بسم الله الرحمن الرحیم

درس نرم افزارهای ریاضی، آشنایی با نرم‌افزارهای متلب و لاتک

مدرس: نجمه حسینی منجزی

دانشگاه اصفهان، دانشکده ریاضی و آمار، گروه ریاضیات کاربردی و علوم کامپیوتر

بخش ۲

بهمن ۱۴۰۰

فهرست مطالب

۱ توابع در MATLAB

نرم افزار MATLAB دارای توابع زیادی است که از قبل نوشته شده‌اند و کاربر براحتی با تایپ کردن اسم آها می‌تواند از آن‌ها استفاده کند. اکثر این توابع را می‌توان برای متغیرهای حقیقی و مختلط و همچنین ماتریس‌ها بکار گرفت. دقت کنید عنوان تمام توابع با حروف کوچک می‌باشد.

$$>> sqrt(x)$$

$$>> pow۲(x)$$

$$>> floor(x)$$

$$>> ceil(x)$$

$$>> abs(x)$$

$$>> min(x)$$

$$>> max(x)$$

$$>> min(min(x))$$

$$>> max(max(x))$$

$$>> log(x)$$

$>> log۱۰(x)$

$>> log۲(x)$

$>> rank(x)$

$>> cos(x)$

$>> sin(x)$

$>> sinh(x)$

$>> cosh(x)$

$>> tan(x)$

$>> cot(x)$

$>> sign(x)$

$>> isprime(x)$

$>> rem(x,y)$

$>> mod(x,y)$

$>> sind(x)$

$>> cosd(x)$

$>> gcd(x,y)$

$>> lcm(x,y)$

$>> primes(x)$

$>> nthroot(x,y)$

$>> round(x)$

$>> nextpow۲(x)$

$>> tand(x)$

به مثال‌های زیر توجه کنید

Command Window

```
>> a=sqrt(4)

a =

     2

>> b=sqrt(2)

b =

    1.4142

>> c=sqrt(sqrt(16))

c =

     2

>> d=pow2(
```

pow2(Y)
pow2(F,E)
More Help...

دقت کنید گاهی یک تابع با ورودی های متفاوت می تواند کارهای متفاوتی انجام دهد. برای مثال

Command Window

```
>> d=pow2(3)      %return 2^3

d =

     8

>> f=pow2(3,4)    %returen3*2^4

f =

    48
```

و

Command Window

```
>> d=pow2(3)     %return 2^3

d =

     8

>> f=pow2(3,4)    %returen3*2^4

f =

    48

>> h=power(3,4)

h =

    81
```

همان گونه که اشاره شد برای توابع معرفی شده می توان ورودی های مختلفی را به کار گرفت. برای مثال

Command Window

```
>> A=[1 2 0 ; 4 1 -1]

A =

     1     2     0
     4     1    -1

>> B=pow2(A)

B =

    2.0000    4.0000    1.0000
   16.0000    2.0000    0.5000

>> C=power(A,2)

C =

     1     4     0
    16     1     1
```

و

Command Window

```
A =

     1     2     0
     4     1    -1

>> B=[0 2 3 ; 4 1 2]

B =

     0     2     3
     4     1     2

>> C=power(A,B)

C =

     1     4     0
   256     1     1

>> D=pow2(A,B)

D =

     1     8     0
    64     2    -4
```

و همچنین

Command Window

```
>> a=sqrt(81)

a =

     9

>> b=nthroot(81,2)

b =

     9

>> b=nthroot(81,3)

b =

    4.3267

>> b=nthroot(8,3)

b =

     2
```

با توجه به ماهیت تابع رعایت بعضی ضوابط علم ریاضیات ضروری است. به مثال زیر توجه کنید

```
Command Window
>> A

A =

     1     2     0
     4     1    -1

>> B

B =

     0     2     3
     4     1     2

>> C=nthroot(A,B)
Error using nthroot (line 31)
If X is negative, N must be an odd integer.

fx >>
```

و حال داریم

Command Window

```
>> A

A =

     1     2     0
     4     1     1

>> B

B =

     0     2     3
     4     1     2

>> C=nthroot(A,B)

C =

       NaN    1.4142         0
    1.4142    1.0000    1.0000

fx >>
```

مثال دیگری از توابع

Command Window

```
>> gcd(3,2)

ans =

     1

>> lcm(3,2)

ans =

     6

>> gcd(16,4)

ans =

     4

>> lmc(16,4)
Undefined function or variable 'lmc'.

Did you mean:
>> lcm(16,4)

ans =

    16
```

در این نرم افزار زاویه به صورت پیش فرض بر اساس رادیان می باشد و برای تبدیل آن به درجه باید در $\frac{pi}{۱۸۰}$ ضرب شود. خروجی دستورهای زیر یکسان است؟

$$>> sin(pi/۴)$$

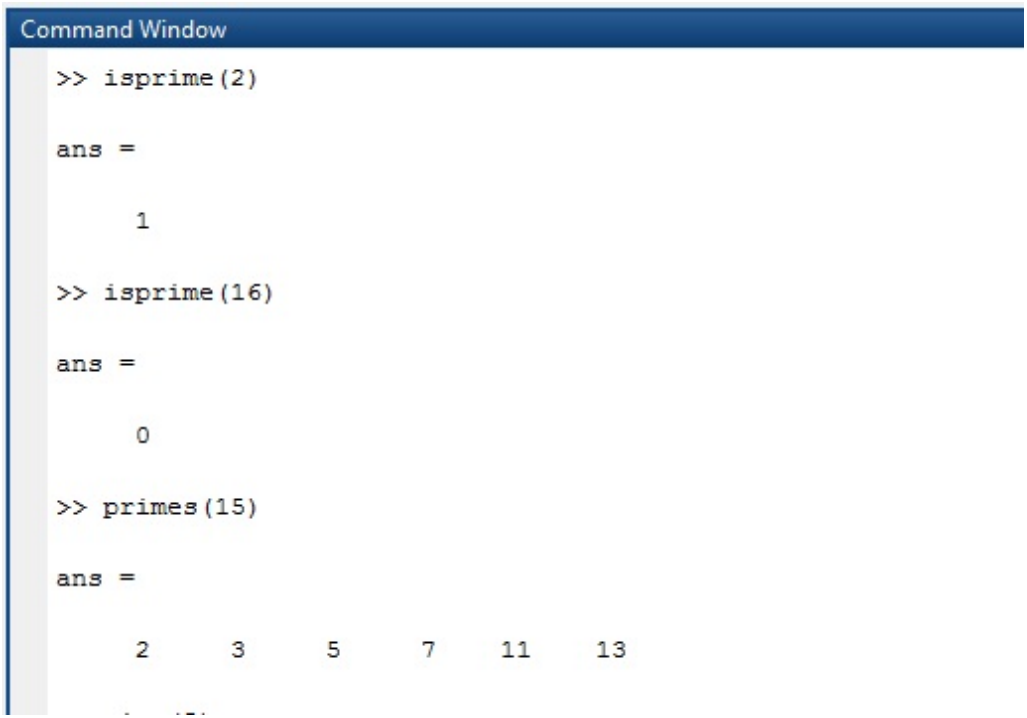
$$>> sin(۴۵)$$

$$>> sind(۴۵)$$

$$>> sin(۴۵ * (pi/۱۸۰))$$


```
Command Window
>> sqrt(2)/2

ans =

    0.7071

>> sin(pi/4)

ans =

    0.7071

>> sin(45)

ans =

    0.8509

>> sind(45)

ans =

    0.7071
```

برای مثال دیگر توجه کنید


Command Window
>> isprime(2)
ans =
1
>> isprime(16)
ans =
0
>> primes(15)
ans =
2 3 5 7 11 13

و

Command Window

```
>> x=-5;
>> a=sign(x)

a =

    -1

>> B=[1 2 -2; 0 1 -5; 2 0 -3]

B =

     1     2    -2
     0     1    -5
     2     0    -3

>> b=sign(B)

b =

     1     1    -1
     0     1    -1
     1     0    -1

fx >>
```

به خطاهای متلب برای توابع زیر دقت کنید

```
Command Window
>> B

B =

     1     2    -2
     0     1    -5
     2     0    -3

>> C=isprime(B)
Error using isprime (line 19)
All entries of X must be nonnegative integers.

>> D=primes(B)
Error using primes (line 16)
N must be a scalar

fx >>
```

نکته

از دستور lookfor ... برای جستجوی دستوراتی مرتبط به یک عمل خاص استفاده می‌شود.

```
Command Window
>> lookfor division
cgdivexpr                - Division Expression object constructor
deconv                   - Deconvolution and polynomial division.
mod                      - Modulus after division.
rem                      - Remainder after division.
idivide                  - Integer division with rounding option.
slash                    - Matrix division.
spcrv                    - Spline curve by uniform subdivision.
gdivide                  - Generalized right division.
nnfcnPerformance         - NNPERFORMANCEFCNINFO Data Division function info.
nnfcnDivision            - NNDIVISIONFCNINFO Data Division function info.
show_et                  - Show table obtained by the Euclidean division algorithm.
fx >>
```

مثال

مقدار عبارت زیر را به ازای $x = ۱۰$ و $y = ۲$ بدست آورید.

$$z = tanh(cosh(x) + tan(\sqrt{x^۲ + y^۷ + log_{۱۰}x^۲ + cot(log_۲ x^۵)})) + \text{باقی مانده}(x/y)$$

۱.۱ ذخیره کردن فعالیت ها

گاهی لازم است کار با MATLAB متوقف شود و کارهای انجام شده حفظ شوند. فایل‌های با پسوند m در پوشه مربوط ذخیره شده‌اند. ولی ممکن است لازم باشد متغیرهای تعریف شده در Command window نیز ذخیره شوند این کار توسط save workspace انجام می‌شود. به این ترتیب می‌نویسیم

$$>> save \text{ matrix or variable of name the}$$

$$>> import \text{ matrix or variable of name the}$$

و اطلاعات به صورت یک فایل mat. ذخیره می‌شوند.

ایجاد کلیدهای میانبر:

در پنجره history Command هر دستوری که مایل باشید از آن کلید میانبر ایجاد کنید می‌توانید با کلیک روی آن و نگه داشتن کلید موس و کشاندن آن به قسمت میله ابزار shortcut و رها کردن آن دکمه میانبر آن دستور را ایجاد کنید.(پس قبل از انجام این کار باید میله ابزار مربوطه را فعال کنید.) در این صورت پنجره‌ای به نام shortcut Add باز می‌شود و در قسمت label یک نام برای آن وارد کنید و سپس بر روی گزینه save کلیک کنید. در این صورت این گزینه را در نوار ابزار shortcut خواهیم داشت و برای حذف آن نیز کلیک راست کرده و delete را انتخاب می‌کنیم.

۲ ماتریس‌ها و بردارها

بردارها و ماتریس‌ها مهمترین متغیرهایی هستند که نرم افزار با آن‌ها کار می‌کند. در واقع MATLAB بر اساس ماتریس‌ها طراحی شده است و بهترین ورودی ست که توابع MATLAB می‌تواند با آن‌ها کند. حتی اعداد را به صورت یک ماتریس یک در یک ذخیره می‌کند. بردارها که می‌توانند به صورت سطری یا ستونی تعریف شوند نیز به صورت یا ماتریس ذخیره می‌شوند.

تصحیح یک فرمان:

اگر در نوشتن یک فرمان اشتباه تایپی داشته باشیم و بخواهیم آن را تصحیحی کنیم لازم نیست از ابتدا آن را تایپ کنیم و با استفاده از کلیدهای ↑ و ↓ می‌توانیم به دستورات قبلی دسترسی داشته باشیم و آن‌ها را دوباره اجرا یا تصحیح کنیم. و با استفاده از کلیدهای → و ← نیز می‌توانیم در دستور تایپ شده به نقطه مورد نظر رفته و تغییرات لازم را ایجاد کنیم.

ماتریس‌ها:

برای تعریف یک بردار به صورت زیر عمل می‌کنیم:

$$x = [۱\ ۲\ ۳\ \ -۴\ ۰\ ۱\ ۹\ \ -۸]$$

$$y = [۲, ۰, -۵, -۶, ۰, ۱, -۱]$$

$$z = [۰; -۵; ۴; -۳; ۲; -۴; ۷; ۸; ۱۰]$$

$$w = [۲; ۱; ۰; -۹; ۸; ۷; ۵; -۵]'$$

برای تعریف یک بردار که مولفه‌های آن به صورت تصاعدی است از روش زیر استفاده می‌کنیم

$$x = a : b : c$$

a مقدار اولیه و c مقدار نهایی و b میزان افزایش است. برای مثال

$$x۱ = ۱ : ۲ : ۱۰ \quad \Longrightarrow \quad x۱ = [۱\ ۳\ ۵\ ۷\ ۹]$$

$$x۲ = ۲ : ۱ : ۷ \quad \Longrightarrow \quad x۲ = [۲\ ۳\ ۴\ ۵\ ۶\ ۷]$$

از طرفی بسادگی می‌توان از ماتریس‌ها ماتریس‌های جدیدی تعریف کرد:

$$y = [۵\ ۲\ ۱۰\ -۱\ ۴۰\ -۱\ -۲\ ۷\ ۶\ -۲]$$

$$y۱ = y(۱ : ۱ : ۵) \quad \Longrightarrow \quad y۱ = [۵\ ۲\ ۱۰\ -۱]$$

$$y۲ = y(۵ : ۲ : end) \quad \Longrightarrow \quad y۲ = [-۱۰\ -۲\ ۶]$$

$$y۳ = y(end : -۱ : ۱) \quad \Longrightarrow \quad y۳ = [-۲\ ۶\ ۷\ -۲\ -۱۰\ ۴\ -۱۰\ ۱\ ۲\ ۵]$$

برای تعریف یک ماتریس به صورت زیر عمل می‌کنیم

$$A = [۱\ ۲\ ۰\ ۴; ۸\ ۰\ ۱\ -۱;\ ۲\ ۱\ ۳\ ۰] \quad \Longrightarrow \quad A = \begin{bmatrix} ۱ & ۲ & ۰ & ۴ \\ ۸ & ۰ & ۱ & -۱ \\ ۲ & ۱ & ۳ & ۰ \end{bmatrix}$$

که یک ماتریس سه در دو می‌باشد.

اعمال روی ماتریس‌ها: اگر بعدهای ماتریس‌ها مناسب باشند می‌توانیم اعمال زیر را انجام دهیم

$$T۱ = A + B$$

$$T۲ = A * C$$

$$T۳ = [A\ B]$$

$$T۴ = [A; B]$$

از جمله عملگرهایی که روی ماتریس‌ها کار می‌کنند به صورت زیر هستند

Operator	Description
+	Addition or unary plus. A+B adds the values stored in variables A and B. A and B must have the same size, unless one is a scalar. A scalar can be added to a matrix of any size.
-	Subtraction or unary minus. A-B subtracts the value of B from A. A and B must have the same size, unless one is a scalar. A scalar can be subtracted from a matrix of any size.

*	Matrix multiplication. C = A*B is the linear algebraic product of the matrices A and B. More precisely, $C(i,j) = \sum_{k=1}^{n} A(i,k)B(k,j)$ For non-scalar A and B, the number of columns of A must be equal to the number of rows of B. A scalar can multiply a matrix of any size.
.*	Array multiplication. A.*B is the element-by-element product of the arrays A and B. A and B must have the same size, unless one of them is a scalar.
/	Slash or matrix right division. B/A is roughly the same as B*inv(A). More precisely, B/A = (A'\B')'.
./	Array right division. A./B is the matrix with elements A(i,j)/B(i,j). A and B must have the same size, unless one of them is a scalar.

\	Backslash or matrix left division. If A is a square matrix, A\B is roughly the same as inv(A)*B, except it is computed in a different way. If A is an n-by-n matrix and B is a column vector with n components, or a matrix with several such columns, then X = A\B is the solution to the equation *AX* = *B*. A warning message is displayed if A is badly scaled or nearly singular.
.\	Array left division. A.\B is the matrix with elements B(i,j)/A(i,j). A and B must have the same size, unless one of them is a scalar.
^	Matrix power. X^p is X to the power p, if p is a scalar. If p is an integer, the power is computed by repeated squaring. If the integer is negative, X is inverted first. For other values of p, the calculation involves eigenvalues and eigenvectors, such that if [V,D] = eig(X), then X^p = V*D.^p/V.
.^	Array power. A.^B is the matrix with elements A(i,j) to the B(i,j) power. A and B must have the same size, unless one of them is a scalar.

'	Matrix transpose. A' is the linear algebraic transpose of A. For complex matrices, this is the complex conjugate transpose.
.'	Array transpose. A.' is the array transpose of A. For complex matrices, this does not involve conjugation.

ممکن است برای بعضی از این عملگرها تابع نیز تعریف شده باشد به عنوان مثال

Function	Description
uplus(a)	Unary plus; increments by the amount a
plus (a,b)	Plus; returns a + b
uminus(a)	Unary minus; decrements by the amount a
minus(a, b)	Minus; returns a - b
times(a, b)	Array multiply; returns a.*b
mtimes(a, b)	Matrix multiplication; returns a* b
rdivide(a, b)	Right array division; returns a ./ b

ldivide(a, b)	Left array division; returns a.\ b
mrdivide(A, B)	Solve systems of linear equations *xA* = *B* for *x*
mldivide(A, B)	Solve systems of linear equations *Ax* = *B* for *x*
power(a, b)	Array power; returns a.^b
mpower(a, b)	Matrix power; returns a ^ b

cumprod(A)	Cumulative product; returns an array of the same size as the array A containing the cumulative product. If A is a vector, then cumprod(A) returns a vector containing the cumulative product of the elements of A. If A is a matrix, then cumprod(A) returns a matrix containing the cumulative products for each column of A. If A is a multidimensional array, then cumprod(A) acts along the first non-singleton dimension.

cumsum(A)	Cumulative sum; returns an array A containing the cumulative sum. If A is a vector, then cumsum(A) returns a vector containing the cumulative sum of the elements of A. If A is a matrix, then cumsum(A) returns a matrix containing the cumulative sums for each column of A. If A is a multidimensional array, then cumsum(A) acts along the first non-singleton dimension.

برای مثال داریم

Command Window

```
>> y=[1 2 3 4 -5]

y =

     1     2     3     4    -5

>> cumprod(y)

ans =

     1     2     6    24  -120

>> cumsum(y)

ans =

     1     3     6    10     5
```

توابع زیادی وجود دارند که مرتبط به ماتریس‌ها هستند و توسط آن‌ها کد نویسی با ماتریس‌ها ساده‌تر می‌شود و بعلاوه با استفاده از آن‌ها می‌توان تا حد امکان از بکار بردن حلقه اجتناب کرد که این امر باعث افزایش سرعت برنامه می‌شود. در ادامه تعدادی از این توابع را معرفی

می‌کنیم: فرض کنیم A یک ماتریس $m \times n$ و x یک بردار سطری یا ستونی باشد.

$$>> [m, n] = size(A)$$

$$>> m۱ = length(x)$$

$$>> m۲ = size(A)$$

$$>> g۱ = A(۲, ۱)$$

$$>> g۲ = A(۳, ۴)$$

$$>> g۳ = A(۱, :)$$

$$>> g۴ = A(:, ۲)$$

$$>> min۰ = min(x)$$

$$>> min۱ = min(A)$$

$$>> minA = min(min(A))$$

$$>> min۲ = min(min۱)$$

$$>> g۵ = A(:, ۳)$$

$$>> g۶ = A(۲ : ۳, ۳ : ۴)$$

$$>> A(۵, :) = []$$

$$>> A(:, ۲) = []$$

$$>> A(:, ۳ : ۴) = []$$

$$>> A(۲, ۳) = ۶$$

$$>> A + ۵$$

$$>> ۲ * A$$

$$>> [l, u] = find(B == ۵)$$

مثال

به مثال زیر درباره تابع find توجه کنید.

Command Window

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

B =

     1     2     3     4
     0     1     1    -2
     1     4     5     7

>> [l,u]=find(B==5)

l =

     3


u =

     3
```

ادامه مثال

Command Window

```
>> [l1,u1]=find(B==1)

l1 =

     1
     3
     2
     2


u1 =

     1
     1
     2
     3
```

ادامه مثال

Command Window

```
>> [l2,u2]=find(B==12)

l2 =

   Empty matrix: 0-by-1


u2 =

   Empty matrix: 0-by-1

>>
```

حال در ادامه مثال دیگری را بیان می‌کنیم و دستورات بیشتری را مرور می‌کنیم.

ادامه مثال

Command Window

```
>> A=[1 2 0 4 5 0 -4; 8 0 1 -1 1 2 3; 2 1 3 0 6 -2 0; -3 2 -1 -1 4 9 11; -2 0 6 4 1 -2 0]

A =

     1     2     0     4     5     0    -4
     8     0     1    -1     1     2     3
     2     1     3     0     6    -2     0
    -3     2    -1    -1     4     9    11
    -2     0     6     4     1    -2     0

>> [m,n]=size(A);
>> m1=length(A);
>> B=A(2:3,3:6);
>> A(3,4)=-7;
>> A(1:2,3:4)=[-1 -1;-2 -2];
>> A+2;
>> A(:,5)=[];
>> A(3:5,1:2)=[0 1; 1 2; 3 4];
>>
```

ادامه مثال

Command Window

```
>> A=[1 2 0 4 5 0 -4; 8 0 1 -1 1 2 3; 2 1 3 0 6 -2 0; -3 2 -1 -1 4 9 11; -2 0 6 4 1 -2 0]

A =

     1     2     0     4     5     0    -4
     8     0     1    -1     1     2     3
     2     1     3     0     6    -2     0
    -3     2    -1    -1     4     9    11
    -2     0     6     4     1    -2     0

>> [m,n]=size(A);
>> m1=length(A);
>> B=A(2:3,3:6);
>> A(3,4)=-7;
>> A(1:2,3:4)=[-1 -1;-2 -2];
>> A+2;
>> A(:,5)=[];
>> A(3:5,1:2)=[0 1; 1 2; 3 4];
>>
```

Command Window

```
>> A(3:5,1:2)=[0 1; 1 2; 3 4];
>> A

A =

     1     2    -1    -1     0    -4
     8     0    -2    -2     2     3
     0     1     3    -7    -2     0
     1     2    -1    -1     9    11
     3     4     6     4    -2     0

>>
```

عملگر . در ماتریس‌ها باعث می‌شود عملیات روی تک تک درایه‌ها انجام شود. فرض کنید A و B دو ماتریس هم سایز باشند.

$$>> A./B \quad \Rightarrow \quad [a_{۱۱}./b_{۱۱},\ a_{۱۲}./b_{۱۲},\ a_{۱۳}./b_{۱۳},\ ...]$$

$$>> A.\hat{}۲$$

$$>> A.\hat{}-۱$$

$$>> ۲.\hat{}A$$

$$>> A'$$

$$>> det(A)$$

$$>> inv(A)$$

$$>> ones(۳)$$

$$>> ones(۳,۵)$$

$$>> zeros(۳)$$

$$>> eye(۳)$$

$$>> eye(۴,۲)$$

ادامه توابع مرتبط با ماتریس‌ها

$$>> W = rand(۵)$$

$$>> w۱ = randn(۴)$$

$$>> w۲ = rand(۳, ۱)$$

$$>> w۳ = randperm(۸)$$

$$>> [xs, ind] = sort(x)$$

$$>> diag(x)$$

$$>> diag(x, ۱)$$

$$>> diag(x, -۲)$$

$$>> a = rand$$

$$>> b = randn$$

$$>> c = randi([۲۰۵۰], ۱, ۵)$$

$$>> d = randi([st], m, n)$$

$$>> e = rand + i * rand$$

در حالتی که n عدد تصادفی در بازه (a, b) بخواهیم به صورت زیر عمل می‌کنیم:

$$r = a + (b - a) * rand(n, ۱)$$

اگر بخواهیم یک ماتریس سه بعدی به صورت تصادفی تولید کنیم که دارای دو سطر و چهار ستون و سه لایه باشد به طریق زیر عمل می‌کنیم:

$$X = rand([۲,\ ۴,\ ۳])$$

و برای اینکه ماتریس قابل نمایش در صفحه کار باشد ماتریس مربوط به هر لایه را بصورت جداگانه نمایش می‌دهد. این ماتریس‌ها مثلا مواقعی که بخواهیم مکان یک جسم را در فضای سه بعدی نشان دهیم کاربرد دارند. چون مکان هر جسم در فضای سه بعدی دارای سه مولفه طول، عرض و ارتفاع می‌باشد.

```
Command Window
>> X=rand([2 4 3])

X(:,:,1) =

    0.6787    0.7431    0.6555    0.7060
    0.7577    0.3922    0.1712    0.0318


X(:,:,2) =

    0.2769    0.0971    0.6948    0.9502
    0.0462    0.8235    0.3171    0.0344


X(:,:,3) =

    0.4387    0.7655    0.1869    0.4456
    0.3816    0.7952    0.4898    0.6463

fx >>
```

معرفی دستور diag

درباره دستور diag قبلا صحبت کرده‌ایم. اگر x یک بردار n تایی باشد دستور diag(x) یک ماتریس $n \times n$ بر می‌گرداند که این بردار قطر اصلی آن می‌باشد. و اگر به صورت $diag(x, ۱)$ بنویسیم یک ماتریس $(n+۱) \times (n+۱)$ بر می‌گرداند که بردار x یک قطر بالای قطر اصلی قرار می‌گیرد. حال اگر A یک ماتریس باشد حاصل diag(A) چه می‌شود؟

توابع دیگر روی ماتریس‌ها:

$$>> W = diag(A)$$

$$>> w۱ = diag(A, ۱)$$

$$>> w۲ = diag(A, -۲)$$

$$>> temp = isdiag(A)$$

$$>> S = tril(A)$$

$$>> S۱ = triu(A)$$

$$>> temp۱ = istril(A)$$

$$>> temp۲ = istriu(A)$$

مثال

ماتریس A را به صورت زیر تعریف کنید و موارد خواسته شده را تعیین کنید.

$$A = \begin{bmatrix} ۱ & ۲ & ۰ & ۴ & ۵ & ۰ & -۴ \\ ۸ & ۰ & ۱ & -۱ & ۱ & ۲ & ۳ \\ ۲ & ۱ & ۳ & ۰ & ۶ & -۲ & ۰ \\ -۳ & ۲ & -۱ & -۱ & ۴ & ۹ & ۱۱ \\ -۲ & ۰ & ۶ & ۴ & ۱ & -۲ & ۰ \end{bmatrix}$$

$$b = A(۳ : -۱ : ۱, ۱ : ۳)$$

$$b۱ = tril(A)$$

$$b۲ = triu(A)$$

$$b۳ = diag(A)$$

$$b۴ = diag(A, ۲)$$

$$b۵ = diag(A, -۱)$$

حاصل می شود

```
Command Window
>> A=[1 2 0 4 5 0 -4; 8 0 1 -1 1 2 3; 2 1 3 0 6 -2 0; -3 2 -1 -1 4 9 11; -2 0 6 4 1 -2 0]

A =

     1     2     0     4     5     0    -4
     8     0     1    -1     1     2     3
     2     1     3     0     6    -2     0
    -3     2    -1    -1     4     9    11
    -2     0     6     4     1    -2     0

>> b=A(3:-1:1,1:3)

b =

     2     1     3
     8     0     1
     1     2     0
```

و

Command Window

```
>> b1=tril(A)

b1 =

     1     0     0     0     0     0     0
     8     0     0     0     0     0     0
     2     1     3     0     0     0     0
    -3     2    -1    -1     0     0     0
    -2     0     6     4     1     0     0

>> b2=triu(A)

b2 =

     1     2     0     4     5     0    -4
     0     0     1    -1     1     2     3
     0     0     3     0     6    -2     0
     0     0     0    -1     4     9    11
     0     0     0     0     1    -2     0
```

ادامه

Command Window

```
>> b3=diag(A)

b3 =

     1
     0
     3
    -1
     1

>> b4=diag(A,2)

b4 =

     0
    -1
     6
     9
     0
```

و

Command Window

```
>> b5=diag(A,-1)

b5 =

     8
     1
    -1
     4

>> b6=diag(b4)

b6 =

     0     0     0     0     0
     0    -1     0     0     0
     0     0     6     0     0
     0     0     0     9     0
     0     0     0     0     0
```

مثال

ماتریس A تعریف شده در قبل را در نظر بگیرید دستور زیر را تعبیر کنید.

$$>> c = [۱,\ ۳,\ ۱,\ ۱]$$

$$>> ac = A(c, c)$$

حاصل به صورت زیر است:

Command Window

```
A =

     1     2     0     4     5     0    -4
     8     0     1    -1     1     2     3
     2     1     3     0     6    -2     0
    -3     2    -1    -1     4     9    11
    -2     0     6     4     1    -2     0

>> c=[1 3 1 1]

c =

     1     3     1     1

>> ac=A(c,c)

ac =

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

ادامه کار با ماتریس‌ها و بردارها

$$>> y = [۱,\ ۵,\ ۹,\ ۷,\ ۸,\ -۶,\ -۴]$$

$$>> x = [y(۱),\ y(۲),\ y(۶),\ y(۷)]$$

$$>> x = y([۱,\ ۲,\ ۶,\ ۷])$$

$$>> y۱ = [-۱,\ ۶,\ ۱۵,\ -۷,\ ۳۱,\ ۲,\ -۴,\ -۵]$$

$$>> indexx = find(y۱ <= ۰) \quad \Longrightarrow \quad indexx = [۱,\ ۴,\ ۷,\ ۸]$$

$$>> s = y۱(indexx) \quad \Longrightarrow \quad s = [-۱,\ -۷,\ -۴,\ -۵]$$

دو دستور انتهایی را می‌توان با یک دستور نیز نوشت

$$>> s = y۱(find(y۱ <= ۰)) \quad \Longrightarrow \quad s = [-۱,\ -۷,\ -۴,\ -۵]$$

تولید ماتریس جادویی:

$$>> z = magic(۴)$$

Command Window

```
>> M1=magic(4)

M1 =

    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1

>>
```

تولید ماتریس‌های پیچیده تر از ماتریس‌های از پیش تعریف شده:

$$>> a = [۱,\ ۲;\ ۳,\ ۴;\ ۵,\ ۶]$$

$$>> m۱ = a(:)$$

$$>> m۲ = reshape(m, ۲, ۳)$$

$$>> m۳ = reshape(m, ۱, ۶)$$

$$>> m۴ = reeshape(m, ۲, ۴)$$

که حاصل به صورت زیر می‌باشد:

Command Window

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

a =

     1     2
     3     4
     5     6

>> m1=a(:)

m1 =

     1
     3
     5
     2
     4
     6
```

Command Window

```
>> m2=reshape(a,2,3)

m2 =

     1     5     4
     3     2     6

>> m3=reshape(a,1,6)

m3 =

     1     3     5     2     4     6

>> m4=reshape(a,2,4)
Error using reshape
To RESHAPE the number of elements must not change.
```

ماتریس‌های تکراری:

فرض کنید x یک بردار باشد r و c دو عدد طبیعی باشند

$$>> a = [۱,\ -۲,\ ۰,\ ۲,\ -۴]$$

$$>> a۱ = repmat(a, ۳, ۱)$$

$$>> a۲ = repmat(a', ۱, ۳)$$

$$>> a۳ = repmat(a, ۲, ۳)$$

که حاصل می شود

Command Window

```
>> a=[1 -2 0 2 -4]

a =

     1    -2     0     2    -4

>> a1=repmat(a,3,1)

a1 =

     1    -2     0     2    -4
     1    -2     0     2    -4
     1    -2     0     2    -4

>> a3=repmat(a,2,3)

a3 =

     1    -2     0     2    -4     1    -2     0     2    -4     1    -2     0     2    -4
     1    -2     0     2    -4     1    -2     0     2    -4     1    -2     0     2    -4
```

و

Command Window

```
>> a2=repmat(a',3,1)

a2 =

     1
    -2
     0
     2
    -4
     1
    -2
     0
     2
    -4
     1
    -2
     0
     2
    -4
```

توسط دو بردار s و t و دستور زیر می‌توان دو ماتریس تولید کرد

$$>> [u,v] = meshgrid(s,t) \quad \Longrightarrow \quad u = repmat(s, length(t), ۱)$$

$$v = repmat(t', ۱, length(s))$$

مثال

ماتریس‌های u و v چه ابعادی دارند

$$>> s = [۱,\ ۲,\ ۳,\ ۴]$$

$$>> t = [-۵,\ -۶,\ -۷]$$

$$>> [u,v] = meshgrid(s,t)$$

حاصل می‌شود

Command Window

```
>> [u,v]=meshgrid(s,t)

u =

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


v =

    -5    -5    -5    -5
    -6    -6    -6    -6
    -7    -7    -7    -7
```

توابع دیگر روی ماتریس‌ها

$>> fix(a)$

$>> fix(A)$

$>> min(A)$

$>> min(a)$

$>> min(min(A))$

$>> sum(a)$

$>> sum(A)$

$>> sum(sum(A))$

$>> sort(a)$

$>> sort(A)$

$>> mean(a)$

$>> mean(A)$

$>> mean(mean(A))$

$>> prod(A)$

$>> prod(a)$

$>> sort(a)$

$>> sort(A)$

$>> trace(A) \iff sum(diag(A))$

$>> dot(a, b)$

$>> dot(A, B)$

تمرین

الف- دو بردار دلخواه x و y با ابعاد یکسان ۱۰ × ۱ بسازید. حال بردارهای $xnew$ و $ynew$ را چنان بسازید که عناصر $xnew$ از کوچک به بزرگ مرتب شده باشند و عناصر $ynew$ متناظر با درایه‌های $xnew$ باشند یعنی هر درایه‌ای که در $xnew$ نسبت به x جایش عوض شد درایه متناظرش در y هم جایش عوض شود.

ب- درایه‌های منفی از بردار x و y را در کنار هم در بردار z ذخیره کنید.

ج- ماتریس زیر را تعیین کنید.

$$d = [۳ : ۲ : ۱۱; linspace(۲۰, ۲۱, ۵); ones(۱, ۵)]$$

د- یک ماتریس جادویی پنج در پنج ایجاد کنید و قطر اصلی آن را صفر قرار دهید.

ه- دستورات زیر چه عملیاتی انجام می‌دهد.

$$>> z = magic(۴)$$

$$>> z۱ = z - diag(diag(z)) + ۵ * eye(۴)$$

$$>> z۲ = z - diag(diag(z)) + diag([-۷, \ -۸, \ ۱۱, \ ۲۴])$$

نکته: سعی کنید عملیات را با کمترین تعداد دستور انجام دهید.

محاسبه مقادیر ویژه یک ماتریس:

$$>> e = eig(A)$$

$$>> [V, D] = eig(A)$$

```
Command Window
>> H=[3 -2 0; 2 -2 0; 0 1 1]

H =

     3    -2     0
     2    -2     0
     0     1     1

>> e1=eig(H)

e1 =

     1
    -1
     2
```

Command Window

```
>> [u,r]=eig(H)

u =

         0   -0.4082   -0.8165
         0   -0.8165   -0.4082
    1.0000    0.4082   -0.4082


r =

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

حل دستگاه $Ax = b$ به صورت زیر می‌باشد

$$x = A \setminus b$$

دستورات زیر چه عملیاتی انجام می‌دهد:

$$>> w۱ = repmat(۴۵/۷۲,\ ۱\ , ۶)$$

$$>> w۲ = repmat(۴۵/۲۷,\ ۲,\ ۳)$$

$$>> a = [۱۲; ۳۴]$$

$$>> w۲ = repmat(a, ۲, ۳)$$

کاربرد علامت‌های $<$ ، $>$ ، $\leq$ ، $\geq$ و $=$ برای ماتریس ها و بردارها بعنوان علامت‌های منطقی

$$>> ۱ > ۰$$

$$>> -۲ > ۰$$

$$>> a > ۰$$

$$>> [۱,\ ۲,\ ۰,\ -۳,\ ۴] > ۰$$

$$>> [۰,\ ۳,\ ۲,\ ۵,\ -۵,\ ۲,\ -۱] <= ۰$$

$$>> a > b$$

و

$$B = \begin{bmatrix} ۱ & ۲ & ۰ & ۴ \\ ۸ & ۰ & ۱ & -۱ \\ ۲ & ۱ & ۳ & ۰ \\ -۳ & ۲ & -۱ & -۱ \end{bmatrix} >= ۰$$

جابجایی درایه‌های یک ماتریس :

$$A = \begin{bmatrix} a_{۱۱} & a_{۱۲} & a_{۱۳} & a_{۱۴} & a_{۱۵} \\ a_{۲۱} & a_{۲۲} & a_{۲۳} & a_{۲۴} & a_{۲۵} \end{bmatrix} \quad \Longrightarrow \quad fliplr(A) = \begin{bmatrix} a_{۱۵} & a_{۱۴} & a_{۱۳} & a_{۱۲} & a_{۱۱} \\ a_{۲۵} & a_{۲۴} & a_{۲۳} & a_{۲۲} & a_{۲۱} \end{bmatrix}$$

$$flipud(A) = \begin{bmatrix} a_{۲۱} & a_{۲۲} & a_{۲۳} & a_{۲۴} & a_{۲۵} \\ a_{۱۱} & a_{۱۲} & a_{۱۳} & a_{۱۴} & a_{۱۵} \end{bmatrix}$$

$$flipud(fliplr(A)) = \begin{bmatrix} a_{۲۵} & a_{۲۴} & a_{۲۳} & a_{۲۲} & a_{۲۱} \\ a_{۱۵} & a_{۱۴} & a_{۱۳} & a_{۱۲} & a_{۱۱} \end{bmatrix}$$

$$>> v = [v_1,\ v_2,\ v_3, ..., v_n]$$

$$>> v1 = cumsum(v) = [\sum_{k=1}^{1} v_k,\ \sum_{k=1}^{2} v_k,\ \sum_{k=1}^{3} v_k,\ ...,\sum_{k=1}^{n} v_k]$$

و برای ماتریس‌ها به صورت زیر می شود

$$>> V = \begin{bmatrix} v_{11} & v_{12} & v_{13} & \dots & v_{1n} \\ v_{21} & v_{22} & v_{23} & \dots & v_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ v_{m1} & v_{m2} & v_{m3} & \dots & v_{mn} \end{bmatrix}$$

$$\Longrightarrow \quad cumsum(V) = \begin{bmatrix} \sum_{k=1}^{1} v_{k1} & \sum_{k=1}^{1} v_{k2} & \sum_{k=1}^{1} v_{k3} & \dots & \sum_{k=1}^{1} v_{kn} \\ \sum_{k=1}^{2} v_{k1} & \sum_{k=1}^{2} v_{k2} & \sum_{k=1}^{2} v_{k3} & \dots & \sum_{k=1}^{2} v_{kn} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum_{k=1}^{m} v_{k1} & \sum_{k=1}^{m} v_{k2} & \sum_{k=1}^{m} v_{k3} & \dots & \sum_{k=1}^{m} v_{kn} \end{bmatrix}$$