Modeling and control of cyber-physical systems
Project II

Merangolo Maria Francesca (s318905), Pisani Fabrizio (s305391),
Zampolini Matilde (s316182), Zulian Matteo (s310384)

a.y. 2022-2023

1 Problem description

1.1 Introduction

In this project activity we were asked to create a distributed controller for a multi-agent system made by 7 magnetic levitators. A magnet is activated by a suitable current flowing into a solenoid. This magnet is going to provide an electromagnetic field which is attracting a ball. The objective is to make this ball levitate and possibly track a desired behavior dictated by a leader. The state variables for each agent are the position and the speed of the ball, while the only variable that is measurable is the output provided by optical sensors.

This is known as cooperative tracking problem and it works under the assumption that all the agents and the leader are identical. In our case the cyber-physical system is modeled by the same matrix A. The matrix C is not the identity matrix. It means state variables are not directly available.

The model of the follower agents is: $\dot{x}_i = Ax_i + Bu_i$, $y_i = Cx_i$

The model of the leader agent is: $\dot{x}_0 = Ax_0$, $y_0 = Cx_0$

1.2 Topology

The agents share information between each other on a communication network. It can be modeled as a weighted directed graph where the root is the leader node. There are some requirements: the leader node must be connected to at least one of the follower agents and there must be a path that connects the leader to each agent, otherwise the problem is infeasible.

It is clear that different topologies will determine different convergence times. For example we will expect that the star topology on the left will be better than the topology on the right because the leader communicates directly to every agent.

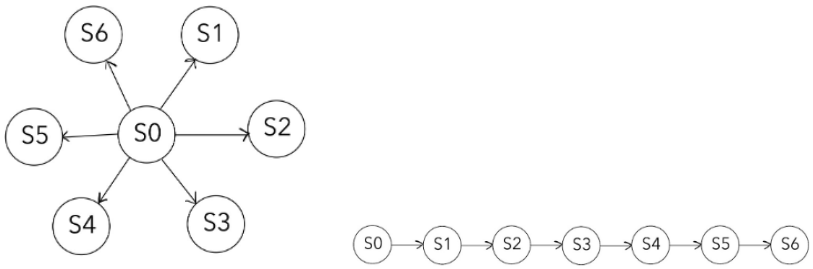

Figure 1: Topology 1 and 2

The graph is described by some matrices:

- Adjacency matrix A_d: a_{ij} contains the weight for the edge that connect agent j to agent i
- In-degree matrix D: d_i is the in-degree of the node i
- Laplacian matrix $L = D - A$

- Pinning matrix G: g_i is the pinning gain, i.e. the weight of the edge that connects the leader to agent i.

```
% Topology 2: Linear topology
Ad = [0 0 0 0 0 0; 2 0 0 0 0 0; 0 6 0 0 0 0; 0 0 1 0 0 0; 0 0 0 1 0 0;
    0 0 0 0 3 0]; % adjacency matrix
D = diag([0 2 6 1 1 3]); % in-degree matrix
G = diag([1 0 0 0 0 0]); % c = 5
% Topology 1: Star topology (each agent receives information only from
    the leader)
Ad = [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];
D = diag([0 0 0 0 0 0]);
G = diag([1 1 1 1 1 1]); % leader connected with all nodes
```

Considering a constant zero output we obtain these results:

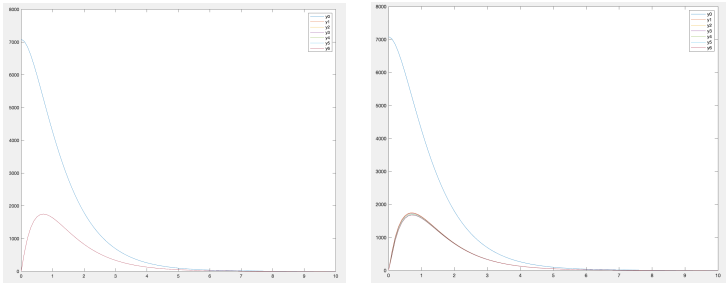

Figure 2: Constant output with topology 1 and 2

All the results shown in this report will be computed using topology 2 because being the most disadvantaged one we can see the real use of the algorithm.

1.3 Cooperative state variables feedback (SBVF) control

Assumption: each node knows its own state variables x and that it can share it with its neighbors. Each node i can compute a neighborhood tracking error

$$z(i) = \sum_{j=1}^{N} a_{ij}(x_j - x_i) + g_i(x_0 - x_i) \tag{1}$$

which is a number that explains how far is the state x_i of node i from the state of its neighbors. It can be used by the agent to correct itself. A large z will mean a large correction to be made. This correction is actuated exploiting the input $u_i = cKz_i$, where the c is the coupling gain and K is the feedback gain matrix.

Exploiting the theory, we obtain a closed-loop system of each node i:

$$\dot{x}_i = Ax_i + cBK\Big(\sum_{j=1}^{N} a_{ij}(x_j - x_i) + g_i(x_0 - x_i)\Big) \tag{2}$$

and the overall global closed-loop dynamics: $\dot{x} = (I_N \otimes A - c(L+G) \otimes BK)x + (c(L+G) \otimes BK)\underline{x_0}$
As said before, our goal is that every follower agent tracks the behavior of our leader node. For this reason are useful these definitions:

- the local disagreement error is the difference between the x_i and the x_0: $\delta_i = x_i - x_0$
- the global disagreement error is nothing but a vector that collect every local disagreement error: $\delta = col(\delta_1, \delta_2, \delta_N) = x - \underline{x_0}$

This problem is solved if the global disagreement error converges to zero when the time goes to infinity. We define the matrix A_c as: $A_c = I_N \otimes A - c(L+G) \otimes BK$.

We know that global disagreement error converges to zero if and only if the matrix A_c is Hurwitz. So, in order to reach our goal, it is clear that we have to act on the two parameters which are not known: the coupling gain c and the feedback gain matrix K.

Thanks to a Theorem, we have theoretical information on the design of c and K:

$$K = R^{-1}B^T P \tag{3}$$

where R is a design matrix, B is known, P is the unique positive definite solution of the control algebraic Riccati equation:

$$0 = A^T P + PA + Q - PBR^{-1}B^T P \tag{4}$$

where Q is a design matrix.

To designing c, the Theorem impose a constraint:

$$c \geq \frac{1}{2\min_{i \in N} Re(\lambda_i)} \tag{5}$$

1.4 Observers

After designing the SVFB protocol, we have to solve the problem of observability: we assume that the state variables of the agents are not directly measurable. For this reason, we have to design observers that estimate an $\hat{x}$ based on the information it has at that moment. At each step it tries to improve its estimation to be more accurate.

For the state estimation of the leader agent we just used a Luenberger observer using the command `Lun = place(A',C',[-10 -20]);` to set the observer eigenvalues at desired locations to estimate the states of the system.

For the follower node observer we use two different approaches: cooperative observers and local observers.

1.4.1 Cooperative observer

From the theory, we define the local neighborhood output estimation error:

$$\zeta_i = \sum_{j=1}^{N} a_{ij}(\tilde{y_j} - \tilde{y_i}) + g_i(\tilde{y_0} - \tilde{y_i}) \tag{6}$$

The cooperative observer for the node i is:

$$\dot{\hat{x}_i} = A\hat{x}_i + Bu_i - cF\zeta_i \tag{7}$$

where c (as said before) is the coupling gain and F is the observer gain matrix.

$$\dot{\hat{x}} = A_0\hat{x} + (I_N \otimes B)u + c[(L+G) \otimes F]y \tag{8}$$

The global state estimation error is $\tilde{x} = x - \hat{x}$ and our goal is that this goes to zero.

We define the matrix A_0 as: $A_0 = I_N \otimes A - c[(L+G) \otimes FC]$

We know that global state estimation error converges to zero if and only if the matrix A_0 is Hurwitz. So, in order to reach our goal, it is clear that we have to act on the two parameters which are not known: the coupling gain c and the observer gain matrix F. The constraint on the coupling gain is the same as before. Thanks to a Theorem, we have theoretical information on the designing of F:

$$F = PC^T R^{-1} \tag{9}$$

and P is (as before) the unique positive definite solution of the ARE.

1.4.2 Local observer

The second approach is easier because instead of compute the neighborhood estimation error, the observer exploits only its local information. Connecting the SVFB control protocol, we obtain:

$$\dot{\hat{x}}_i = A\hat{x}_i + Bu_i - cF\tilde{y}_i \tag{10}$$

A Lemma tells us that in order to drive the state estimation error $\tilde{x}$ to 0 asymptotically we have just to design the matrix F such that $(A + cFC)$ is Hurwitz.

In order to do that we can compute F as before but with a minus sign:

$$F = -PC^T R^{-1} \tag{11}$$

However, by reasoning in terms of the eigenvalues that we want to impose on the observer, we can obtain better results because we have greater control over the system:

```
Fc=place(A',C',[-10 -20]); % Fc = c*F
F=Fc/c;
F=-F';
```

2 Outputs

At this moment we have completed the controller and the observers, so we can try to simulate the system. We notice that the free response of the maglev diverges to infinity, that's because the eigenvalues of A don't all have negative real part.

We can't act on the input of S0, so we must change the eigenvalues of the matrix A to change the free response of the system.

Studying the modal analysis in control system theory, we learned how to create different output by selecting the right set of eigenvalues and the appropriate initial conditions.

- Constant zero output: we need a set of λ with all strictly negative real part because the mode $e^{Re(\lambda)t}$ will eventually converge to zero, the magnitude of λ and the initial conditions will determine the convergence time.
- Constant output C: we need $Re(\lambda_1) \leq 0$ and $Re(\lambda_2) = 0$. C value will be influenced by x_0.

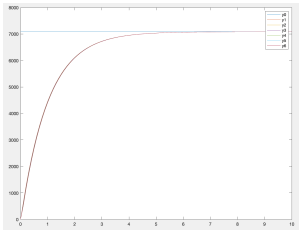

Figure 3: Constant output

- Sinusoidal output with amplitude A and frequency w: $Re(\lambda) = 0$ and $Im(\lambda)$ complex conjugate, w is chosen by the magnitude of the complex part of λ while A is determined by x_0.

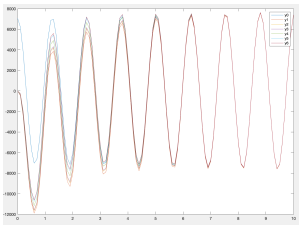

Figure 4: Sinusoidal output

- Ramp output with slope R: all $\lambda = 0$ and R depends on x_0.

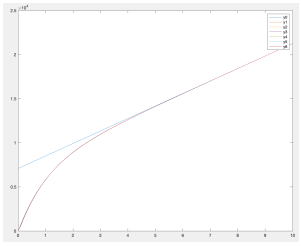

Figure 5: Ramp output

We can start to simulate our system using the new A, computed like in a closed loop $A - BK$, where K is generated by the command `place` with the desired $[\lambda_1, \lambda_2]$. This operation was possible because the rank of the controllability matrix is maximum, so the system is completely controllable and we were able to change both the eigenvalues of A.

3 Effect of the coupling gain c

The coupling gain c is a parameter that directly changes the energy of the input $u = cBKz_i$ and the contribute of the observer $cF\zeta_i$. This means that with increasing c, the corrections actuated by each agent will increase accordingly, improving the velocity to reach the consensus between the agents. However, we notice a lost in precision during the tracking of the leader.

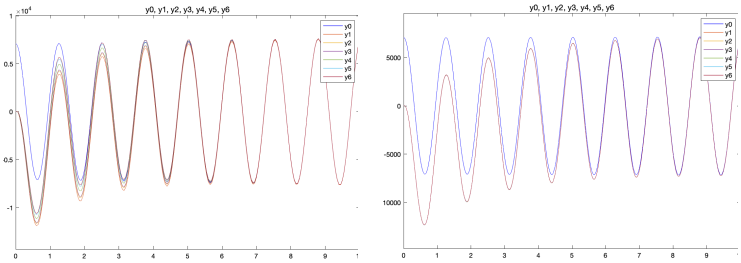

Figure 6: Effect of the coupling gain in a sinusoidal ouput (c=c*100 in the right figure)

In the right figure signals from y_1 to y_6 are coincident almost immediately to reach y_0, while they are slower in the left figure.

By looking at the definition of the coupling gain:

```
lam_g = eig(L+G);
c = 10/(2*min(real(lam_g)));
```

we see that the network topology affect quite significantly the effect of the coupling gain because it depends on the eigenvalues of the matrix $(L + G)$. Therefore we can also increase it by selecting a different topology. With the linear topology defined before we obtained $c = 5$.

With this other kind of topology:

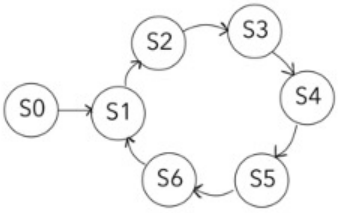

Figure 7: Coupling gain in a different topology

```
Ad = [0 0 0 0 0 1; 2 0 0 0 0 0; 0 6 0 0 0 0; 0 0 1 0 0 0; 0 0 0 3 0 0;
     0 0 0 0 1 0];
D = diag([1 2 6 1 1 3]);
G = diag([1 0 0 0 0 0]);
```

the coupling gain c is almost 27.

4 Choice of Q and R

Our algorithm has to find the best tradeoff between two requirements: minimize the energy of the state (related to matrix Q) and minimize the energy of the command input (related to matrix R). The only thing that matter is how big the weights of the states are with respect to the weight of the command input, so we have to analize the ratio `q/r`. We begin with this choice:

```
q = 1;
r = q/10;
Q = q*eye(n);
R = r;
```

If `r` is smaller than `q` it means that we want to focus more on perfomance than on saving energy (because we are penalizing more `r`).

The ratio `q/r` will determine an increase in both consensus of the agents and accuracy with respect of the leader:

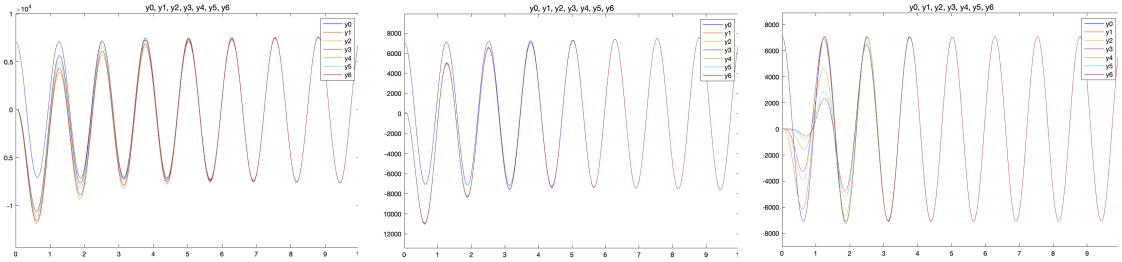

Figure 8: Effect of the ratio q/r. From left to right: r=q/10, r=q/1000, r=q*100

5 Noise

In real case studies it happens that the sensors are affected by noise that could disturb the functioning of the algorithm if its effect is too strong.

In this section we analize the different effects of the noise in the cooperative observer and the local observer, in presence of a sinusoidal output, because it's the most difficult to track.

5.1 No noise

When there is no noise the two types of observers show no difference in their behavior.

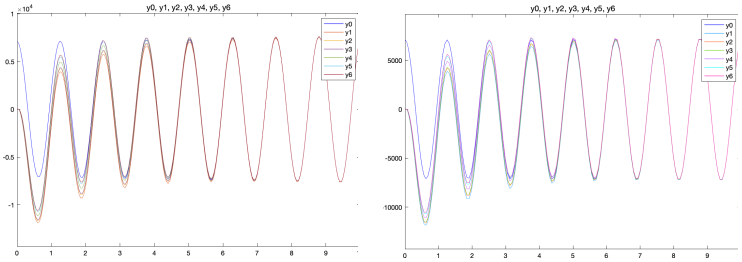

Figure 9: Distributed and local observers with no noise

5.2 Medium noise applied to all the agents

The outputs are in the order of 10^3. Firstly, we applied a constant noise with magnitude in the order of 10^1 to all the agents, to see if there was any difference.

How the local observer is computed and more specifically, the magnitude of the eigenvalues used to compute the observer gain matrix F, makes the local observer more or less accurate than the cooperative one. Here some examples:

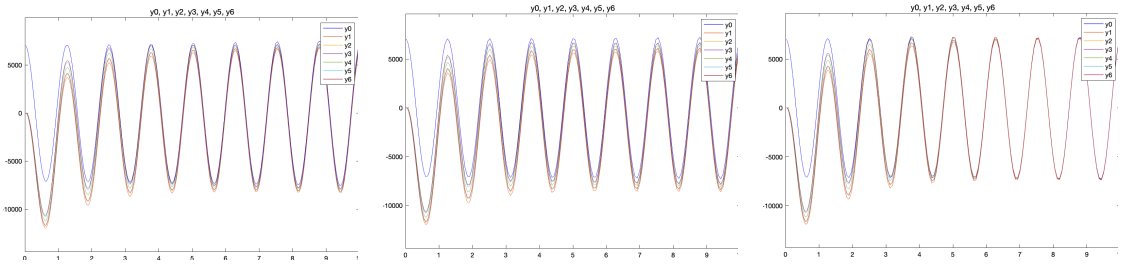

Figure 10: Cooperative observer - Local observer with eig $= [-1, -2]$ - Local observer with eig $= [-10, -20]$

We can see that the cooperative observer struggles to reach the consensus, but its output looks bounded. Instead, the local observer with little eigenvalues has worst performances and its output looks like will never reach consensus. On the other hand, with bigger eigenvalues, the performances are highly improved, and the agents can track correctly the leader node, so, in this case, a well-designed local observer is better than a cooperative one, otherwise is the opposite.

5.3 Strong noise on one agent using linear topology

In this particular case we applied a 10^2 constant noise only to the fourth agent to see how the observers reacted. First of all, we expect that y_0, y_1, y_2 and y_3 are not influenced by the noise, because of the linear topology they don't depend on y_4, instead of y_5 and y_6 which are descendants of y_4.

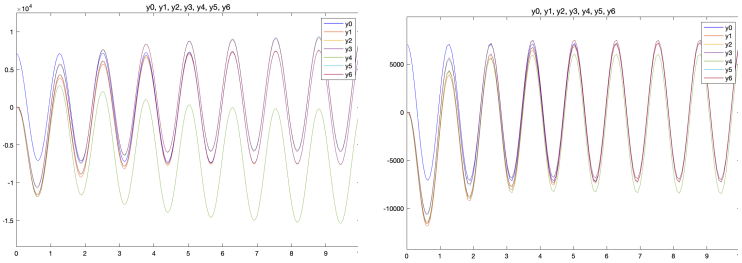

Figure 11: Distributed and local observers with strong noise

In Figure 11 we can see that y_4 goes off track in both cases, in the local one the effect of the noise is attenuated a little, but the value of y_4 is still not satisfying.

The other agents look the same in both cases but with a closer look we can see that in the cooperative observer the noise has spread to y_4's neighborhood preventing y_5 and y_6 to track correctly

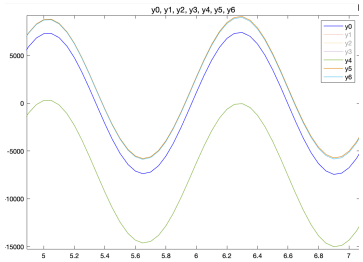

Figure 12: Closer look on distributed observer

y_0 while in the local they reached consensus and tracked the leader without being influenced by the wrong values of y_4.

With this particular case we could see the strength of a local algorithm where a failure of a single device doesn't prevent the overall processing.