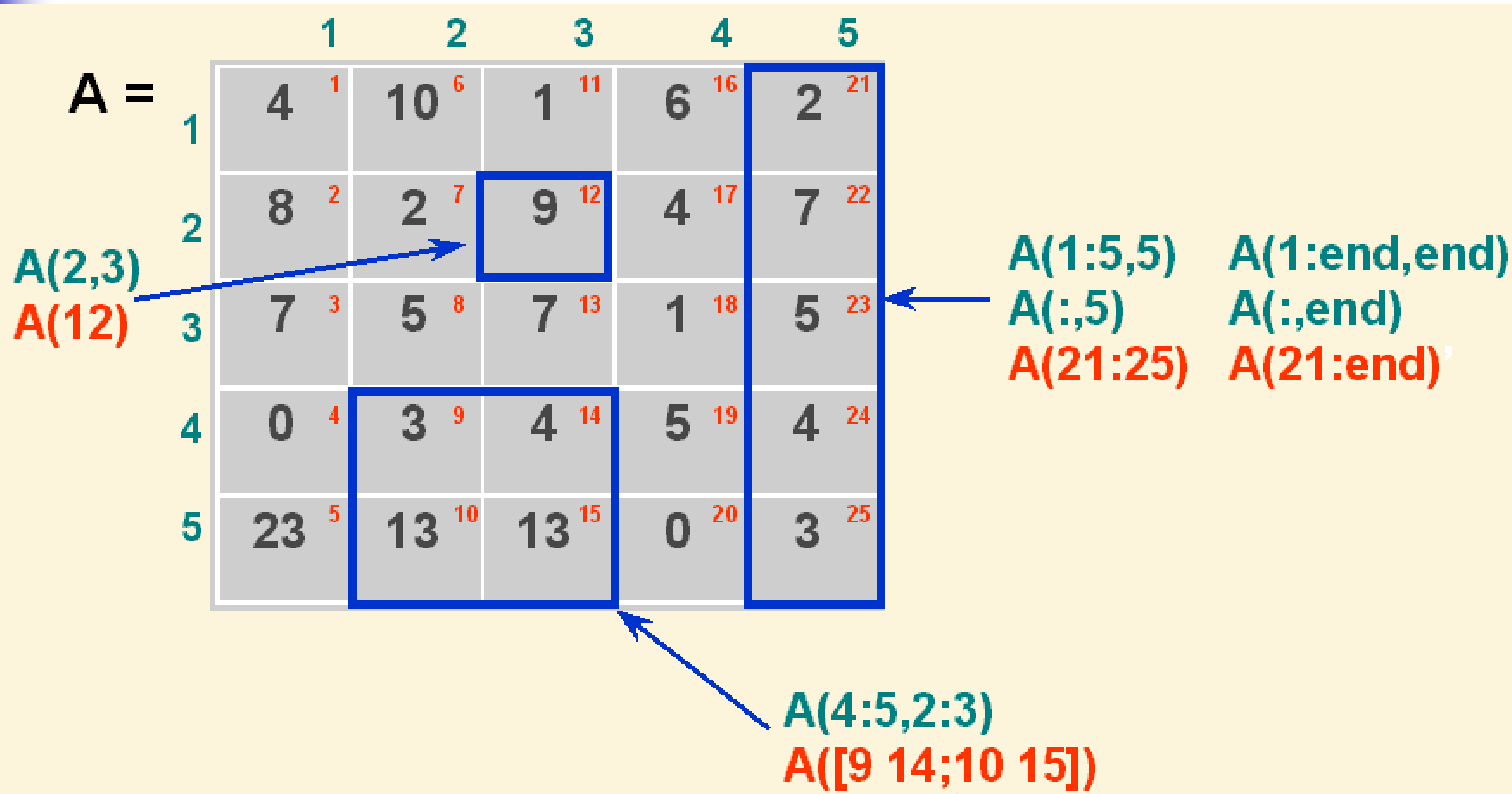
第九章:
矩陣的處理與運算

張智星 清大資工系
補充內容：方煒 台大生機系
小幅修改：吳俊仲 長庚機械系

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~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 (matrix01.m)



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） (matrix02.m)
  - 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 (matrix04.m)
  - 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 (matrix05.m)
  - 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 (matrix06.m)
  - A(:, [2 4 5]) = [] - 刪除 A 矩陣的第二、四、五直行(matrix07.m)

矩陣的索引或下標

- 可依次把矩陣 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(matrix08.m)
- 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(matrix09.m)
- 用 reshape 指令來改變一個矩陣的維度 (matrix10.m)
  - C = reshape(B, 2, 8) - 將矩陣 B 排成 2x8 的新矩陣 C
- 注意!! 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

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(m, n)	產生維度為 m×n 的 Vandermonde 矩陣
hilb(n)	產生維度為 n×n 的 Hilbert 矩陣
rand(m, n)	產生 [0, 1] 均勻分佈的亂數矩陣，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 and 魔方陣

- hilb(n) 指令可以產生 n×n 的 Hilbert 矩陣

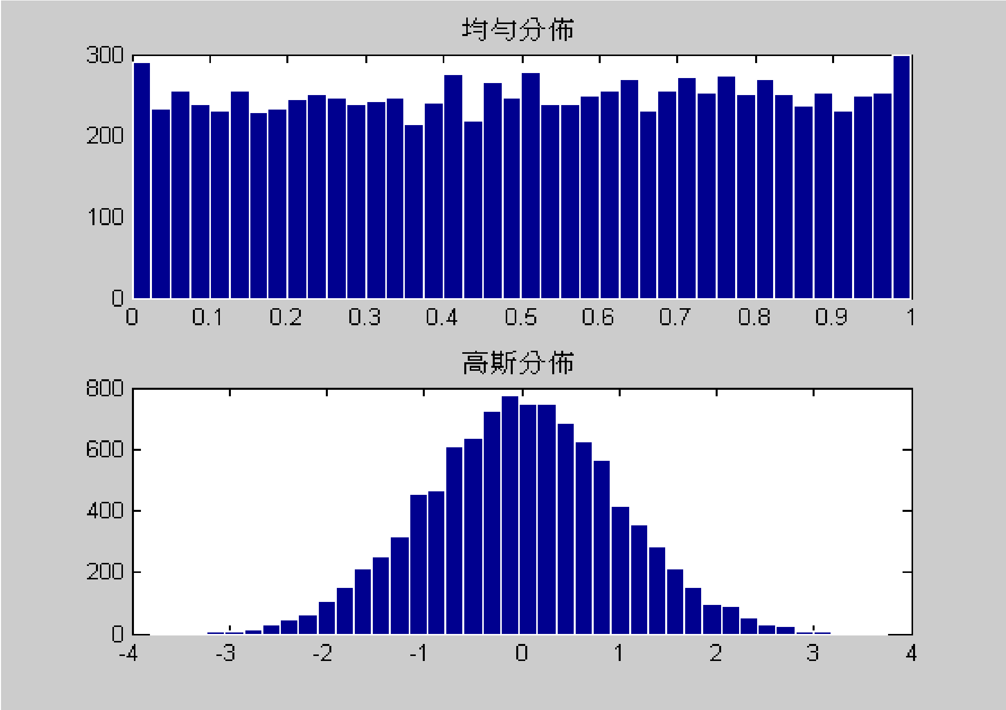
$$[\mathbf{H}]_{\mathrm{i,j}} = \frac{1}{i+j-1}$$

  - 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

均匀和高斯分布

- 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('均匀分佈');
subplot(2,1,2); hist(x2, 40); title('高斯分佈');
set(findobj(gcf, 'type', 'patch'), … 'EdgeColor',
'w');           % 改邊線為白色
```



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: matrix13.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

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
- x(index) 即等於 sorted 向量
- 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
- 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(：))

Learning Linear Algebra

補充內容

Vector Products, Dot and Cross

- a=[1,2,3];b=[3,2,1];
- C=a*b
- D=a.*b
- E=dot(a,b)
- F=cross(a,b)

- (ex7_0.m)

Ex7_1 Solve a Linear System (ex7_1.m)

- A=[1, 0,1;-1, 1,1;1,-1,1];
- b=[4;4;2];
- % now solve for x
- x=A\b
- %we obtain [1;2;3]

$$
\begin{aligned}
x + z &= 4 \\
-x + y + z &= 4 \\
x - y + z &= 2\,,
\end{aligned}
$$

Ex7_2 Max and Min (ex7_2.m)

- x=0:.01:5;
- y=x.*exp(-x.^2);
- % take a look at the function so we know what it looks like
- plot(x,y)
- % find the max and min
- ymin=min(y)
- ymax=max(y)
- % find the max and min along with the array indices imax and imin
- % where they occur
- [ymin,imin]=min(y)
- [ymax,imax]=max(y)

Ex7_3 Matrix Inverse (ex7_3.m)

- A=[1,0,-1;-1,1,1;1,-1,1]
- % load C with the inverse of A
- C=inv(A)
- % verify by matrix multiplication
- % that A*C is the identity matrix
- A*C

Ex7_5a Special Matrices

- % eye:
- % load I with the 4x4 identity matrix (the programmer who invented this
- % syntax must have been drunk)
- I=eye(4,4)
- % zeros:
- % load Z with a 5x5 matrix full of zeros
- Z=zeros(5,5)
- % ones:
- % load X with a 3x3 matrix full of ones
- X=ones(3,3)
- % rand:
- % load Y with a 4x6 matrix full of random numbers between 0 and 1
- % The random numbers are uniformly distributed on [0,1]
- Y=rand(4,6)
- % And to load a single random number just use
- r=rand
- % randn:
- % load Y with a 4x6 matrix full of random numbers with a Gaussian
- % distribution with zero mean and a variance of 1
- Y=randn(4,6)

Ex7_6 Determinant

- Ex7_6 Determinant
- %Find the determinant of a square matrix this way
- det(A)

Ex7_8 Sum the Elements

- %For arrays the command sum adds up the elements of the array:
- % calculate the sum of the squares of the reciprocals of the
- % integers from 1 to 10,000
- n=1:10000;
- sum(1./n.^2)
- % compare this answer with the sum to infinity, which is pi^2/6
- ans-pi^2/6
- For matrices the sum command produces a row vector which is made up of the sum of the
- columns of the matrix.
- A=[1,2,3;4,5,6;7,8,9]
- sum(A)