0;
s=x;
while s~=1
    q=floor(s/2);
    R(count+1)=s-q*2;
    s=q;
    count=count+1;
end
R(count+1)=1;
R=fliplr(R);
disp('nargin is')
nargin
disp('nargout is')
nargout
```

تمرین‌ها و اهداف فصل :

۱. تابعی بنویسیدکه اعدد x و a را از طریق ارگومان ورودی خوانده، عدد x را در مبنای a نمایش دهد.(این برنامه تعمیمی از تابع مذکور در مثال ۳ است)

```
 function R=base_a(x,a)
% this function turn x into base a
count=0;
s=x;
while s>=a
    q=floor(s/a);
    R(count+1)=s-q*a;
    s=q;
    count=count+1;

end
R(count+1)=s;
R=fliplr(R);
```

۲. تابعی بنویسد که عددی را از طریق ارگومان ورددی خوانده و تعیین کند آیا عدد مذکور اول بوده یا خیر.(خروجی تابع داده boolian است)

```
function z=is_prime(x)
% z=1 if x is a prime number otherwise z=0
z=1;
for ii=2:floor(x/2)
    if ~rem(x,ii)
        z=0;
        return;
    end
end
```

۳. برنامه‌ای بنویسید که حاصلجمع مقسوم علیه‌های اول یک عدد را محاسبه کند، و حاصلجمع مقسوم علیه‌های اول، مقسوم علیه‌های عدد، و مقسوم علیه‌های اول عدد را

تنظیم: لعیا افضلی پور و علی دوری

به‌صورت بردارهای مجزا در قالب آرگومان خروجی تولید کند. برای تشخیص اول بودن از تمرین قبل کمک بگیرید.(استفاده از حلقه‌ها و دستورات شرطی وتمرین فراخوانی توابع دست‌نویس)

```
function [p_d,d,s_pd]=sumiation_prime_denominator(x)

%p_d is prime_denominators of x
%d is denominators of x
%s_pd is sumiation_prime_denominator s of x
d_count=1;
count=1;
for ii=2:floor(x/2)
    if ~rem(x,ii)
        d(d_count)=ii;
        d_count=d_count+1;
        if is_prime(ii)
            p_d(count)=ii;
            count=count+1;
        end
    end
end
d(d_count)=x;
if is_prime(x)
    p_d(count)=x;
end
s_pd=sum(p_d);

disp('_____________________________________________')
disp('nargin is')
nargin
disp('nargout is')
nargout
disp('_____________________________________________')
```

تنظیم: لعیا افضلی پور ؛ علی داوری

۴. برنامه زیر تابع ریاضی را از ورودی خوانده، ریشه آن را به دو روش تنصیف و نیوتن محاسبه و نمایش میدهد. به ساختار دستور منو در این مثال توجه کنید.(کاربرد دستور switch و فراخوانی زیر توابع، زیر تابع نویسی)

```
function guid

disp(' * welcom :' )
disp(' * This program performe 2 method for Finding root of specific functions
.')

while(1)
    main=menu('This Program include a Newton method for Finding root :','Newton
','Bisection ','Exit');

    switch(main)
        case 1
             Newton1
        case 2
            Bisection
        case  3
              disp(' *  Program ende . ')
            break;

    end

end
%**************************************************************************

 function Newton1()
% This function find root of f(x)by Newton method and primer guess x0.
%work by string
disp('** *** *** *** ** :')
disp(' This function find root of f(x)by newton method and primer guess x0' )

fA=input(' please enter your  favor function : ','s');

x0=input('\n please enter your favor primer guess x0:');
e=input('\n Please enter your favore epsilone :' );

n=1 ;% n-1 count the number of iteration
ff=subs(fA,x0);
```

```
dFA=diff(sym(fA));


dF=subs(dFA,x0);
x1(n)=x0;
    while(abs(ff)>e)
        if(abs(dF)<0.00009 ||isnan(dF))
            fprintf('\n Thanks of diff f(x%d)=0 the result of this method is
invalid.',n)
            fprintf('\n The x%d is : %f',n,x1)
            fprintf('\n The f(x%d) is : %f',n,ff )
            fprintf('\n The Df(x%d) is : %f',n,dF )
            fprintf('\n ')
            disp(' *! You can exam this function by Tendineous method .')
            return;
        end
            x1(n+1)=x1(n)-(ff/dF);
            n=n+1;
            ff=subs(fA,x1(n));
            dF=subs(dFA,x1(n));

    end

        fprintf('\n The proximate root is found & it is : %f',x1(n))
        fprintf('\n The error value of appriximate root is : %f ', ff )
        fprintf('\n The number of iteration is : %d ',n-1)
        fprintf('\n')

        t=x1(n)-10*e:e:x1(n)+10*e;
        yy=subs(fA,t);
        plot(t,yy,' r')
        grid on

%*********************************************************************
function  Bisection()
% This function find root of f(x)in a limitation that f in first and end point
has different sing by Bisection method.

disp('** welcom :')
disp(' This function find root of f(x)in a limitation that f in first and end')
disp(' point has different sing by Bisection method.')

syms x

f=input(' please enter your  favor function : ');
disp(' ');
```

```
while(1)
    disp('* please enter your favor [a,b]as row vector *.')
    l=input(' ! Notice in first & end point,f should has different sing :');
     fa=subs(f,l(1));
     fb=subs(f,l(2));
     if(fa*fb<0)
         break;
     end
      disp(' *!!!!! You wrong !' )

      disp(' *! Notice in first & end point,f should has different sing .' )
      k2=menu('  Do you continue & reset [a b]?','Yes','No');
           if(k2==2)
               disp(' *! Program is failed ')
               return;
           end
end

e=input('\n Please enter your favore epsilone :' );
a=l(1);
b=l(2);

n=0;% count number of iteration

while(1)
    x1=(a+b)/2;
    fa=subs(f,a);
    fx=subs(f,x1);
    n=n+1;

    if(abs(fx)<=e)
        fprintf('\n* The proximate root is found & it is : %f',x1)
        fprintf('\n The error value of appriximate root is : %f ', fx )
        fprintf('\n The number of iteration is : %d ',n)
        fprintf('\n')

        break;
    end
    k=fa*fx;

    if(k<0)
        b=x1;
    elseif (k>0)
        a=x1;
    else
        fprintf('* The Exact root is found & it is : %f',x1)
        fprintf('\n the number of iteration is : %d \n ',n)
```

تنظیم: یعیا افضلی پور ؛ علی داوری

```
    end
end

k=menu('Would you like to survey the graph of function ?','Yes','No');
if(k==1)
    t=x1-1:0.1:x1+1;
    yt=subs(f,t);
    plot(t,yt,'m')
    xlabel(' x ')
    ylabel(' y ')
else
    disp('! program was ended')
    return;
end
```

۵. برنامه‌ای بنویسید که ماتریسی را از ورودی خوانده وتجزیه LU این ماریس را بدست آورد. (بعد از ورود ماتریس شرط تجزیه پذیر بودن بررسی می‌شود.)

```
function [l,u,c]=lu_factorization

while 1
    A=input('* Please enter your favor matrix : ');
    [m,n]=size(A);
    if(m==n)
        A1=A;
        detAi(1)=det(A1);
        for ii=m:-1:1
            A1(:,ii)=[];
            A1(ii,:)=[];
            detAi(m-ii+2)=det(A1);
        end
        if(~all(abs(detAi)>=10^-4))
            disp('* Thanks of some det of submatrix of supposed matrix is 0
there is not lu factorization .');
            k=menu('Would you like try new matrix ?','Yes ','NO');
            if(k==2)
                disp('* program is failed .')
                c=0;
                return ;
            end
        else
            break;
        end % related to if(size(detAi,2)~=size(find(detAi)))

    else % related if m==n
        disp(' ! you wrong ,suppose matrix should be square .')
```

```
        k=menu('would you llike try again ?','Yes ','No ');
        if k==2
            disp('* program is failed .')
            c=0;
            return;
        end
    end% related if m==n
end % while 1

% main algorithm
c=1;
l=zeros(m,m);
u=zeros(m,m);

u(eye(m))=1;

% computing firrst clumn of l & first row of u

l(1:m,1)=A(1:m,1);
u(1,1:m)=A(1,1:m)/l(1,1);

% computing next roe & column
for ii=2:m
    for jj=ii: m

        s1= l(jj,1:ii-1)*u(1:ii-1,ii);
        l(jj,ii)=A(jj,ii)-s1;

        s2= l(ii,1:ii-1)*u(1:ii-1,jj);
        u(ii,jj)=(A(ii,jj)-s2)/l(ii,ii);
    end
end
```

تنظیم: لعیا افضلی پور - علی داوری

تنظیم: لعیا فاضلی پور؛ علی داوری

فصل ششم

ساختمان داده‌ها در Matlab

ساختمان داده‌ها ابزاری برای ذخیره ونگهداری داده‌های مرتبط تحت یک عنوان واحد است.

- **آرایه‌های پراکنده:** هنگامی که آرایه معمولی بیان می شود، Matlab یک فضا برای هر یک از المان‌های آرایه در نظر می‌گیرد. تابع $a = eye(10)$، صد المان در یک ساختار 10×10 قرار گرفته‌اند، ایجاد می‌کند. در این آرایه، عناصر روی قطر ۱ و مابقی ۰ هستند، یعنی ۹۰ عنصر آن صفر است. به‌چنین آرایه‌ای، آرایه پراکنده یا ماتریس پراکنده گویند.

ذخیره و کار با ماتریس‌های پراکنده بزرگ که در آن‌ها تعداد زیادی از عناصر صفراند، بخش عمده‌ای از حافظه و CPU را اشغال می‌کند. بسیاری از مسائل واقعی ماتریس‌های پراکنده ایجاد می‌کند. به عنوان مثال سیستم‌های قدرت بزرگ می‌توانند دارای هزاران گذرگاه در نیروگاه تولید، ایستگاه‌های تولید و پخش باشد. اگر لازم باشد جریان، ولتاژ و قدرت جریان یافته در سیستم محاسبه شود، ابتدا باید ولتاژ هر گذرگاه محاسبه شود. حل یک سیستم با ۱۰۰ گذرگاه، شامل ۱۰۰۰ معادله با ۱۰۰۰مجهول خواهد بود که معادل یک ماتریس با ۱۰۰۰۰۰۰ درایه است. حل این دستگاه شامل بیش از میلیون‌ها عمل جبری است. با این حال، هر گذرگاه در یک سیستم قدرت به طور متوسط تنها به سه گذرگاه دیگر متصل می‌شود. بنابراین ۹۹۷ تا از ۱۰۰۰عنصر در هر ستون ماتریس صفر خواهد بود و اغلب عملکردها با ارقام صفر انجام می‌شود. در صورتیکه بتوان در مراحل حل، صفرها را حذف کرد، محاسبه ولتاژ در سیستم قدرت کاراتر و ساده‌تر خواهد بود.

Matlab دارای نوع خاصی از داده‌ها است که برای کار با آرایه‌های پراکنده طراحی شده‌اند. فرق داده‌های double و داده‌های sparse این است که در داده پراکنده تنها المان‌های غیرصفر یک آرایه در حافظه قرار می‌گیرد. داده‌های پراکنده برای هر المان غیر صفر سه مقدار را ذخیره می‌کند. مقدار آن المان و سطر و ستونی که آن عدد در آن قرار دارد. اگرتعداد عناصر غیر صفر زیاد باشد، استفاده از این روش ممکن است حافظه بیشتری را نسبت به ذخیره‌ی خود ماتریس اشغال کند، زیرا در این حالت آدرس‌ها هم ذخیره می‌شود.

Commands for Managing sparse Martix	
issparse(S)	Returns logical 1if S is sparse and logical 0 otherwise
sparse(A)	Converts a full matrix to sparse form
speye(n)	Form an n-by-n sparse matrix with 1s on the main diagonal
full(S)	Converts a sparse matrix S to full storage organization.
sprand(S)	Has the same sparsity structure as S, but uniformly distributed random entries
sprandn(S)	Has the same sparsity structure as S, but normally distributed random entries with mean 0 and variance
spones(S)	Generates a matrix with the same sparsity structure as S, but with 1's in the nonzero position
nnz(S)	Returns the number of nonzero elements in matrix
sprandsym(S)	Returns a symmetric random matrix whose lower triangle and diagonal have the same structure as S. Its elements are normally distributed, with mean 0 and .variance 1

جدول ۱: توابع کاربردی در کنترل آرایه‌های پراکنده

آرایه سلولی:

آرایه سلولی، نوع خاصی از آرایه در Matlab است که المان‌های آن سلولهایی هستند، که خود می‌توانند شامل آرایه‌هایی دیگر باشد. برای مثال، یک سلول از آرایه سلولی ممکن است شامل یک آرایه اعداد حقیقی باشد، یکی آرایه رشته‌ای، و یکی دیگر نیز آرایه‌ای از اعداد مختلط باشد. در برنامه‌نویسی، هر المان از آرایه سلولی یک اشاره‌گر به ساختار داده‌های دیگر است، که آن ساختارها می‌توانند انواع مختلفی باشد. آرایه‌های سلولی راه خوبی برای جمع‌آوری اطلاعات از یک مسئله است، زیرا تمام اطلاعات می توانند با هم نگهداری شده و با یک اسم واحد در دسترس قرار گیرند.

در آرایه‌های سلولی از آکولاد {} به جای () برای دسترسی به محتویات سلول‌ها استفاده می‌کنند. این تفاوت به خاطر این است که آرایه‌های سلولی شامل ساختارهای داده‌اند نه داده. عبارت $a(ii,jj)$ ساختار داده‌های درون سلول ii,jj ام را نشان می‌دهد در حالیکه $a\{ii,jj\}$ محتوای داده‌های موجود در سلول را نشان می‌دهد.

Commands for Managing cell-Array	
$cell(m,n)$	Creates an m-by-n cell array of empty matrices
X = C(s)	Returns the cells of array C that are specified by subscripts s
X = C{s}	Returns the contents of the cells of C that are specified by subscripts s
x = = C{s}(t)	References one or more elements of an array that resides within a cell. (Subscript s selects the cell, and subscript t selects the array element(s)

cellplot(c)	.Displays a figure window that graphically represents the contents of c.
celldisp(C)	Recursively displays the contents of a cell array
iscell(A)	Returns logical 1 if A is a cell array and logical 0 otherwise

آرایه‌های ساختاری:

ساختار یک نوع داده است که المان‌های مختلف آن دارای نام‌های مختلف‌اند. المان‌های مختلف ساختار به عنوان یک حوزه شناخته می‌شوند. حوزه‌ها در یک ساختار می‌توانند انواع مختلفی داشته باشند. حوزه‌های مختلف به‌صورت ترکیبی از نام ساختار به‌همراه نام حوزه که با یک نقطه از یکدیگر جدا می‌شوند، آدرس‌دهی می‌شوند.

یک آرایه ساختاری، آرایه‌ای از ساختارهاست که هر ساختار در آرایه اسمی مشابه حوزه خواهد داشت. ولی داده‌های ذخیره شده در حوزه می تواند متفاوت باشد.

یک ساختار را می توان را به‌طریق ایجاد کرد،

- یک حوزه بوسیله عبارت‌های جایگزین ایجاد شود
- تمامی آن‌ها با تابع struct ایجاد می‌شود.

ساخت یک ساختار بوسیله عبارت‌های جایگزین:

یک حوزه از ساختار را می‌توان با عبارت‌های جایگزین ایجاد کرد. هر بار که مقادیر به حوزه نسبت داده می‌شود، آن حوزه به‌طور اتوماتیک ایجاد می‌شود.

مثال: ساختار مربوط به ذخیره اطلاعات دانشجویان می‌باشد.

```
Function student=data_structur_Arrey_struct
clc

% by 88dding88ion
student.name='John Doe';
student.address='123 Main street';
student.city='Anytown';
student.zip=71211;

student
disp('----------------------------------------')
% 88dding next person to this structure
student(2).name='Jane Q. Public';
student(2).address='225 second Main street';
student(2).zip=2561;

student
student(2)

disp('----------------------------------------')
```

هنگامی که یک آرایه ساختاری بیش از یک المان داشته باشد، تنها نام حوزه‌ها لیست می‌شود نه محتوای آن‌ها. محتوای هر المان را می‌توان با فراخوانی آن المان به طور جداگانه مشاهده کرد.

ایجاد آرایه ساختاری به کمک تابع struct :

با استفاده از تابع struct می‌توان آرایه از ساختارها را مقداردهی کرد.فرم این تابع به‌صورت زیر است :

$$str_array = struct(fields)$$

تابع struct هر حوزه در آرایه را به ماتریس خالی مقداردهی می‌کند. حوزه‌ها را می‌توان به‌صورتی که ذکر شد مقداردهی کرد. این شکل از تابع به‌صورت زیر است:

$$str_array = struct('field1',val,'field2',val,...)$$

اضافه کردن حوزه‌ها به ساختارها:

اگر یک حوزه جدید در مورد هر یک از المان‌های یک آرایه ساختاری تعریف شود، حوزه جدید به‌طور اتوماتیک به تمامی المان‌های موجود در آرایه اضافه می‌شود. برای مثال فرض کنید که در مثال قبل به Jane Q public چند نمره امتحانی اضافه شود.

```
student(2).exams=[90 82 88]
```

حذف یک حوزه از ساختار:

یک حوزه را می‌توان با استفاده از تابع rmfield از ساختار حذف کرد.فرم این دستور به‌صورت زیر است :

$$struct2 = rmfield(struct_array,'field')$$

مثال: می‌توان zip را از ساختار student حذف کرد.

تنظیم: لیلا افضلی پور و علی داوری

lafzalipour@hotmail.com

l.afzalipour@gmail.com

تنظیم: لعیا افضلی پور؛ علی داوری

فصل هفتم

رشته ها و سیستم فایل

رشته ها :

رشته در Matlab یک آرایه از نوع char است. هر کاراکتر در دو بایت از حافظه ذخیره می‌شود. لیستی از پرکاربردترین ابزارهای پردازش رشته‌ها در جدول ۵ ضمیمه است.

Commands for Managing Strings	
$ischar(a)$	Returns l if a is a character array and 0 otherwise
$double(s)$	Returns an array the same size as str containing 1 where the elements of s are letters of the alphabet and 0 where they are not
$char(X)$	converts array X of nonnegative integer codes into a character array
$strcat(s_1,s_2)$	Output can be combinations of single s_1, s_2
$strvcat(s_1,s_2)$	It is same as strcat, but in vertical direction
$strcmp(s_1,s_2)$	compares the strings s_1 and s_1 and returns 1 if they are identical, and returns 0 otherwise
$strcmpi(s_1,s_2)$	It is same as strcmp, but not case sensetive
$strncmp(s_1,s_2,n)$	compares the first n characters of strings s_1 and s_2 and returns 1if they are identical, and returns 0 otherwise
$strncmpi(s_1,s_2,n)$	It is same as strncmp, but not case sensetive
$isletter(s)$	returns an array the same size as str containing 1 where the elements of str are letters of the alphabet and 0 where they are not
$isspace(s)$	returns an array the same size as 'str' containing 1 where the elements of s are whitespace characters, and 0 where they are not

$findstr(s, pat)$	searches the s for occurrences of a shorter string, pat, and returns the starting index of each such occurrence in the double array
$strrep(s, pat1, pat2)$	replaces all occurrences of the string pat1 within string s with the string pat2
$[tok, re] = strtok(s)$	parses input string s from left to right, returning part or all of that string in token. Using the white-space character as a delimiter,
$upper(s)$	converts any lowercase characters in the string str to the corresponding uppercase characters to upper case
$lower(s)$	returns the string formed by converting any uppercase characters to lowercase characters
$num2str(x, n)$	converts numbers to their string representations
$int2str(x)$	converts integers to their string representations
$mat2str(A)$	converts matrix A into a string
$eval(s)$	executes s
$blanks(n)$	is a string of n blanks
$deblank(s)$	removes all trailing whitespace and null characters from the end of character string s

جدول ۱: ابزار پردازش رشته‌ها

سیستم فایل :

در فصل ۳ چگونگی بازیابی و ذخیره داده‌ها توسط دو دستور load , save و همچنین چاپ داده‌ی فرمت شده با فراخوانی تابع fprintf، در محیط Matlab مرور شد. در این فصل مطالب بیشتری پیرامون قابلیت‌های توابع ورودی/ خروجی Matlab ارائه می‌شود.

- تابع **textread**:

این تابع برای خواندن فایل‌های نوع اسکی که به صورت ستون‌هایی از داده‌های فرمت شده می‌باشند، طراحی شده‌است. هر ستون در این داده‌ی فرمت شده، می‌تواند از نوع متفاوتی باشد. این فرمان برای وارد کردن جداولی از داده‌ها که توسط برنامه‌های کاربردی دیگر ایجاد شده‌اند، بسیار مفید است.

شکل این فرمان به صورت:

```
[a,b,c,...]=textread (filename, format, n)
```

که درآن filename نام فایلی است که باید باز شود وformat رشته‌ای است که شامل توضیحی در مورد نوع داده‌ها در هر ستون می‌باشد n نیز تعداد خطهایی است که باید خوانده شود. اگر n نوشته نشود کل فایل خوانده می‌شود. توجه کنید که تعداد آرگومان‌های خروجی باید با تعداد ستون‌هایی که خوانده می‌شود، سازگار باشد.

```
[firstn,lastn,blood,score,age,answer]=textread('file_read',...
    '%s %s %s %f %d %s',2)

disp('_______  May be ignor some _____________________________')
% ignor some column
[firstn,lastn,score]=textread('file_read','%s %s %*s %f %*d %*s')
```

- تابع **xlsread**:

این تابع در بارگذاری داده‌ها از فایل‌های صفحه گسترده کاربرد دارد.

فرم کلی این دستور به فرم :

```
[num,txt,raw] = xlsread(filename,sheet,range)
```

که filename نام فایل مورد نظر، sheet رشته ای محتوی نام صفحه مورد نظر و range رشته‌ای حاوی اطلاعات بلوک‌ای است که باید در صفحه و فایل مذکور بارگذاری شود. num حاوی اطلاعات عددی و txt ارایه ای سلولی حاوی اطلاعات رشته‌ای خوانده شده و raw ارایه سلولی است که شامل کل اطلاعات بارگذاری شده به فرمت رشته می‌باشد.

- تابع **xlswrite** :

این تابع در درج اطلاعات در صفحه گسترده کاربرد دارد. فرم کلی این دستور به فرم :

```
xlswrite(filename,A,sheet,range)
```

که filename نام فایل مورد نظر، sheet رشته ای محتوی نام صفحه مورد نظر و range رشته‌ای حاوی اطلاعات بلوک‌ای است که باید در صفحه و فایل دادهای ماتریس A درج شود.

پردازش فایل‌ها در **Matlab** :

تنظیم: لعیا افضلی پور و علی داوری

در محیط نرم‌افزار Matlab برای کنترل و پردازش هر فایل بهنگام بازگشایی هر فایل یک file id به آن اختصاص داده می‌شود. این عدد برای همه‌ی عملیات خواندن، نوشتن، و کنترل مورد استفاده قرار می‌گیرد. شناسه فایل یک عدد صحیح مثبت است. دو شناسه فایل همیشه باز هستند شناسه ۱ که بافر خروجی استاندارد استو شناسه ۲ که بافر پردازش خطای استاندارد مربوط به کامپیوتری است که Matlab در ان در حال اجرا است. به محض باز کردن فایل، شناسه منحصر بفردی به آن اختصاص داده و با بستن فایل آن شناسه آزاد می‌شود. مجموعه‌ای از دستورات کاربردی پردازش فایل‌ها در جدول ۲ ضمیمه شده‌اند.

I/O Commands in Matlab	
fopen(fname,p)	Opens the file with the specified permission.
fclose(id)	Closes an open file which is identified by id.
fread(id,sizeA)	Reads sizeA elements into A and positions the file pointer after the last element read.
fwrite(id, A)	Writes the elements of array A to a binary file in column order.
fscanf(id,format)	Reads and converts data from a text file into array A in column order
fprintf(id,format,A)	Applies the format to all elements of array A , and writes the data to a text file
ftell(id)	Returns the current position in the specified file
fgetl(id)	Returns the next line of the specified file, removing the newline characters
fgets(id,nchar)	Reads the next line of the specified file, including the newline characters.

ferror(id)	Returns the error message for I/O operation on the specified file. If the operation was successful, message is an empty string
feof(id)	Returns 1 if a previous operation set the end-of-file indicator for the specified file. Otherwise 0.
fseek(id, P, O)	Sets the file position indicator P bytes from origin in the specified file
frewind(id)	Sets the file position indicator to the beginning of a file
tempdir	Returns the name of the system's temporary folder, if one exists
tempname	Returns a unique string, suitable for use as a temporary filename

جدول ۲: ابزار پردازش فایل‌ها

داده‌ها می‌تواند به دو روش در یک فایل نوشته شده و یا از ان خوانده شود: به عنوان داده‌ی باینری یا به عنوان داده‌ی کاراکتری(txt) فرمت شده. داده‌ی باینری شامل الگوی واقعی بیت‌ها می‌باشد، که برای ذخیره داده در حافظه کامپیوتر استفاده شده است. خواندن و نوشتن داده‌س باینری کارایی بسیاری دارد، اما خود کاربر نمی‌تواند داده ذخیره شده در فایل را بخواند. این در حالی است که داده در فایل فرمت شده به کاراکترهایی ترجمه می‌شود که کاربر به درک آن‌ها خواهدبود. در هر صورت عملیات I/O فرمت شده نسبت به I/O باینری دارای سرعت و کارایی کمتری است. اگر می-خواهید یک فایل در حالت متن باز شود، یک t به رشته permission اضافه شود. (برای مثال rt یا +rt) . برای گشودن فایل در حالت باینری نیز، باید یک b به permission اضافه شود. (برای مثال wb).

Acceptable permission string in I/O Operations

‘r’	Open existed file for reading (default).
‘r+’	Open existed file for reading and writing
‘w’	Open or create new file for writing. Discard existing contents.
‘w+’	Open or create new file forreading and writing. Discard existing contents, if any
‘a’	Open or create new file for writing. Append data to the end of the file.
‘a+’	Open or create new file for reading writing. Append data to the end of the file.

جدول ۳: مقادیر مجاز رشته **permission**

Acceptable Origin string in I/O Operations	
bof or -1	Beginning of file
cof or 0	Current position in file
eof or 1	End of file

جدول ۴: مقادیر مجاز رشته **origin** در شروع مکان دریک فایل

Acceptable foemat string in fprinft and fscanf function	
%c	Single character
%s	String of characters
%d	Base 10
%o	Base 8 (octal)
%x	Base 16 (hexadecimal), lowercase letters a–f
%f	Fixed-point notation
%e	Exponential notation, such as 3.141593e+00
%g	The more compact of %e or %f, with no trailing zeros
"	Single quotation mark
٪٪	Percent character
\\	Backslash
\a	Alarm
\b	Backspace
\n	New line

\t	Horizontal tab
\v	Vertical tab
\f	Form feed

جدول۵: مقادیر مجاز رشته **format** در تابع **fprintf**

پسوندهای مرسوم در ذخیره سازی داده‌ها txt و bin می باشد. مثال‌های این فصل را می توانید در m –file ، به‌نام file_system مرور کنید.

تنظیم: لعیا افضلی پور ؛ علی داوری

lafzalipour@hotmail.com

l.afzalipour@gmail.com

تنظیم: لعیا افضلی پور؛ علی داوری

تنظیم: لیلا افضلی پور و علی داوری

فصل هشتم

پرکاربردترین مفاهیم ریاضی مطرح در سایر علوم

معادلات خطی و غیر خطی:

❖ معادلات خطی:

مقصود از معادلات خطی سیستم $Ax=b$ است. جواب این سیستم x است که از طریق روابط جبری و یا روش‌های عددی محاسبه می‌شود. A می‌تواند مربعی یا مستطیلی باشد.

مثال: سیستم خطی $\begin{cases} x+2y+3z=2, \\ 4x+y+2z=3 \end{cases}$ باشد.

روش‌های جبری: با استفاده از دستور solve می‌توان یک سیستم خطی را به‌صورت جبری حل کرد.

```
% linear equation A is rectangular type
disp('---------  A ia recangular  ------------------')
[x,y,z]=solve('x+2*y+3*z-2','4*x+y+2*z-3','x,y,z')
z1=1;
sol1=eval([x,y,z])
```

روش‌های عددی: با استفاده از حل کننده عددی \ می‌توان یکی از جواب‌های سیستم فوق را محاسبه کرد.

```
disp('_____________ second way _____________________')
A=[1 2 3;4 1 2];
b=[2;3];
sol2=A\b;
disp('solution is ')
disp(sol2)
```

تنظیم: نعیما افضلی پور ؛ علی دارش

معادلات غیرخطی:

مقصود از یک سیستم غیر خطی $\begin{cases} f_1(X)=0 \\ f_2(X)=0 \\ \vdots \\ f_n(x)=0 \end{cases}$ است، که توابع $f_1, f_2, \cdots, f_n$ بعضاً غیرخطی اند. در اینجا بردار تصمیم X است، که می‌تواند تک مولفه و یا چندمولفه‌ای باشد.

مثال: برای محاسبه جواب سیستم $\begin{cases} f1=(x-2)^2+(y-4)^2-r^2, \\ f2=y-(x/2)-1 \end{cases}$ می‌توان از روش‌های جبری و عددی کمک گرفت:

- روش‌های **جبری**: با استفاده از دستور solve می‌توان یک سیستم غیرخطی را حل کرد.

```
disp('*********** Non Linear part *****************')
f1='(x-2)^2+(y-4)^2-r^2';
f2='y-(x/2)-1';
[xs,ys]=solve(f1,f2)
r=2;
eval([xs,ys])
```

- روش‌های **عددی**: با استفاده از حل کننده‌عددی fsolve می‌توان یکی از جواب‌های سیستم فوق را بدست آورد.

```
 disp('_____________ second way _____________________')
x0=[0;0];
options=optimset('Algorithm','levenberg-marquardt');
[xsol,fval,exitflag]=fsolve(@fun1,x0,options)
```

تنظیم: لعیا افضلی پور ؛ علی داودی

بهینه‌سازی:

$$\begin{aligned} Min \ & f(x) \\ s.t \ & G(x) \le 0, \\ & H(x) = 0, \\ & x \in X . \end{aligned}$$

مقصود از یک سیستم بهینه سازی عبارت است از :

توابع $H(x), G(x), f(x)$ می‌توانند خطی باشند– در این حالت سیستم را خطی نامند،– و یا بعضاً غیرخطی نیز باشند که در این حالت سیستم را غیرخطی نامند. توابع $G(x), H(x)$ می‌توانند صفر باشند، چنانچه مجموعه $X = IR^n$ باشد در این‌صورت سیستم بهینه‌سازی یک سیستم نامقید خواهدبود. بسته به خطی، غیر خطی و یا نامقید بودن سیستم حل کننده‌ی خاصی برای حل سیستم پیشنهاد می‌شود.

- **سیستم خطی** : همانطور که گفته شد، در این حالت توابع $H(x), G(x), f(x)$ خطی هستند، و سیستم کلی تر را می توان به‌فرم ساده‌تری که فرم استاندارد نامیده می‌شود، بازنویسی کرد:

$$\begin{aligned} Min \ & c^T . x \\ s.t \ & Ax \le b, \\ & Aeqx = beq, \\ & lb \le x \le ub, \quad (lb \text{ allowed to be } -\inf, ub \text{ allowed to be } \inf) \end{aligned}$$

برای حل یک سیستم خطی استاندارد حل کننده linprog مناسب است.

مثال: جواب سیستم

$$\begin{aligned}\min\quad & 20x_1+30x_2+15x_4\\ s.t\quad & .3x_1+.2x_2+.4x_3+.2x_4=.2,\\ & .2x_1+.6x_2+.3x_3+.4x_4\ge .3,\\ & .4x_1+.15x_2+.25x_3+.3x_4\ge .2,\\ & x_1\le .3,\\ & x_2\le .4,\\ & x_2,x_3,x_4\ge 0.\end{aligned}$$

را می‌توان با فراخوانی‌های زیر بدست آورد.

```
function Optimization_system

% linear system
disp(' ------------  Linaer system -------------')
disp(' _______________ Details are sorted as follow _____________')
C=[20;30;20;15]
A=[-.2 -.6 -.3 -.4;-.4 -.15 -.25 -.3]
b=[-.3;-.2]
Aeq=[.3 .2 .4 .2]
beq=[.2]
lb=[-inf;0;0;0]
ub=[.3;.4;inf;inf]
[x,fval,exitflag,output,lambda]=linprog(C,A,b,Aeq,beq,lb,ub)
```

- سیستم غیرخطی مقید: حل کننده fmincon برای سیستم غیرخطی مقید استاندارد مناسب است.

مثال : سیستم استاندارد غیر خطی زیر را در نظر بگیرید.

$$\begin{aligned}\min\quad & f(x)=(x_1-x_2)^2+(x_2-x_3)^3+(x_3-x_4)^4+(x_4-x_5)^4\\ s.t\quad & x_1+x_2+x_3^2=3,\\ & x_2-x_3^2+x_4=1,\\ & x_1x_5=1.\end{aligned}$$

```
disp(' ------------ NonLinaer constrained system -------------')
dimx=5;
A=[];
b=[];
Aeq=[];
```

```
beq=[];
lb=-inf(dimx,1);
ub=inf(dimx,1);
x0=2*ones(dimx,1);

[x,fval,exitflag,output,lambda]=fmincon(@fun,x0,A,b,Aeq,beq,lb,ub,@nonlcon)
disp(' ------------ New try ------------------------------------------')
options=optimset('Algorithm','sqp');
[x,fval,exitflag,output,lambda]=fmincon(@fun,x0,A,b,Aeq,beq,lb,ub,@nonlcon,op
tions)
%-------------------------------------------------------------------------
function z=fun(x)
z=(x(1)-x(2))^2+(x(2)-x(3))^3+(x(3)-x(4))^4+(x(4)-x(5))^4;

%_____________________________________
function [c,ceq]=nonlcon(x)
c=[];% Compute nonlinear inequalities at x
ceq=[x(1)+x(2)^2+x(3)^3-3;x(2)-x(3)^2+x(4)-1;x(1)*x(5)-1]; % Compute
nonlinear equalities at x.
```

با استفاده از دستور optimset می‌توان مقادیر پیش‌فرض در حا کننده را تغییر داد. به پیغام هشدار که پس از اولین فراخوانی دستور fmincon ظاهر می‌شود دقت کنید.

- سیستم غیرخطی نامقید: در این سیستم مجموعه قیود سیستم تهی است. حل کننده fminunc مناسب است.

مثال: مسئله غیرخطی و نامقید زیر را در نظر بگیرید.

$$\min \quad 2x_1^2 - x_1x_2 + x_2^2 - 3x_1 + e^{(2x_1+x_2)}.$$

```
disp(' ------------ NonLinaer unconstrained system -------------')
dimx=2;
x0=zeros(dimx,1);
[x,fval,exitflag,output,lambda]=fminunc(@fun2,x0)

% ----------------------------
function z=fun2(x)
z=2*x(1)^2-x(1)*x(2)+x(2)^2-3*x(1)+exp(-2*x(1)-x(2));
```

آمار: اغلب داده‌ها و پارامترها مورد مطالعه علم آمار قرار می‌گیرند.

- پارامترهای مهم آماری در جدول ۱ ضمیمه می‌باشند.

Statistic essential parameter	
$mean(A)$	Returns the mean values of the elements along different dimensions of an array
$median(A)$	Returns the median values of the elements along different dimensions of an array
$mode(A)$	the most frequently occurring value in A
$std(A)$	Returns the standard deviation
$var(A)$	Returns the variance of A for vectors
$cov(x, y)$	Is the covariance matrix
corrcoef(x,y)	Correlation coefficients calculated from x,y
$sum(A)$	Returns sums along different dimensions of an array
cumsum(A)	Returns the cumulative sum along different dimensions of an array
$prod(A)$	Returns the products along different dimensions of an array
cumprod(A)	Returns the cumulative product along different dimensions of an array
$max(A)$	Returns the largest elements along different dimensions of an array.
$min(A)$	Returns the smallest elements along different dimensions of an array
$sort(A)$	Sorts the elements along different dimensions of an array, and arranges those elements in ascending order

جدول ۱:پارامترهای مهم آماری

- **توزیع‌های احتمالاتی و اعداد تصادفی:**

لیست منتخب از اعداد تصادفی براساس توزیع‌های احتمالاتی در جدول ۲ ضمیمه است. جزئیات سایر توزیع‌ها را با جستجوی کلمات کلیدی distribution Reference در help نرم‌افزار دنبال کنید.

probability Distribution and random number	
$rand(m,n)$	Returns an m-by-n matrix containing pseudorandom values drawn from the standard uniform distribution on the open interval (0,1)
$randn(m,n)$	Returns an n-by-n matrix containing pseudorandom values drawn from the standard normal distribution
$randi(i\ \max, m, n)$	Returns an m-by-n matrix containing random integer values drawn from the discrete uniform distribution on 1:imax
exprnd(μ, m, n)	Random numbers from the exponential distribution with μ.
binornd(n,p,dim1,dim2)	Random numbers from the binomial distribution with n and p.
normrnd(mu,sigma,m,n)	Generates random numbers from the normal distribution with mean parameter mu and standard deviation parameter sigma
pdf(name,X,A,B,C)	computes the probability density function for three-parameter families of distributions
random(name,A,B,C,[m,n])	Returns random numbers from a three-parameter family of distributions.
cdf('name',X,A,B,C)	computes the cumulative distribution function for three-parameter families of distributions

جدول ۲: اعداد تصادفی و توزیع‌های احتمالاتی منتخب

برای آگاهی از حالت‌های ممکن name در تابع random و pdf و cdf از help نرم‌افزار کمک بگیرید. جزئیات تابع توزیع تجمعی احتمال را با جستجوی عبارت cumulative Distribution در help نرم‌افزار مرور کنید. جزئیات تابع چگالی احتمال با جستجوی عبارت probability density در help نرم‌افزار دنبال کنید.

- **وارون توزیع‌های احتمالاتی:**

لیستی از وارون‌های توزیع‌های احتمالاتی منتخب در جدول ۳ ضمیمه است. جزئیات بیشتر را با جستجوی عبارت کلیدی Inverse Cumulative Distribution در help نرم‌افزار پیگیری کنید.

Inverse Cumulative Distribution	
x=icdf(name,Y,A,B,C)	Computes the inverse cumulative distribution function for three-parameter families of distributions,
x=binoinv(Y,N,P)	Returns the smallest integer X such that the binomial cdf evaluated at X is equal to or exceeds Y
x= expinv(P,mu)	Computes the inverse of the exponential cdf with parameters mu for the corresponding probabilities in P
x = norminv(P,mu,sigma)	computes the inverse of the normal cdf using the mu and sigma at the corresponding probabilities in P
x=poissinv(P,lambda)	Returns the smallest value X such that the Poisson cdf evaluated at X equals or exceeds P, lambda
x=unifinv(P,A,B)	computes the inverse of the uniform cdf with parameters A and B (the minimum and maximum values, respectively) at the corresponding probabilities in P

جدول ۳: وارون توزیع‌های احتمالاتی

وارون‌های احتمالاتی در مباحث شبیه‌سازی کاربرد دارد. در محاسبات ریاضیات مالی نیز نقش پر-رنگی ایفا می‌کنند.

معادلات دیفرانسیل معمولی :

حل سیستم‌های معادلات دیفرانسیل در بسیاری از علوم مطرح است.

معادله ای که در آن عملگر دیفرانسیل ظاهر شود، یک معادله دیفرانسیل است.یک معادله دیفرانسیل شامل مشتق‌های متغیر وابسته و متغیر مستقل است.

یک معادله دیفرانسیل مرتبه اول به صورت $\frac{dy}{dt}=f(t,y)$ است. یک معادله دیفرانسیل مرتبه دوم به صورت $\frac{d^2y}{dt^2}=f(t,y,\frac{dy}{dt})$ است. حل عمومی این معادلات تابع $y=f(x)$ است که در معادلات مذکور صدق می‌کند.

- **حل جبری** : با استفاده دستور dsolve می‌توان یک حل عمومی ویا حتی در صورت داشتن شرایط مرزی حل خصوصی یک سیستم دیفرانسیل معمولی را بدست آورد. فرم کلی این دستور به یکی از دو صورت زیر است.

```
sol = dsolve('eq')
sol = dsolve('eq','var')
```

eq همان معادله مورد نظر است.

❖ از هر حرف کوچک و بزرگ برای معرفی متغیر وابسته می‌توان استفاده کرد به‌غیر از D.

در دستور فوق متغیر مستقل با تایپ به صورت رشته درمکان var تعیین می‌شود. در فرم اول به صورت پیش‌فرض متغیر مستقل t در نظر گرفته می‌شود. تایپ حرف D، عمل دیفرانسیل گیری را مشخص می‌کند. اگر t متغیر مستقل و y متغیر وابسته باشد، Dy علامت

اختصاری $\frac{dy}{dt}$ است. مشتق مرتبه دوم به صورت D2 و مشتق مرتبه سوم D3 و ... تایپ می‌شود.

مثال : یک جواب عمومی از حل ODE درجه ۲، $\frac{d^2y}{dt^2}+2\frac{dy}{dt}+y=0$ به صورت زیر است:

```
function Differential_system
clc
% ODE
disp('---------  General ODE --------------')
eq='D2y+2*Dy+y=0';
sol=dsolve(eq)
```

به تفاوت جواب‌ها فراخوانی‌های فرم اول و دوم در مثال زیر دقت کنید.

```
disp('_____________________________________________________________')
disp('---------  forn 1 --------------')
z1=dsolve('Ds=a*x^2')
disp('---------  form 2 respected to x --------------')
z2=dsolve('Ds=a*x^2','x')
disp('---------  form 2 respected to a --------------')
z3=dsolve('Ds=a*x^2','a')
```

- **حل خصوصی** :همان‌طور که گفته شد با وجود شرایط اولیه و مرزی می‌توان یک جواب خصوصی برای یک معادله ODE بدست آورد. شکل کلی دستور dsolve بسته به درجه معادله ODE به صورت زیر است:

$$sol = dsolve('eq','cond1','cond2',\cdots,'var')$$

اگر درجه معادله دیفرانسیل n باشد، برای بدست آوردن جواب خصوصی نیاز به n عبارت cond است. شرایط مرزی به شکل رشته و معمولاً به شکل زیر معادل سازی می‌شود.

دستور Matlab	شکل ریاضی

$y(a)=A$	$'y(a)=A'$
$y'(a)=A$	$'Dy(a)=A'$
$y''(a)=A$	$'D2y(a)=A'$

جدول ۴: معادل‌سازی شرایط مرزی

مثال: معادله مرتبه دوم $\frac{d^2y}{dt^2}-2\frac{dy}{dt}+2y=0$ با شرایط اولیه $y(0)=1$ و $\frac{dy}{dt}|_{t=0}=0$ در Matlab به صورت زیر حل می‌شود.

```
disp('_____________________  particular solution _____________________')

p_sol=dsolve('D2y-2*dy+2*y=0','y(0)=1','Dy(0)=0')
```

➢ **حل دستگاه معادلات:**

دستگاه معادلات $\begin{cases} x_1' = x_1+2x_2+1 \\ x_2 = 4x_1+3x_2-1 \end{cases}$ و شرایط مرزی $x_1(0)=0, x_2(0)=2$ را در نظر بگیرید.

```
disp('--------------- linear Differential_system-----------------------------
')
sol=dsolve('Dx1=-x1+2*x1+1,Dx2=4*x1+3*x2-1','x1(0)=0','x2(0)=2')
disp(' ! solution is a structur ....')
disp(' ----  x1 field is : ')
sol.x1
disp(' ----  x2 field is : ')
sol.x2
```

❖ اگر Matlab نتواند یک معادله ODE را با توجه به شرایط داده نشده حل کند پیغام Warning : expliciet solution could not be found نمایش داده می‌شود.

```
% exolicit type
sol=dsolve('Dx1=-x2+t,Dx2=x1*sin(t)','x1(0)=1','x2(0)=0')
```

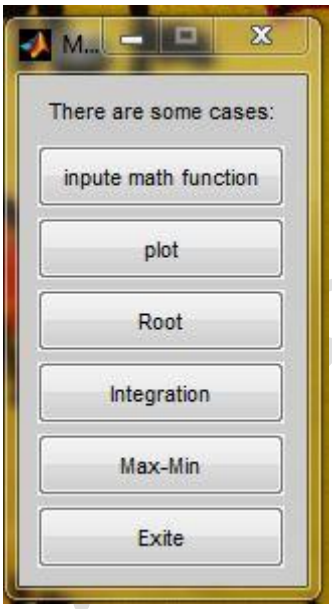
تنظیم: لعیا افضلی پور؛ علی داوری

lafzalipour@hotmail.com

l.afzalipour@gmail.com

تمرینات تکمیلی پایان دوره

۱. منویی شامل موارد ذیل طراحی کنید که نسبت به انتخاب هر کلید عملکرد مناسب را اتخاذ کند. رفتار هر کلید با فراخوانی یک زیر تابع کنترل شود.



۱. بردار های

$x = 0:.1:1$

$y = [0 \quad .099 \quad .198 \quad .295 \quad .389 \quad .479 \quad .564 \quad .644 \quad .717 \quad .783 \quad .841]$

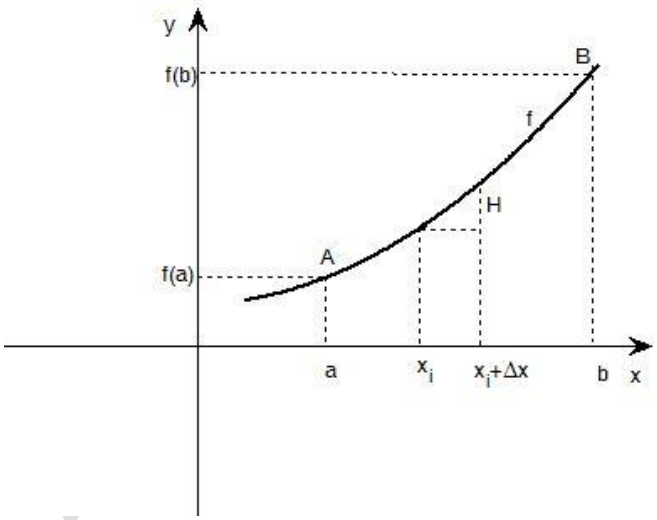
مفروض اند. $\int_0^1 ydx$ کدام است؟ (کاربرد چندجمله‌ای درونیاب . interpolation را در help جستجو کرده ، از تابع مذکور در تعریف تابع دست نویس فراخوانی شونده در quad بهره گیرید.)

۲. معادلات زیر مفروض اند جواب دستگاه کدام است؟

$$x_1 + x_2 = 3,$$
$$x_{i+1} - x_{i+2} = 1; \quad \forall i = 1, \cdots, n-2,$$
$$x_n + x_{n+1} = 3.$$

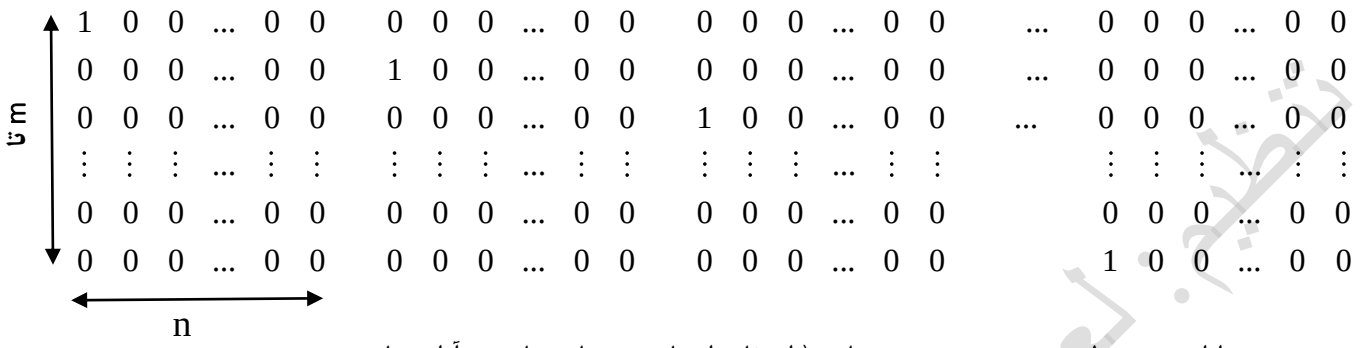
در حل موردی $n = 10$ در نظر گرفته شود.

۳. نمودار زیر را رسم کنید.(ابزار پنجره ترسیم)



۴. با استفاده از توابع int*egral*3 (سه بعدی) int*egral*2 (دو بعدی) امکان محاسبه انتگرال سه گانه و دوگانه روی نواحی عام با رویکردی عددی میسر است. از طریق help نرم افزار این روند را دنبال کرده و مقدار انتگرال $\int_{-1}^{1}\int_{0}^{\sqrt{1-x^2}} dydx$ را محاسبه کنید. بدیهی است این عدد باید برابر مقدار عددی $\frac{\pi}{2}$ باشد.

۵. m,n مفروض اند. دستوری بنویسید که ماتریس زیر را با الگوی داده شده تولید کند.



ماتریس مصور دارای m سطر و $m\times n$ ستون می باشد.(استفاده از ماتریس های خاص و آرایه های ماسک)

تنظیم: لیلا افضلی پور ؛ علی داوری

lafzalipour@hotmail.com

l.afzalipour@gmail.com

تنظیم: لیا افضلی پور؛ علی داوری