CS319 Week 12 - Graph Applications: PageRank

Table of Contents


```
clear;
```

1. Directed Graphs

Last week we studied (undirected) graphs: the edge (u, v) is the same as (v, u). You can think of this as a two-way street: if a road can take you from point u to point v, then it can take you back again. The adjacency matrix is always symmetric.

Now we'll look at **directed** graphs, also called **digraphs**. This is made up of one-way streets. Every edge needs a direction.

In MATLAB, the class representing a directed graph is called `digraph()`. The syntax is essentially the same as for a graph. For example, suppose we want to build a graph with edges $1 \to 2$, $2 \to 3$ and $3 \to 1$, we could it like this:

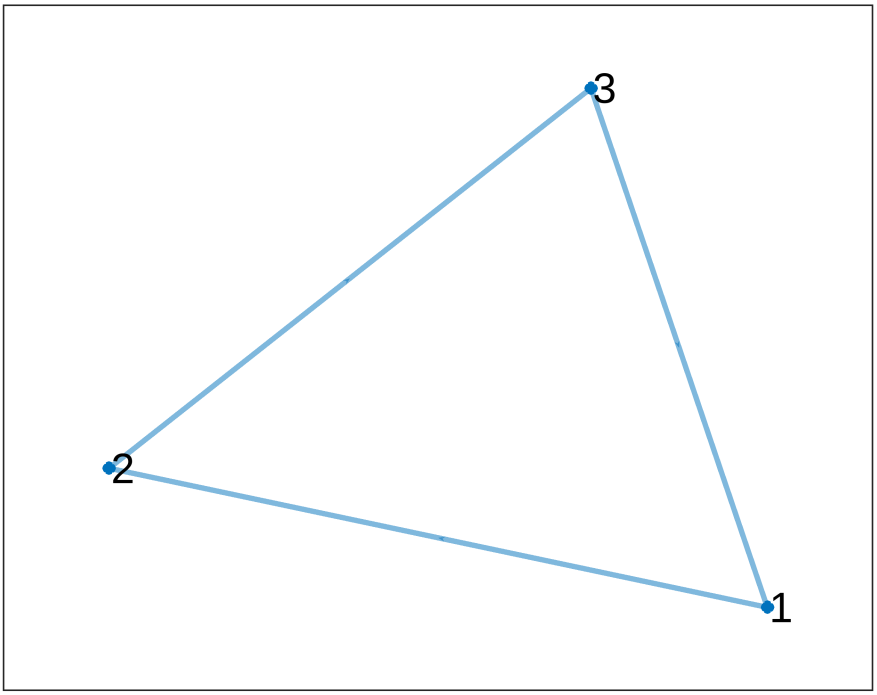
```
a = [1 2 3]
```

```
a = 1×3
     1     2     3
```

```
b = [2 3 1]
```

```
b = 1×3
     2     3     1
```

```
D1 = digraph(a,b);
plot(D1, 'LineWidth', 2, 'ArrowSize',16, 'NodeFontSize', 16);
```



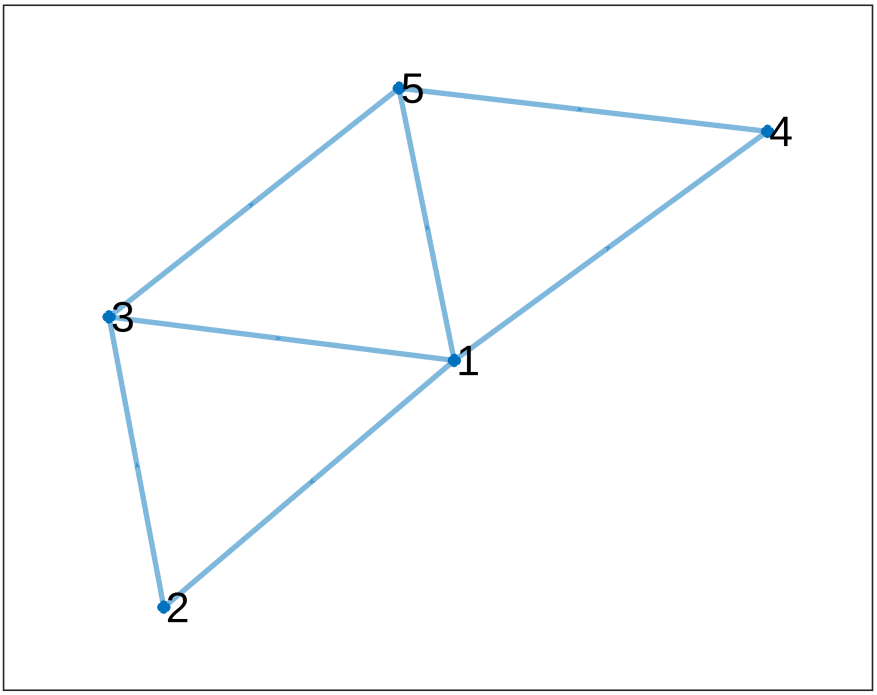
The adjacency matrix of a directed graph is no necessarily directed.

```
full( D1.adjacency() )
```

```
ans = 3×3
     0     1     0
     0     0     1
     1     0     0
```

All the `graph` methods that we use in Week 11 also apply to `digraphs, e.g, addedge(), addnode(), plot()`. `addedge()` works slightly differently for digraphs: `D1.addedge(u,v)` adds a directed edge from u to v. Example:

```
D1=D1.addedge([1,1,3,5],[4,5,5,4]);
plot(D1, 'LineWidth', 2, 'ArrowSize',16, 'NodeFontSize', 16);
```



Other methods that are different:

- `d=G.degree()`: applies only to **graphs**. Sets d to be a vector where $d(i)$ is the number of edges incident to node i.
- `d=D.indegree()`: applies only to **digraphs**. Sets d to be a vector where $d(i)$ is the number of edges starting at i.
- `d=D.outdegree()`: applies only to digraphs. Sets to be a vector where $d(i)$ is the number of edges ending at i.

```
D1.indegree()'
```

```
ans = 1×5
     1     1     1     2     2
```

```
D1.outdegree()'
```

```
ans = 1×5
     3     1     2     0     1
```

2. Building and visualising a more complex graph

Let's make the digraph that was shown on the cover page of the Week 11 slides.

```
s = [1 1 2 2 3 3 4 5 6]
```

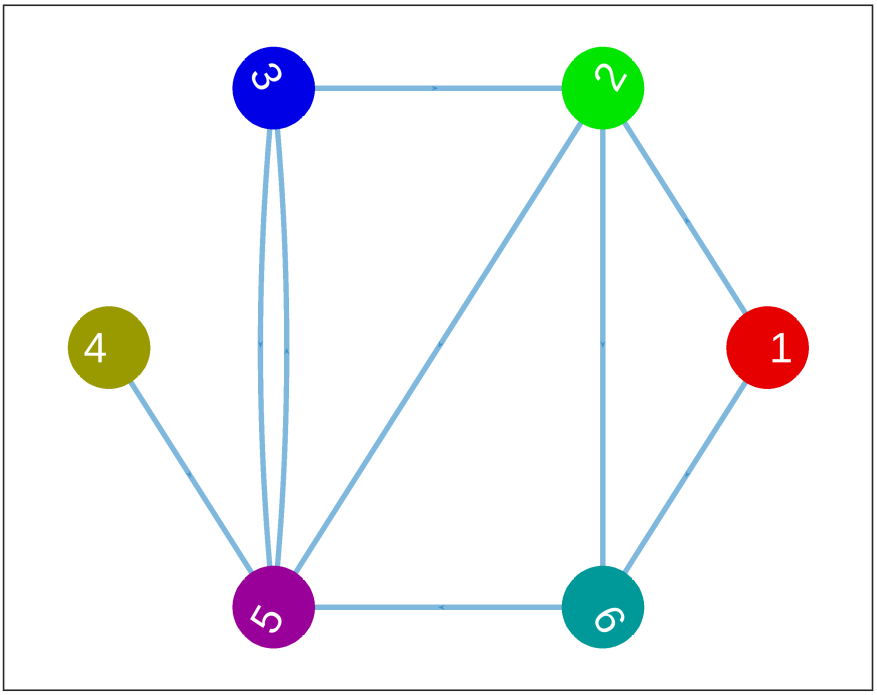
```
s = 1×9
```

```
     1     1     2     2     3     3     4     5     6
```

```
t = [2 6 5 6 2 5 5 3 5]
```

```
t = 1×9
     2     6     5     6     2     5     5     3     5
```

```
D = digraph(s,t);
plot(D, 'Layout','circle', 'MarkerSize', 30, ...
   'NodeColor', [.9 0 0; 0 .9 0; 0 0 0.9; .6 .6 0; .6 0 .6; 0 .6 .6], ...
   'LineWidth', 2, 'ArrowSize',16, 'NodeFontSize',16,...
   'NodeLabelColor', 'white');
```



3. PageRank

See notes from class.

```
A = full(D.adjacency)
```

```
A = 6×6
     0     1     0     0     0     1
     0     0     0     0     1     1
     0     1     0     0     1     0
     0     0     0     0     1     0
     0     0     1     0     0     0
     0     0     0     0     1     0
```

```
N = length(A);
S = A;
```

```
for i=1:N
   S(i,:)  = S(i,:)/sum(S(i,:));
end
disp(S)
```

```
         0    0.5000         0         0         0    0.5000
         0         0         0         0    0.5000    0.5000
         0    0.5000         0         0    0.5000         0
         0         0         0         0    1.0000         0
         0         0    1.0000         0         0         0
         0         0         0         0    1.0000         0
```

```
sigma = 0.7;
G = (sigma*S + (1-sigma)/N)';
u = ones(N,1)/N;
v = zeros(N,1);
d = norm(u-v);
TOL = 1.0e-3;
k=0; % iteration count
while(d > TOL)
   k=k+1;
   v = u;
   u = G*u;
   d = norm(u-v);
end
fprintf("Power method tool %d iterations\n", k);
```

```
Power method tool 10 iterations
```

```
[ranked_score,ranking]=sort(u, 'descend');
fprintf('The ranking is: ');
```

```
The ranking is:
```

```
for i=1:D.numnodes
   fprintf('Rank %d : Vertex %d (Value %5.3f)\n', ...
      i, ranking(i), ranked_score(i));
end
```

```
Rank 1 : Vertex 5 (Value 0.329)
Rank 2 : Vertex 3 (Value 0.280)
Rank 3 : Vertex 2 (Value 0.165)
Rank 4 : Vertex 6 (Value 0.126)
Rank 5 : Vertex 1 (Value 0.050)
Rank 6 : Vertex 4 (Value 0.050)
```

```
node_size= 3*(1+log(u/min(u)))
```

```
node_size = 6×1
    3.0000
    6.5900
    8.1723
    3.0000
    8.6490
    5.7611
```

```
plot(D, 'Layout','circle', 'MarkerSize', node_size, ...
```

```
    'NodeColor', [.9 0 0; 0 .9 0; 0 0 .9; .6 .6 0; .6 0 .6; 0 .6 .6], ...
    'LineWidth', 2, 'ArrowSize',16, 'NodeFontSize',16);
```