Previous Lecture:
- Discrete vs. continuous; finite vs. infinite
- Linear interpolation
- Vectorized operations

Today’s Lecture:
- 2-d array—matrix

Announcements:
- Discussion this week in the classrooms as listed in the roster
- Prelim 1 tonight at 7:30pm
  - Last names A-O: Uris Auditorium (room G01)
  - Last names P-Z: Upson Auditorium (room B17)
Storing and using data in *tables*

A company has 3 factories that make 5 products with these costs:

<table>
<thead>
<tr>
<th></th>
<th>C</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>36</td>
<td>22</td>
<td>15</td>
<td>62</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>35</td>
<td>20</td>
<td>12</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>37</td>
<td>21</td>
<td>16</td>
<td>59</td>
<td></td>
</tr>
</tbody>
</table>

What is the best way to fill a given purchase order?
2-d array: matrix

- An array is a named collection of like data organized into rows and columns.
- A 2-d array is a table, called a matrix.
- Two indices identify the position of a value in a matrix, e.g.,

  \[ \text{mat}(r,c) \]

  refers to component in row \( r \), column \( c \) of matrix \( \text{mat} \).
- Array index starts at 1.
- Rectangular: all rows have the same #of columns.
2-d array: **matrix**

- An array is a **named** collection of **like** data organized into rows and columns.
- A 2-d array is a table, called a **matrix**.
- Two *indices* identify the position of a value in a matrix, e.g.,

  \[ \text{mat}(r,c) \]

  refers to component in row \( r \), column \( c \) of matrix \( \text{mat} \).
- Array index starts at 1.
- **Rectangular**: all rows have the same # of columns.
Creating a matrix

- **Built-in functions:** `ones`, `zeros`, `rand`
  - E.g., `zeros(2,3)` gives a 2-by-3 matrix of 0s
- “Build” a matrix using square brackets, `[]`, but the dimension must match up:
  - `[x  y]` puts `y` to the right of `x`
  - `[x; y]` puts `y` below `x`
  - `[4 0 3; 5 1 9]` creates the matrix
  - `[4 0 3; ones(1,3)]` gives
  - `[4 0 3; ones(3,1)]` doesn’t work
Working with a matrix: 

**size** and individual components

**Given a matrix M**

\[
\begin{array}{cccc}
2 & -1 & .5 & 0 & -3 \\
3 & 8 & 6 & 7 & 7 \\
5 & -3 & 8.5 & 9 & 10 \\
52 & 81 & .5 & 7 & 2 \\
\end{array}
\]

\[
[nr, nc]= \text{size}(M) \quad \% \text{ nr is # of rows,} \\
\quad \% \text{ nc is # of columns}
\]

nr = size(M, 1) \quad \% \text{ # of rows}

nc = size(M, 2) \quad \% \text{ # of columns}

\[
M(2,4)= 1; \\
disp(M(3,1)) \\
M(1,nc)= 4;
\]
Example: minimum value in a matrix

function val = minInMatrix(M)

% val is the smallest value in matrix M
Example: minimum value in a matrix

function val = minInMatrix(M)

% val is the smallest value in matrix M
[nr, nc] = size(M);
val = M(1, 1);
for r = 1:nr
    % At row r
    for c = 1:nc
        % At col c (at row r)
        if M(r, c) < val
            val = M(r, c);
        end
    end
end
Pattern for traversing a matrix $M$

$$
\begin{array}{l}
[nr, nc] = \text{size}(M) \\
\text{for } r = 1:nr \\
\quad \% \text{At row } r \\
\quad \text{for } c = 1:nc \\
\quad \quad \% \text{At column } c \text{ (in row } r) \\
\quad \quad \% \\
\quad \quad \% \text{Do something with } M(r,c) \ldots \\
\quad \text{end} \\
\text{end}
\end{array}
$$
% Given an nr-by-nc matrix M.
% What is A?
for r = 1: nr
    for c = 1: nc
        A(c,r) = M(r,c);
    end
end

A is M with the columns in reverse order
B is M with the rows in reverse order
C is the transpose of M
D A and M are the same
% Given an nr-by-nc matrix M.
% What is A?
for r = 1: nr
    for c = 1: nc
        A(c,r) = M(r,c);
    end
end

M
\[
\begin{array}{cccc}
0 & 3 & 2 & 5 \\
4 & 13 & 20 & 6 \\
11 & 26 & 9 & 1 \\
\end{array}
\]

A
\[
\begin{array}{ccc}
0 & 4 & 11 \\
3 & 13 & 26 \\
2 & 20 & 9 \\
5 & 6 & 1 \\
\end{array}
\]
Matrix example: Random Web

- N web pages can be represented by an N-by-N Link Array $A$.
- $A(i,j)$ is 1 if there is a link on webpage $j$ to webpage $i$.
- Generate a random link array and display the connectivity:
  - There is no link from a page to itself.
  - If $i \neq j$ then $A(i,j) = 1$ with probability $\frac{1}{1+|i-j|}$.
  - There is more likely to be a link if $i$ is close to $j$. 


function A = RandomLinks(n)
% A is n-by-n matrix of 1s and 0s
% representing n webpages

A = zeros(n,n);
for i=1:n
    for j=1:n
        r = rand(1);
        if i~=j && r<= 1/(1 + abs(i-j))
            A(i,j) = 1;
        end
    end
end
Random web
N = 20
Represent the web pages graphically…

100 Web pages arranged in a circle. Next display the links….
Bidirectional links are blue. Unidirectional link is black as it leaves page j, red when it arrives at page i.
for i = 1:n
    for j = 1:n
        end
    end

Is there another way? See ShowRandomLinks.m
for i = 1:n
    for j = 1:n
        if A(i,j) == 1 && A(j,i) == 1
            % Blue
        elseif A(i,j) == 1
            % Black-Red
            j → mid, mid → i
    end
end

Somewhat inefficient: each blue line gets drawn twice.
See ShowRandomLinks.m
Given an n-by-m matrix A.

What is this operation?

```matlab
for g = 1: n
    for h = 1: floor(m/2)
        A(g,h) = A(g, m-h+1);
    end
end
```

- Reflect the right half of A onto the left half
- Reflect the bottom half of A onto the top half
% Given an nr-by-nc matrix A.
% What is this operation?
for  r= 1: nr
    for  c= 1: floor(nc/2)
        A(r,c)= A(r, nc-c+1);
    end
end

a. Reflect the right half of A onto the left half
b. Reflect the bottom half of A onto the top half