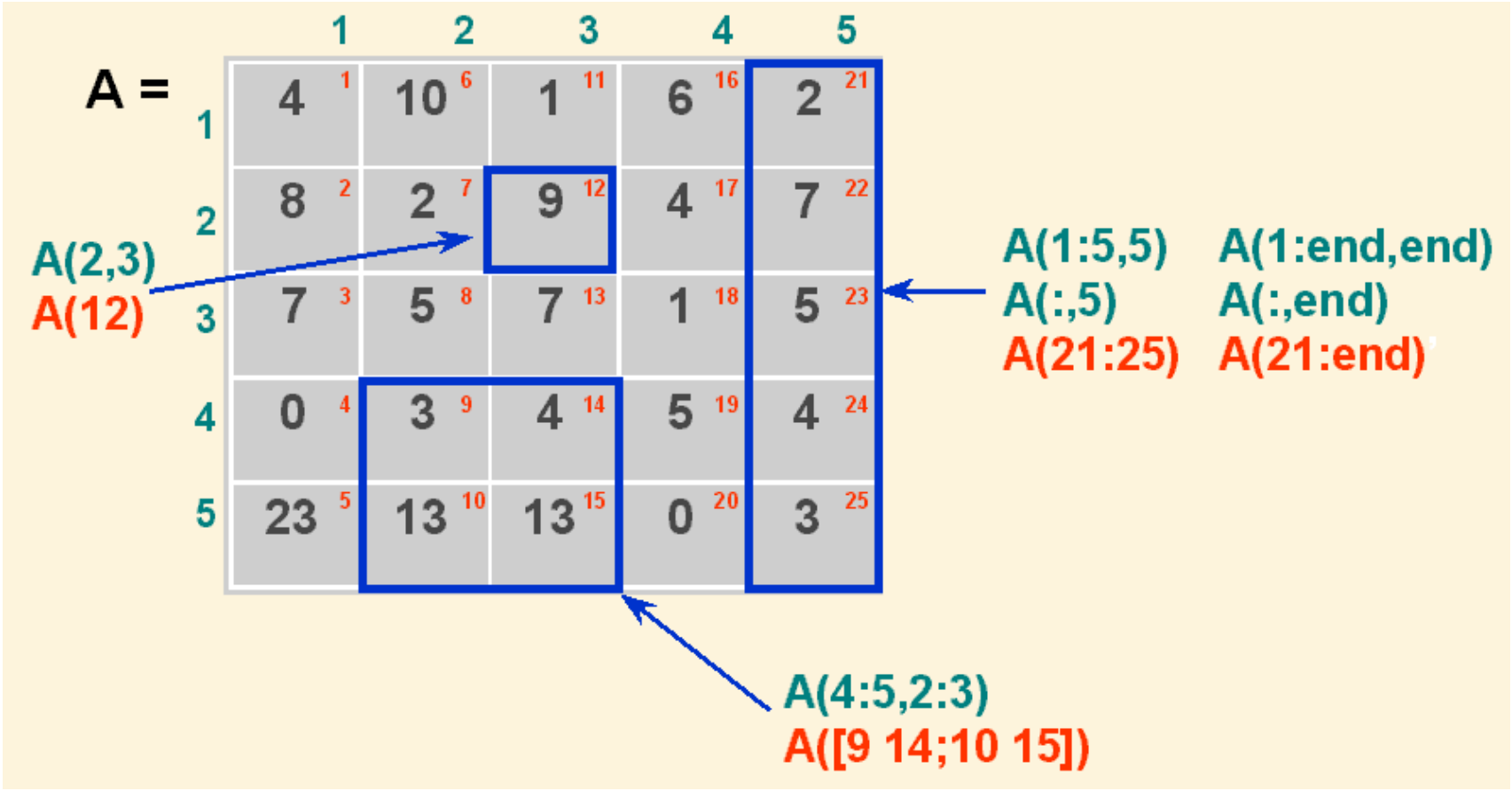
Chapter 9
矩陣的處理與運算

9-1 矩陣的索引或下標

- 矩陣 A 中，位於第 i 橫列、第 j 直行的元素可表示為 A(i, j) 。
  - i 與 j 即是此元素的下標（Subscript）或索引（Index）
- MATLAB 中，所有矩陣的內部表示法都是以直行為主的一維向量。
  - A(i, j) 和 A(i+(j-1)*m) 是完全一樣的，其中m為矩陣A的列數
- 我們可以使用一維或二維下標來存取矩陣。

矩陣的索引或下標


A =
1 2 3 4 5
1 4 1 10 6 1 11 6 16 2 21
2 8 2 2 7 9 12 4 17 7 22
3 7 3 5 8 7 13 1 18 5 23
4 0 4 3 9 4 14 5 19 4 24
5 23 5 13 10 13 15 0 20 3 25
A(2,3)
A(12)
A(1:5,5)
A(:,5)
A(21:25)
A(1:end,end)
A(:,end)
A(21:end)
A(4:5,2:3)
A([9 14;10 15])

矩陣的索引或下標

- 可以使用矩陣下標來進行矩陣的索引(Indexing)
  - A(4:5,2:3) -取出矩陣 A 的 第四、五 橫列與 二、三 直行所形成的部份矩陣。
  - A([9 14; 10 15]) - 用一維下標的方式來達到同樣目的。
- 用冒號（:）, 取出一整列或一整行
  - A(:, 5) -取出矩陣 A 的第五個直行
- 用 end 這個保留字來代表某一維度的最大值
  - A(:, end) - 矩陣 A 的最後一個直行
- 可以直接刪除矩陣的某一整個橫列或直行
  - A(2, :) = [] – 刪除A矩陣的第二列
  - A(:, [2 4 5]) = [] - 刪除 A 矩陣的第二、四、五直行

矩陣的索引或下標

- 可把矩陣 A 和其倒數「並排」起來，得到新矩陣 B
  - B = [A 1./A]　　% 1./A 是矩陣 A 每個元素的倒數(和反矩陣有何不同?)

- 用 diag 指令取出矩陣的對角線各元素
  - D = diag(B)　　% 取出矩陣 B 的對角線元素
  - D = diag(v)　　% 產生以向量 v 為主對角線的方陣
  - E = A*diag(v)　　% 將矩陣A的每個行向量乘上向量v的元素
  - E = diag(v)*A　　% 將矩陣A的每個列向量乘上向量v的元素

- 用 reshape 指令來改變一個矩陣的維度
  - B = B(1:4, 1:4);
  - C = reshape(B, 2, 8)　　% 將矩陣 B 排成 2x8 的新矩陣 C
  - MATLAB 會先將矩陣 B 排成一個行向量（即 MATLAB 內部的矩陣表示法），再將此行向量塞成 2x8 的新矩陣。
  - 如果是reshape(B,2,3)呢?

練習9-1

- 給定矩陣A=[1 2 3; 4 5 6; 7 8 9; 10 11 12]及向量v=[13, 14, 15]
  - 計算A的倒數矩陣
  - 取出A的第二行(column)存入x向量
  - 除了A(4,:)外，可用哪個指令取出A的最後一列(row)
  - 形成對角線矩陣D，其中D的對角線元素為v的元素
  - 將A的維度改成3x4
  - 可不可能將A改成5個列(row)?
  - 請將v放入A形成A的第五列(row)

9-2 特殊用途矩陣

- 產生各種特殊用途矩陣的好用指令：

指令	說明
zeros(m, n)	產生維度為 m×n ，構成元素全為 0 的矩陣
ones(m, n)	產生維度為 m×n ，構成元素全為 1 的矩陣
eye(n)	產生維度為 n×n ，對角線的各元素全為 1 ，其他各元素全為 0 的單位矩陣
pascal(m, n)	產生維度為 m×n 的 Pascal 矩陣
vander(v)	產生Vandermonde矩陣，其中每一個行向量都是向量 v 的冪次
hilb(n)	產生維度為 n×n 的 Hilbert 矩陣
rand(m, n)	產生均勻分佈於 [0, 1] 的亂數矩陣，其維度為 m×n
randn(m, n)	產生 μ = 0, σ= 1 的正規分佈亂數矩陣，其維度為 m×n
magic(n)	產生維度為 n×n 的魔方陣，其各個直行、橫列及兩對角線的元素和都相等

Hilbert矩陣 & 魔方陣

- hilb(n) 指令可以產生 n×n 的 Hilbert 矩陣

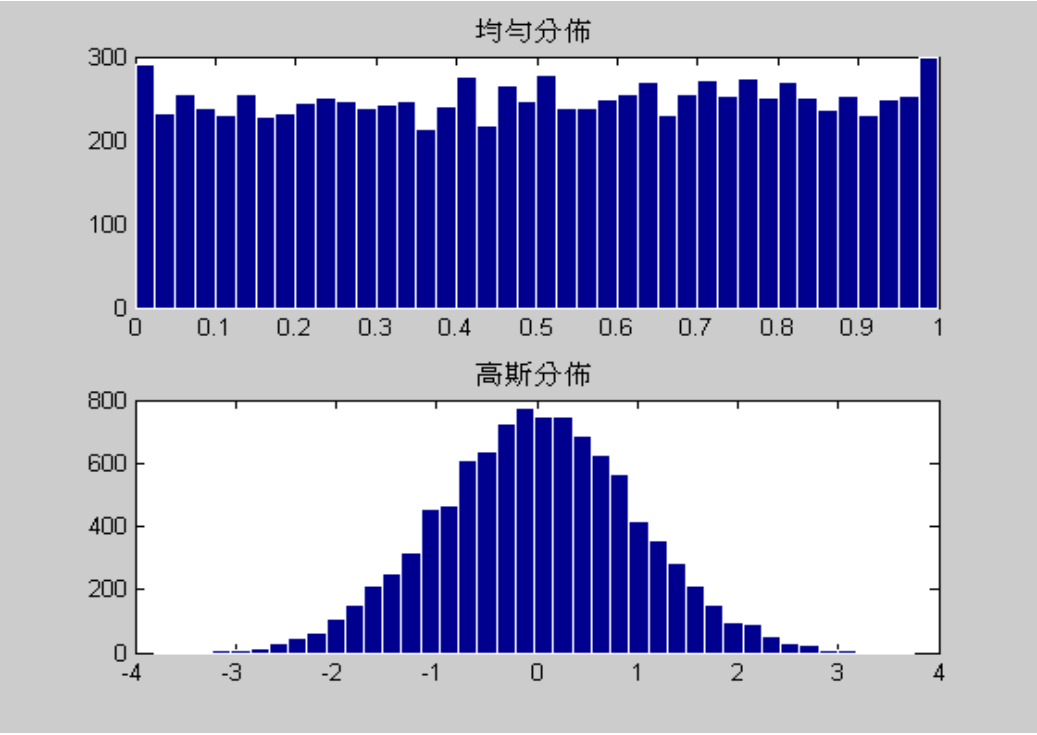
$$[\mathbf{H}]_{i,j} = \frac{1}{i+j-1}$$

  - Hilbert 矩陣的特性: 當矩陣變大時，其反矩陣會接近 Singular（即矩陣的行列式會接近於 0）
  - Hilbert 矩陣常被用來評估各種反矩陣計算方法的穩定性
- magic(n) 可以產生一個 n×n 的魔方陣（Magic Matrix），
  - 其各個直行、橫列及兩對角線的元素值總和都相等
  - n應該大於2

均勻和高斯分布

- rand 指令及 randn 指令則常用於產生亂數矩陣
  - 範例9-11: matrix11.m

```
x1 = rand(10000, 1);
x2 = randn(10000, 1);
subplot(2,1,1); hist(x1, 40); title('均勻分佈'); %分成40堆
subplot(2,1,2); hist(x2, 40); title('高斯分佈');
```



9-3 矩陣的數學運算

- 矩陣的加減與一般純量（Scalar）的加減類似
- 相加或相減的矩陣必需具有相同的維度
  - 範例9-12: matrix12.m

```
A = [12 34 56 20];
B = [1 3 2 4];
C = A + B
```

  - C =
    13 37 58 24

- 矩陣與純量可以直接進行加減，MATLAB 會直接將加減應用到每一個元素

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

矩陣的乘法與除法

- 純量對矩陣的乘或除，可比照一般寫法

```
>> A = [123 , 442];
>> B = 2*A
B =
   246   884
```

```
>> C = A/3
C =
   41.0000  147.3333
```

- 欲進行矩陣相乘，必需確認第一個矩陣的直行數目（Column Dimension）必需等於第二個矩陣的橫列數目（Row Dimension）
  - 範例9-13: matrix12.m

```
A = [1; 2];
B = [3, 4, 5];
C = A*B
```

```
C =
   3    4    5
   6    8   10
```

- 矩陣的除法，常藉由反矩陣或解線性方程式來達成

矩陣的次方運算

- 矩陣的次方運算，可由「^」來達成，但矩陣必需是方陣，其次方運算才有意義。
  - 範例9-14: matrix14.m

```
A = magic(3);
B = A^2
```

```
B =
   91   67   67
   67   91   67
   67   67   91
```

- 在「*」，「/」及「^」之前加上一個句點，MATLAB 將會執行矩陣內「元素對元素」(Element-by-element)的運算。

```
A = [12; 45];
B = [2; 3];
C = A.*B                    % 注意「*」前面的句點
D = A./B                    % 注意「/」前面的句點
E = A.^2                    % 注意「^」前面的句點
```

轉置和「共軛轉置」矩陣

- 複數矩陣 z，其「共軛轉置」矩陣（Conjugate Transpose）可表示成矩陣 z'。
  - 範例9-16: conjTranspose01.m

```
i = sqrt(-1);                    % 單位虛數
z = [1+i, 2; 3, 1+2i];
w = z'                           % 共軛轉置（注意 z 後面的單引號）
```

```
w =
   1.0000-1.0000i   3.0000
   2.0000           1.0000-2.0000i
```

- 想得到任何矩陣 z 的轉置(Transpose)，則可表示成矩陣 z.'。
  - 範例9-17: transpose01.m

```
i = sqrt(-1);                    % 單位虛數
z = [1+i, 2; 3, 1+2i];
w = z.'                          % 單純轉置（注意 z 後面的句點及單引號）
```

```
w =
   1.0000+1.0000i   3.0000
   2.0000           1.0000+2.0000i
```

- 若 z 為實數，則 z' 和 z.' 的結果是一樣的

練習9-2

- 給定矩陣A=[1+i 2 3; 2 3 4-i; 3 4+i 5]，B=[1 2 3; 2 3 4; 3 4 5]
  - 計算兩個矩陣相加
  - 計算兩個矩陣相乘
  - 計算A矩陣的平方
  - 將A與B矩陣同位置的元素相除
  - 將B矩陣的每個元素開庚號
  - 計算A的轉置
  - 計算A的共軛轉置

Sort指令

- sort 指令可對向量元素進行排序（Sorting）
  - 範例9-20: sort01.m

```
x = [3 5 8 1 4];
[sorted, index] = sort(x)                    % 對矩陣 x 的元素進行排序
```

```
sorted =
   1   3   4   5   8
index =
   4   1   5   2   3
```

- sorted 是排序後的向量，index 則是每個排序後的元素在原向量 x 的位置，因此 x(index) 即等於 sorted 向量。
- 如果向量元素有相同的值呢?
- 如何使用 sort 指令加上前例中的 sorted 及 index 來求得原先的向量 x？

矩陣的最大元素

- 找出一矩陣最大元素的位置
  - 範例9-21: max01.m

```
x = magic(5);
[colMax, colMaxIndex] = max(x)
```

```
colMax =
    23    24    25    21    22
colMaxIndex =
     2     1     5     4     3
```

- colMax 代表每一直行的最大值，colMaxIndex 則是每一直行出現最大值的位置

矩陣的最大元素

- 求得 x 的最大元素的位置
  - 範例9-22: max02.m

```
x = magic(5);
[colMax, colMaxIndex] = max(x);
[maxValue, maxIndex] = max(colMax);
fprintf('Max value = x(%d, %d) = %d\n', colMaxIndex(maxIndex), maxIndex,
maxValue);
```

Max value = x(5, 3) = 25

- x 的最大元素即是 maxValue，發生位置為 [colMaxIndex(maxIndex), maxIndex] = [5 , 3]

- 若只要找出一矩陣 x 的最大值，可輸入 max(max(x))或是 max(x(：))

練習9-3

- 若A是一個矩陣(非一維向量)，則sort(A)會產生什麼結果?
- 給定向量[5 3 7 4 2 8 1]，請將其排序並顯示元素原先的位置 。
- 給定向量[5 3 7 4 2 8 1]，請求出最大元素之值及其所在位置 。
- 給定矩陣1./magic(5)，請求出該矩陣最大元素的值及其所在的位置 。
- 已知sorted = [1 2 3 4 5 7 8]及index = [1 3 6 4 7 5 2]，請求出原來的向量。

9-4 矩陣的內部資料型態

- 一般矩陣的內部資料型態都是 double（雙精準浮點數），但在 MATLAB 5.3 版之後，也支援不同長度的整數與浮點數資料態

指令	說明
uint8	轉換成不帶正負號、 8 位元的整數，其值域為 [0,255]
uint16	轉換成不帶正負號 、16 位元的整數，其值域為 [0,65535]
uint32	轉換成不帶正負號 、32 位元的整數，其值域為 [0,2^32-1]
int8	轉換成帶正負號 、 8 位元的整數，其值域為 [-128,127]
int16	轉換成帶正負號、16 位元的整數，其值域為 [-32768,32767]
int32	轉換成帶正負號、32 位元的整數，其值域為 [-2^31,2^31-1]
single	轉換成 single（單精準浮點數），佔用 32 位元（4 bytes）
double	轉換成double（雙精準浮點數），佔用 64 位元（8 bytes）
char	轉換成字元或字串，每個字元佔用（16 位元）（2 bytes）

不同資料的儲存

- 我們要節省記憶體空間，可以依矩陣元素值的範圍，選用不同的資料來儲存
  - 範例9-23: datatype01.m

```
clear all                                   % 清除所有工作空間的變數
x_double = magic(10);
x_single = single(x_double);
x32 = uint32(x_double);
x16 = uint16(x_double);
x8 = uint8(x_double);
whos
```

```
Name         Size              Bytes  Class
  x16        10x10               200  uint16 array
  x32        10x10               400  uint32 array
  x8         10x10               100  uint8 array
  x_double   10x10               800  double array
  x_single   10x10               400  single array
Grand total is 500 elements using 1900 bytes
```

- uint8 來儲存變數所佔的空間只有 double 的八分之一！

資料儲存的注意事項

- 整數資料型態的範圍有限，若超過此範圍，則超出部分將會被「裁掉」。

```
>> uint8(300)              % uint8 的最大值為 255
ans =
   255
>> int8(-500)              % int8 的最小值為 -128
ans =
  -128
```

- 整數資料型態可以比較大小，亦可直接進行數學運算，但必須注意其資料型態的自動轉換：

```
>> uint8(20)== 20          % 可比較大小
ans =
    1
>> z=uint8(magic(3))
>> z*2.5
```

(Please try it by yourself to get the conversion rule!) 下一頁作業

- 若要進行精準的數學運算，需先用 double 指令將整數型態之變數轉成雙倍精準浮點數。

練習9-4

- 於matlab鍵入

```
>> z=uint8(magic(3))
>> z*2.5
```

比較2.5*magic(3)

請觀察結果並得出結論。

- 給定矩陣magic(20)，
  - 下表中那些資料型態可以正確表示出此矩陣
  - 這些資料型態各別花費多少bytes
  - 哪種型態最省記憶體

指令	說明
uint8	轉換成不帶正負號、 8 位元的整數，其值域為 [0,255]
uint16	轉換成不帶正負號 、16 位元的整數，其值域為 [0,65535]
uint32	轉換成不帶正負號 、32 位元的整數，其值域為 [0,2^32-1]
int8	轉換成帶正負號 、 8 位元的整數，其值域為 [-128,127]
int16	轉換成帶正負號、16 位元的整數，其值域為 [-32768,32767]
int32	轉換成帶正負號、32 位元的整數，其值域為 [-2^31,2^31-1]
single	轉換成 single（單精準浮點數），佔用 32 位元（4 bytes）
double	轉換成double（雙精準浮點數），佔用 64 位元（8 bytes）
char	轉換成字元或字串，每個字元佔用（16 位元）（2 bytes）