Distributed Autonomous Systems 2023-24

Course Project

The project consists in two main tasks. The first one involves a data analytics application, while the second one deals with the control for multi-robot systems in ROS 2.

Task 1: Distributed Classification via Logistic Regression

Suppose to have N agents that want to cooperatively determine a nonlinear classifier for a set of points in a given feature space. A two-dimensional example is shown in Figure 1.

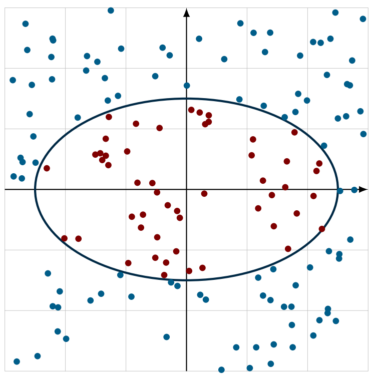

Figure 1: Example of points that cannot be separated by a linear classifier.

Task 1.1 – Distributed Optimization

1. As a preliminary task, implement the *Gradient Tracking* algorithm to solve a consensus optimization problem in the form

$$\min_z \sum_{i=1}^{N} \ell_i(z)$$

with $z \in \mathbb{R}^d$ while $\ell_i : \mathbb{R}^d \mapsto \mathbb{R}$ is a quadratic function, for all $i \in \{1, \dots, N\}$. You can extend the code provided during lectures.

2. Run a set of simulations to test the effectiveness of the implementation. Moreover, provide a set of solutions that includes different weighted graph patterns (e.g., cycle, path, star) whose weights are determined by the Metropolis-Hastings method. Finally, for each simulation, plot the evolution of the cost function and of the norm of the gradient of the cost function across the iterations.

Task 1.2 – Centralized Classification

1. Generate a dataset of $\mathcal{M} \in \mathbb{N}$ points $\mathcal{D}^m \in \mathbb{R}^d$, $m = 1, \dots, \mathcal{M}$.

2. Label the points with a binary label $p^m \in \{-1, 1\}$ for all $m = 1, \dots, \mathcal{M}$. The separating function is given in the form

$$\{x \in \mathbb{R}^d \mid w^\top \varphi(x) + b = 0\},$$

with $\varphi : \mathbb{R}^d \to \mathbb{R}^q$ a nonlinear function and $w \in \mathbb{R}^q,\ b \in \mathbb{R}$ parameters. Hence,

$$\begin{aligned} w^\top \varphi(\mathcal{D}^m) + b \geq 0, &\quad \text{if } p^m = 1, \text{ and} \\ w^\top \varphi(\mathcal{D}^m) + b < 0, &\quad \text{if } p^m = -1. \end{aligned}$$

3. Implement a (centralized) Gradient Method to minimize a Logistic Regression Function in order to classify the points. The resulting optimization problem is

$$\min_{w,b} \sum_{m=1}^{\mathcal{M}} \log\Big(1 + \exp(-p^m(w^\top \varphi(\mathcal{D}^m) + b))\Big)$$

   where $(w, b) \in \mathbb{R}^q \times \mathbb{R}$ is the optimization variable.

4. Run a set of simulations to test the effectiveness of the implementation. Moreover, provide a set of solutions that includes, e.g., different dataset patterns. Finally, for each simulation, plot the evolution of the cost function and of the norm of the gradient of the cost function across the iterations.

Hint. In a two-dimensional example, assuming $\mathcal{D} = ([\mathcal{D}]_1, [\mathcal{D}]_2) \in \mathbb{R}^2$, the separating function can be an ellipse. To this end, define

$$\varphi(\mathcal{D}) := \begin{bmatrix} [\mathcal{D}]_1 \\ [\mathcal{D}]_2 \\ ([\mathcal{D}]_1)^2 \\ ([\mathcal{D}]_2)^2 \end{bmatrix}$$

Task 1.3 – Distributed Classification

1. Split (randomly) the dataset in N subsets, one for each agent $i \in \{1, \dots, N\}$.

2. Implement the *Gradient Tracking* algorithm to classify the dataset in a distributed fashion (you can extend the code designed in Task 1.1).

3. Generate a set of simulations testing different dataset sizes and patterns, showing the convergence of the distributed algorithm to a stationary point of the optimization problem. Moreover, plot the evolution of the cost function and of the norm of the gradient of the cost function across the iterations.

4. Evaluate the quality of the obtained solution by computing the percentage of misclassified points.

Hint: you are allowed to use the files provided during the exercise lectures as a template for the implementation.

Task 2: Aggregative Optimization for Multi-Robot Systems

Consider a team of N robots in a two-dimensional environment. Denote the position of robot $i \in \{1, \dots, N\}$ at iteration $k \in \mathbb{N}$ with $z_i^k \in \mathbb{R}^2$ and denote with $z^k \in \mathbb{R}^{2N}$ the stack vector of the locations of the whole team.

Task 2.1 – Problem Set-up

The goal of this task is to implement a distributed control algorithm that allows the robots to keep the formation tight, and, at the same time, be close to some private targets. An illustrative example is shown in Figure 2.

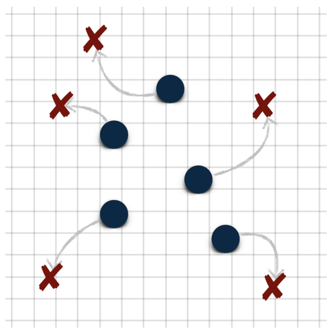

Figure 2: Blue circle represent the team of robots. Red crosses represent targets. Each robot want to move towards a certain target, keeping the formation tight.

This scenario can be formalized as the following aggregative optimization problem

$$\min_{z \in \mathbb{R}^d} \sum_{i=1}^{N} \ell_i(z_i, \sigma(z)),$$
$$\sigma(z) \triangleq \frac{\sum_{i=1}^{N} \phi_i(z_i)}{N}.$$

where $\ell_i : \mathbb{R}^2 \times \mathbb{R}^d \to \mathbb{R}$ and $\phi_i : \mathbb{R}^2 \to \mathbb{R}^d$. In this scenario, agent i is aware only of the decision variable z_i and of the functions ℓ_i and ϕ_i. The global aggregative variable $\sigma(z)$ can be seen as the barycenter of the team, and, hence, $\phi_i(z_i) = z_i$ for all $i = 1, \dots, N$.

1. Implement on a Python script, a version of the *Aggregative Tracking* algorithm designing the most suitable cost functions ℓ_i, which allows the robots to
   (a) keep the formation tight;
   (b) move towards targets.
2. Run a set of simulations showing how the team adapts its behavior by choosing
   - diverse tuning parameters of the cost function;
   - different target locations;
   - (optional) moving target locations.

   Moreover, plot the evolution of the cost function and of the norm of the gradient of the cost function across the iterations.
3. Provide an animated visualization of the team behavior (you can use templates provided during lectures).

Task 2.2 – ROS 2 Implementation

1. Create a ROS 2 package written in Python implementing the *Aggregative Tracking* algorithm previously designed.

2. Run the same simulation of Task 2.1 (here, you can also use the RVIZ).

Task 2.3 – Moving inside a corridor

1. Design and implement a new cost function that also includes a term that allows the team to enter in a corridor without colliding with the walls. An illustrative example is shown in Figure 3.

2. Run a set of experiments in ROS 2 showing the effectiveness of the new cost function. Moreover, plot the evolution of the cost function and of the norm of the gradient of the cost function across the iterations.

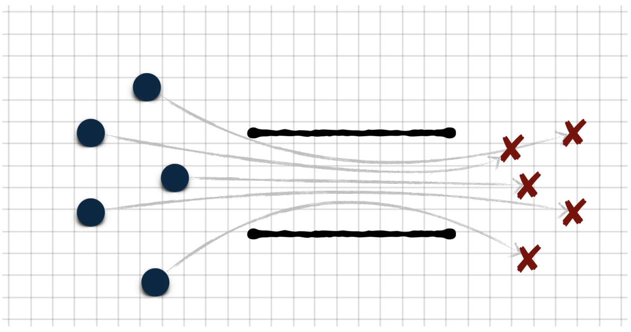

Figure 3: The team reaches the targets while keeping the formation tight.

Notes

1. Each group must be composed of at most 3 students.
2. Any other information and material necessary for the project development will be given during project meetings.
3. Each group must attend 3 meetings with the tutor.
4. The project report must be written in LaTeX and must follow the main structure of the provided template.
5. Any email for project support must have the subject:
   "[DAS2024] Group X: *support request*".
6. All the emails exchanged **must be cc-ed** to prof. Notarstefano, prof. Notarnicola, dr. Pichierri, and all the other group members (if any).

IMPORTANT: Instructions for the Final Submission

1. The final submission **deadline** is **one** week before the exam date.
2. One member of the group must send an email with subject
   "[DAS2024] Group X: Submission"
3. The email **must** include a link to a **OneDrive** folder, shared with prof. Notarstefano, prof. Notarnicola, dr. Pichierri, and all the other group members (if any).
4. The final submission folder must contain:

   - `README.txt`
   - `report_group_XX.pdf`
   - `report` – a folder containing the LaTeX code and a `figs` folder (if any)
   - `task_1` – a folder containing the code relative to Task 1, including `README.txt`
   - `task_2` – a folder containing the code relative to Task 2, including `README.txt`