فصل ۳

عملیات ریاضی با آرایه ها

MATLAB An Introduction With Applications, 6th Edition
Dr. Amos Gilat
The Ohio State University

Slide deck by
Dr. Greg Reese
Miami University

در فصول قبل به اسکالرها (اعداد تکی) پرداختیم. در این فصل با آرایه ها که در حالت کلی شامل چند عدد هستند کار خواهیم کرد.

این فصل شامل مقدمات استفاده از آرایه ها است.

آرایه – چیدمان مستطیلی از اعداد که میتواند یک بعد یا بیشتر داشته باشد

بردار – آرایه ای با یک ردیف یا ستون

اسکالر – آرایه با تنها یک ردیف و یک ستون، یعنی یک عدد تکی

نکته – «ماتریس» و «آرایه» معمولاً به جای هم به کار می روند

- از + برای جمع دو آرایه یا جمع یک اسکالر با یک آرایه استفاده کنید
- از – برای کم کردن یک آرایه از دیگری یا کم کردن یک اسکالر از یک آرایه استفاده کنید
  - اگر بخواهید دو آرایه را جمع یا از هم کم کنید، باید دارای ابعاد یکسان باشند (تعداد ردیف و ستون برابر)
  - بردارها نه فقط تعداد درایه برابر، بلکه باید ابعاد (تعداد ردیف و ستون) یکسان داشته باشند

زمانی که دو آرایه A و B را جمع میکنید، MATLAB درایه های متناظر را جمع میکند، یعنی

- درایه ردیف اول و ستون اول A را با درایه ردیف اول و ستون اول B جمع میکند
- درایه ردیف اول و ستون دوم A را با درایه ردیف اول و ستون دوم B جمع میکند، و به همین ترتیب تا آخر

این روند جمع درایه ای نامیده می شود

زمانی که دو آرایه A و B را از هم کم میکنید، MATLAB درایه های متناظر را از هم کم میکند.

به طور کلی، هر عملیاتی با دو آرایه که روی درایه های در محل یکسان از هر دو انجام شود، عملیات درایه ای نامیده می شود.

مثال

اگر $A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \end{bmatrix}$ و $B = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \end{bmatrix}$

$$A + B = \begin{bmatrix} A_{11} + B_{11} & A_{12} + B_{12} & A_{13} + B_{13} \\ A_{21} + B_{21} & A_{22} + B_{22} & A_{23} + B_{23} \end{bmatrix}$$

$$A - B = \begin{bmatrix} A_{11} - B_{11} & A_{12} - B_{12} & A_{13} - B_{13} \\ A_{21} - B_{21} & A_{22} - B_{22} & A_{23} - B_{23} \end{bmatrix}$$

زمانی که یک اسکالر را با یک آرایه جمع میکنید، MATLAB آن عدد را با هر درایه آرایه جمع میکند

زمانی که یک اسکالر را از یک آرایه کم میکنید، MATLAB آن عدد را از هر درایه آن آرایه کم میکند

مثال

اگر c یک اسکالر و $A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \end{bmatrix}$

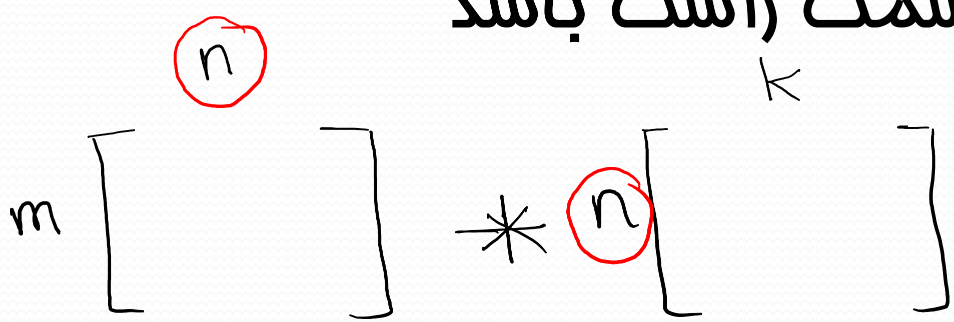
$$A + c = \begin{bmatrix} A_{11} + c & A_{12} + c & A_{13} + c \\ A_{21} + c & A_{22} + c & A_{23} + c \end{bmatrix}$$

$$A - c = \begin{bmatrix} A_{11} - c & A_{12} - c & A_{13} - c \\ A_{21} - c & A_{22} - c & A_{23} - c \end{bmatrix}$$

دو راه برای ضرب ماتریس ها وجود دارد – ضرب ماتریسی و ضرب درایه های متناظر

ضرب ماتریسی

- همان ضربی که در جبر خطی انجام میشود
- MATLAB آن را با ستاره (*) نشان می دهد
- تعداد ستونهای ماتریس سمت چپ باید برابر تعداد ردیفهای ماتریس سمت راست باشد



۳×۲
۲×۴ → ۳×۴

For example, if A is a 4×3 matrix and B is a 3×2 matrix:

$$A = \begin{array}{cc} & \text{٣} \\ \text{٤} & \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \\ A_{41} & A_{42} & A_{43} \end{bmatrix} \end{array} \quad \text{and} \quad B = \begin{array}{cc} & \text{٢} \\ \text{٣} & \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \end{bmatrix} \end{array} \longrightarrow \begin{array}{cc} & \text{٢} \\ \text{٤} & \begin{bmatrix} \quad & \quad \end{bmatrix} \end{array}$$

then the matrix that is obtained with the operation $A*B$ has dimensions 4×2 with the elements:

$$\begin{bmatrix} (A_{11}B_{11} + A_{12}B_{21} + A_{13}B_{31}) & (A_{11}B_{12} + A_{12}B_{22} + A_{13}B_{32}) \\ (A_{21}B_{11} + A_{22}B_{21} + A_{23}B_{31}) & (A_{21}B_{12} + A_{22}B_{22} + A_{23}B_{32}) \\ (A_{31}B_{11} + A_{32}B_{21} + A_{33}B_{31}) & (A_{31}B_{12} + A_{32}B_{22} + A_{33}B_{32}) \\ (A_{41}B_{11} + A_{42}B_{21} + A_{43}B_{31}) & (A_{41}B_{12} + A_{42}B_{22} + A_{43}B_{32}) \end{bmatrix}$$

A numerical example is:

$$\begin{bmatrix} 1 & 4 & 3 \\ 2 & 6 & 1 \\ 5 & 2 & 8 \end{bmatrix} \begin{bmatrix} 5 & 4 \\ 1 & 3 \\ 2 & 6 \end{bmatrix} = \begin{bmatrix} (1 \cdot 5 + 4 \cdot 1 + 3 \cdot 2) & (1 \cdot 4 + 4 \cdot 3 + 3 \cdot 6) \\ (2 \cdot 5 + 6 \cdot 1 + 1 \cdot 2) & (2 \cdot 4 + 6 \cdot 3 + 1 \cdot 6) \\ (5 \cdot 5 + 2 \cdot 1 + 8 \cdot 2) & (5 \cdot 4 + 2 \cdot 3 + 8 \cdot 6) \end{bmatrix} = \begin{bmatrix} 15 & 34 \\ 18 & 32 \\ 43 & 74 \end{bmatrix}$$

مثال:

```
>> A = [ 1 4 3; 2 6 1; 5 2 8 ]
A = 1  1    4    3
       2    6    1
    3  5    2    8
>> B = [ 5 4; 1 3; 2 6 ]
B =  1     2
     5     4
     1     3
     2     6
>> A * B
ans =  15     34
       18     32     ans(1,1)
       43     74     ans(3,2)
```

3x2 3X3

مثال:

دقت کنید که B*A تعریف نشده است، زیرا تعداد ستونهای B برابر تعداد ردیفهای A نیست.

اگر بخواهید B*A را به دست آورید منجر به پیغام خطا خواهد شد:

```
>> B * A
Error using  *
Inner matrix dimensions must agree.
```

زمانی که دو ماتریس مربعی را در هم ضرب کنید

- باید دارای ابعاد یکسان باشند
- حاصلضرب ماتریسی با همان ابعاد خواهد بود
- به طور کلی، ضرب از قانون جابجایی پیروی نمی کند، یعنی $A * B \neq B * A$

```
>> A = randi(3,3)
A =
     3     3     1
     3     2     2
     1     1     3
>> B=randi(3,3)
B =
     3     3     1
     1     2     2
     3     3     3
>> AB = A*B
AB =
    15    18    12
    17    19    13
    13    14    12
```

```
>> BA = B*A
BA =
    19    16    12
    11     9    11
    21    18    18
>> AB == BA
ans =
     0     0     1
     0     0     0
     0     0     0
```

زمانی که دو بردار را در هم ضرب کنید

- باید دارای اندازه برابر باشند
- یکی باید سطری و دیگری ستونی باشد
- اگر بردار سمت چپ سطری باشد، حاصلضرب یک اسکالر خواهد بود
- اگر بردار سمت راست سطری باشد، حاصلضرب یک ماتریس مربعی خواهد بود که اندازه ضلع آن با اندازه بردارها برابر است

$$\underset{1\times 3}{\begin{bmatrix} 2 & 4 & 6 \end{bmatrix}} \underset{3\times 1}{\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}} = 4$$

```
>> h = [ 2 4 6 ]
h =
      2        4        6
>> v = [ -1 0 1 ]'
v =
     -1
      0
      1
```

```
>> h * v
ans =
      4
>> v * h
ans =
     -2       -4       -6
      0        0        0
      2        4        6
```

$$\underset{3\times 1}{\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}} \underset{1\times 3}{\begin{bmatrix} 2 & 4 & 6 \end{bmatrix}} =$$

dot(a,b) ضرب داخلی یا ضرب نقطه ای را حساب میکند

- a و b باید دارای اندازه برابر باشند
- هر ترکیبی از بردارهای سطری و ستونی مجاز است
- حاصل همیشه اسکالر خواهد بود

مثال

```
>> h = [ 2 4 6 ]
h =
     2     4     6
>> v = [ -1 0 1 ]'
v =
    -1
     0
     1
>> dot(h,v)
ans =
     4
>> dot(v,h)
ans = 4
```

Linear algebra rules of array multiplication provide a convenient way for writing a system of linear equations. For example, the following system of three equations with three unknowns:

$$\begin{aligned} A_{11}x_1 + A_{12}x_2 + A_{13}x_3 &= B_1 \\ A_{21}x_1 + A_{22}x_2 + A_{23}x_3 &= B_2 \\ A_{31}x_1 + A_{32}x_2 + A_{33}x_3 &= B_3 \end{aligned}$$

can be written in a matrix form by:

$$\overset{3\times3}{\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}} \overset{3\times1}{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}} = \overset{3\times1}{\begin{bmatrix} B_1 \\ B_2 \\ B_3 \end{bmatrix}}$$

and in matrix notation by:

$$AX = B \quad \text{where } A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}, \; X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \text{ and } B = \begin{bmatrix} B_1 \\ B_2 \\ B_3 \end{bmatrix}.$$

ماتریس واحد

- یک ماتریس مربعی است که درایه های روی قطر آن یک و بقیه صفر هستند
  - قطر اصلی از بالا سمت چپ به پایین سمت راست است
- زمانی که ماتریس واحد در یک ماتریس یا بردار ضرب شود، آن ماتریس یا بردار تغییر نخواهد کرد
  - ماتریس واحد می تواند در سمت چپ یا راست ضرب باشد
- دستور eye(n) یک ماتریس واحد $n \times n$ می سازد

ماتریس معکوس :

ماتریس B معکوس ماتریس A است اگر حاصلضرب A و B ماتریس واحد I باشد

- هر دو ماتریس باید مربعی و با ابعاد یکسان باشند
- ضرب می تواند از هر طرف انجام شود، یعنی

$$BA = AB = I$$

مثال

$$\begin{bmatrix} 2 & 1 & 4 \\ 4 & 1 & 8 \\ 2 & -1 & 3 \end{bmatrix} \begin{bmatrix} 5.5 & -3.5 & 2 \\ 2 & -1 & 0 \\ -3 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 5.5 & -3.5 & 2 \\ 2 & -1 & 0 \\ -3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 4 \\ 4 & 1 & 8 \\ 2 & -1 & 3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

در ریاضیات، معکوس ماتریس A را با A^{-1} نشان می دهند

در MATLAB، معکوس را با `A^-1` یا `inv(A)` به دست آورید

```
>> A=[2 1 4; 4 1 8; 2 -1 3]
A =
     2     1     4
     4     1     8
     2    -1     3
>> B=inv(A)
B =
    5.5000   -3.5000    2.0000
    2.0000   -1.0000         0
   -3.0000    2.0000   -1.0000
>> A*B
ans =
     1     0     0
     0     1     0
     0     0     1
```

Creating the matrix A.

Use the `inv` function to find the inverse of A and assign it to B.

Multiplication of A and B gives the identity matrix.

دترمینان:

دترمینان تابعی مرتبط با ماتریسهای مربعی است

- در ریاضیات، دترمینان A را به صورت $det(A)$ یا $|A|$ نشان می دهند
- در MATLAB، دترمینان A را با `det(A)` حساب کنید
- یک ماتریس تنها زمانی معکوس پذیر است که مربعی باشد و دترمینان آن صفر نباشد

اگر با دترمینان آشنا نیستید، به کتابهای جبر خطی مراجعه کنید

تقسیم بر چپ، \:

تقسیم بر چپ یکی از دو نوع تقسیم آرایه ها در MATLAB است

- برای حل معادله ماتریسی $AX=B$ به کار میرود
  - A یک ماتریس مربعی و X و B بردارهای ستونی هستند
  - پاسخ $X = A^{-1}B$ است

در MATLAB حل معادله با استفاده از عملگر تقسیم بر چپ (\) انجام میشود، یعنی

```
>> X = A \ B
```

برای حل دستگاه معادلات از تقسیم بر چپ، و نه ماتریس معکوس استفاده کنید. یعنی از `X=A\B` استفاده کنید، نه `X=inv(A)*B`

تقسیم بر چپ

- ۲-۳ برابر سریع تر است
- اغلب خطای کمتری نسبت به `inv()` دارد
- بعضی وقتها `inv()` جوابهای نادرست می دهد

تقسیم بر راست، /:

تقسیم بر راست نوع دیگر تقسیم آرایه ها در MATLAB است

- برای حل معادله ماتریسی $XC=D$ به کار میرود
  - C یک ماتریس مربعی و X و D بردارهای سطری هستند
  - پاسخ $X = D \cdot C^{-1}$ است

در MATLAB برای حل آن از عملگر تقسیم بر راست (/) استفاده کنید، یعنی

```
>> X = D / C
```

نوع دیگر عملیات درایه ای، عملیات درایه به درایه است

- جمع و تفاضل آرایه ها همیشه درایه ای است
- ضرب، تقسیم و به توان رساندن آرایه ها می تواند درایه ای باشد
- هر دو آرایه باید ابعاد یکسان داشته باشند

ضرب، تقسیم و توان درایه ای را با قرار دادن یک نقطه در جلوی عملگر حسابی انجام دهید

Symbol	**Description**	**Symbol**	**Description**
.*	Multiplication	./	Right division
.^	Exponentiation	.\	Left Division

If two vectors a and b are a=[a_1 a_2 a_3 a_4] and b=[b_1 b_2 b_3 b_4], then element-by-element multiplication, division, and exponentiation of the two vectors gives:

$$a.*b = [\,a_1*b_1 \;\; a_2*b_2 \;\; a_3*b_3 \;\; a_4*b_4\,]$$

$$a./b = [\,a_1/b_1 \;\; a_2/b_2 \;\; a_3/b_3 \;\; a_4/b_4\,]$$

$$a.\hat{}\,b = \left[(a_1)^{b_1} \;\; (a_2)^{b_2} \;\; (a_3)^{b_3} \;\; (a_4)^{b_4}\right]$$

If two matrices A and B are

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \text{ and } B = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix}$$

then element-by-element multiplication and division of the two matrices give:

$$A .* B = \begin{bmatrix} A_{11}B_{11} & A_{12}B_{12} & A_{13}B_{13} \\ A_{21}B_{21} & A_{22}B_{22} & A_{23}B_{23} \\ A_{31}B_{31} & A_{32}B_{32} & A_{33}B_{33} \end{bmatrix} \qquad A ./ B = \begin{bmatrix} A_{11}/B_{11} & A_{12}/B_{12} & A_{13}/B_{13} \\ A_{21}/B_{21} & A_{22}/B_{22} & A_{23}/B_{23} \\ A_{31}/B_{31} & A_{32}/B_{32} & A_{33}/B_{33} \end{bmatrix}$$

Element-by-element exponentiation of matrix A gives:

$$\mathrm{A} .\hat{}\, n = \begin{bmatrix} (A_{11})^n & (A_{12})^n & (A_{13})^n \\ (A_{21})^n & (A_{22})^n & (A_{23})^n \\ (A_{31})^n & (A_{32})^n & (A_{33})^n \end{bmatrix}$$

Element-by-element multiplication, division, and exponentiation are demonstrated in Tutorial 3-2.

ضرب درایه ای

- برای ضرب درایه ای از *. استفاده کنید (دقت کنید که نقطه قبل از ستاره است)
- هر دو ماتریس باید ابعاد یکسانی داشته باشند

```
>> A = [1 2; 3 4];
>> B = [0 1/2; 1 -1/2];
>> C = A .* B
>> C =
      0  1
      3 -2
```

در ضرب درایه ای، اگر ماتریسها دارای ابعاد یکسان نباشند MATLAB پیغام خطا می دهد

```
>> A = [ 1 2; 3 4];
>> B = [1 0]';
>> A .* B % Meant matrix multiplication!
??? Error using ==> times
Matrix dimensions must agree.
>> A * B % this works
ans =
     1
     3
```

مراقب باشید – زمانی که ماتریسهای مربعی را ضرب می کنید

- هر دو نوع ضرب همیشه درست است
- اگر از عملگر نادرست استفاده کنید، MATLAB محاسبه نادرست را انجام می دهد و پیغام خطایی در کار نخواهد بود!
  - پیدا کردن این نوع اشتباه سخت است

مثال

```
>> A = [1 2; 3 4];
>> B = [0 1/2; 1 -1/2];
>> A .* B
>> ans
      0  1
      3 -2
>> A * B
ans =
      2.0000  -0.5000
      4.0000  -0.5000
```

محاسبات درایه ای برای محاسبه مقدار یک تابع به ازای مقادیر مختلف ورودی مناسب هستند

```
>> x=[1:8]

x =
    1    2    3    4    5    6    7    8

>> y=x.^2-4*x

y =
   -3   -4   -3    0    5   12   21   32
>>
```

Create a vector x with eight elements.

Vector x is used in element-by-element calculations of the elements of vector y.

توابع داخلی MATLAB میتوانند آرایه ها را به عنوان ورودی بگیرند

- زمانی که ورودی آرایه باشد، نتیجه به دست آمده آرایه ای با اندازه برابر خواهد بود که هر درایه آن حاصل عمل تابع روی درایه ورودی متناظر است

```
>> x=[0:pi/6:pi]

x =
     0    0.5236    1.0472    1.5708    2.0944    2.6180    3.1416

>>y=cos(x)

y =
   1.0000    0.8660    0.5000    0.0000   -0.5000   -0.8660   -1.0000
>>
```

مثال برای ورودی آرایه ای

```
>> d=[1 4 9; 16 25 36; 49 64 81]
d =
     1     4     9
    16    25    36
    49    64    81
>> h=sqrt(d)
h =
     1     2     3
     4     5     6
     7     8     9
```

Creating a 3×3 array.

h is a 3×3 array in which each element is the square root of the corresponding element in array d.

برداری کردن قابلیت MATLAB در استفاده از آرایه ها به عنوان ورودی توابع است. محاسبات برداری بسیار کارآمد هستند

MATLAB توابع بسیاری برای کار با آرایه ها دارد. برای بردار v

- mean(v) – میانگین درایه ها
- max(v) – مقدار بیشینه
- min(v) – مقدار کمینه
- sum(v) – جمع درایه ها
- sort(v) – مرتب کردن درایه ها به ترتیب صعودی

- median(v) – میانه درایه ها
- std(v) – انحراف معیار درایه ها
- det(A) – دترمینان ماتریسهای مربعی
- dot(v,w) – ضرب داخلی دو بردار •
- cross(v,w) – ضرب خارجی دو بردار؛ هر دو باید سه درایه داشته باشند •
- inv(A) – معکوس ماتریس مربعی

جدول ۳-۱ کتاب را برای جزئیات توابع بیان شده ببینید

دقت کنید که در تمام توابع جدول ۳-۱، به جز `det(A)` و `inv(A)`، A بردار است، نه ماتریس. به جز این دو تابع، جدول برای ماتریس ها صدق نمی کند

اعداد تصادفی معمولاً در کاربردهای مهندسی MATLAB استفاده می شوند

- شبیه سازی نویز
- کاربرد در محاسبات ریاضی خاص، مثل شبیه سازی مونت کارلو

MATLAB سه دستور برای ایجاد توابع تصادفی دارد: `rand`, `randn`, `randi`

- هر سه می توانند اسکالر، بردار یا ماتریسی از اعداد تصادفی بسازند

rand اعداد تصادفی با توزیع یکنواخت بین صفر و یک تولید میکند

- برای به دست آوردن اعداد بین a و b، خروجی rand را در b-a ضرب و با a جمع کنید، یعنی:
  (b-a)*rand + a

For example, a vector of 10 elements with random values between –5 and 10 can be created by ($a = -5$, $b = 10$):

```
>> v=15*rand(1,10)-5
v =
   -1.8640    0.6973    6.7499    5.2127    1.9164    3.5174
6.9132   -4.1123    4.0430   -4.2460
```

جدول ۳-۲ کتاب را برای حالتهای مختلف rand ببینید

randi اعداد صحیح تصادفی با توزیع یکنواخت در یک بازه مشخص تولید میکند. مثلاً برای ساختن یک ماتریس ۴×۳ از اعداد تصادفی بین ۵۰ و ۹۰

```
>> d=randi( [50 90],3,4)
d =
     57   82   71   75
     66   52   67   61
     84   66   76   67
```

جدول ۳-۳ کتاب را برای حالتهای مختلف randi ببینید

randn اعداد تصادفی با توزیع نرمال و میانگین صفر و انحراف معیار یک تولید میکند

```
>> d=randn(3,4)

d =
   -0.4326    0.2877    1.1892    0.1746
   -1.6656   -1.1465   -0.0376   -0.1867
    0.1253    1.1909    0.3273    0.7258
```

از randn مشابه rand، همان طور که در جدول ۳-۲ کتاب آمده است استفاده می شود

برای به دست آوردن اعداد با توزیع نرمال با میانگین μ و انحراف معیار σ، خروجی `randn` را در μ ~~ضرب~~ جمع و با σ ~~جمع~~ ضرب کنید، مثلاً

```
>> A = randn( 100, 100 ); % mean=0, std dev = 1
>> mu = 20;
>> sigma = 3;
>> B = sigma * A + mu; % mean = 20, std dev = 3
```

برای به دست آوردن اعداد صحیح با توزیع نرمال، تابع round را روی رابطه قبل اعمال کنید، یعنی

```
round( sigma * rand + mu )
```

مثال

```
>> w = round(4*randn(1,6)+50)
w =
51  49  46  49  50  44
```

شماره تمرین های منتخب

- ۱۳
- ۱۵
- ۱۷
- ۲۰
- ۲۱
- ۲۳
- ۲۴
- ۲۶
- ۲۷
- ۲۹
- ۳۰
- ۳۱
- ۳۲
- ۳۴
- ۳۵