UNIVERSITÀ DI BOLOGNA

School of Engineering

Master Degree in Automation Engineering

Distributed Autonomous Systems

TITLE

Professors:
Giuseppe Notarstefano
Ivano Notarnicola

Students:
Valerio Costa
Tian Cheng Xia

Academic year 2024/2025

Abstract

Contents

Introduction

Motivations

Contributions

Chapter 1

Multi-Robot Target Localization

1.1 Gradient tracking with quadratic functions

The first part of the task consists of implementing the gradient tracking algorithm generalized in $\mathbb{R}^d$ and then experiment with the implementation using quadratic functions, which we define in the usual way as:

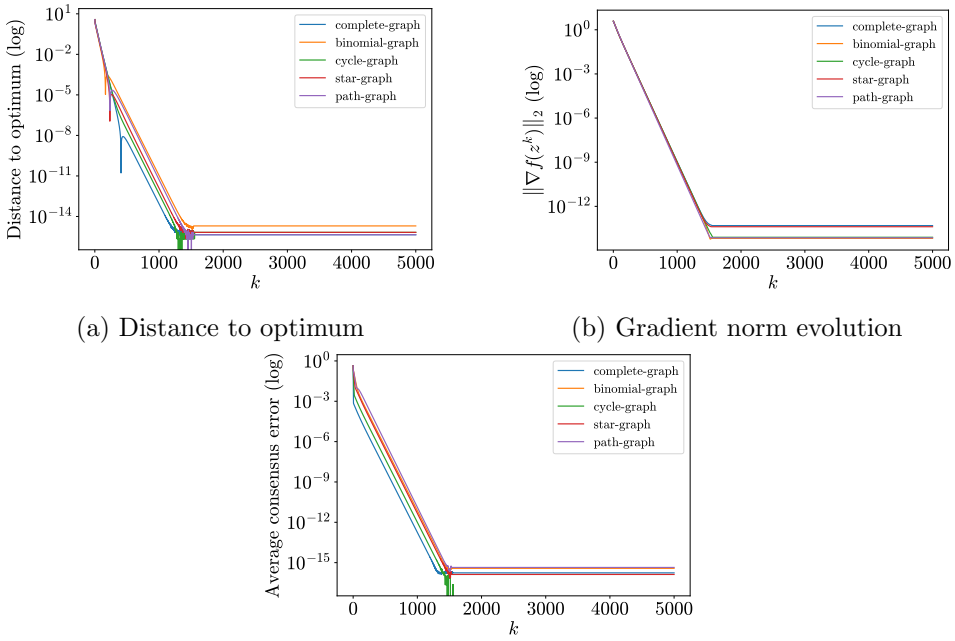
$$f(\boldsymbol{z}) = \frac{1}{2}\boldsymbol{z}^T\boldsymbol{Q}\boldsymbol{z} + \boldsymbol{r}^T\boldsymbol{z} \quad \nabla f(\boldsymbol{z}) = \boldsymbol{Q}\boldsymbol{z} + \boldsymbol{r}$$

where $\boldsymbol{z} \in \mathbb{R}^d$, $\boldsymbol{Q} \in \mathbb{R}^{d\times d}$ is positive definite, and $\boldsymbol{r} \in \mathbb{R}^d$.

We analyzed the behavior with quadratic functions through the definition of different problems with different kinds of graph patterns (in particular complete, binomial, cycle, star, and path graph). The configurations we tested are the following:

- A small problem (5 agents in $\mathbb{R}^3$),
- A problem with higher dimensionality (5 agents in $\mathbb{R}^{15}$),
- A problem with many agents (15 agents in $\mathbb{R}^3$), and
- A problem with many agents in higher dimensionality (15 agents in $\mathbb{R}^{15}$).

In addition, we performed a comparison between the distributed gradient tracking algorithm and the centralized one. For compactness in the discussion, in the rest of this report we show the results with a single initialization seed and, if not specified, it indicates that the results are consistent across different initializations. Also, for readability, for quadratic functions we report the distance to the optimum in semi-logarithmic scale instead of the cost itself which can be negative.



(a) Distance to optimum

(b) Gradient norm evolution

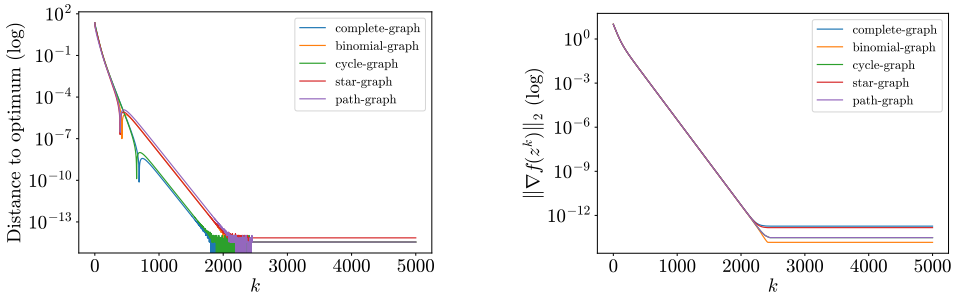
(c) Consensus error

Figure 1.1: Quadratic function minimization with 5 agents in $\mathbb{R}^3$

1.1.1 Comparison between different graph patterns

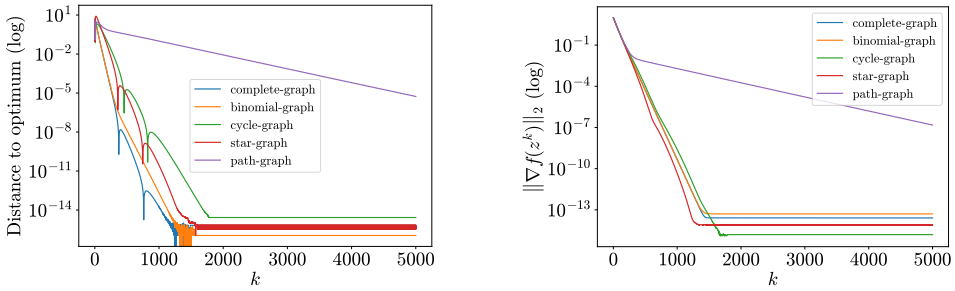
For the starting small problem, we can observe from Figure 1.1 a relatively smooth improvement of the cost function and an exponentially decreasing gradient in all cases. Moreover, a result that can be expected and is consistently persistent in all the other experiments is that consensus is reached slightly faster with a complete graph. Next, by experimenting with higher dimensionality, we can observe from Figure 1.2 that the behavior of both the cost and its gradient are very similar to the previous case with the only difference that more iterations are required to reach full convergence, indicating that the dimensionality is marginal in changing the difficulty of the problem.

In the case of many agents with lower dimensionality, we can observe from Figure 1.3 that the cost function does not reach the optimum within the given number of iterations with the configuration using the path graph, indicating that connectivity is important for larger numbers of agents. This can be explained by the fact that, by adding more agents, the overall problem includes more local losses and becomes more difficult to solve in a distributed way. From Figure 1.4, we observe the same convergence behavior as in the previous case and also confirm that, with higher dimensionality, the problem is not significantly affected.



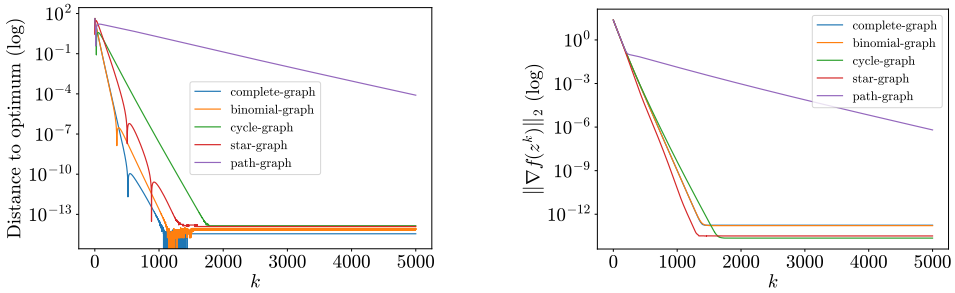
(a) Cost evolution (b) Gradient norm evolution

Figure 1.2: Quadratic function minimization with 5 agents in $\mathbb{R}^{15}$



(a) Cost evolution (b) Gradient norm evolution

Figure 1.3: Quadratic function minimization with 15 agents in $\mathbb{R}^{3}$

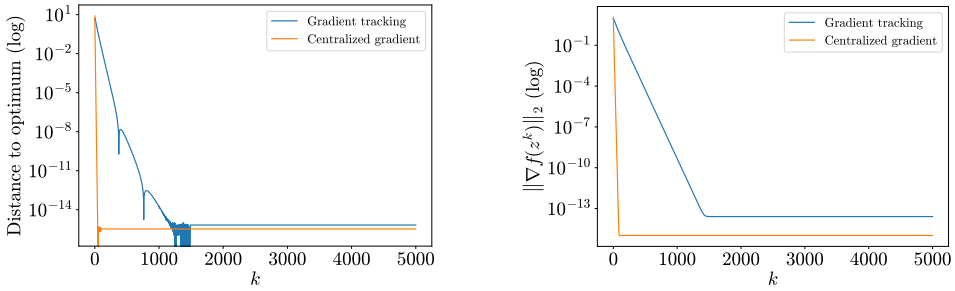


(a) Cost evolution (b) Gradient norm evolution

Figure 1.4: Quadratic function minimization with 15 agents in $\mathbb{R}^{15}$

1.1.2 Comparison with centralized gradient

Following the previous results, we select the configuration using the complete graph for the comparison with the centralized gradient method. In Figure 1.5, we can observe the results with 15 agents, but the overall behavior is the same for all configurations. It can be seen that, as one could expect, the centralized gradient method is faster to converge compared to a



(a) Distance to optimum

(b) Gradient norm evolution

Figure 1.5: Quadratic function minimization with 15 agents in $\mathbb{R}^3$ compared to centralized gradient

distributed algorithm as it has available all the global information and does not rely on estimates and information exchange with the neighbors.

1.2 Cooperative multi-robot target localization

The second part of the task involves applying the gradient tracking algorithm for estimating the position of N_T fixed targets in a distributed way through N_R tracking robots. Each robot is located at position $\boldsymbol{p}_i \in \mathbb{R}^2$ and it is assumed that the distance measured from each robot is noisy. Given the positions $\boldsymbol{p}_i$ and $\boldsymbol{p}_\tau$ of the i-th robot and the τ-th target, respectively, we model the measured noisy distance $d_{i,\tau}$ as follows:

$$d_{i,\tau} = \|\boldsymbol{p}_i - \boldsymbol{p}_\tau\| + \varepsilon \cdot \texttt{noise}$$

where `noise` is drawn from some distribution and ε is the noise rate.

The local loss each robot i uses is the following:

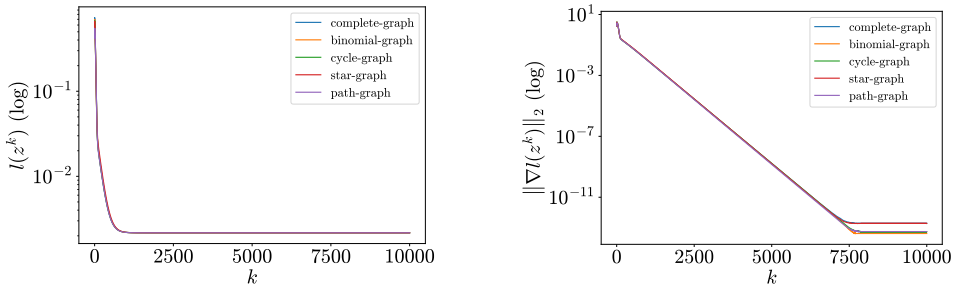
$$l_i(\boldsymbol{z}) = \sum_{\tau=1}^{N_T} \left(d_{i,\tau}^2 - \|\boldsymbol{z}_\tau - \boldsymbol{p}_i\|^2\right)^2 \qquad \nabla l_i(\boldsymbol{z}) = (\nabla l_{i,1}(\boldsymbol{z}_1), \ldots, \nabla l_{i,N_T}(\boldsymbol{z}_{N_T}))$$

$$\nabla l_{i,j}(\boldsymbol{z}_j) = -4\left(d_{i,j}^2 - \|\boldsymbol{z}_j - \boldsymbol{p}_i\|^2\right)(\boldsymbol{z}_j - \boldsymbol{p}_i)$$

where $\boldsymbol{z} = (\boldsymbol{z}_{\tau_1}, \ldots, \boldsymbol{z}_{\tau_{N_T}}) \in \mathbb{R}^{2N_T}$ is the stack of decision variables of robot i containing the estimated positions of the targets $\boldsymbol{z}_\tau \in \mathbb{R}^2$ and $\nabla l_i(\boldsymbol{z}) \in \mathbb{R}^{2N_T}$ is the concatenation of the gradients computed with respect to each target.

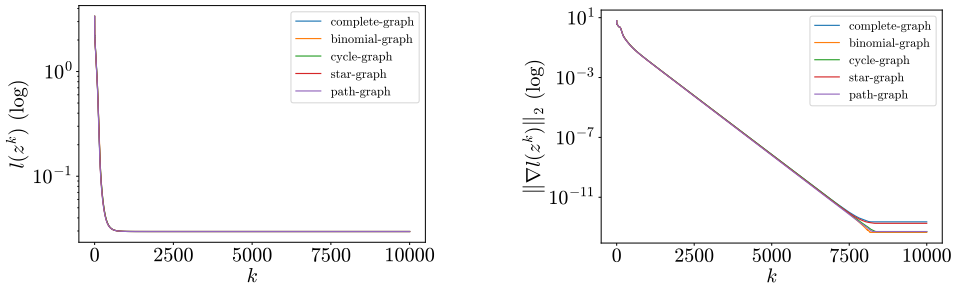
We approach the experimentation of such algorithm by trying different graph patterns and comparing with the centralized gradient method, similarly to the previous case. At first, we evaluated the performance of the algorithm in the following cases, all with the same type of noise:

- Network of 5 robots and 1 target,



(a) Loss evolution (b) Gradient norm evolution

Figure 1.6: Tracking with 5 robots and 1 target



(a) Loss evolution (b) Gradient norm evolution

Figure 1.7: Tracking with 5 robots and 3 targets

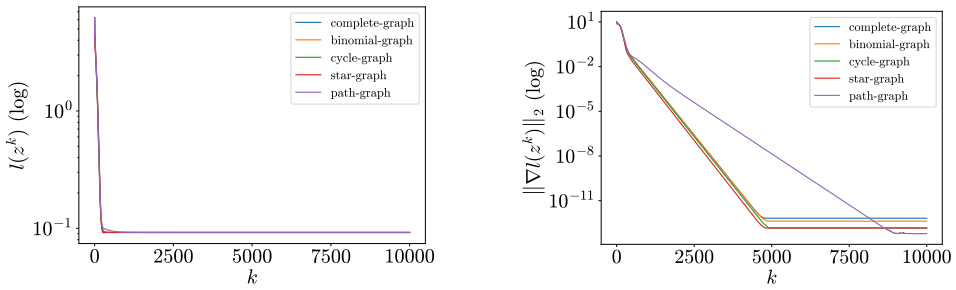
- Network of 5 robots and 3 targets, and
- Network of 15 robots and 3 targets.

Then, the focus switched to observe how much the performance changes in terms of noise. By fixing the problem configuration, we experimented with varying Gaussian noises, Poisson noises, and noise rates.

1.2.1 Comparison between different graph patterns

In terms of graph pattern, we can observe from Figure 1.6 and Figure 1.7 that with the same number of robots and increasing number of targets, the number of iterations required to converge is roughly the same. This makes sense as each target is independent to the others and can be tracked in parallel.

Instead, by increasing the number of tracking robots, we can see from Figure 1.8 that the plateau is reached in fewer number of iteration, which intuitively means that more tracking robots help in reaching a faster convergence.



(a) Loss evolution (b) Gradient norm evolution

Figure 1.8: Tracking with 15 robots and 3 targets

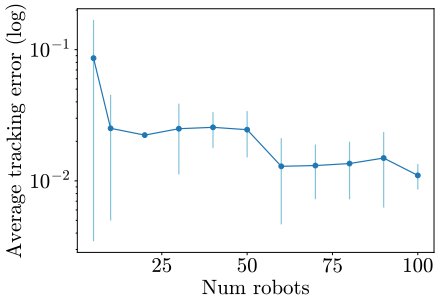


Figure 1.9: Average tracking error for different number of robots (average of 5 runs). Vertical bars represent the standard deviation.

Moreover, as the total loss is a summation, we must note that the overall loss is higher in the case of more agents or targets, but this does not indicate worse tracking results. We report in Figure 1.9 the average distance between the estimated and real target positions for a fixed configuration with varying number of robots. It can be seen, as intuition would suggest, that on average the tracking error becomes smaller by increasing the number of tracking robots.

Finally, an observation consistent in all experiments is that, as shown in Figure 1.10, the overall behavior of this system is to reach an approximate consensus (i.e., consensus error around 10^{-4}) in the first few iterations and then optimize the loss while improving and preserving consensus. This can also be observed in Figure 1.11 where a few frames of the animation of the scenario are shown.

1.2.2 Comparison with centralized gradient

Compared with the centralized gradient algorithm, the plots in Figure 1.12 confirm what we observed before in the case of quadratic functions. The convergence speed is faster and more accurate in a centralized approach,

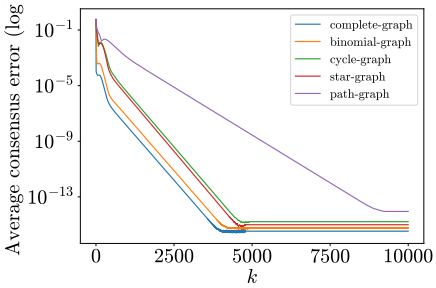


Figure 1.10: Tracking with 15 robots and 3 targets

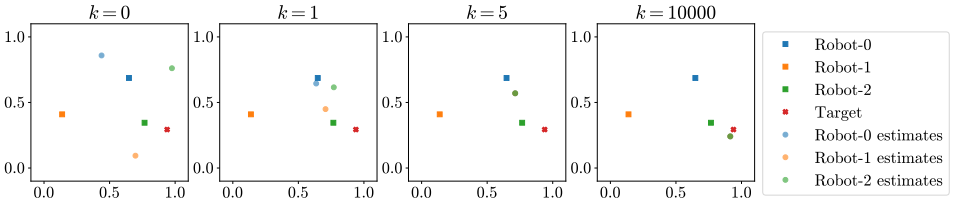
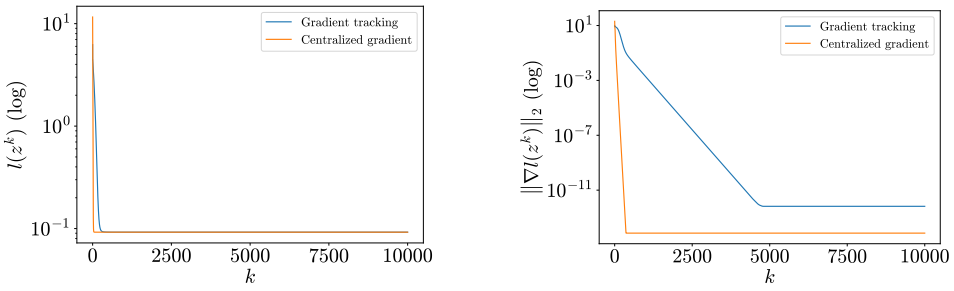


Figure 1.11: Tracking animation with 3 robots and 1 target



(a) Loss evolution

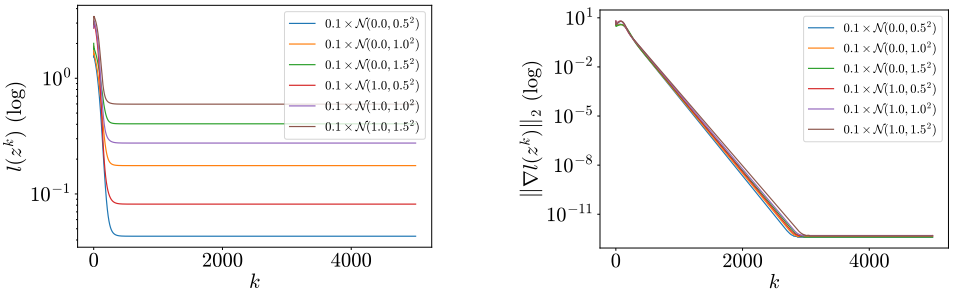
(b) Gradient norm evolution

Figure 1.12: Tracking with 15 robots and 3 targets with centralized gradient

which also results in a lower average tracking error.

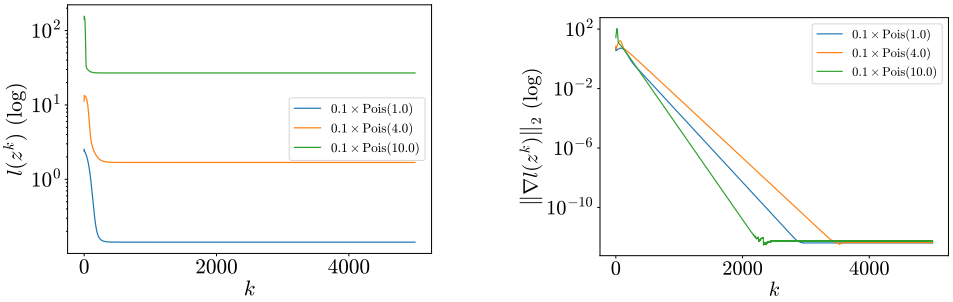
1.2.3 Different noises

From the experiments with different noises, we observed a worsening in performance that is proportional to the amount of noise injected into the distance measurement, implying as one could expect that more noise leads to worse results. This behavior is consistent with noise drawn from different distributions as in Figure 1.13 and Figure 1.14, and also when the noise rate is increased as in Figure 1.15.



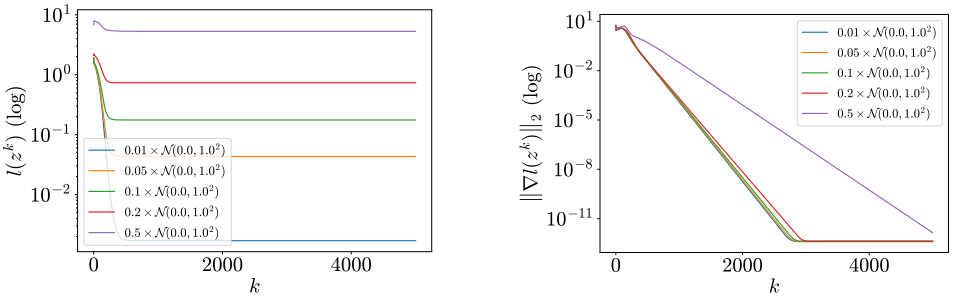
(a) Loss evolution (b) Gradient norm evolution

Figure 1.13: Tracking with 15 robots and 3 targets with noise drawn from Gaussian distributions



(a) Loss evolution (b) Gradient norm evolution

Figure 1.14: Tracking with 15 robots and 3 targets with noise drawn from Poisson distributions



(a) Loss evolution (b) Gradient norm evolution

Figure 1.15: Tracking with 15 robots and 3 targets with different rates of Gaussian noise

Chapter 2

Aggregative Optimization for Multi-Robot Systems

2.1 Problem formulation

Given N agents with positions $\boldsymbol{z}_i$ and each associated to a private target $\boldsymbol{r}_i$, we want to solve the problem of positioning the agents close to its own target while staying close to the rest of the agents.

We solve this optimization problem using the following loss:

$$\begin{aligned} l_i(\boldsymbol{z}_i, \sigma(\boldsymbol{z})) &= \gamma_1 \frac{1}{2}\|\boldsymbol{z}_i - \boldsymbol{r}_i\|^2 + \gamma_2 \frac{1}{2}\|\boldsymbol{z}_i - \sigma(\boldsymbol{z})\|^2 \\ \nabla_1 l_i(\boldsymbol{z}_i, \sigma(\boldsymbol{z})) &= \gamma_1(\boldsymbol{z}_i - \boldsymbol{r}_i) + \gamma_2(\boldsymbol{z}_i - \sigma(\boldsymbol{z})) \\ \nabla_2 l_i(\boldsymbol{z}_i, \sigma(\boldsymbol{z})) &= -\gamma_2(\boldsymbol{z}_i - \sigma(\boldsymbol{z})) \end{aligned}$$

where γ_1 weighs the importance of being close to the targets and γ_2 weighs the importance of being tight to the fleet.

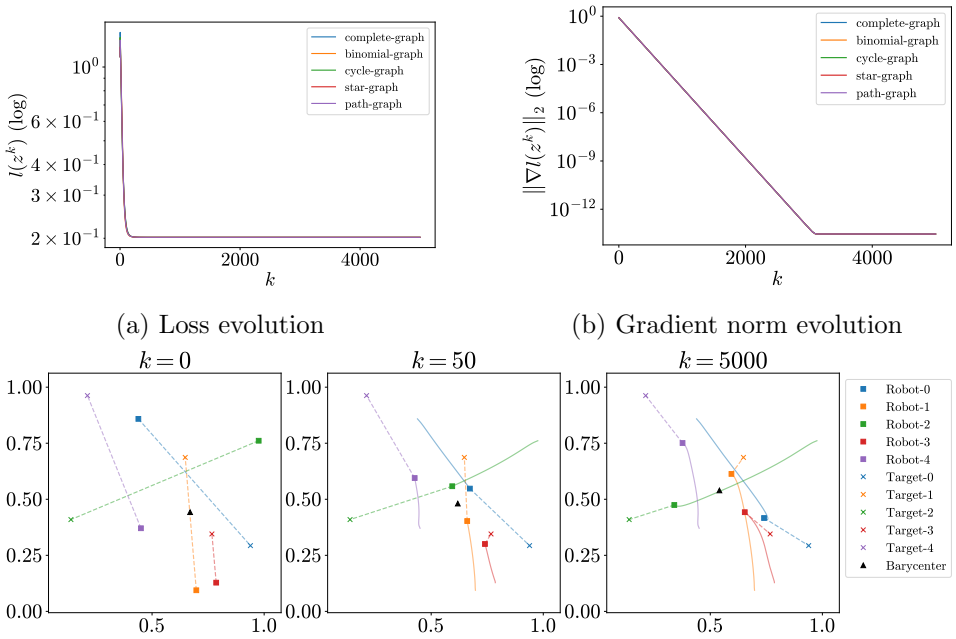
In order to compute the global aggregation function $\sigma(\boldsymbol{z})$, each agent contributes to the barycenter as the expression of the linear function ϕ_i:

$$\phi_i(\boldsymbol{z}_i) = \alpha_i \boldsymbol{z}_i \qquad \sigma(\boldsymbol{z}) = \frac{1}{N}\sum_i^N \phi_i(\boldsymbol{z}_i)$$

With $\alpha_i = 1$ for all i, we obtain the standard formulation of the problem in which $\sigma(\boldsymbol{z})$ represents the barycenter of the fleet. With different α_i (with $\sum_i^N \alpha_i = N$), we obtain a $\sigma(\boldsymbol{z})$ that is a weighted average and can be interpreted as a barycenter that is biased towards specified agents.

2.2 Comparison with different graph patterns

As in the previous task, we first experiment this setup with different graph patterns and different number of agents.



(a) Loss evolution

(b) Gradient norm evolution

(c) Animation frames. Dashed lines connect robots to the private targets and solid lines are the trajectories.

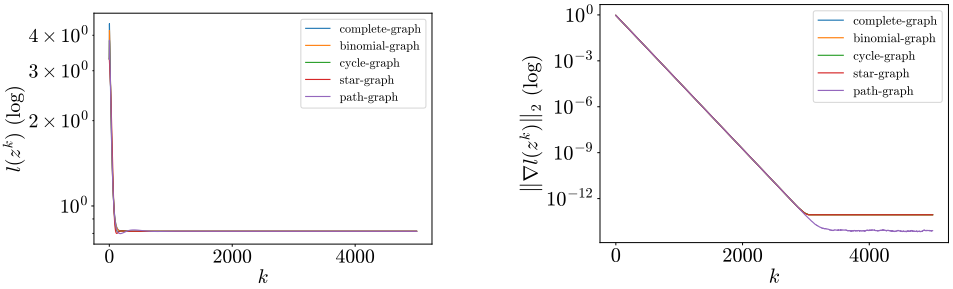
Figure 2.1: Positioning with 5 robots

From the plots in Figure 2.1, we can observe an identical trend in terms of loss and gradient for every graph pattern we have experimented with. In all cases, we can observe that the algorithm converges, and also, visually, we can see that the final positions of the agents tends to reach an expected behavior.

By adding more agents, we cannot detect any relevant changes in behavior as they all reach convergence in the same way as in the previous experiment. The only thing we can underline is that the trend for the path graph has a little variation at convergence, most likely due to numerical instability.

2.3 Comparison with different loss configurations

In terms of different loss functions, we experiment our formulation with different weights to prioritize target vicinity (higher γ_1), barycenter vicinity (higher γ_2), and different agents' importance (different α_i). Results are presented in Figure 2.3, Figure 2.4, and Figure 2.5, respectively. In all cases the positions at convergence are intuitively the expected ones, showing that the algorithm allows many degrees of freedom.



(a) Loss evolution (b) Gradient norm evolution

Figure 2.2: Positioning with 15 robots

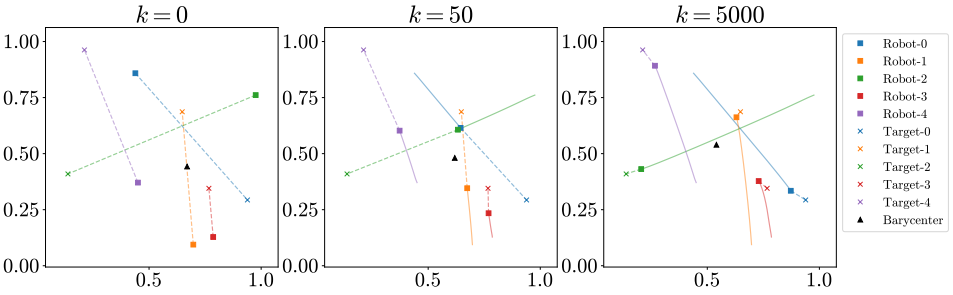


Figure 2.3: Frames with 5 robots and loss that prioritizes the private targets. Dashed lines connect robots to the targets and solid lines are the trajectories.

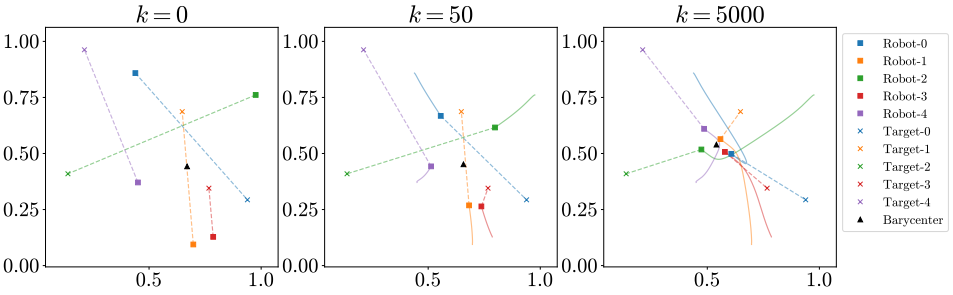


Figure 2.4: Frames with 5 robots and loss that prioritizes the barycenter. Dashed lines connect robots to the targets and solid lines are the trajectories.

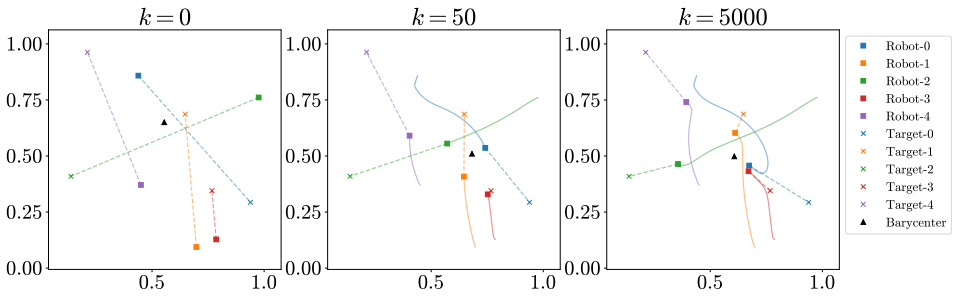


Figure 2.5: Frames with 5 robots and barycenter that prioritizes robot 0. Dashed lines connect robots to the targets and solid lines are the trajectories.

Conclusions

Bibliography