Modeling and Control of Cyber-Physical Systems
Project I

Fabio Veroli
fabio.veroli@studenti.polito.it

Dario Lupo
s336550@studenti.polito.it

Fabio Brugiafreddo
fabio.brugiafreddo@studenti.polito.it

Department of Control and Computer Engineering
Politecnico di Torino

1 Task 1: Secure State Estimation of a Static CPS with Sparse Sensor Attacks

The first objective of this report is to implement and compare the **Iterative Soft-Thresholding Algorithm (ISTA)** and the **Inertial Jacobi Alternating Minimization (IJAM)** to solve the P-LASSO optimization problem for estimating the state of CPS under sparse sensor attacks.

The P-LASSO model is defined as:

$$y = C\widetilde{x} + \widetilde{a} + \eta. \tag{1}$$

where:

- $\widetilde{x} \in \mathcal{R}^n$ is the unknown state vector,
- $\widetilde{a} \in \mathcal{R}^q$ is the unknown sparse attack vector,
- $\eta \in \mathcal{R}^q$ is a possible measurement noise.

The goal is to estimate both the state and the indices of the attacked sensors, using:

- A sensing matrix C (randomly generated from a normal distribution),
- 15 states,
- 30 sensors,
- and 2 randomly selected sensor attacks, where attack values are drawn from $\widetilde{a}_i \in [-5, -4] \cup [4, 5]$.

The state vector components are generated uniformly in the range $\widetilde{x}_j \in [-3, -2] \cup [2, 3]$. The remaining parameters used are:

- Measurement noise $\eta \sim \mathcal{N}(0, \sigma^2), \sigma = 10^{-2}$
- Stopping criterion: first iteration $T_{\max}$ such that $|x(T_{\max} + 1) - x(T_{\max})|_2^2 < \delta$, with $\delta = 10^{-10}$
- Regularization parameter: $\lambda = 0.1$
- Step sizes:
  - ISTA: $\nu = \frac{0.99}{||G||_2^2}$ where $G = (CI)$
  - IJAM: $\nu = 0.7$

To evaluate the accuracy of the estimated state $x(k)$, we compute the state estimation error as:

$$\frac{||x(k) - \widetilde{x}||_2}{||\widetilde{x}||_2} \tag{2}$$

We ran 20 simulations of both ISTA and IJAM. The performance results, averaged over the runs, are shown in Figures 1 and 2. For visualization purposes, a fixed number of iterations $T_{\max}$ was used instead of the stopping criterion, ensuring both algorithms had enough time to converge.

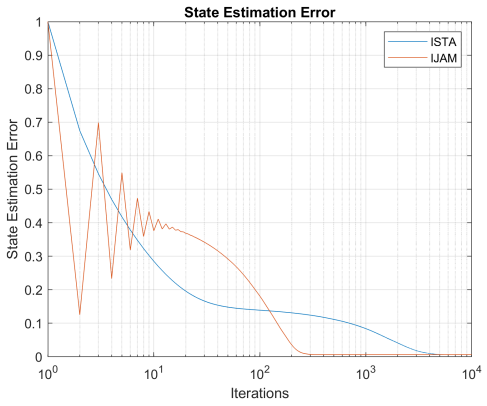

Figure 1: State estimation error

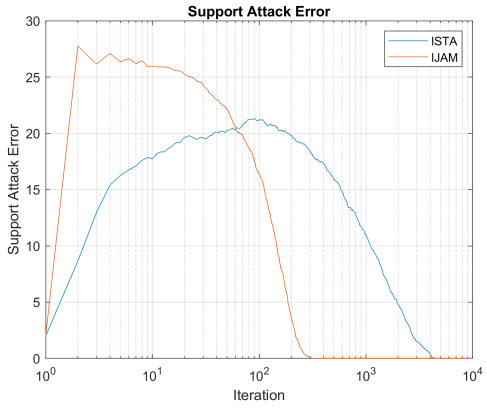

Figure 2: Support attack error

1.1 Recovery Performance Metrics and Convergence Rate

Both, ISTA and IJAM achieve the same asymptotic recovery performance. That is, given sufficient iterations, both algorithms are capable of accurately recovering the state, even in the presence of sparse attacks and measurement noise.

As expected, IJAM converges significantly faster than ISTA:

- IJAM quickly reduces both the state estimation error and the support error within ≈ 100 iterations.
- ISTA exhibits a more gradual convergence, particularly in the initial and mid-phase (from 10 to 1000 iterations), with a slower but smoother descent.

1.2 Variation of λ

To study the effect of the regularization parameter λ, we varied its value across $0.1, 0.2, 0.5, 1, 1.5$ while keeping the step size ν fixed. This allowed us to assess how the sparsity-inducing term affects convergence and estimation quality.

We observed that:

- Increasing λ initially improves convergence, as the soft-thresholding operator more strongly promotes sparsity.
- This suppresses small noise components and enhances support recovery performance.

However, for large values of λ, both the support attack error and the state estimation error worsen (Figures 3 and 4). This happens because excessive regularization shrinks not only noise but also legitimate attack components, leading to:

- Missed detection of attacks (false negatives),
- Poor support recovery,

- Degraded state estimates due to misattribution of attack effects to system dynamics.

Thus, choosing λ involves a trade-off between enforcing sparsity and preserving true attack signals.

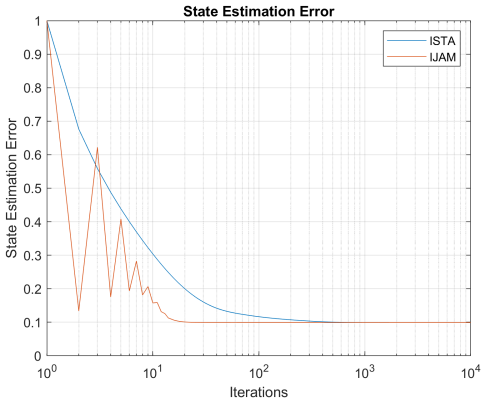

Figure 3: State Estimation Error $\lambda = 1.5$

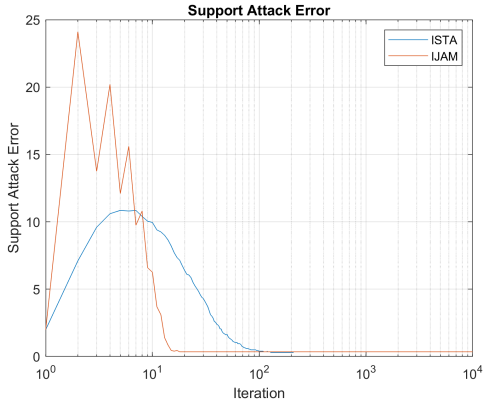

Figure 4: Support Attack Error $\lambda = 1.5$

1.3 Variation of ν

To analyze the impact of the step-size parameter ν, we tested several values in the range $(0, 1]$, keeping λ fixed. The step size ν controls how aggressively the algorithm updates its estimates at each iteration.

- In ISTA, ν influences both the state and attack updates through the gradient step. A larger ν typically speeds up convergence, but values that are too large can cause instability or oscillation.
- In IJAM, since the state is updated via the pseudo-inverse of C, ν only affects the sparse recovery (i.e., the thresholding step for a). Again, higher ν accelerates convergence, but overly large values can result in false positives.

The difference arises because ISTA applies ν globally in the gradient step, whereas IJAM restricts it to the thresholding operation. A good strategy is to begin with $\nu \in [0.5, 0.7]$ and adjust based on the speed-stability trade-off.

1.4 Resilience to Attacks

We evaluated the robustness of ISTA and IJAM by increasing the number of attacked sensors, denoted as h. According to Proposition 1, h attacks are correctable if and only if:

$$\forall z \in R^n, \quad ||Cz||_0 \geq 2h + 1 \tag{3}$$

In our case, the C matrix is full rank (i.e., `rank(C)=n`) so the necessary condition becomes

$$h \leq \frac{q - n}{2} \tag{4}$$

Given the default configuration, this implies $h \leq 7.5$, so we expect that up to 7 attacks can be corrected in our system. We tested both algorithms with h=7 (Figures 5 and 6) and h=12 (Figures 7 and 8). As expected:

- For $h = 7$, the algorithms still recover both the state and the support with acceptable accuracy.

- For $h = 12$, both the state estimation error and support identification error increase significantly.

This confirms that exceeding the correctable threshold leads to algorithm failure, as the system no longer has sufficient redundancy to distinguish attacks from legitimate signals.

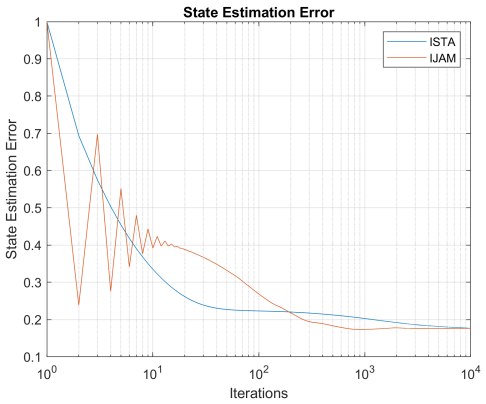

Figure 5: State Estimation Error with $h = 7$

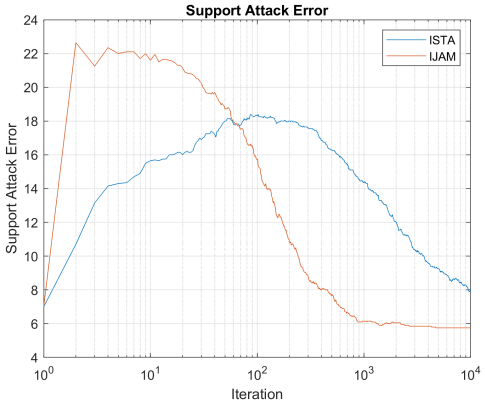

Figure 6: Support Attack Error with $h = 7$

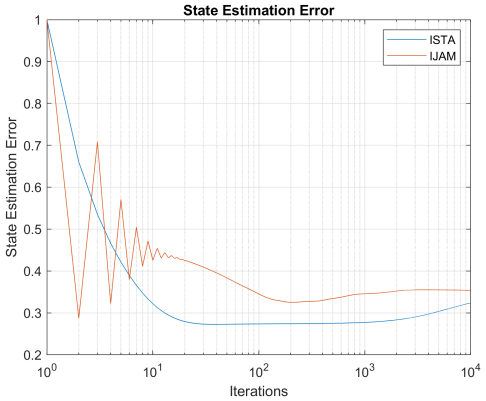

Figure 7: State estimation error with $h = 12$

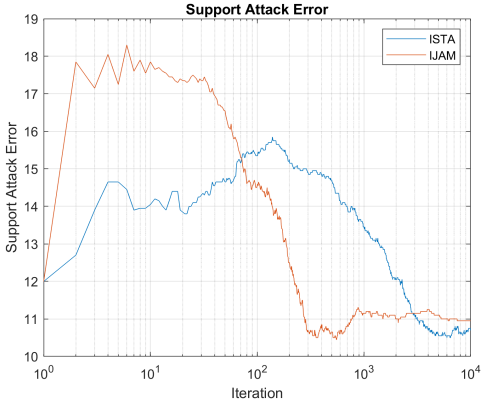

Figure 8: Support attack error with $h = 12$

2 Task 2: Target Localization Under Sparse Sensor Attacks

This experiment addresses the problem of indoor localization using a Received Signal Strength (RSS) fingerprinting approach, in the presence of adversarial sensor tampering. The objectives are:

- To estimate the location of a single target within a predefined indoor area;
- To detect sensors that have been compromised by adversarial interference.

The localization environment is a $100m^2$ indoor area, discretized into $n = 100$ cells. A sensor network composed of $q = 20$ sensors is randomly deployed throughout the environment. During the run-time phase, the system collects RSS measurements, which may be corrupted by adversarial attacks on some sensors. The dataset used in the experiment includes the dictionary matrix D and the measurement vector y, both of which are known.

To jointly estimate the target location and identify the attacked sensors, we use a sparse signal recovery approach based on the Iterative Soft Thresholding Algorithm (ISTA). Specifically, we solve a weighted Lasso optimization problem. The measurement matrix is defined as:

$$G = \text{normalize}\begin{pmatrix} D & I \end{pmatrix} \tag{5}$$

It is normalized so that each column has zero mean and unit variance, ensuring numerical stability and proper scaling.

We solve the following problem:

$$\min_{x \in \mathcal{R}^n, a \in \mathcal{R}^n} ||G \begin{pmatrix} x \\ a \end{pmatrix} - y||_2^2 + \lambda_1 ||x||_1 + \lambda_2 ||a||_1 \tag{6}$$

The parameters used are $\lambda_1 = \lambda_2 = \lambda = 10$ and the ISTA step size is $\nu = ||G||_2^{-2}$.

We implemented the ISTA algorithm with 10,000 iterations. The estimated target position corresponds to the maximum value in the recovered position vector x. The results are:

- Estimated target position: cell 37
- Detected attacked sensors: 1, 10, 14, 16, 17

2.1 Analysis of Results for Different Regularization Values

The experiment was repeated for different combinations of λ_1 and λ_2 to verify how they influence target localization and identification of sensors under attack. Results are summarized in Table 1.

2.1.1 Conclusion

From the analysis of the table, it can be observed that:

- **Effect of λ_1:** As λ_1 increases, the solution of the state vector x becomes more accurate. For $\lambda_1 = 1$ and $\lambda_2 \geq 5$, the solution tends to become excessively sparse to the point of being completely zero.
- **Effect of λ_2:** As λ_2 increases, the number of sensors identified as attacked decreases. For very low λ_2 (e.g., 1), almost all sensors are incorrectly classified as attacked (false positives).

It's notable that for a similar value of λ_1 and λ_2 the results are satisfactory, while for non-uniform values we obtain non-reliable data. The best trade-off between localization accuracy, correct attack detection, and sparsity is observed for $\lambda_1 = 20$ and $\lambda_2 = 15$, where the position is accurate, the attack set is correct, and sparsity is minimal.

Table 1: Results of ISTA algorithm for different regularization values

λ_1	λ_2	Target Pos.	Attacked Sensors	Sparsity x
1	1	37	1, 3, 4, 7, 10, 14, 15, 16, 17	9
1	5	24	16, 17	32
1	10	24	–	35
1	15	24	–	35
1	20	24	–	35
5	5	37	1, 10, 14, 16, 17	3
5	10	37	1, 10, 14, 16, 17	5
5	15	56	1, 8, 10, 16, 17	14
5	20	24	16	18
10	5	37	1, 6, 10, 12, 14, 16, 17, 20	1
10	10	37	1, 10, 14, 16, 17	3
10	15	37	1, 10, 14, 16, 17	4
10	20	37	1, 10, 14, 16, 17	6
15	10	37	1, 10, 12, 14, 16, 17	2
15	15	37	1, 10, 14, 16, 17	3
15	20	37	1, 10, 14, 16, 17	3
20	10	37	1, 6, 10, 12, 14, 16, 17, 20	1
20	15	37	1, 10, 14, 16, 17	2
20	20	37	1, 10, 14, 16, 17	3

3 Task 3: Secure State Estimation of a Dynamic CPS with Sparse Sensor Attacks

This task focuses on secure state estimation and attack detection in a dynamic CPS, where sensor measurements are affected by sparse and constant attacks. The system evolves according to the following discrete-time linear model:

$$x(k+1) = Ax(k)$$
$$y(k) = Cx(k) + a$$

where $x(k) \in \mathbb{R}^n$ is the system state at time step k, $y(k) \in \mathbb{R}^q$ is the measurement vector, and $a \in \mathbb{R}^q$ is a sparse attack vector corrupting a subset of the sensor readings. The specific parameters of the experimental setup are:

- Number of sensors: $q = 30$
- Number of sensors under attack: $h = 3$
- State dimension: $n = 15$

According to theoretical results on the *Observability of Dynamic CPSs with Constant Attacks*, such systems are not observable with traditional observers (like the Luenberger observer), even if the attacks are known to be constant. This is because the attack vector, although fixed, introduces an unobservable component that breaks observability unless extra structure (like sparsity) is exploited.

To address this, we use two estimation algorithms that incorporate prior knowledge of the sparsity of the attack vector a:

- **Sparse Soft Observer (SSO)**: an iterative gradient-based method inspired by ISTA, which alternates between state and attack estimation using soft-thresholding.
- **Deadbeat Sparse Soft Observer (D-SSO)**: an observer inspired by IJAM, which uses a deadbeat-like update for the state estimate, while refining the attack estimate similarly.

The following hyper-parameters are used throughout the experiment:

- Regularization parameter: $\lambda = 0.1$
- Step size for SSO: $\nu = \frac{0.99}{\|G\|_2^2}$, where $G = (C \quad I)$
- Step size for D-SSO: $\nu = 0.7$

We use the following performance metrics to compare them:

- State Estimation Error: $\frac{\|\hat{x}(k) - x(k)\|_2}{\|x(k)\|_2}$
- Support Attack Error: $\sum_j ||1(a_j \neq 0) - 1(\hat{a}_j(k) \neq 0)||$, where $1(v) = 1$ if v is true and 0 otherwise.

As shown in Figures 9 and 10, both algorithms are able to identify attacks in finite time. As expected from their counterparts (ISTA and IJAM), both observer algorithms converge to the correct state estimate and identify the support of the attack vector. While D-SSO converges more quickly, SSO, which follows a smoother trajectory, achieves a lower state estimation error and more accurate recovery of the state.

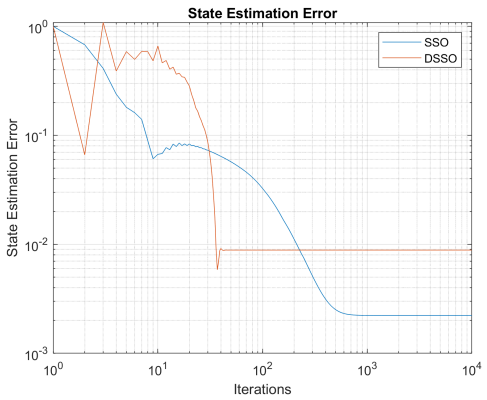

Figure 9: State Estimation Error in SSO and D-SSO

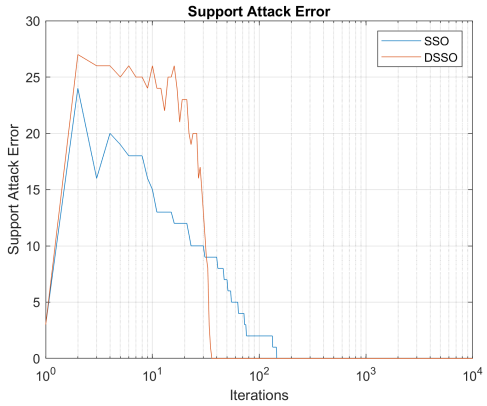

Figure 10: Support Attack Error in SSO and D-SSO

To further improve state estimation accuracy in the presence of sparse sensor attacks, we implemented a refined strategy based on the online identification of the attack support. Once the observer's attack estimate stabilizes (approximately after 100 iterations in our case), the rows of the measurement matrix C corresponding to the sensors identified as compromised are set to zero. This effectively excludes unreliable measurements from the estimation process. For the D-SSO observer, the observer gain L is recomputed using the modified measurement matrix to maintain proper convergence properties.

This refinement leverages the knowledge of the attack locations: by ignoring the contributions of corrupted sensors, the observer relies solely on trustworthy data, which leads to a significant reduction in the state estimation error, down to machine precision in some cases (as shown in Figures 11 and 12). As a result, the secure observer is able to achieve highly accurate state reconstruction despite the presence of sparse sensor attacks.

4 Task 4: Target Tracking under Sparse Sensor Attacks

In this task, we consider an indoor tracking problem involving a single moving target using an RSS fingerprinting setup. We assume that some sensors are compromised by adversarial attacks, and our goal is to identify these tampered sensors. The system model is described by the following equations:

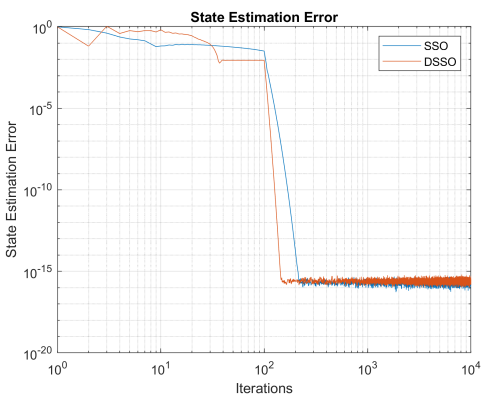

Figure 11: Refined State Estimation Error

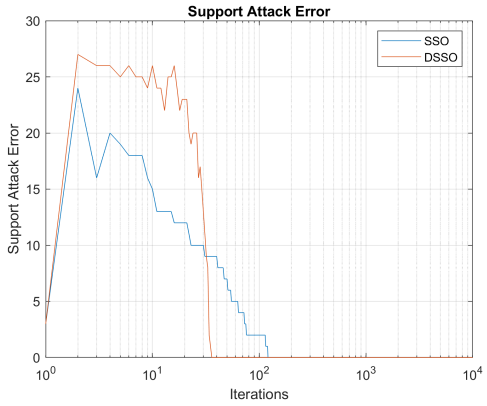

Figure 12: Refined Support Attack Error

$$x(k+1) = Ax(k) \tag{7}$$
$$y(k) = Dx(k) + \tilde{a} + \eta \tag{8}$$

where η is a measurement noise. The matrices A, D, and the output $y(k)$ are assumed to be known, while the attack vector $\tilde{a}$ and the initial state $x(0)$ are given for the final analysis. The environment is discretized into $n = 36$ cells, and a sensor network of $q = 15$ sensors is randomly deployed within the room.

To track the target and detect attacked sensors, we implement the Sparse State Observer (SSO) algorithm. Since the state vector is sparse, we modify the algorithm by applying a soft-thresholding operator also on the state update, promoting sparsity in both state and attack estimates.

Considering the attack-free system described by (A, D), it is observable because the observability matrix

$$O_n = \begin{pmatrix} D \\ DA \\ \dots \\ DA^{n-1} \end{pmatrix}$$

is full rank (rank$(O_n) = n$) according to the Kalman observability criterion. However, the augmented system that includes constant attacks, represented by $\begin{pmatrix} A & 0 \\ 0 & I \end{pmatrix}$ and G is not observable since its observability matrix lacks full rank. This lack of observability implies that a classic Luenberger observer cannot be used directly. Furthermore, one eigenvalue of A is equal to 1, which, according to the observability conditions for dynamic CPS with constant attacks, confirms that the augmented system is not observable.

In this setting, we cannot apply D-SSO to solve the target localization problem under sparse sensor attacks because the number of sensors q is smaller than the number of cells n, i.e., $q < n$. D-SSO is based on the IJAM algorithm, which requires computing the pseudo-inverse of the measurement matrix C. This operation is only valid when C has full column rank, which implies $q \geq n$. However, when $q < n$, the system becomes under-determined and admits infinitely many solutions for the state. As a consequence, D-SSO cannot reliably recover the target's position.

We evaluate the performance of the algorithm using the following metrics:

- Support Attack Error: $\sum_j |1(|\tilde{a_j}| \geq \epsilon) - 1(|\hat{a_j}(k)| \geq \epsilon)|$, where $1(v) = 1$ if v is true, and 0 otherwise, $\epsilon = 1$.
- Support State Error: $\sum_j |1(|\tilde{x_j}| \geq \epsilon) - 1(|\hat{x_j}(k)| \geq \epsilon)|$, where $1(v) = 1$ if v is true, and 0 otherwise, $\epsilon = 1$.

As shown in Figures 13 and 14, the SSO algorithm effectively tracks the target and identifies the sensors under attack. The attack support is fully and stably recovered after an initial convergence period, while the state support is mostly accurate, with only a few isolated estimation errors.

In our implementation, we observed slight discrepancies in the support state and attack errors compared to the values reported in the task assignment. We attribute this to the fact that the soft-thresholding operator was applied on the concatenated vector $[\hat{x}, \hat{a}]$ as a whole, rather than separately on $\hat{x}$ and then on $\hat{a}$.

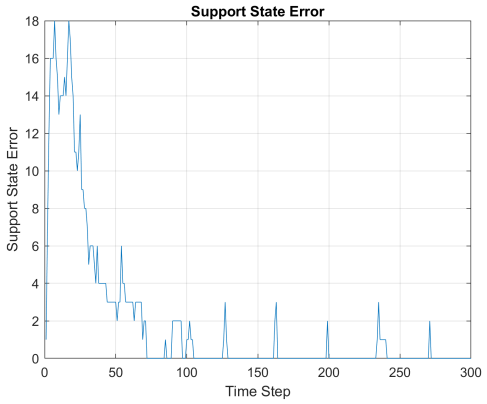

Figure 13: Support State Error

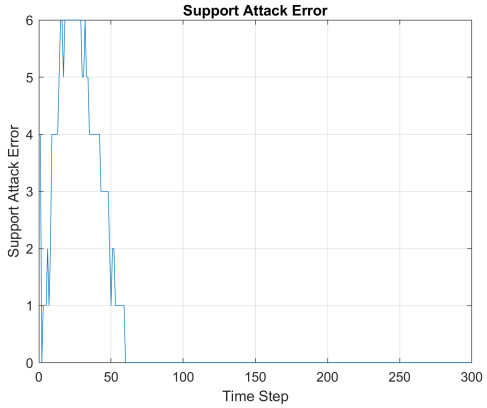

Figure 14: Support Attack Error

5 Task 5: Distributed Target Localization under Sparse Sensor Attacks

In this task, we revisit the target localization problem introduced in Task 2 and solve it in a distributed manner using Distributed ISTA (DISTA). The aim is to localize a single target in the presence of sparse sensor attack. The measurement model is given by:

$$y = D\tilde{x} + \eta + \tilde{a} \in \mathbb{R}^q \tag{9}$$

where $\eta \in \mathbb{R}^q$ is a measurement noise and $\tilde{a} \in \mathbb{R}^q$ is the attack vector. These attacks correspond to physical tampering of sensor measurements during the runtime phase, while the training phase is assumed to be reliable.

Data for y and D are provided. During the training phase, we normalize the matrix G as normalize$(D\ I)$. At runtime, we consider a distributed setting in which each sensor node $i \in 1, \ldots, q$ stores its own local data G_i and measurement $y_i \in \mathbb{R}$, without sharing them with other nodes.

The DISTA algorithm was tested on both star and ring network topologies, illustrated in Figures 15 and 16.

The DISTA algorithm successfully localizes the target position and identifies attacked sensors in both network topologies, achieving results consistent with those obtained by the centralized approach in Task 2.

The two topologies are characterized by doubly stochastic weight matrices. According to the sufficient conditions for average consensus using such matrices, the ring topology guarantees convergence to average consensus, while the star topology does not satisfy these conditions and thus does not guarantee consensus. Despite this, when running the algorithm on the provided data, the system reaches consensus on the target localization and on the support of the attack vector $\tilde{a}$. However, consensus is not reached on the exact estimated values of $\tilde{a}$ and $\tilde{x}$; each agent may maintain different estimates of these variables.

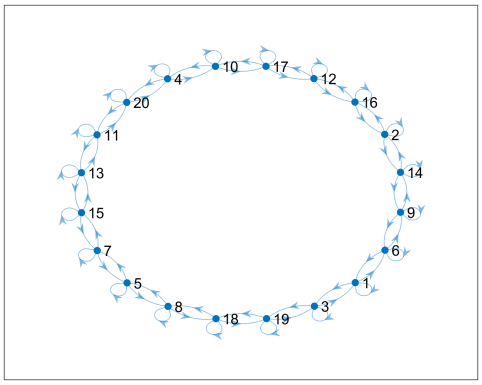

Figure 15: Ring Topology

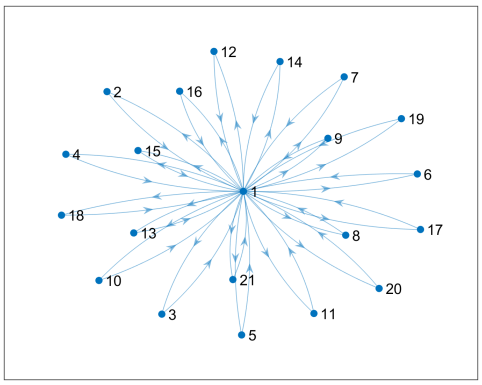

Figure 16: Star Topology

A distributed implementation of IJAM is not feasible because it requires sharing the full D and y information among all nodes, which is prohibited due to privacy and communication constraints. Hence, each node can only access its local data.