Homework 2 Sol. (OCDR)

Universidad Nacional de Colombia

By:

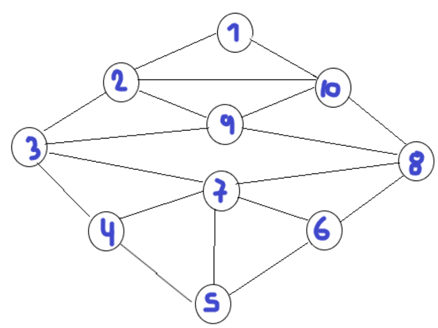
- Santiago Cadena Álvarez (Electrical Engineering student)

To:

Eng. Eduardo A. Mojica - Nava

1) Consensus Protocol: Continuous-time

Construct a connected graph of 10 nodes with ramdon structure, i.e., the links are randomly selected, but the graph should not be fully connected.



1. Derive the consensus equation. Show the degree matrix, adjancency matrix, and Laplacian matrix, and then the full equation in matrix form.

The degree matrix is the matrix which has the node degrees of G inside the diagonal:

$$\Delta(\mathcal{G}) = \begin{bmatrix} d(v_1) & 0 & \cdots & 0 \\ 0 & d(v_2) & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & d(v_n) \end{bmatrix}$$

$$\Delta(G) = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 \end{bmatrix}$$

The adjacency matrix is defined by

$$[Y(\mathcal{G})]_{ij} = \begin{cases} 1 & si\ v_i v_j \in E, \\ 0 & De\ otra\ manera. \end{cases}$$

$$Y(\mathcal{G}) = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & 0 \end{bmatrix}$$

where v_i is the current node and v_j is any of the adjacent nodes of v_i

$$Y(G) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

The Laplacian is defined:

$$\boxed{L(\mathcal{G}) = \Delta(\mathcal{G}) - Y(\mathcal{G})}$$

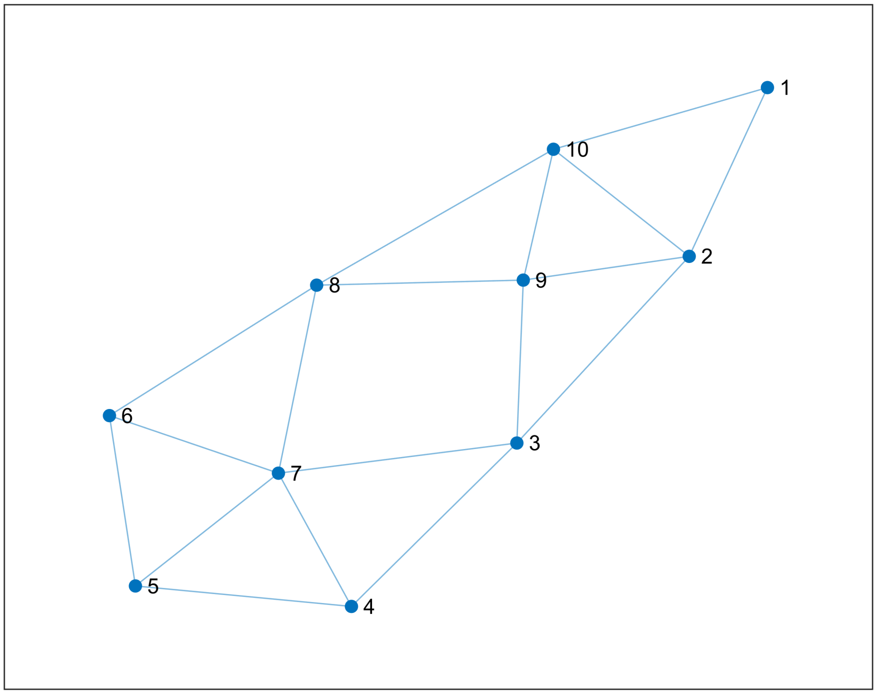
$$L(G) = \begin{bmatrix} 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ -1 & 4 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \\ 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 3 & -1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 & -1 & -1 & 5 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 4 & -1 & -1 \\ 0 & -1 & -1 & 0 & 0 & 0 & 0 & -1 & 4 & -1 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 4 \end{bmatrix}$$

Then, the full equation form is:

$$\dot{x} = L(G)x$$

$$
\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \\ \dot{x}_8 \\ \dot{x}_9 \\ \dot{x}_{10} \end{bmatrix}
=
\begin{bmatrix}
2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
-1 & 4 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \\
0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & -1 & 0 \\
0 & 0 & -1 & 3 & -1 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 \\
0 & 0 & -1 & -1 & -1 & -1 & 5 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & -1 & 4 & -1 & -1 \\
0 & -1 & -1 & 0 & 0 & 0 & 0 & -1 & 4 & -1 \\
-1 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 4
\end{bmatrix}
\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{bmatrix}
$$

```
Y = [0, 1, 0, 0, 0, 0, 0, 0, 0, 1;
     1, 0, 1, 0, 0, 0, 0, 0, 1, 1;
     0, 1, 0, 1, 0, 0, 1, 0, 1, 0;
     0, 0, 1, 0, 1, 0, 1, 0, 0, 0;
     0, 0, 0, 1, 0, 1, 1, 0, 0, 0;
     0, 0, 0, 0, 1, 0, 1, 1, 0, 0;
     0, 0, 1, 1, 1, 1, 0, 1, 0, 0;
     0, 0, 0, 0, 0, 1, 1, 0, 1, 1;
     0, 1, 1, 0, 0, 0, 0, 1, 0, 1;
     1, 1, 0, 0, 0, 0, 0, 1, 1, 0];
G = graph(Y);
plot(G)
```



2. Plot using Matlab the eigenvalues and comment about the distribution of the nodes connectivity and the localization of the eigenvalues.

The eigen values and eigen vectors come from:

$$L(G)\overrightarrow{X} = \lambda\overrightarrow{X}$$

$$(L(G) - \lambda I)\overrightarrow{X} = 0$$

Where λ are the eigenvalues corresponding to the eigenvectores X

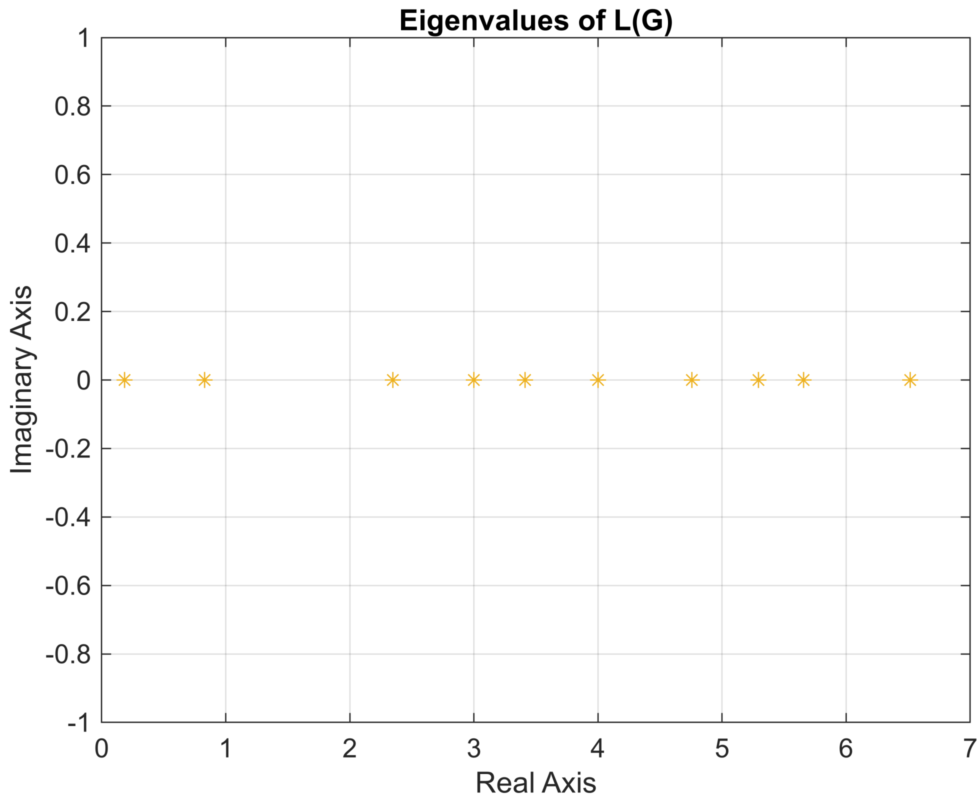
```
L= [2, -1, 0, 0, 0, 0, 0, 0, 0, -1;
    -1, 4, -1, 0, 0, 0, 0, 0, -1, -1;
     0, -1, 4, -1, 0, 0, -1, 0, -1, 0;
     0, 0, -1, 3, -1, 0, -1, 0, 0, 0;
     0, 0, 0, -1, 3, -1, -1, 0, 0, 0;
     0, 0, 0, 0, -1, 3, -1, -1, 0, 0;
     0, 0, -1, -1, -1, -1, 5, -1, 0, 0;
     0, 0, 0, 0, 0, 1, -1, 4, -1, -1;
     0, -1, -1, 0, 0, 0, 0, -1, 4, -1;
    -1, -1, 0, 0, 0, 0, 0, -1, -1, 4];

eigenvalues=eig(L)
```

```
eigenvalues = 10×1
   0.185248195377493
   0.830505268473319
   6.515559835252567
   2.346825696196454
   5.657523380709324
   5.293968839647963
   4.757751628789638
   3.999999999999996
   3.000000000000007
   3.412617155553241
```

```
[eigenvectors,D]=eig(L)
```

```
eigenvectors = 10×10
   0.360502184723704   0.357274292030795  -0.016704619018228   0.657037562663227 ⋯
   0.345549905134386   0.175595850058795   0.026268421068708  -0.082317505075289
   0.343808056532594  -0.160912740552553  -0.401782986633406  -0.184733369620316
   0.351770616693913  -0.471037839118041  -0.037129063684675   0.235519161410847
   0.337321837125526  -0.528477257426240  -0.186011819592510   0.334337177790159
   0.288689548599496  -0.342966674262348  -0.027259505709945  -0.021369964189201
   0.309017084494244  -0.332524112337508   0.718324251236293   0.004231256117084
   0.166250506276317   0.116936976862357  -0.436480008241063  -0.352526745388831
   0.305204797778927   0.117954017979915   0.303245535042489  -0.462829372163992
   0.308672085163311   0.242234552181145   0.049162285633201  -0.145560005022605
D = 10×10
   0.185248195377493                   0                   0                   0 ⋯
                   0   0.830505268473319                   0                   0
                   0                   0   6.515559835252567                   0
                   0                   0                   0   2.346825696196454
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
```

```
plot(eigenvalues,zeros(10),'*')
grid on;
ylabel('Imaginary Axis')
xlabel('Real Axis')
title('Eigenvalues of L(G)')
```



As the Laplacian of the graph L(G) is simetric and is defined semipositive, hence all of the the eigen values are real positive and can be ordered as:

$\lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \ldots \lambda_{10}$

We see that's true. It also can be seen that the Fiedler Eigenvalue (λ_2) is positive and, therefore **the Graph is connected,** i.e that exist at least one way go from a node v_i to any other node v_j inside the graph. Since no node is zero, then there's no node appart from the others, or a component isolated from the rest. Finally, the substantial variation among the eigenvalues highlights marked distinctions, suggesting that certain nodes in the graph exhibit significantly stronger connections than others.

3. Simulate the consensus protocol with distributed initial conditions from the interval I = [−5 5]. What is the agreement set? Obtain the time constant of the consensus equation and verify it in a plot of the time response (x_i vs $t, \forall i$)

In continuous time, the solution of the system of lineal equations of first order is :

$x(t) = e^{-L(G)t}x_0$

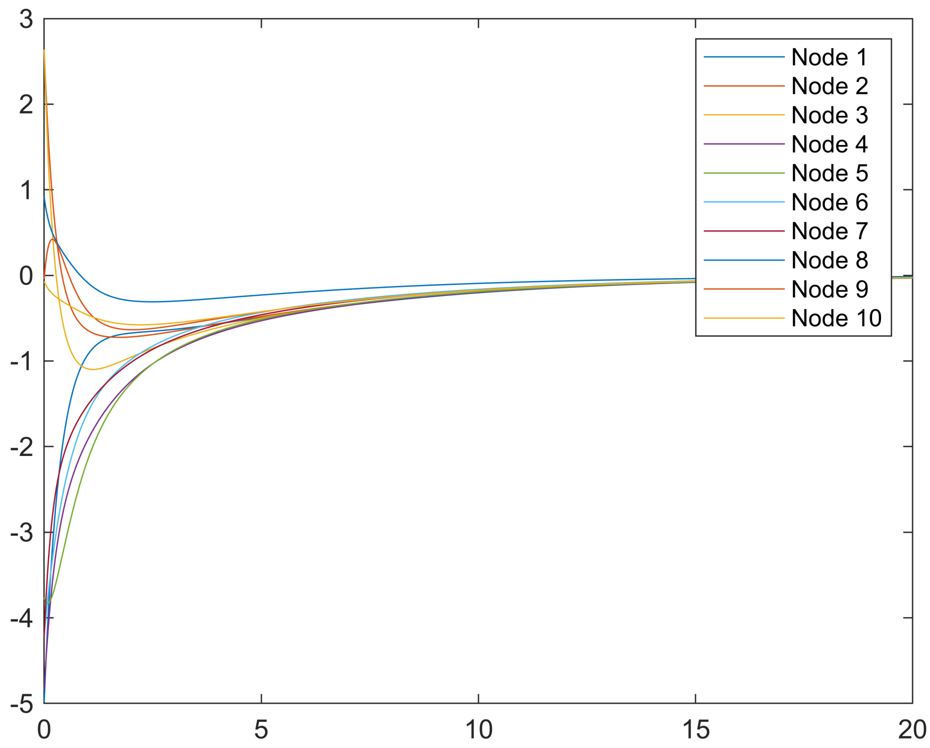
Since this cannot be computed in matlab this way given that L(G) is a matrix, i will use a function solver named 'ode45' which resolves differential equations using the intermediate method.

```
% First, it's necessary to define the time span:
```

```
tspan = [0, 20]; %t in seconds

% Number of nodes (10)
n = size(L, 1);

% Generate distributed initial conditions randomly in the interval [-5, 5]
x0 = -5 + 10 * rand(n, 1);

% Define the consensus equation dx_i/dt = -L*x_i
consensus_equation = @(t, x) -L * x;

% Numerically simulate the consensus equation for each node
[t, x] = ode45(consensus_equation, tspan, x0);
```

The agreement set is defined as the set of nodes inside a graph that reach the concensus, i.e the nodes try to reach to a commoun value, obbiusly with a certain tolerance. In the code, I'm comparing the value of the last nodes states with the value of the last node state of the first node x1. Then if all of the last states are equal to the x1 last state, so the agreement set is reached.

```
% Find the agreement set (nodes that reach consensus)
tolerance = 1e-3;
if all(abs(x(end, :) - x(end, 1)) < tolerance)
    agreement_set=x(end,1) %0
end

% Plot the time response for each node
for i = 1:n
    plot(t, x(:, i));
    hold on;
end
legend('Node 1','Node 2','Node 3','Node 4','Node 5','Node 6','Node 7','Node 8','Node 9','Node 1
```



It's showed that the constant is zero and it goes accordly to the theory:

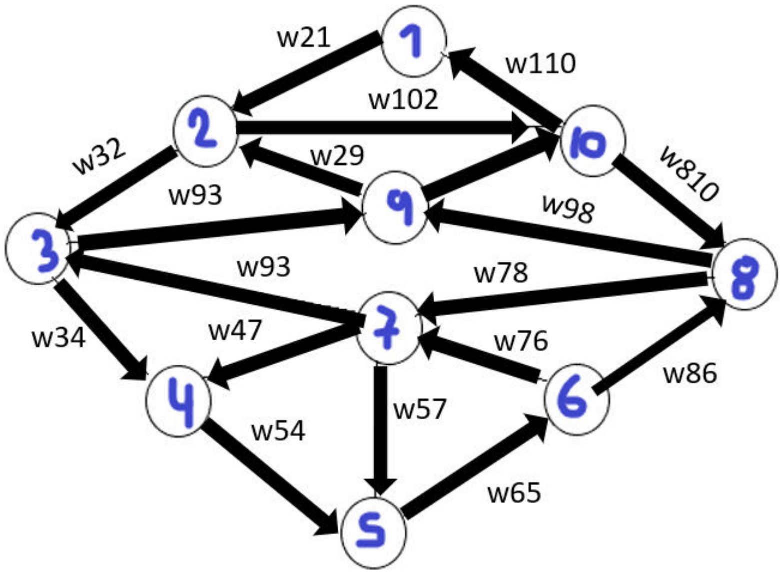
$$\lim_{t \to \infty} x(t) = 0$$

4. Add some weights and directions to the graph. You should guarantee that the new agreement protocol reach consensus. Explain what have you considered to guarantee convergence to an agreement set.x.

When there's a diagraph and it has unitary weights, then in the compact concencus problem we have to account the directions that enter into the node and the respectively weight.

$$\dot{x} = L(D)x$$

Stablishing a the weights and the directions arbitrary of this way:



We have:

$$L(G)=\begin{bmatrix} w_{110} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -w_{110} \\ -w_{21} & w_{21}+w_{29} & 0 & 0 & 0 & 0 & 0 & 0 & -w_{29} & 0 \\ 0 & -w_{32} & w_{32}+w_{37} & 0 & 0 & 0 & -w_{37} & 0 & 0 & 0 \\ 0 & 0 & -w_{43} & w_{43}+w_{47} & 0 & 0 & -w_{47} & 0 & 0 & 0 \\ 0 & 0 & 0 & -w_{54} & w_{54}+w_{57} & 0 & -w_{57} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -w_{65} & w_{65} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -w_{76} & w_{76}+w_{78} & -w_{78} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -w_{86} & 0 & w_{810}+w_{86} & 0 & -w_{810} \\ 0 & 0 & -w_{93} & 0 & 0 & 0 & 0 & -w_{98} & w_{93}+w_{98} & 0 \\ 0 & -w_{102} & 0 & 0 & 0 & 0 & 0 & 0 & -w_{910} & w_{102}+w_{910} \end{bmatrix}$$

Having in account the next theorem

Theorem 3.12. *For a digraph $\mathcal{D}$ containing a rooted out-branching, the state trajectory generated by (3.14), initialized from x_0, satisfies*

$$\lim_{t\to\infty} x(t) = (p_1 q_1^T)\, x_0,$$

where p_1 and q_1, are, respectively, the right and left eigenvectors associated with the zero eigenvalue of $L(\mathcal{D})$, normalized such that $p_1^T q_1 = 1$. As a result, one has $x(t) \to \mathcal{A}$ for all initial conditions if and only if $\mathcal{D}$ contains a rooted out-branching.

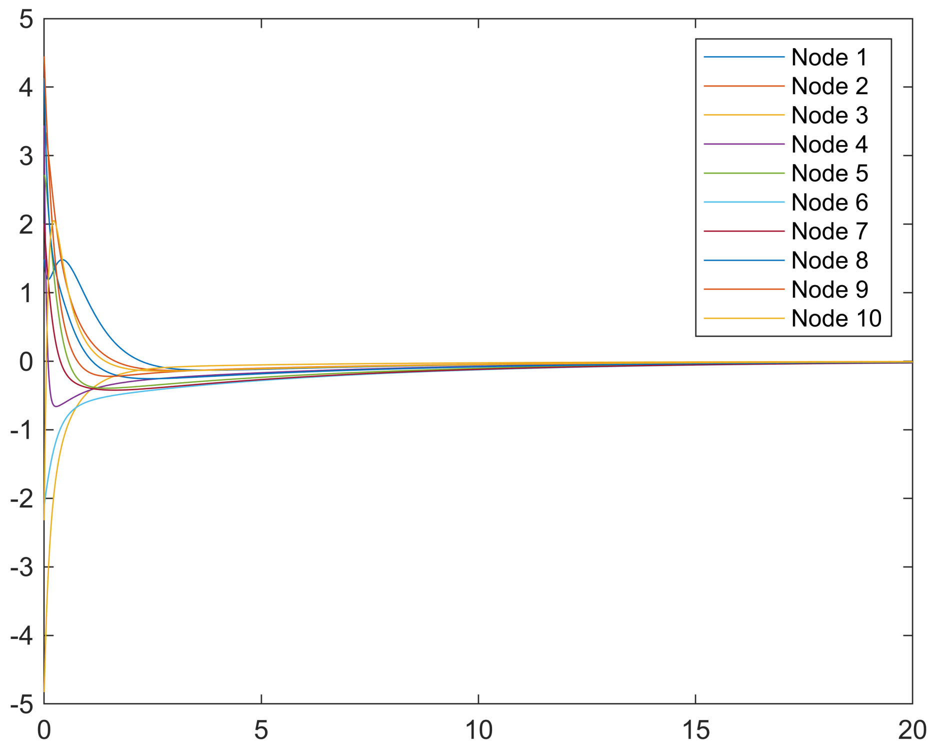
And the next propostion is accomplished:

Proposition 3.8. *A digraph $\mathcal{D}$ on n vertices contains a rooted out-branching as a subgraph if and only if* **rank** $L(\mathcal{D}) = n - 1$. *In that case,* $\mathcal{N}(L(\mathcal{D}))$ *is spanned by the vector of all ones.*

Choosing arbitrary values

$$L(G) = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 \\ -3 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & -5 & 0 \\ 0 & -4 & 7 & 0 & 0 & 0 & -3 & 0 & 0 & 0 \\ 0 & 0 & -8 & 17 & 0 & 0 & -9 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 & 5 & 0 & -3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -3 & 4 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & 0 & 5 & 0 & -3 \\ 0 & 0 & -2 & 0 & 0 & 0 & 0 & -1 & 3 & 0 \\ 0 & -6 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & 10 \end{bmatrix}$$

```
L = [2, 0, 0, 0, 0, 0, 0, 0, 0, -2;
    -3, 8, 0, 0, 0, 0, 0, 0, -5, 0;
    0, 4, 7, 0, 0, 0, -3, 0, 0, 0;
    0, 0, -8, 17, 0, 0, -9, 0, 0, 0;
    0, 0, 0, -2, 5, 0, -3, 0, 0, 0;
    0, 0, 0, 0, -1, 1, 0, 0, 0, 0;
    0, 0, 0, 0, 0, -3, 4, -1, 0, 0;
    0, 0, 0, 0, 0, -2, 0, 5, 0, -3;
    0, 0, -2, 0, 0, 0, 0, -1, 3, 0;
    0, -6, 0, 0, 0, 0, 0, 0, -4, 10];

% First, it's necessary to define the time span:
tspan = [0, 20]; %t in seconds
% Number of nodes (10)
n = size(L, 1);
% Generate distributed initial conditions randomly in the interval [-5, 5]
x0 = -5 + 10 * rand(n, 1);
% Define the consensus equation dx_i/dt = A*x_i (for graphs with direction and weights)
consensus_equation = @(t, x) -L * x;
% Numerically simulate the consensus equation for each node
[t, x] = ode45(consensus_equation, tspan, x0);
% Plot the time response for each node
figure
for i = 1:n
    plot(t, x(:, i));
    hold on;
end
legend('Node 1','Node 2','Node 3','Node 4','Node 5','Node 6','Node 7','Node 8','Node 9','Node
```



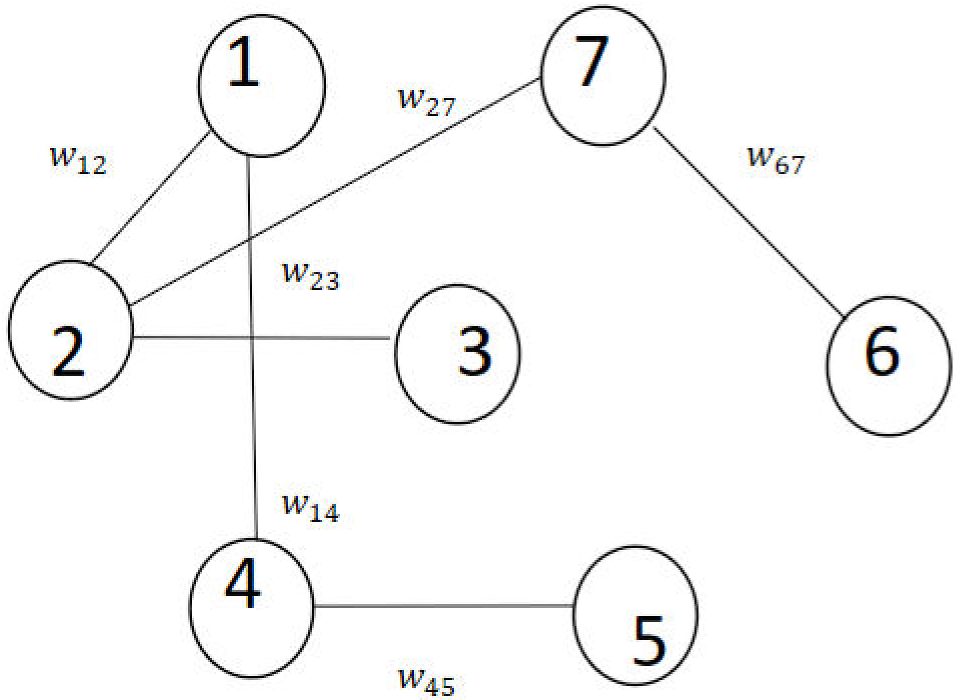

```
eig(L)
```

```
ans = 10×1 complex
 17.014942188626346 + 0.000000000000000i
  0.162342442784495 + 0.000000000000000i
  1.858043480922473 + 0.000000000000000i
  2.234560216834665 + 0.000000000000000i
  4.537613725679439 + 2.588743730168143i
  4.537613725679439 - 2.588743730168143i
  9.140024164438959 + 0.199568343655513i
  9.140024164438959 - 0.199568343655513i
  6.687417945297618 + 0.316569180684525i
  6.687417945297618 - 0.316569180684525i
```

As the previous exercise, the eigenvalues are all positive and the rank is 10, that is to say that the maximum number of columns are lineary independent.

1) Consensus Protocol: Discrete - Time

Consider an undirected graph of 7 nodes (no fully connected) as your network of agents.



1. Find a weighted matrix satisfying the assumptions for convergence (doubly-stochastic, positive entries, etc.)

a) Each A(t) is a **stochastic** matrix what is compliant with the graph G_t i.e: $\mathrm{A}_{\mathrm{ij}} > 0 \text{ for } (j,i) \in E_t$ for all t.

b) (Aperiodicity) The diagonal entries of each A(t) are positive, $\mathrm{A}_{\mathrm{ii}}(t) > 0 \text{ for } t \text{ and } i \in V$

c) (Uniform Positivity) There is a scalar β > 0 such that $\mathrm{A}_{\mathrm{ij}}(t) > \beta$ whenever $\mathrm{A}_{\mathrm{ij}}(t) > 0$

d) (Irreducibility) Each graph G_t is strongly connected.

The Laplacian is:

$$\boxed{L(\mathcal{G}) = \Delta(\mathcal{G}) - Y(\mathcal{G})}$$

$$L = \begin{bmatrix} w_{12}+w_{14} & -w_{12} & 0 & -w_{14} & 0 & 0 & 0 \\ -w_{12} & w_{12}+w_{23}+w_{27} & -w_{23} & 0 & 0 & 0 & -w_{27} \\ 0 & -w_{23} & w_{23} & 0 & 0 & 0 & 0 \\ -w_{14} & 0 & 0 & w_{14}+w_{45} & -w_{45} & 0 & 0 \\ 0 & 0 & 0 & -w_{45} & w_{45} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{67} & -w_{67} \\ 0 & -w_{27} & 0 & 0 & 0 & -w_{67} & w_{67}+w_{27} \end{bmatrix}$$

$A = (I - \alpha L)$

$$A = \begin{bmatrix} 1-\alpha(w_{12}+w_{14}) & \alpha w_{12} & 0 & \alpha w_{14} & 0 & 0 & 0 \\ \alpha w_{12} & 1-\alpha(w_{12}+w_{23}+w_{27}) & \alpha w_{23} & 0 & 0 & 0 & \alpha w_{27} \\ 0 & \alpha w_{23} & 1-\alpha w_{23} & 0 & 0 & 0 & 0 \\ \alpha w_{14} & 0 & 0 & 1-\alpha(w_{14}+w_{45}) & \alpha w_{45} & 0 & 0 \\ 0 & 0 & 0 & \alpha w_{45} & 1-\alpha w_{45} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1-\alpha w_{67} & \alpha w_{67} \\ 0 & \alpha w_{27} & 0 & 0 & 0 & \alpha w_{67} & 1-\alpha(w_{67}+w_{27}) \end{bmatrix}$$

We see that the matrix is already stochastic by rows and stochastic by columns independent of the α value and the weights.

To acomplish the first 3 assumptions **a) b) and c)** of the matrix, all of the elements of the matrix must be grater than 0. So choosing $\alpha = \frac{1}{7}$,we have:

$$w_{67} < \frac{1}{\alpha} \rightarrow w_{67} = 4$$

$$w_{27} < \frac{1}{\alpha} - w_{27} \rightarrow w_{27} = 1$$

$$w_{45} < \frac{1}{\alpha} \rightarrow w_{45} = 1$$

$$w_{14} < \frac{1}{\alpha} - w_{45} \rightarrow w_{14} = 5$$

$$w_{23} < \frac{1}{\alpha} \rightarrow w_{23} = 4$$

$$w_{12} < \frac{1}{\alpha} - w_{14} \rightarrow w_{12} = 0.5$$

$$A = \begin{bmatrix} \frac{3}{14} & \frac{1}{14} & 0 & \frac{5}{7} & 0 & 0 & 0 \\ \frac{1}{14} & \frac{1}{14} & \frac{4}{7} & 0 & 0 & 0 & \frac{1}{7} \\ 0 & \frac{4}{7} & \frac{3}{7} & 0 & 0 & 0 & 0 \\ \frac{5}{7} & 0 & 0 & \frac{1}{7} & \frac{1}{7} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{7} & \frac{6}{7} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{3}{7} & \frac{4}{7} \\ 0 & \frac{1}{7} & 0 & 0 & 0 & \frac{4}{7} & \frac{2}{7} \end{bmatrix}$$

And at last, the irreducibility **d)** is fulfilled, as there exists a path from every node to reach any other node in the graph.

2. Consider a scalar state x ∈ R, propose a consensus protocol in discrete-time or some initial conditions. Analyze the convergence from a theoretical point of view and in simulation (implement the protocol using Matlab (or Python) and show the results).

Beacause the ireative process is distributed according to the local agent. Also the algorithm will converge, but it´s tricky to find the rate of conversion due to it depends on the weight matrix A, and the conversion depends on the in A as well.

From the theorical point of view, the solution converges when it satisfies the following:

$$\lim_{n \to \infty} A^n = \text{constant}$$

As the elements of the matrix are all smaller than 1, so this limit it's true and therefore the solution converges. Also all of the eigenvalues acomplish the Ruth Hurwitz criterion ($|\lambda_i|$<1) for completely convergence.

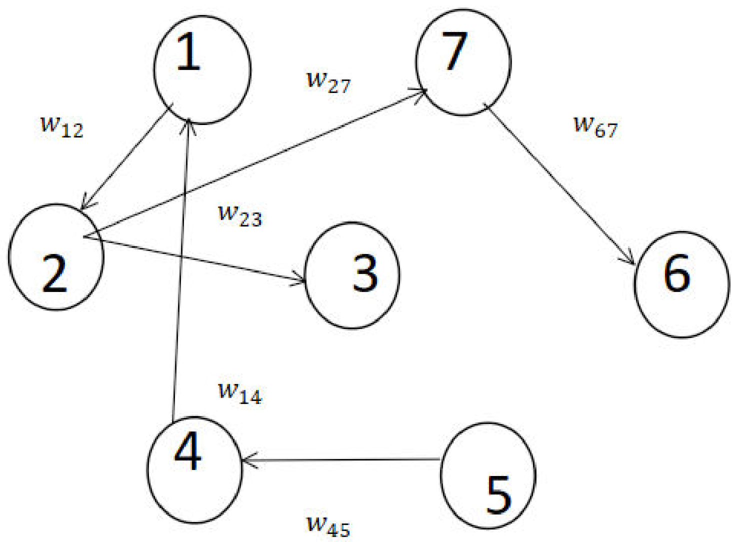
```
A = [3/14, 1/14, 0, 5/7, 0, 0, 0;
     1/14, 1/14, 4/7, 0, 0, 0, 1/7;
     0, 4/7, 3/7, 0, 0, 0, 0;
     5/7, 0, 0, 1/7, 1/7, 0, 0;
     0, 0, 0, 1/7, 6/7, 0, 0;
     0, 0, 0, 0, 0, 3/7, 4/7;
     0, 1/7, 0, 0, 0, 4/7, 2/7];
A^inf %0
```

```
ans = 7×7
     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     0     0     0     0
     0     0     0     0     0     0     0
     0     0     0     0     0     0     0
```

```
eig(A) %<1
```

```
ans = 7×1
  -0.553800448901036
  -0.388942395378279
  -0.179263771239329
   0.773166785427477
   0.829678311361490
   0.960119319080658
   0.987613628220448
```

3. Consider the same graph but with directed edges having a root-outbranching node. Find the consensus protocol and analyze the convergence from a theoretical point of view and in simulation (implement the protocol using Matlab (or Python) and show the results).



In this case the root out branching node is the node number 5.

The matrix L is:

$$L = \begin{bmatrix} w_{14} & 0 & 0 & -w_{14} & 0 & 0 & 0 \\ -w_{12} & w_{12} & 0 & 0 & 0 & 0 & 0 \\ 0 & -w_{23} & w_{23} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_{45} & -w_{45} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{67} & -w_{67} \\ 0 & -w_{27} & 0 & 0 & 0 & 0 & w_{27} \end{bmatrix}$$

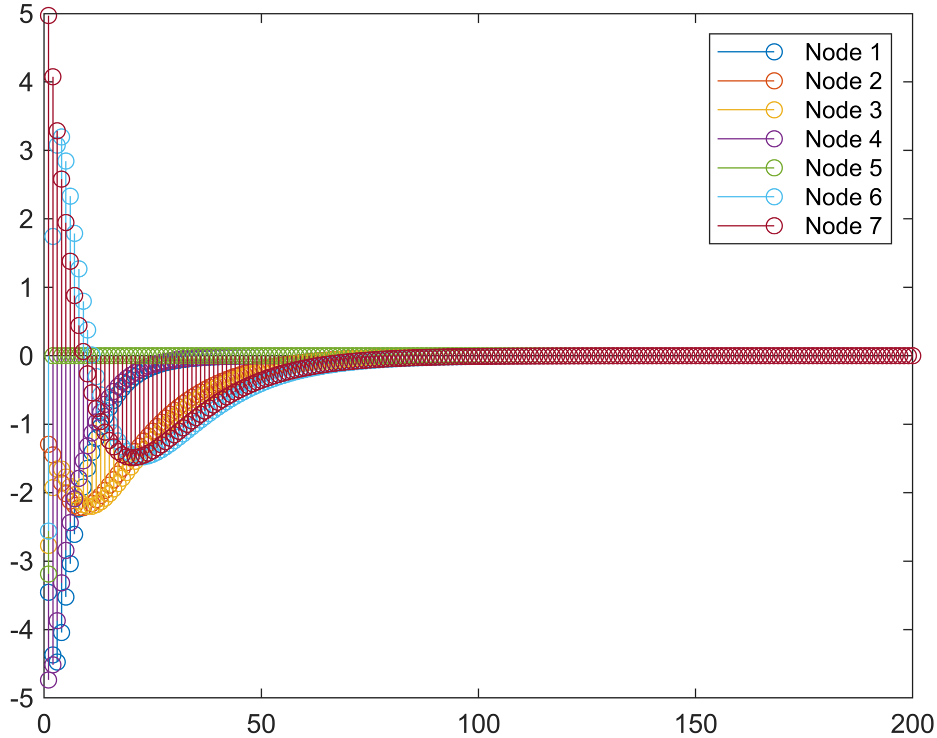
$$A = \begin{bmatrix} 1-\alpha w_{14} & 0 & 0 & \alpha w_{14} & 0 & 0 & 0 \\ \alpha w_{12} & 1-\alpha w_{12} & 0 & 0 & 0 & 0 & 0 \\ 0 & \alpha w_{23} & 1-\alpha w_{23} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1-\alpha w_{45} & \alpha w_{45} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1-\alpha w_{67} & \alpha w_{67} \\ 0 & \alpha w_{27} & 0 & 0 & 0 & 0 & 1-\alpha w_{27} \end{bmatrix}$$

$$A = \begin{bmatrix} \frac{2}{7} & 0 & 0 & \frac{5}{7} & 0 & 0 & 0 \\ \frac{1}{14} & \frac{13}{14} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{4}{7} & \frac{3}{7} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{6}{7} & \frac{1}{7} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{3}{7} & \frac{4}{7} \\ 0 & \frac{1}{7} & 0 & 0 & 0 & 0 & \frac{6}{7} \end{bmatrix}$$

$x[n+1] = \mathrm{Ax}[n]$

We see the matrix is stochastic in rows, but not in columns with the previouly values of α and the weights, it means that it reaches the semi-convergence, i.e that the value of the nodes could be near, but neccesary and not in all the cases. We can verify this on simulation:

```
A = [2/7, 0, 0, 5/7, 0, 0, 0;
     1/14, 13/14, 0, 0, 0, 0, 0;
     0, 4/7, 3/7, 0, 0, 0, 0;
     0, 0, 0, 6/7, 1/7, 0, 0;
     0, 0, 0, 0, 0, 0, 0;
     0, 0, 0, 0, 0, 3/7, 4/7;
     0, 1/7, 0, 0, 0, 0, 6/7];

% Number of nodes (10)
m = size(A, 1);
% Generate distributed initial conditions randomly in the interval [-5, 5]
x0 = -5 + 10 * rand(m, 1);
% The solution to the concensus equation in discrete time is iterative
x=zeros(m,200);
x(:,1)=x0;
for n=1:199
   x(:,n+1)=A*x(:,n);
end
% Plot the discrete time response for each node

figure
for i = 1:m
    stem(1:200,x(i,:))
    hold on;
end
%}
legend('Node 1','Node 2','Node 3','Node 4','Node 5','Node 6','Node 7');
```



4. Consider again the undirected graph but a time-varying weighted matrix A(t). Find the consensus protocol and analyze the convergence from a theoretical point of view and in simulation (implement the protocol using Matlab (or Python) and show the results).

When there's a time-varying weighted matrix, the concensus is by the way:

$$x_i(t+1) = \sum_{j \in N_i(t)} A_{ij}(t) x_j(t)$$

Which can be written as:

$$\boxed{x(t+1) = A(t)x(t)} \quad \forall t \geq 0,$$

$x(t)$ will converge to a concensus if the matrixes $A(t)A(t-1)..A(1)A(0)$ converge to a rank one matrix as $t \to \infty$

Using the distributed subgradient method, the concensus - like step :

$$\mathbf{w}_i(t+1) = \sum_{j=1}^{m} A_{ij} \mathbf{x}_j(t) \qquad (a_{ij} = 0 \text{ when } j \notin N_i).$$

Followed by a local subgradient step:

$$\mathbf{x}_i(t+1) = \mathcal{P}_X[\mathbf{w}_i(t+1) - \alpha(t)\tilde{\nabla} f_i(\mathbf{w}_i(t+1))]$$

where $\mathcal{P}_X[y]$ is the Euclidean projection of y on X and $\alpha(t) > 0$ is a stepsize.

Using a code for the paper "Spectral-like gradient method for distribuited optimization" by Jakovetic, i use a random geometric graph model of 7 nodes.

Firtst, it's necesary to run the Generator_random_Probabilities_Disc_Graph.m, then the generateDataVectorQuadratic.m for finding the xOpt and finally the MainSpectral.m to see the response in time.