# Announcements

- Assignment 1 due tonight @ 11:59 pm
  - Submit what you have by the deadline; do not resubmit until resubmissions are open in CMS

| Su | M | Tu | W | Th | F | Sa |
|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 12 | **13** | 14 | 15 | 16 | 17 | 18 |
| 19 | 20 | 21 | 22 | 23 | 24 | 25 |
| 26 | 27 | 28 | 29 | 30 | | |

## Agenda

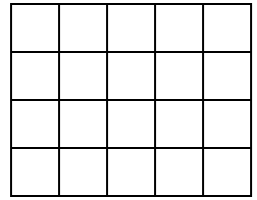
- How to search for "the best"
- How to model probabilities
- How to iterate over triangular matrices
- How to write subfunctions

# Pattern for traversing a matrix ("row-major")

```
[nr, nc] = size(M)
for r= 1:nr
    % Start of row r
    for c= 1:nc
        % At column c (in row r)
        % Do something with M(r,c) ...
    end
    % End of row r
end
```

Where should end-of-column logic go?

# Example:  minimum value in a matrix



```
function val = minInMatrix(M)
% val is the smallest value in matrix M
```

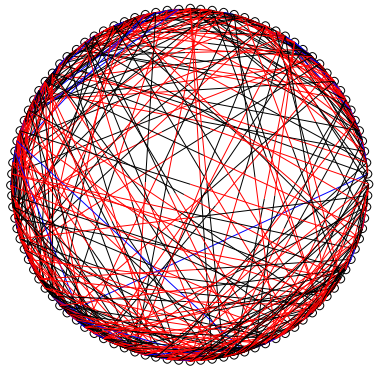
**minInMatrix.m**

```
function val = minInMatrix(M)
% val is the smallest value in matrix M

[nr,nc]= size(M);
% ??? (A)
for r = 1:nr
    for c = 1:nc
        % ??? (B)



    end
end
% ??? (C)
```

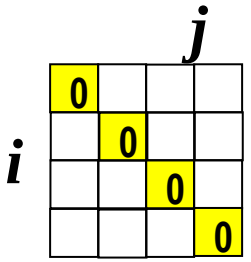
Algorithm: Finding the best in a set

```
Init bestSoFar
Loop over set
   if current is better than bestSoFar
      bestSoFar ← current
   end
end
```

# 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≠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);  % initialize to 0s
for r = 1:n
  for c = 1:n
    % if A(r,c) not on diagonal,
    % assign 1 with some probability



  end
end
```

# An event happens with probability p, $0 \le p \le 1$

```
% Flip a fair coin
x= rand();
if x <  0.5
   disp('heads')
else
   disp('tails')
end
```

p

```
% Unfair coin: shows heads
% twice as often as tails
x= rand();
if x <  2/3
   disp('heads')
else
   disp('tails')
end
```

p

```
% Event Y happens with probability p
x= rand();
if x < p
   % Code for event Y
end
```

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

A= zeros(n,n);
for r = 1:n
   for c = 1:n
      x= rand();
      if r ~= c && x < 1/(1 + abs(r-c))
          A(r,c)= 1;
      end
   end
end
```

# Random web: N=20


2

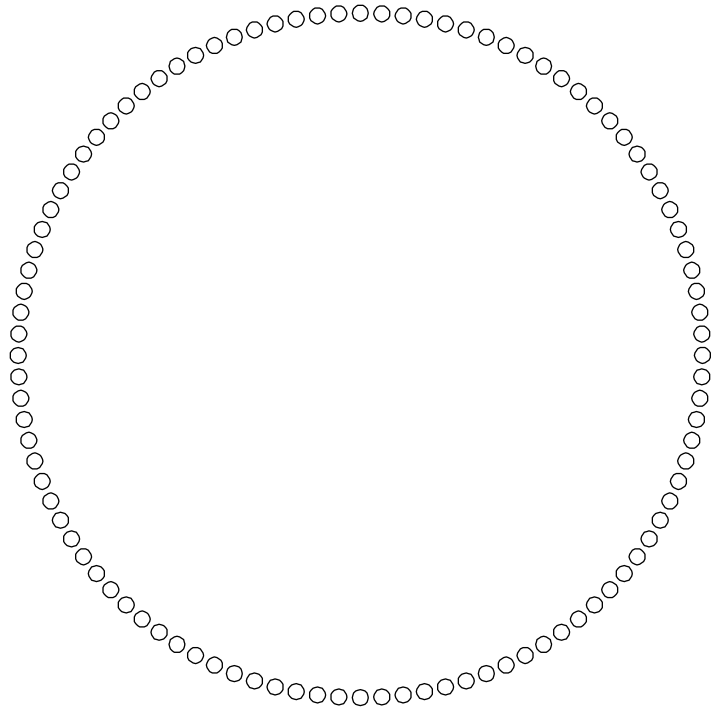
M(3,2)


3

M(2,3)

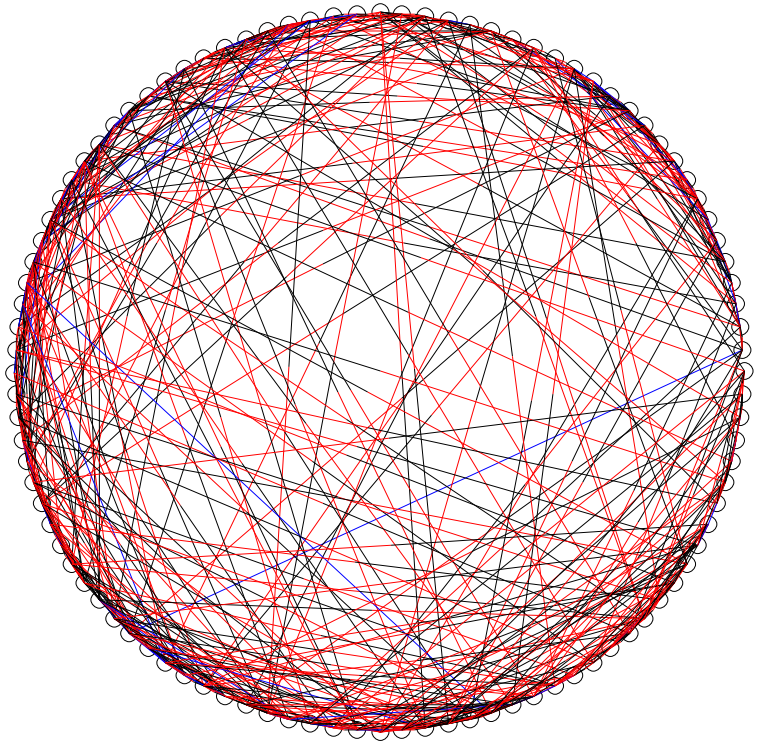
```
0 1 1 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 1 1 1 0 0 0 0 0 0 1 0 0
0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0
0 1 1 1 1 1 0 0 0 1 0 1 1 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 1 1
0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0
0 0 0 0 0 0 0 1 1 0 1 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 1
0 0 0 1 0 0 0 0 1 1 0 1 0 1 1 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0
0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 1
0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 1 0
0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 1 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0
```

Represent the web pages graphically…



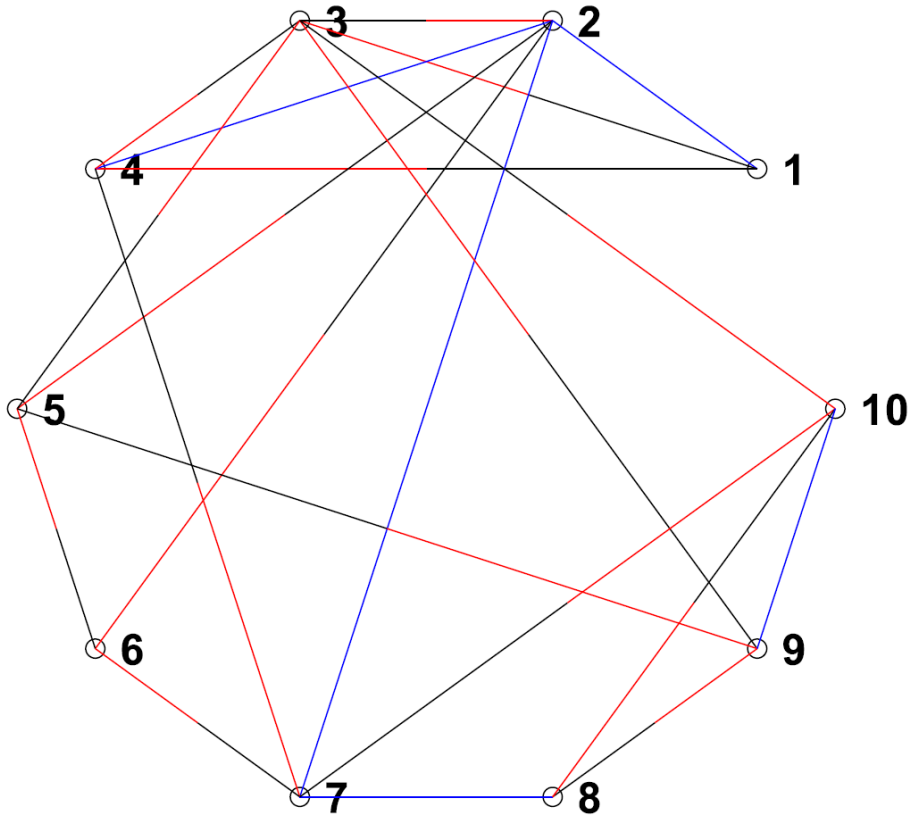
100 Web pages arranged in a circle.
Next display the links….

Represent the web pages graphically…



Bidirectional links are blue. Unidirectional link is black as it leaves page c, red when it arrives at page r.

Represent the web pages graphically…

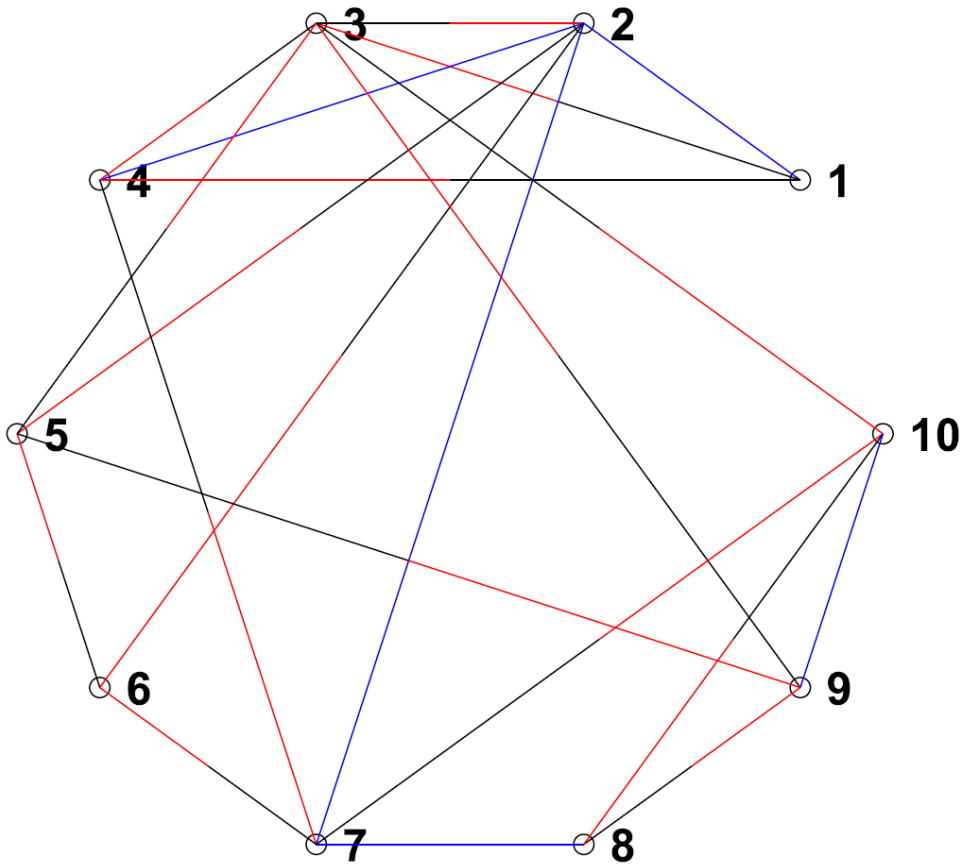


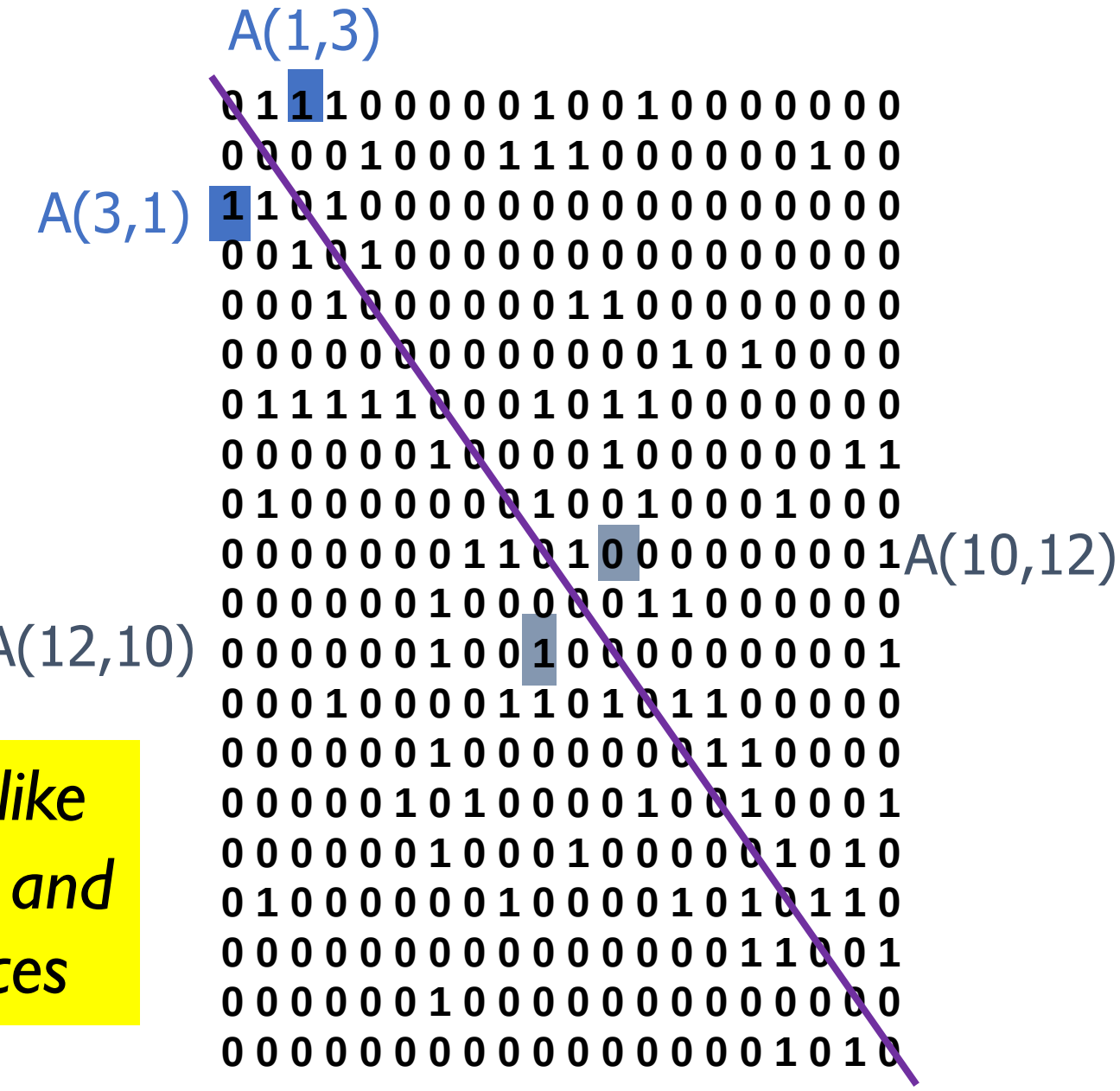

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | **1** | 0 | 1 | **1** | 0 | 0 | **1** | 0 | 0 | 0 |
| 3 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 4 | 1 | **1** | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 6 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 7 | 0 | **1** | 0 | 1 | 0 | 0 | 0 | **1** | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 1 |
| 9 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | **1** |
| 10 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | **1** | 0 |

Bidirectional links are blue. Unidirectional link is black as it leaves page c, red when it arrives at page r.

# Outline

1. Get coordinates of points on circle
2. Iterate over all links
    1. Determine color to draw
    2. Draw line(s) between points

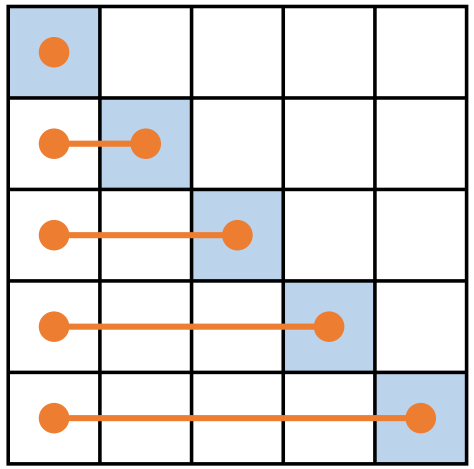




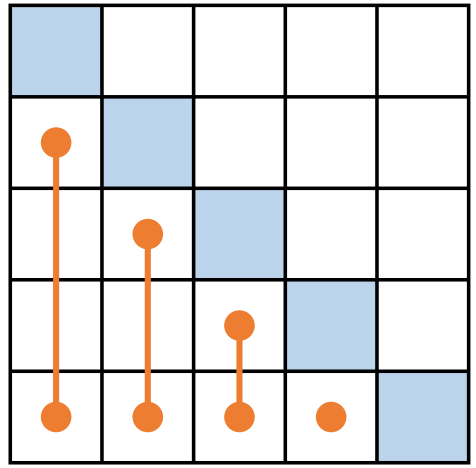

*Transpose—like switching row and column indices*

# Triangular traversal

```
[nr, nc] = size(M);
for A = B:C
    for D = E:F
        disp(M(r,c))
    end
end
```



Case 1



Case 2