Matrices (2D Arrays)

Lecture 11 (Mar 1)
CS100M – Spring 2007

Topics

- Reading: CFile 9, Section 9.1
- Recall
  - Matlab vectors (1D arrays)
  - Characters & Strings
- Plans for today
  - Matrices (2D arrays)

2D Arrays (Matrices)

- Recall: An *array* is a named collection of data values organized into rows and/or columns
- A 2D array is a table, called a matrix
- Two indices are used to identify the position of a item in a matrix
  - M(r, c) refers to the item in row r, column c
  - Just like vectors, indices for matrices start at 1
  - Example: M(2, 3) refers to 6

7	0	5
2	4	6
3	8	1

Creating a Matrix

- Matlab makes it easy to create a matrix
  - Use brackets
  - Comma or space separates items in *same* row
  - Semicolon ";" indicates a new row
  - Example: M = [7 0 5; 2 4 6; 3 8 1] creates

7	0	5
2	4	6
3	8	1

- The vector-creating functions can also create matrices
  - zeros(2, 3) % 2-by-3 matrix of zeros
  - ones(3, 2) % 3-by-2 matrix of ones
  - rand(3, 4) % 3-by-4 matrix of random numbers

Creating a Matrix, Continued

- You can build a new matrix out of smaller matrices (or vectors) — as long as all the dimensions match up
  - [ones(1,4); 1:4] works
  - [ones(1,3); 1:4] doesn't
  - [ones(2,4); 1:4] works

1	1	1	1
1	2	3	4

- If you start filling a matrix, Matlab will create it for you (unspecified values are set to 0)
  - Example: B(2, 3) = 77

0	0	0
0	0	77

Transpose of a Matrix

- If A is a matrix then A' is the transpose of A
  - The transpose of a matrix just swaps the rows and the columns
    - An item at position (r, c) becomes an item at position (c, r)
  - Example: The transpose of [1:3; 4:6] is

1	2	3
4	5	6

transpose

1	4
2	5
3	6

Finding the Dimensions of a Matrix

- Matlab provides a function for this: size(M)
- Examples

```
[nr, nc] = size(M)   % Both # of rows and # of columns
nr = size(M, 1)      % # of rows
nc = size(M, 2)      % # of columns
```

Example: Finding Min Value in a Matrix

- Function header

```
function val = minInMatrix(M)
% Return min value in matrix M
```

- Pseudocode:

```
Initialize val
Loop through all items in M
    Update val at each item
```

- Resulting Code

```
function val = minInMatrix(M)
% Return min value in matrix M
val = M(1,1);
[nr, nc] = size(M);
for r = 1:nr
   for c = 1:nc
      val = min(val, M(r,c));
   end
end
```

Pattern for Traversing a Matrix M

```
[nr, nc] = size(M);
for r = 1:nr
   for c = 1:nc
      % Do something with M(r, c)
   end
end
```

Submatrices

- Matlab colon notation can be used to easily create a *submatrix* of a matrix
- Example: Let M = [7 0 5; 2 4 6; 3 8 1]

7	0	5
2	4	6
3	8	1

  - M(1:2, 1:3) is

7	0	5
2	4	6

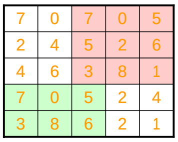
  - M(2:3, 1:2) is

2	4
3	8

- A single colon ":" can be used to represent "all indices"
  - Thus M(1:2, :) is the same as M(1:2, 1:3)

Neighborhood of a Cell

- We define the *neighborhood of a cell* to be the cell itself and all adjacent cells (including diagonally adjacent)



7	0	7	0	5
2	4	5	2	6
4	6	3	8	1
7	0	5	2	4
3	8	6	2	1

The neighborhood of cell(2,4)

The neighborhood of cell(5,2)

Min of a Neighborhood

- Goal:
  Write a function minInNeighborhood(M, row, col) that reports the minimum value in neighborhood of cell(row, col) in matrix M
- Function header

```
Function val = minInNeighborhood(M, row, col)
% Return min in neighborhood of (row, col) in M
```