```
ans =
     7
     8
     8
     9
```

ملاحظة:

يوفر برنامج MATLAB الدالتين max ، min الذين يوجدان اكبر واصغر عنصر في المصفوفة ومواقعهما.

في حالة المصفوفة الأحادية:

```
>> v = rand (1, 6)
v =
   0.3046   0.1897   0.1934   0.6822   0.3028   0.5417
>> max (v)
ans =
      0.6822
>> [mx, i] = max (v)
mx =
     0.6822
i =
   4
>> min (v)
ans =
      0.1897
>> [mn, j] = min (v)
mn =
     0.1897
j =
```

2

في حالة كون المصفوفة ثنائية البعد:

```
>> A = rand (4, 6)
A =
    0.1509   0.8537   0.8216   0.3420   0.7271   0.3704
    0.6979   0.5936   0.6449   0.2897   0.3093   0.7027
    0.3784   0.4966   0.8180   0.3412   0.8385   0.5466
    0.8600   0.8998   0.6602   0.5341   0.5681   0.4449
>> [mx, r] = max (A)
mx =
     0.8600   0.8998   0.8216   0.5341   0.8385   0.7027
r =
    4   4   1   4   3   2
```

ملاحظة:

```
>> max (A');          (اكبر عنصر لكل سطر)
>> [mn, r] = min (A)
mn =
     0.1509   0.4966   0.6449   0.2897   0.3093   0.3704
r =
    1   3   2   2   2   1
```

ملاحظة:

```
>> min (A');          (اصغر عنصر لكل سطر)
```

<<ملاحظة: اكبر عنصر في مصفوفة ثنائية البعد.

```
mmx = max (mx)
mmx =
        0.8998
>> [mmx, i] = max (A (:))
```

```
mmx =
        0.8998
i =
    8
```

ملاحظة: توجد طريقة أخرى:

```
>> z = max (max (A));
>> z = min (min (A));
```

ملاحظة: نفس الشيء لحساب المجموع sum.

```
>> z = sum (sum (A));
```

توابع التعامل مع المصفوفة

يزودك برنامج MATLAB، بالإضافة إلى عنونة المصفوفات والمقدرة على التعامل مع المصفوفات التي شرحناها سابقاً، بعمليات التعامل مع المصفوفات، وهي سهلة التطبيق مثل:

```
>> A = [1    2    3; 4    5    6; 7    8    9]
A =
    1    2    3
    4    5    6
    7    8    9
>> flipud (A)                قلب المصفوفة باتجاه up-down
ans =
      7    8    9
      4    5    6
      1    2    3
>> fliplr (A)                قلب المصفوفة باتجاه left-right
ans =
      3    2    1
      6    5    4
      9    8    7
```

```
>> triu (A)          (upper) استخلاص الجزء المثلية العليا
ans =
     1   2   3
     0   5   6
     0   0   9
>> tril (A)          (lower) استخلاص الجزء المثلية السفلى
ans =
     1   0   0
     4   5   0
     7   8   9
>> g = det (A);      حساب محدد المصفوفة (قيمة)
>> h = inv (A);      حساب معكوس المصفوفة (مصفوفة)
>> i = eig (A);      حساب القيم الذاتية للمصفوفة
>> j = eye (3)       حساب مصفوفة الوحدة
j =
   1   0   0
   0   1   0
   0   0   1
>> trace (A);        حساب مجموع عناصر القطر الرئيسي
```

حجم المصفوفة

إذا أردت أن تعرف حجم أو بعد مصفوفة أحادية أو ثنائية أو ثلاثية البعد غير معروفين وكنت بحاجة لحجمها لإجراء بعض العمليات الرياضية، فان برنامج MATLAB يمكنك من خلال الايعاز length و size و numel واليك الأمثلة التالية:

```
>> A = [1   2   3   4; 5   6   7   8]
A =
   1   2   3   4
   5   6   7   8
```

```
>> S = size (A)
S =
    2    4
```

يعبر العنصر الأول عن عدد الأسطر (2) بينما يعطي العنصر الثاني عدد الأعمدة (4).

```
>> [r, c] = size (A)
r =
     2
c =
     4
>> r = size (A, 1)
r =
     2
>> c = size (A, 2)
c =
     4
```

يعيد الايعاز numel العدد الكلي لعناصر مصفوفة فمثلاً:

```
>> numel (A)
ans =
        8
```

بينما يعيد الايعاز length عدد العناصر الموجودة ضمن البعد الأطول للمصفوفة، كما يلي:

```
>> length (A)
ans =
        4
>> B = -3: 3
B =
    -3    -2    -1    0    1    2    3
>> length (B)
```

```
ans =
     7
>> min (size (A))          A مصفوفة ثنائية
ans =
     2
```

ملاحظة: طريقة توليد مصفوفة بالدمج.

```
>> x = [1    2; 3    4];
>> y = [x    x .^ 2; x .^ 3    x .^ 4];
```

المصفوفات متعددة الإبعاد

لقد شرحنا في الفصل السابق المصفوفات أحادية وثنائية الأبعاد والعمليات التي تجري عليها. يدعم برنامج MATLAB المصفوفات متعددة الأبعاد (أي n-D arrays) وذلك نفس الايعازات وتقنيات العنونة المطبقة على المصفوفات أحادية وثنائية البعد. وبشكل عام، يرقم البعد الثالث عبر صفحات (pages)، ولذلك تمتلك المصفوفات ثلاثية البعد اسطرا وأعمدة وصفحات، حيث تتألف كل صفحة من مصفوفة ثنائية البعد ذات اسطر وأعمدة، ويجب أن تمتلك كل صفحة عددا متساويا من الأسطر والأعمدة والعكس بالعكس في كل صفحة.

ليس هناك حد لعدد الأبعاد في المصفوفات، ولكننا سنستخدم مصفوفات ثلاثية الأبعاد في هذا الفصل بسبب سهولة تخيلها وإظهارها.

تركيب المصفوفة

يمكن إنشاء المصفوفة المتعددة الأبعاد بطرق مختلفة، واليك بعضها:

```
>> A = zeros (4, 3, 2)
A (:, :, 1) =
     0    0    0
     0    0    0
     0    0    0
     0    0    0
A (:, :, 2) =
```

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

تتألف هذه المصفوفة الصفرية من أربعة اسطر وثلاثة أعمدة وصفحتين، ولقد ظهرت الصفحة الأولى ثم الصفحة الثانية.

مثال:

```
>> B (:, :, 1) = zeros (2, 3);
>> B (:, :, 2) = ones (2, 3);
>> B (:, :, 3) = 4;
>> B
B (:, :, 1) =
     0     0     0
     0     0     0
B (:, :, 2) =
     1     1     1
     1     1     1
B (:, :, 3) =
     4     4     4
     4     4     4
```

يمكن استخدام الايعاز reshape لتحويل المصفوفة من مصفوفة ثنائية الأبعاد إلى مصفوفة ثلاثية الأبعاد وكالاتي:

```
>> C = [B (:, :, 1), B (:, :, 2), B (:, :, 3)]
C =
    0    0    0    1    1    1    4    4    4
    0    0    0    1    1    1    4    4    4
>> reshape (C, 2, 3, 3)
```

```
ans (:, :, 1) =
     0   0   0
     0   0   0
ans (:, :, 2) =
     1   1   1
     1   1   1
ans (:, :, 3) =
     4   4   4
     4   4   4
```

حجم المصفوفة

الايعاز size يعيد بعد المصفوفة وفق كل أبعادها كما شرحنا سابقاً.

```
>> [r, c, p] = size (C)
r =
    2
c =
    3
p =
    3
```

وإذا لم نعرف عدد إبعاد المصفوفة أو كانت ابعادها متغيرة، عندما نستطيع استخدام الايعاز ndims كما يلي:

```
>> ndims (C)
ans =
      3
>> numel (C)              إيجاد عدد عناصر المصفوفة
ans =
      18
>> length (size (C))      إيجاد طول اكبر بعد بالمصفوفة
```

```
ans =
     3
```

مصفوفة الخلايا Cell Arrays

تعتبر مصفوفات الخلايا مصفوفات في لغة MATLAB تكون عناصرها عبارة عن خلايا، وتتضمن كل خلية نوعاً من البيانات قد تكون مصفوفات عددية أو رمزية أو كائنات بسيطة أو مصفوفات خلايا أخرى. فمثلاً قد تحوي خلية من مصفوفة الخلية مصفوفة عددية وتحوي الخلية الأخرى مصفوفة رمزية، بينما تحوي الثالثة على أعداد عقدية (يسمح باستخدام مصفوفات بأنواع بيانية مختلفة (غير متجانسة)) كما ويمكن إنشاء مصفوفات الخلايا بأي بعد كان كما هي الحال مع المصفوفات العددية، ولكن معظم مصفوفات الخلايا تكون عبارة عن مصفوفات أحادية البعد.

تنشأ مصفوفات الخلايا عبر استخدام تعابير الإسناد أو عبر إعادة تقسيم المصفوفة بالإيعاز cell، ثم نقوم بإسناد البيانات إلى الخلايا.

هناك طريقتان مختلفتان للوصول إلى الخلايا. وإذا أردت استخدام رموز مصفوفة قياسية للدلالة على المصفوفة، يجب عليك أن تحيط الخلية بأقواس مجموعة { }، إذ إن برنامج MATLAB يستخدم هذه الأقواس لتعريف مصفوفات الخلايا، واليك الأمثلة التالية:

```
>> A (1, 1) = {[1   2   3; 4   5   6; 7   8   9]};
>> A (1, 2) = {2 + 3i};
>> A (2, 1) = {'Ali Ahmed'};
>> A (2, 2) = {12: -2: 0};
>> A
A =
    [3×3 double]       [2.0000 + 3.0000i]
```

'Ali Ahmed' [1×7 double]

لاحظ إن برنامج MATLAB يظهر المصفوفة كمصفوفة خلية بعداها 2×2 ولكن ذلك لا يظهر محتويات الخلية، وإنما يظهر البرنامج محتويات الخلية بشكل أساسي إذا لم تأخذ هذه المحتويات حجماً كبيراً، كما ويوصف محتويات الخلية إذا أخذت حجماً معقولاً. إن وجود أقواس مجموعات على الجانب الأيمن من المساواة يدل على إن المشار إليه هو خلية وليس قيما عددية وهذا ما يسمى *بفهرسة الخلية (cell indexing)*، وسينشئ التعابير التالية مصفوفة الخلية نفسها.

ملاحظة:

يخبر كلا التعبيرين A (i, j) = {x} و A {i, j} = x برنامج MATLAB بأن يضع المتغير x في العنصر (i, j) من مصفوفة الخلية A.

يدعى التعبير A (i, j) *بفهرسة الخلية (cell indexing)*، بينما يدعى التعبير A {i, j} *بعنونة المحتوى (content addressing)* أي تدل أقواس المجموعة { } على محتوى الخلية، بينما تعرف الأقواس العادية ( ) الخلايا دون النظر إلى محتواها.

مثال:

```
>> y = {1, 'hello', 1 > 5}
y =
    [1]    'hello'    [0]
>> y {1}
ans =
      1
>> y {2}
ans =
      hello
>> y {3}
ans =
      0
```

مثال: