■ Previous Lecture:
  ■ 2-d array—matrix

■ Today’s Lecture:
  ■ More examples on matrices
  ■ Optional reading: contour plot (7.2, 7.3 in Insight)

■ Announcement:
  ■ Prelim 1 tonight at 7:30pm
    ■ Last names A-L in Uris Hall Room G01
    ■ Last names M-Z in Baker Lab Room 200
Storing and using data in *tables*

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

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

What is the best way to fill a given purchase order?
Pattern for traversing a matrix $M$

\[
[nr, nc] = \text{size}(M)
\]

\[
\text{for } r = 1:nr
\]
\[
\text{\quad \% At row } r
\]
\[
\text{\quad for } c = 1:nc
\]
\[
\text{\quad \quad \% At column } c \text{ (in row } r)\n\]
\[
\text{\quad \quad \%}
\]
\[
\text{\quad \quad \% Do something with } M(r,c) \text{ \ldots}
\]
\[
\text{\quad end}
\]
\[
\text{end}
\]
% 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
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
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.
Somewhat inefficient: each blue line gets drawn twice.

See `ShowRandomLinks.m`
A Cost/Inventory Problem

- A company has 3 factories that make 5 different products
- The cost of making a product varies from factory to factory
- The inventory/capacity varies from factory to factory
Problems

A customer submits a purchase order that is to be filled by a single factory.

1. How much would it cost a factory to fill the order?
2. Does a factory have enough inventory/capacity to fill the order?
3. Among the factories that can fill the order, who can do it most cheaply?
Cost Array

<table>
<thead>
<tr>
<th></th>
<th>10</th>
<th>36</th>
<th>22</th>
<th>15</th>
<th>62</th>
</tr>
</thead>
<tbody>
<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>

The value of $C(i, j)$ is what it costs factory $i$ to make product $j$. 
Inventory (or Capacity) Array

The value of $\text{Inv}(i,j)$ is the inventory in factory $i$ of product $j$. 

<table>
<thead>
<tr>
<th>Inv</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>99</td>
</tr>
<tr>
<td>34</td>
</tr>
<tr>
<td>42</td>
</tr>
<tr>
<td>82</td>
</tr>
<tr>
<td>19</td>
</tr>
<tr>
<td>83</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>42</td>
</tr>
<tr>
<td>51</td>
</tr>
<tr>
<td>29</td>
</tr>
<tr>
<td>21</td>
</tr>
<tr>
<td>56</td>
</tr>
<tr>
<td>87</td>
</tr>
</tbody>
</table>
Purchase Order

The value of $PO(j)$ is the number of product $j$'s that the customer wants.
**Cost for factory 1:**

\[1 \times 10 + 0 \times 36 + 12 \times 22 + 29 \times 15 + 5 \times 62\]
Cost for factory 1:

\begin{align*}
\text{s} &= 0; \quad \% \text{Sum of cost} \\
\text{for } j=1:5 \\
\text{s} &= \text{s} + \text{C}(1,j) \times \text{PO}(j) \\
\text{end}
\end{align*}
Cost for factory 2:

\[
s = 0; \quad \text{%Sum of cost}\n\]
\[
\text{for } j=1:5
\]
\[
\quad s = s + C(2,j) \times PO(j)
\]
\[
\text{end}
\]
**Cost for factory i:**

\[ s = 0; \quad \text{%Sum of cost} \]

\[ \text{for } j=1:5 \]
\[ s = s + C(i,j) \times \text{PO}(j) \]
\[ \text{end} \]
function  TheBill = iCost(i,C,PO)
% The cost when factory i fills the
% purchase order

nProd = length(PO);
TheBill = 0;
for j=1:nProd
    TheBill = TheBill + C(i,j)*PO(j);
end
\textbf{Finding the Cheapest}

\begin{verbatim}
iBest = 0; minBill = inf;
for i=1:nFact
    iBill = iCost(i,C,PO);
    if iBill < minBill
        % Found an Improvement
        iBest = i; minBill = iBill;
    end
end
\end{verbatim}
inf – a special value that can be regarded as positive infinity

\[ x = \frac{10}{0} \quad \text{assigns inf to } x \]
\[ y = 1 + x \quad \text{assigns inf to } y \]
\[ z = \frac{1}{x} \quad \text{assigns 0 to } z \]
\[ w < \text{inf} \quad \text{is always true if } w \text{ is numeric} \]
Inventory/Capacity Considerations

What if a factory lacks the inventory/capacity to fill the purchase order?

Such a factory should be excluded from the find-the-cheapest computation.
### Who Can Fill the Order?

<table>
<thead>
<tr>
<th></th>
<th>38</th>
<th>5</th>
<th>99</th>
<th>34</th>
<th>42</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Inv</strong></td>
<td>82</td>
<td>19</td>
<td>83</td>
<td><strong>12</strong></td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>51</td>
<td>29</td>
<td>21</td>
<td>56</td>
<td>87</td>
</tr>
</tbody>
</table>

| **PO** | 1  | 0  | 12 | **29** | 5  |

- Cell with red background is **Yes**.
- Cell with green background is **Yes**.
- Other cells are **No**.
Wanted: A True/False Function

DO is “true” if \textit{factory} i can fill the order.
DO is “false” if \textit{factory} i cannot fill the order.
Example: Check inventory of factory 2

<table>
<thead>
<tr>
<th>Inv</th>
<th>38</th>
<th>5</th>
<th>99</th>
<th>34</th>
<th>42</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>82</td>
<td>19</td>
<td>83</td>
<td>12</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>51</td>
<td>29</td>
<td>21</td>
<td>56</td>
<td>87</td>
</tr>
<tr>
<td>PO</td>
<td>1</td>
<td>0</td>
<td>12</td>
<td>29</td>
<td>5</td>
</tr>
</tbody>
</table>

Method 1: check the inventory for every product
## Initialization

<table>
<thead>
<tr>
<th>Inv</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>38</td>
<td>5</td>
<td>99</td>
<td>34</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>82</td>
<td>19</td>
<td>83</td>
<td>12</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>51</td>
<td>29</td>
<td>21</td>
<td>56</td>
<td>87</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PO</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>12</td>
<td>29</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>
Still True...

\[
\begin{array}{ccccc}
  & \text{Inv} & & & \\
 38 & 5 & 99 & 34 & 42 \\
\text{82} & 19 & 83 & 12 & 42 \\
51 & 29 & 21 & 56 & 87 \\
\end{array}
\]

\[
\begin{array}{ccccc}
  & & & & \text{DO} \\
 1 & 0 & 12 & 29 & 5 \\
\end{array}
\]

\[
\text{DO} = \text{DO} && ( \text{Inv}(2,1) \geq \text{PO}(1) )
\]
Still True…

<table>
<thead>
<tr>
<th>Inv</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>5</td>
<td>99</td>
<td>34</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>19</td>
<td>83</td>
<td>12</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>29</td>
<td>21</td>
<td>56</td>
<td>87</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PO</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>12</td>
<td>29</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

\[ \text{DO} = \text{DO} \land (\text{Inv}(2,2) \geq \text{PO}(2)) \]
Still True…

\[
\begin{array}{cccccc}
38 & 5 & 99 & 34 & 42 \\
82 & 19 & 83 & 12 & 42 \\
51 & 29 & 21 & 56 & 87 \\
\end{array}
\]

**DO**

\[
\begin{array}{cccccc}
1 & 0 & 12 & 29 & 5 \\
\end{array}
\]

\[
DO = DO \land (\text{Inv}(2,3) \geq \text{PO}(3))
\]
No Longer True...

\[
\begin{array}{cccccc}
\text{Inv} & 38 & 5 & 99 & 34 & 42 \\
82 & 19 & 83 & 12 & 42 \\
51 & 29 & 21 & 56 & 87 \\
\end{array}
\]

\[
\begin{array}{cccccc}
\text{PO} & 1 & 0 & 12 & 29 & 5 \\
\end{array}
\]

\[
\text{DO} = \text{DO} \land (\text{Inv}(2,4) \geq \text{PO}(4))
\]
Stay False…

\[
\text{DO} = \text{DO} \land \left( \text{Inv}(2, 5) \geq \text{PO}(5) \right)
\]
Encapsulate…

function \ DO = iCanDo(i,Inv,PO) \\
% DO is true if factory \textit{i} can fill \n% the purchase order. Otherwise, false
\n\texttt{nProd = length(PO);} \\
DO = 1; \\
\texttt{for \ j = 1:nProd} \\
\quad \texttt{DO = DO && ( Inv(i,j) \geq \ PO(j) );} \\
\texttt{end}