Modelling of Cyber-Physical Systems Project Report

Davide Fricano, Davide Sipione, Giacomo Caciagli

Nowadays Cyber-Physical Systems are becoming more and more relevant in everyday world. A CPS is defined as a set of devices with computation capability, which are able to communicate between each other through a network with the physical world via sensors and actuators. This paper is concerned with the state estimation of Cyber-Physical Systems, modeled by Discrete-Time Linear Time Invariant systems, even in the case in which some of the sensors are corrupted by an attacker. In particular, we discuss the application and behaviour of estimation algorithms in real world scenarios, through mathematical simulations run on MATLAB.

1. Implementation of Iterative Shrinkage-Thresholding

Firstly, we analyze the Iterative Shrinkage-Thresholding Algorithm (IST) used to solve the LASSO problem that consists of the optimization of the sum of a Least Square, to find a state which minimizes the difference between the real and estimated output, with an additive term $\lambda \|x\|_1$, that is here generalized as a weighted l_1-norm $\Lambda^\top |x|$ which represents the number of non-zero elements to be minimized in order to reduce disturbances (in reality the l_0-norm got relaxed to l_1 for the convexity and differentiability properties) which minimizer is the best state estimation of the related CPS linear model.

$$\min_{x \in \mathbb{R}^p} \|Cx - y\|_2^2 + \Lambda^\top |x|$$

The IST is basically a variant of Gradient Descend algorithm, with learning rate τ, in which x_t estimation is computed applying a vector-valued function which achieve further corrections with respect to a coefficient $\gamma_i = \tau\lambda_i$ that realizes a shrinkage or a thresholding on the parameter, depending on the value assumed by each element, whether or not it is outside the interval $(-\gamma_i, +\gamma_i)$. The i-th IST algorithm step computes

$$x_{t+1} = \mathbb{S}_{\tau\Lambda}\left[x_t + \tau\, C^\top (y - C\, x_t)\right]$$

where each element of x_{t+1} is calculated as

$$\mathbb{S}_{\gamma_i}(x_i) := \begin{cases} x_i - \gamma_i & if \quad x_i > \gamma_i \implies shrink \\ x_i + \gamma_i & if \quad x_i < -\gamma_i \implies shrink \\ 0 & if \quad |x_i| \leqslant \gamma_i \implies threshold \end{cases}$$

Provided data and hyper-parameters given by the assignement project, after the execution of 1000 algorithm runs, the IST manages to estimate correctly the state 922 times with a success rate of 92.2 % (a success is defined here as a complete guess of the support state estimation).
Considering the convergence time is measured by the number of steps required by every run to reach the stop criterion, the average number of iterations is 1565.6 with a maximum of 73605 and a minimum of 92.
It's possible to reach an accuracy of 100 % by increasing the number of sensors q and in detail when q is approximately close to 40, according to the asymptotic behaviour shown in Figure 1.

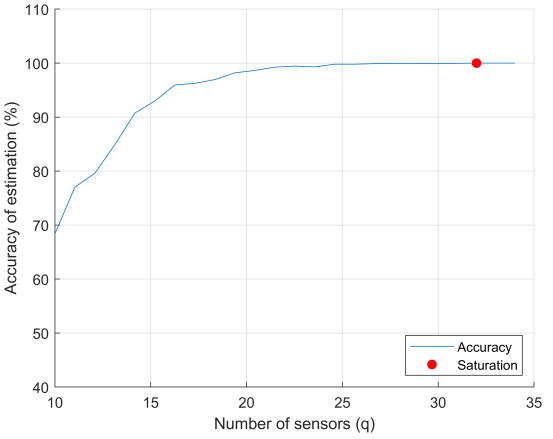

FIGURE 1. IST state estimation accuracy with respect to the number of sensors

Changing the learning rate τ, by keeping a fixed $\lambda = \frac{1}{100\tau}$, while the shrinkage-thresholding coefficient $\gamma_i = \tau\lambda_i$ remains constant, the only difference is the input parameter passed to the function $\mathbb{S}_{\gamma_i}(x_i)$, where τ affects the error between the real output and the intermediate state estimation multiplied by C. In particular if τ decreases the estimation accuracy, measured as the mean squared error between the estimated and the real state, decreases (Figure 2) and the convergence time increases (Figure 3). This happens because the algorithm does not reach the convergence point due the fact that the learning rate becomes so small that the effect of the gradient disappears.

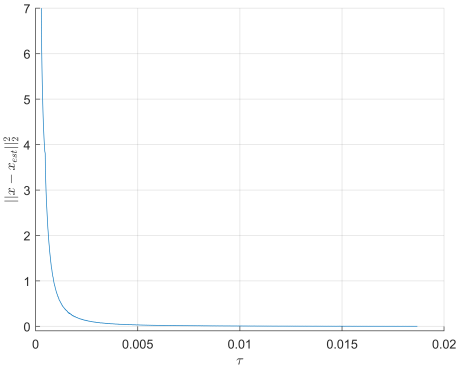

FIGURE 2. Mean Squared Error between real and estimated state based on the learning rate

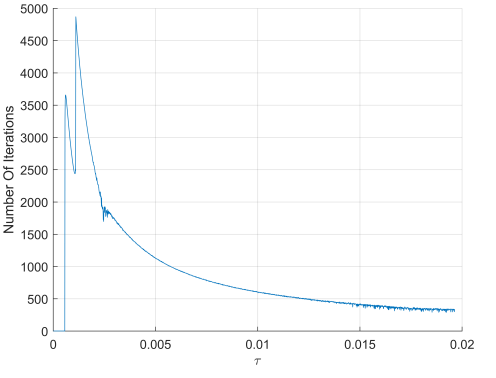

FIGURE 3. Inverse proportionality of IST convergence time versus the learning rate

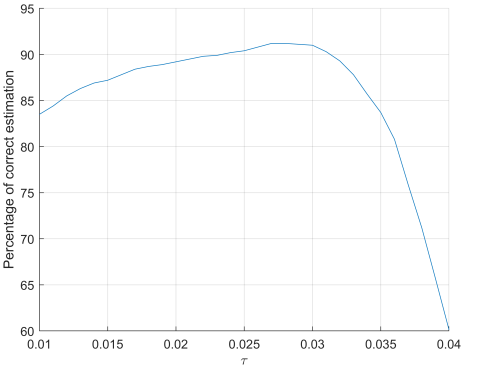

FIGURE 4. Percentage of correct estimation w.r.t increasing learning rate

As long as τ increases it is observable a huge improvement in the percentage of correct state estimations because the gradient obtains more importance but if it gets excessively high the IST can no longer converge due to the bigger and bigger corrections on the estimate which leads to the oscillation around the exit condition (Figure 4).

Keeping a constant learning rate τ and changing the sparsity coefficient λ will result in a change of the shrinkage-thresholding coefficient $\gamma_i = \tau\lambda_i$. By increasing the value of the sparsity coefficient the convergence time and the estimation accuracy decrease until the latter reaches the maximum squared error, which coincides with l_2-norm squared of the state itself. This is due to the fact that the growth of the shrinkage-thresholding coefficient leads to a greater parameter correction performed by $\mathbb{S}$, which brings the algorithm to converge faster since the i-th parameter element falls earlier in the threshold window $(-\gamma_i, +\gamma_i)$, moreover enlarged by a λ factor, and consequently the i-th estimated element is immediately brought to zero when λ is large enough. In particular we observed, after 1000 runs, that λ must be at least equal to 6 to obtain the worst state estimation, according to the piecewise function shown in Figure 5, so it is suggested to keep a low λ to minimize the sparsity of the state vector and achieve a better estimation, but not too much in order to avoid incredibly longer convergence times as shown in Figure 6.

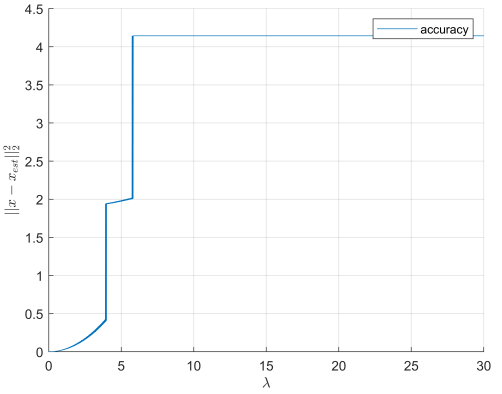

FIGURE 5. Asimptotically esponential MSE with respect to the sparsity coefficient

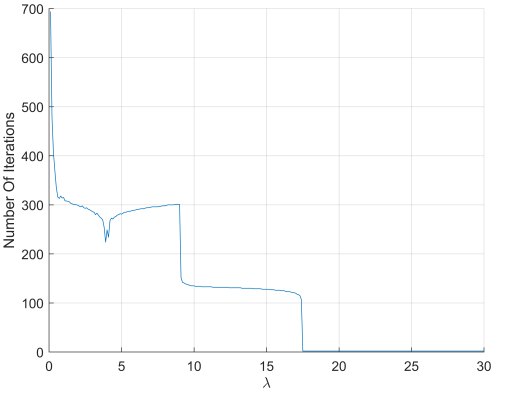

FIGURE 6. Decrease of the convergence time by increasing the attack sparsity coefficient

2. Secure estimation under sparse sensor attack

Now we analyze the ISTA performance used to solve the problem of state estimation, represented by a non-sparse vector $\tilde{x}$, in the presence of attacks a and noise η on sensors, which lead to wrong measurements y, distinguishing the case in which the attack is aware (which depends on the measured value) and unaware (uniformly distributed in $[-2, -1] \cup [1, 2]$ interval). In addition the simulation was realized considering the all four possible combinations obtained taking into account the noise-free and the white-noise cases.
To accomplish the minimization problem to estimate both the state and the attack a variation of the LASSO problem is needed, which becomes a partial LASSO, in which the output matrix is obtained with the concatenation $G = (C \ I) \in \mathbb{R}^{q,n+q}$ and where the sparsity hyper-parameter $\Lambda \in \mathbb{R}^{n+q}$ is created such as $\Lambda^\top = (0, ..., 0, 1, ..., 1)$ to comply with the sparsity hypothesis of $(x \ \ a)^\top$ vector. Furthermore this reformulation allows to minimize only the sparsity of the attack, instead of the state assumed to be non-sparse.

$$\min_{x \in \mathbb{R}^p, a \in \mathbb{R}^q} \left\| G \begin{pmatrix} x \\ a \end{pmatrix} - y \right\|_2^2 + \Lambda^\top \left| \begin{pmatrix} x \\ a \end{pmatrix} \right|$$

The following table shows the simulation results after 1000 runs for each configuration. Notice that the first two columns separate the different cases, the next two are referred to attack support estimations, where a hit is still defined as a complete guess of all the $h = 2$ attacked sensors in a boolean way, so the success percentage, while the third column is referred to the average Mean Squared Error between the real and estimated state.

case		attack		state
noise	attack	number of hits	success percentage	average MSE
No	Unaware	829	82.90 %	6.75×10^{-5}
Gaussian	Unaware	801	80.10 %	7.01×10^{-5}
No	Aware	716	71.60 %	6.60×10^{-5}
Gaussian	Aware	667	66.70 %	6.34×10^{-5}

It is possible to observe that the best conditions are obtained without any noise and in the presence of bounded random attacks, which brings the algorithm to estimate all the attacks correctly above the 80% of the runs. When the noise becomes white, the hit rate drops by a few percentage points but it remains comparable to the previous case. However the estimation attack performances drop by 10 percentage points with the introduction of attacks proportional to the values measured by the sensors and worsen with the addition of Gaussian distributed noise which lowers the attack estimation accuracy by a further 5%.
Regarding the state estimation it can be noticed that the accuracy, inversely proportional to the mean squared error, which is kept in the same order of magnitude in all the cases, stays high enough. In particular it is not possible to derive a correlation for the estimation accuracy with respect to the impact of the disturbances on the measured values.

3. Target localization with real data

In this paragraph we apply the IST algorithm to solve a real fingerprint indoor localization in two scenarios, without any disturbances and in the presence of an aware single attacks on each sensor. In detail, the goal is localizing a target in a two dimensional space which is divided in a grid composed by p cells, where the estimation of the target position is achieved by matching online measurements with the closest a-priori location fingerprints.
In the training phase (offline) the target is placed each time in a different cell of the grid from where it broadcasts a signal measured by the q sensors, each one of them collects the relative Received Signal Strength fingerprint. Thus the result of this phase is a so-called dictionary, that consists of a $q \times p$ matrix containing all the RSS-measurements acquired by sensor i when the device is placed in the cell j.
Given the signature map D and the online measurement taken by sensors (contained in y), the aim of the runtime phase (online) is to evaluate the position of the target x, minimizing the error between the online output and the matrix product of the state estimate with the normalized dictionary, by setting a proper LASSO formulation. In particular, the only variation with respect to the original version (paragraph 1) consists in considering the normalized dictionary D_n itself as the previous output matrix C, assuming the sparsity coefficient as a unit vector $\Lambda = (1\,1\,...\,1)^\top \in \mathbb{R}^q$.

$$\min_{x \in \mathbb{R}^p} \|D_n x - y\|_2^2 + \Lambda^\top |x|$$

Adding a single attack that perturbs y_i of a quantity equal to $y_i/5$ in turn for each sensor measurement i, in order to estimate the target position and which sensor got attacked, it is necessary to formulate the following extended LASSO which sparsifies both the state and the attack array, setting the sparsity vector $\Lambda = (1\,1\,...\,1)^\top \in \mathbb{R}^{p+q}$.

$$\min_{x\in\mathbb{R}^p, a\in\mathbb{R}^q} \left\| \begin{pmatrix} D_n & I \end{pmatrix} \begin{pmatrix} x \\ a \end{pmatrix} - y \right\|_2^2 + \Lambda^\top \left| \begin{pmatrix} x \\ a \end{pmatrix} \right|$$

In both scenarios the minimization is performed by the Iterative Shrinkage-Thresholding algorithm as seen in the first paragraph, with the specific variations on the output matrix and the previous considerations on the hyperparameters.
The following table summarizes the outcomes of the different cases, the one without any attack, represented by the first column, and the other ones with single attacked sensors, represented by the next six columns. For each scenario we evaluate the ability to correctly detect the attacked sensor and the estimated target position, which is obtained taking into account only the largest value of the array x (where each non-null element index represents the cell where one or more targets are placed).

	attacked sensor						
statistics	none	1	2	3	4	5	6
estimated attacked sensor	–	1	2	4	3	5	6
attack correctly detected	–	✓	✓	✗	✗	✓	✓
estimated target position	7	7	7	7	7	7	7

Looking at the above mentioned results it should be noticed, excluding the first case, that the algorithm manages to recover which sensor got corrupted 4 times over 6, as the attack on the third sensor is mistaken for the fourth and viceversa.
As far as the estimate of the position is concerned, however, the algorithm proves to be very accurate in all the cases, since it is capable to find a particular cell in which the vector element has a far greater magnitude (equivalent to the value "1" of the binary linear regression formulation where, before relaxation, $x \in \{0,1\}^p$) and thanks to this it is able to perfectly localize the target, which is found to be in the 7-th cell of the grid.

4. Sparse observer

The previous IST formulation can be re-proposed to design an online observer able to localize moving targets and detect eventual attacks on the sensors. The number of targets is increased to 4 and they are now free to move around $p = 100$ cells, in accordance with the system dynamics described by the A matrix, while $q = 25$ sensors take the measurements which are sent to the Fusion Center that performs an estimation of the targets locations both in the presence of unaware and aware attacks. Notice that, although the state changes in time, regarding the attacked sensors we assume that the support of $a(k)$ is constant over time, otherwise it becomes tough to recover correctly the attack. To accomplish this goal, defining $z = (x \;\; a)^\top \in \mathbb{R}^{p+q}$ as the concatenation of the positions $x \in \mathbb{R}^p$ and attacks $a \in \mathbb{R}^q$ arrays to estimate, we can work it out using an online gradient descend approach, but since the estimate is assumed sparse, it is possible to solve the minimization problem with the IST algorithm. This is performed splitting the single iteration in two steps: in the first one the IST algorithm is used to generate an intermediate prediction, informally denoted here as $z\left(k+\frac{1}{2}\right)$, of both the targets position $x\left(k+\frac{1}{2}\right)$ and the attacks $a\left(k+\frac{1}{2}\right)$,

thus after time k the shrinkage operator computes

$$z\left(k+\frac{1}{2}\right) = \mathbb{S}_{\tau\Lambda}\left[z(k) + \tau G^{\top}\left(y(k) - Gz(k)\right)\right]$$

while, in the second step, it is possible to estimate the next positions value, multiplying the state found before with the dynamics matrix A, and to extract the attack, at $k+1$

$$x\left(k+1\right) = A\, x\left(k+\frac{1}{2}\right)$$

$$a\left(k+1\right) = a\left(k+\frac{1}{2}\right)$$

This formulation can be interpreted from two equivalent points of view: one that considers the estimation as a dynamical Iterative Shrinkage-Thresholding version in which the reference output is updated simultaneously with increasing iterations, and the other one as a state observer that performs the estimation running the IST algorithm just once a time stopping after a single iteration, providing a rough but efficient real time approximation. Below it is possible to observe the visual representation of the Sparse Observer at work in two videos:

- Unaware attacks: `https://youtu.be/XnhWdHDG8QI`
- Aware attacks: `https://youtu.be/0OzCtU2MasY`

The performances of the resulting Sparse Observer are represented following by the Figure 7 and 8. As concerns the accuracy of the target position estimate it is measured at each time step k as a mean over 100 algorithm simulations for 100 time instants. The same criterion is applied for the evaluation of the estimate of the corrupted sensors, but rather considering the specific values of the array we take into account just the support of the attack vector and the a-priori known number of attacked sensors.

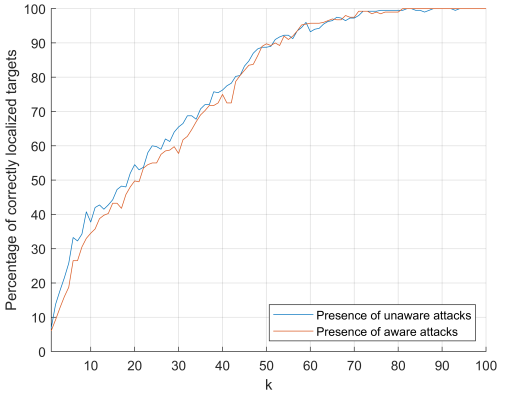

FIGURE 7. Accuracy of the target position estimation w.r.t. discrete-time k over 100 runs

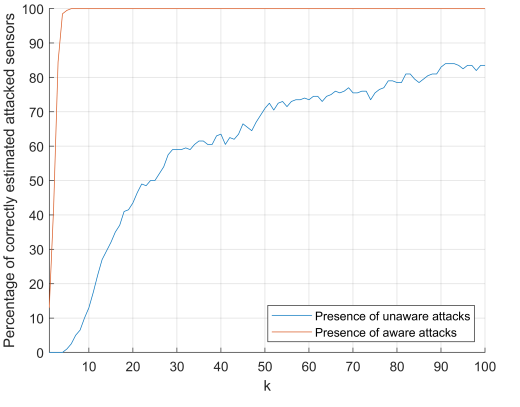

FIGURE 8. Precision of the attacked sensor estimation by increasing time k over 100 runs

As it can be seen the accuracy of the estimation of the targets position tends to increase after each step, following almost the same negative exponential trend in Figure 7, no matter what type of attack is performed on the sensors. Meanwhile, the accuracy of the estimation of the attacked sensors differs as the kind of the attacks changes: in the presence of aware attacks, which are greater in magnitude, the estimation tends to be more precise in much less time compared to the unaware attack case, which turns out to be slower to reach the correct estimate and even after 100 instants it will not reach an accuracy of 100% (Figure 8).

5. Distributed estimation

In this last paragraph we propose a distributed state estimation in the presence of noise and unaware sparse attacks, where the estimate is obtained in-network using a distributed version of the IST algorithm (DIST), which means that each sensor with its embedded computation capability runs the IST algorithm on its own, based on the information that it measures through a distributed acquisition, which results after a local distributed processing are shared with its neighbours. In particular, at each iteration k, every node shrinks the sum of the weighted mean of the estimated state received from all its neighbours plus the gradient of its own estimated state and then spreads it to its connected nodes, so the k-th DIST algorithm step is computed for each node i like

$$z^{(i)}(k+1) = \mathbb{S}_{\tau\Lambda}\left[\sum_{j=1}^{q} Q_{i,j} z^{(j)}(k) + \tau G_i^{\top}\left(y_i - G_i z^{(i)}(k)\right)\right]$$

where the estimate z is the concatenation of the state $x \in \mathbb{R}^n$ with the attack $a \in \mathbb{R}^q$ array, the matrix $G \in \mathbb{R}^{q\times(n+q)}$ is defined as $(C \;\; I)$ and the matrix $Q \in \mathbb{R}^{q\times q}$ represents the edge weights of the network, which i-th row contains the weights of the income links for the i-th node from all q agents.
The sufficient condition for a state matrix Q such that $x(k+1) = Q\,x(k)$, to reach global consensus solved by Q if $\exists\,\alpha \in \mathbb{R} : \lim_{k\to\infty} x(k) = \alpha\mathbf{1}$, making all the nodes converge to the same estimate regardless of the number or the initial state of its neighbours, is such that Q has a maximum unitary eigenvalue $\lambda_{PF} = 1$ and the others with smaller absolute value $|\lambda_{i\neq 1}| < \lambda_{PF}$.

We analyze the DIST algorithm performance using four different network topologies, represented by the relative matrices given by the assignement, with homogeneous weighted links between $q = 20$ non-isolated agents, where the first, the second and the last topologies have random configurations while the third follows a ring topology.
Firstly, taking into account the previous considerations, it is possible to say a-priori that the consensus on the state, for all the given networks, is achievable since they are all double-stochastic matrices with a single maximum unitary eigenvalue (its associated eigenvector represents a probability vector that does not change when the system evolves).

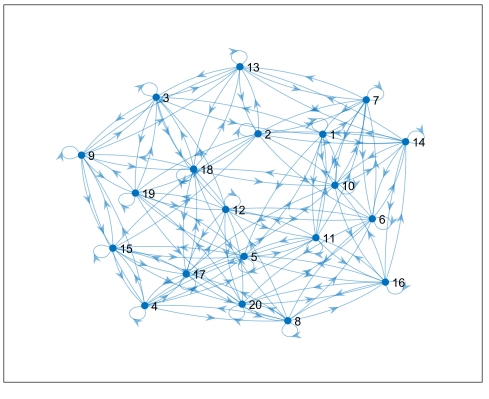

Figure 9. Q1

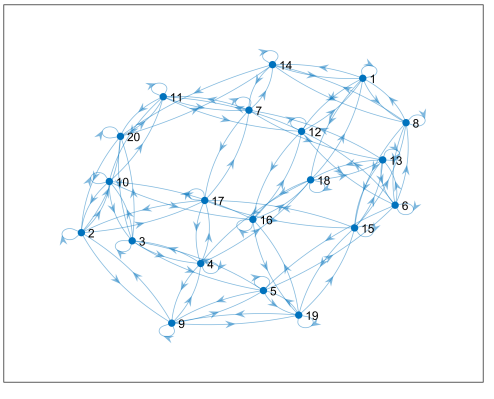

Figure 10. Q2

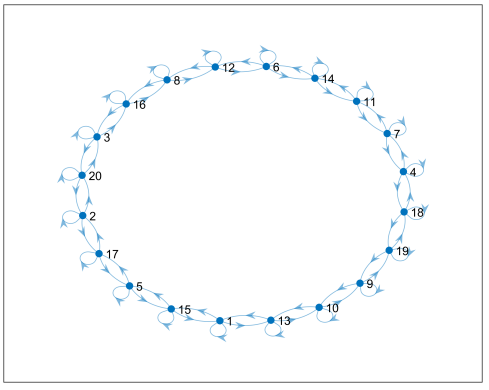

FIGURE 11. Q3

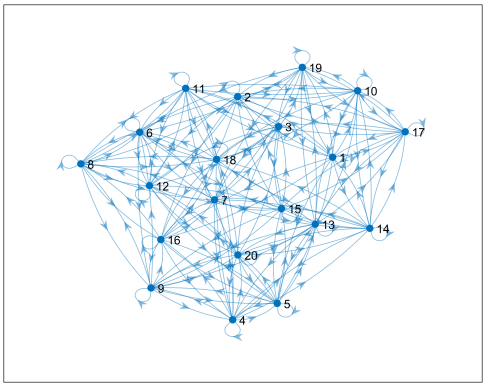

FIGURE 12. Q4

The following table summarizes the average performance measured according to the main statistics of the consensus problem, over 20 runs of the DISTA algorithm for each topology.

	network topology			
statistics	1	2	3	4
number of links	80	60	40	100
essential spectral radius	0.5857	0.7373	0.9674	0.5103
average convergence time	22355	27874	42124	19277
average consensus deviation	6.60×10^{-6}	1.02×10^{-5}	2.56×10^{-4}	5.50×10^{-6}
times of achieved consensus	19	20	20	19
average attack accuracy	92.50%	97.50%	93.12%	95.00%
average state MSE	0.1155	0.0481	0.0764	0.0793

In practice the simulations show that, provided a fixed stop criterion $T_{max} = 10^5$, it is possible to empirically demonstrate the global agreement on the state is reached almost always thanks to a small bounded average variance, but in some cases it may occasionally happen that the algorithm does not converge within the given maximum iteration time, thus consensus cannot guaranteed to be reached at least with low uncertainty. It is observable that the smaller the essential spectral radius $esr(Q)$ is, the more the algorithm ends up with a lower convergence rate, since it is possible to show theoretically that the matrix Q raised to the instant k, according to the system dynamics, tends to take more time to remain with just a unitary eigenvector if the second largest eigenvalue is close to the spectral radius $\rho = 1$, due to the fact it goes asymptotically to zero in a longer transient.
Although it is not present a correlation between the average attack estimation accuracy and the characteristics of the topology, the precision on detecting the attacked sensors is quite high in all the cases, exceeding at least 90% of right attack support hits.
Regarding the state estimation, evaluated according to the average error over all the runs, each one computed as the squared euclidian distance between the real $\tilde{x}$ and the average estimated state $\bar{x}$ by all the agents, it is possible to observe that the algorithm proves to be rather accurate for the nodes in estimating the state of all the others in a distributed way and regardless of the topology, given that the overall error does not exceed 10% of the maximum value assumed by $\tilde{x}$ itself.

Finally, we proved that DISTA results to be an extremely powerful algorithm capable of providing, for each agent in the network, an effective way to independently estimate on

a common agreement the state of each of the other nodes and to detect eventual attacked sensors. Furthermore, we practically demonstrated that the topology does not affect the ability to reach consensus, both for the estimation of the state and the support of the attack, but the weights of the edges can affect the convergence time, which if turns out to be limited in order to improve the efficiency of the algorithm, can prevent global agreement from being reached, thus not guaranteeing a unique estimate.

Conclusion

In this paper we analyzed the performance of the IST algorithm which is an approach easy to implement to solve different estimation problems fitting a large amount of scenarios. Looking at the results of the simulations it is possible to see that the IST guarantees really high success rate and precision although it works only if the state to estimate is sparse, which means that its efficiency is proportional to the number of sensors. Hence the IST algorithm proves to be a suitable and flexible choice for the state estimation of Cyber-Physical systems.