53rd IEEE Conference on Decision and Control
December 15-17, 2014. Los Angeles, California, USA

Event-Triggered Projected Luenberger Observer for Linear Systems Under Sparse Sensor Attacks

Yasser Shoukry and Paulo Tabuada

Abstract— **We consider the problem of designing a Luenberger-like observer for linear systems whose sensor measurements are corrupted by a malicious attacker. The attacker capabilities are limited in the sense that only a subset of all the sensors can be attacked although this subset is unknown. This leads to the problem of reconstructing the system state when the measurements are corrupted by sparse noise and we propose an observer that recursively updates the state estimate as new measurements become available. We show that by utilizing event-triggered techniques, the proposed observer is computationally more efficient than previously reported solutions to the secure state reconstruction problem.**

I. Introduction

Aiming at a reduction of computation and communication resources, the event-triggered control paradigm arose as an alternative to the classical periodic control paradigm. Although early results [1], [2] suggested through examples that event-triggered control could reduce resource usage while maintaing adequate levels of control performance, the interest in these techniques grew substantially once it was shown in [3], [4] how to systematically construct event-triggered implementations of feedback control laws. For a recent survey on event-triggered control, the reader is referred to [5].

In this paper, we consider the use of event-triggering in the design of observers. Several results on event-triggered observers have recently appeared in the literature. Theses results focus on utilizing the event-triggered mechanism to adaptively choose the sampling point aiming to optimize specific system design critera [6], [7], [8], [9], [10]. Unlike the previously reported results, we propose to utilize event-triggering to enhance the computational performance of periodically-triggered observers.

We are motivated by the situation where an adversarial attack is able to sparsely corrupt sensor measurements. An increasing number of such attacks has been reported in the last few years. Examples of control systems which were reported vulnerable to these attacks are SCADA systems [11], smart grids [12], and automotive systems [13].

Sparse/noise attack is a natural model to describe the effect of a malicious attacker that has the ability to alter the measurements of a subset of sensors in a feedback control loop. While measurements coming from un-attacked sensors are attack free, measurements from attacked sensors can be arbitrary: we make no assumption regarding its magnitude, statistical description, or temporal evolution.

A solution to this state reconstruction problem under adversarial attacks was previously proposed in [14], [15], [16] by the authors and co-workers and its experimentally validation discussed in [17]. However, this solution is computationally intensive and we focus in this paper on *efficient* state reconstruction algorithms in the sense of being implementable on computationally limited platforms.

The previously proposed algorithms reconstructs the state of the system using a batch approach based on measured data collected over a period of time. On computationally restricted platforms we may be faced with the difficulty of having to process new measurements before being able to finish all the computations in the previous batch. It would then be preferable to reconstruct the system state using a recursive algorithm that can incorporate new measurements as they become available.

We propose a Luenberger-like observer that reconstructs both the state and the sparse attack signal. The proposed observer utilizes ideas from event-triggered control [3] to enhance the computational efficiency. That is, due to the adaptive nature of the event-triggered observer, the computation time is reduced by an order of magnitude compared to the previously proposed solutions based on $L_r \backslash L_1$ algorithms [14], [15], [16].

This paper is organized as follows. Section II formally introduces the problem under consideration. The notions of s-observability and restricted eigenvalues are introduced in Section III. The main contribution of this paper which is the Event-Triggered Projected Luenberger Observer, along with its convergence properties, is introduced in Section IV. Simulation results for the proposed algorithm is shown in Section V. Finally Section VI concludes this paper. For the sake of space, all proofs are omitted. We point the reader to [18] where detailed proofs can be found.

II. The Secure State Reconstruction Problem

A. Notation

The symbols $\mathbb{N}_0, \mathbb{R}$, and $\mathbb{R}^+$ denote the set of natural, real, and positive real numbers, respectively. Given two vectors $x \in \mathbb{R}^{n_1}$ and $y \in \mathbb{R}^{n_2}$, we denote by $(x, y) \in \mathbb{R}^{n_1+n_2}$ the vector $\begin{bmatrix} x^T & y^T \end{bmatrix}^T$. If S is a set, we denote by $|S|$ the cardinality of S. The support of a vector $x \in \mathbb{R}^n$, denoted by $\text{supp}(x)$, is the set of indices of the non-zero

Y. Shoukry and P. Tabuada with Electrical Engineering Department, UCLA, yshoukry@ucla.edu, tabuada@ucla.edu

This work was partially sponsored by the NSF award 1136174 and DARPA under agreement number FA8750-12-2-0247. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government.

elements of x. We call a vector $x \in \mathbb{R}^n$ s-sparse, if x has at most s nonzero elements, i.e., if $|\text{supp}(x)| \leq s$. A vector $z = (x_1, x_2, \ldots, x_p) \in \mathbb{R}^{np}$ is called cyclic s-sparse, if each $x_i \in \mathbb{R}^n$ is s-sparse for all $i \in \{1, 2, \ldots, p\}$ and $\text{supp}(x_1) = \text{supp}(x_2) = \ldots = \text{supp}(x_p)$. With some abuse of notation, we use s-sparse to denote cyclic s-sparse.

For a vector $x \in \mathbb{R}^n$, we denote by $\|x\|_2$ the 2-norm of x. For a matrix M, we denote by $M_i \in \mathbb{R}^{1\times n}$ the ith row of M. For a set $\Gamma \subseteq \{1, \ldots, m\}$, we denote by $M_\Gamma \in \mathbb{R}^{|\Gamma|\times n}$ the matrix obtained from M by removing all the rows except those indexed by Γ. We also denote by $M_{\overline{\Gamma}} \in \mathbb{R}^{(m-|\Gamma|)\times n}$ the matrix obtained from M by removing the rows indexed by the set Γ, for example, if $m = 4$, and $\Gamma = \{1, 2\}$, then:

$$M_\Gamma = \begin{bmatrix} M_1 \\ M_2 \end{bmatrix} \text{ and } M_{\overline{\Gamma}} = \begin{bmatrix} M_3 \\ M_4 \end{bmatrix}.$$

For any finite set $S = \{s_1, s_2, \ldots, s_k\} \subset \mathbb{N}$ we denote by rS, $r \in \mathbb{N}$, the set $rS = \{rs_1, rs_2, \ldots, rs_k\}$. Finally we denote the maximum eigenvalue of a symmetric matrix M by $\lambda_{\max}\{M\}$.

B. Dynamics and Attack Model

Consider the following linear discrete-time control system where $x(t) \in \mathbb{R}^n$ is the system state at time $t \in \mathbb{N}_0$, $u(t) \in \mathbb{R}^m$ is the system input, and $y(t) \in \mathbb{R}^p$ is the observed measurement:

$$x(t+1) = Ax(t) + Bu(t), \tag{II.1}$$
$$y(t) = Cx(t) + a(t). \tag{II.2}$$

The matrices A, B, and C have appropriate dimensions and $a(t) \in \mathbb{R}^p$ is a s-sparse vector modeling how an attacker changed the sensor measurements at time t. If sensor $i \in \{1, \ldots, p\}$ is attacked then the ith element in the vector $a(t)$ is non-zero otherwise the ith sensor is not attacked. Hence, s describes the number of attacked sensors. Note that we make no assumptions on the vector $a(t)$ either than being s-sparse. In particular, we do not assume bounds, statistical properties, nor restrictions on the time evolution of the elements in $a(t)$. The value of s is also not assumed to be known and the results in this paper relate the possibility of reconstructing the state to the value of s and the properties of (II.1) and (II.2). The set of sensors the attacker has access to is assumed to remain constant over time (and has cardinality at most s). However, the attacker has complete freedom in deciding which sensor or sensors in this set are attacked and when, including the possibility of attacking all of them at all times.

Our objective is to simultaneously construct a delayed version of the state, $x(t-\tau+1)$, and the *cyclic s-sparse* attack vector $E(t) = (a(t-\tau+1), a(t-\tau+2), \ldots, a(t))$ from the measurements $y(t-\tau+1), y(t-\tau+2), \ldots, y(t)$.

It is worth explaining the cyclic sparse nature of $E(t)$ in more detail. Consider the attack vector $E(t) = (a(t-\tau+1), a(t-\tau+2), \ldots, a(t))$ and reformat this data as the matrix $\widetilde{E}(t) \in \mathbb{R}^{p\times\tau}$ where column i is given by $a(t-i+1)$. Since we assume that the set of sensors under attack does not change over time, the sparsity pattern appears in $\widetilde{E}(t)$. The rows corresponding to the un-attacked sensors will only have zeros, while the rows corresponding to the attacked sensors will have arbitrary (zero or non-zero) elements.

Example 2.1: Consider a system with 4 sensors, $\tau = 4$, and an attack on the second and third sensors. The attack matrix $\widetilde{E}(t)$ will be of the form:

$$\widetilde{E}(t) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 5 & 6 & 10 \\ 4 & 8 & 0 & 12 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

The cyclic sparsity structure appears once the attack matrix $\widetilde{E}(t)$ is reshaped as the vector $E(t)$:

$$E(t) = \begin{bmatrix} 0 & 2 & 4 & 0 & 0 & 5 & 8 & 0 & \ldots \end{bmatrix}^T.$$

Since cyclic s-sparse vectors pervade this paper, it is convenient to denote them by a special symbol.

*Definition 2.2 (**Cyclic s-sparse set** $\mathbb{S}_s$):* The subset of $\mathbb{R}^{p\tau}$ consisting of the vectors that are cyclic r-sparse for all $r \in \{0, 1, \ldots, s\}$, $s \leq p$, is denoted by $\mathbb{S}_s$.

C. Problem Formulation

By collecting $\tau \in \mathbb{N}$ observations with $\tau \leq n$, we can rewrite the dynamics of $z(t) = (x(t-\tau+1), E(t)) \in \mathbb{R}^n \times \mathbb{R}^{p\tau}$, using the equality $a(t) = y(t) - Cx(t)$, as follows:

$$z(t) = \overline{A}z(t-1) + \overline{B}\bar{u}(t-1)$$
$$Y(t-1) = Qz(t-1),$$

where:

$$z(t) = \begin{bmatrix} x(t-\tau+1) \\ E(t) \end{bmatrix}, \quad \bar{u}(t-1) = \begin{bmatrix} U(t-1) \\ y(t) \end{bmatrix},$$

$$E(t) = \begin{bmatrix} a(t-\tau+1) \\ a(t-\tau+2) \\ \vdots \\ a(t) \end{bmatrix}, \quad U(t) = \begin{bmatrix} u(t-\tau+1) \\ u(t-\tau+2) \\ \vdots \\ u(t) \end{bmatrix},$$

$$\overline{A} = \left[\begin{array}{cccc} A & 0 & \ldots & 0 \\ \hline 0 & I & \ldots & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \ldots & I \\ -CA^\tau & 0 & \ldots & 0 \end{array}\right],$$

$$\overline{B} = \left[\begin{array}{ccccc} B & 0 & \ldots & 0 & 0 \\ \hline 0 & 0 & \ldots & 0 & 0 \\ \vdots & & & \vdots & \vdots \\ -CA^{\tau-3}B & -CA^{\tau-2}B & \ldots & -CB & I \end{array}\right],$$

$$F = \begin{bmatrix} 0 & 0 & \ldots & 0 & 0 \\ CB & 0 & \ldots & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ CA^{\tau-2}B & CA^{\tau-3}B & \ldots & CB & 0 \end{bmatrix},$$

$$Y(t) = \widetilde{Y}(t) - FU(t), \quad Q = \begin{bmatrix} \mathcal{O} & I \end{bmatrix},$$

$$\widetilde{Y}(t) = \begin{bmatrix} y(t-\tau+1) \\ y(t-\tau+2) \\ \vdots \\ y(t) \end{bmatrix}, \mathcal{O} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{\tau-1} \end{bmatrix}.$$

The problem addressed in this paper can now be formally stated by using this dynamics and the notion of cyclic sparsity.

Problem 2.3: **Luenberger-like Observer**
For the linear control system defined by (II.1) and (II.2) construct a dynamical system:

$$\hat{z}(t+1) = f(\hat{z}(t), U(t), Y(t)),$$

such that:

$$\lim_{t\to\infty} (z^*(t) - \hat{z}(t)) = 0,$$

where $z^*(t) = (x^*(t-\tau+1), E^*(t)) \in \mathbb{R}^n \times \mathbb{S}_s$, $x^*(t)$ is the solution of (II.1) under the inputs $U(t)$, and $Y(t)$ is the sequence of the last τ observed outputs corrupted by $E^*(t)$.

Note that this problem asks for the reconstruction of a delayed version of the state $x(t-\tau+1)$. However, we can always reconstruct of the current state $x(t)$ from $x(t-\tau+1)$ by recursively rolling the dynamics forward in time. Alternatively, when A is invertible, we can directly recover $x(t)$ by re-writing the measurement equation as a function of $x(t)$.

As a final remark we note that it follows from the Cayley-Hamilton theorem that there is no loss of generality in taking the number of collected measurements τ to be no greater than the number of states n.

III. s-Sparse Observability and Restricted Eigenvalues

In this section we define the notions of s-sparse observability and s-restricted eigenvalues. These notions plays important role in characterizing the performance of the proposed algorithm.

A. s-Sparse Observability

We start by introducing the following notion of observability.

Definition 3.1: **s-Sparse Observable System**
The linear control system defined by (II.1) and (II.2) is said to be s-sparse observable if for every set $\Gamma \subseteq \{1, \ldots, p\}$ with $|\Gamma| = s$, the pair $(A, C_{\overline{\Gamma}})$ is observable.

In other words, a system is s-sparse observable if it remains observable after eliminating any choice of s sensors.

B. s-Restricted Eigenvalue

A more quantitative version termed the *s-restricted eigenvalue property* [19] is discussed next. This property is directly related to the possibility of solving Problem 2.3 by Luenberger-like observer.

Definition 3.2: **(s-Restricted Eigenvalue of a Linear Control System)** For a given set $\widetilde{\Gamma}_s \subseteq \{1, \ldots, p\}$ with $|\widetilde{\Gamma}_s| = p - s$, let Γ_s be defined by:

$$\Gamma_s = \widetilde{\Gamma}_s \cup p\widetilde{\Gamma}_s \cup 2p\widetilde{\Gamma}_s \cup \ldots \cup (\tau-1)p\widetilde{\Gamma}_s \tag{III.1}$$

and define (with some abuse of notation) the matrix $Q_{\overline{\Gamma}_s} \in \mathbb{R}^{p\tau \times (n+s\tau)}$ as the matrix obtained from $Q = \begin{bmatrix} \mathcal{O} & I \end{bmatrix}$ by removing from I the columns indexed by Γ_s, i.e.:

$$Q_{\overline{\Gamma}_s} = \begin{bmatrix} \mathcal{O} & (I_{\overline{\Gamma}_s})^T \end{bmatrix}. \tag{III.2}$$

The s-restricted eigenvalue of the control system defined by (II.1) and (II.2) is the the smallest eigenvalue of all the matrices $Q^T_{\overline{\Gamma}_s} Q_{\overline{\Gamma}_s}$ obtained by considering all the different sets $\widetilde{\Gamma}_s$.

The s-restricted eigenvalue of a control system can be related to the s-observability as follows.

Proposition 3.3: **(Non-zero Restricted Eigenvalue)** Let the linear control system, defined by (II.1) and (II.2), be $2s$-sparse observable. There exists a $\delta_{2s} \in \mathbb{R}^+$ such that for every $z = (x, E) \in \mathbb{R}^n \times \mathbb{S}_{2s}$ the $2s$-restricted eigenvalue is no smaller than δ_{2s}.
In other words, although Q has a non-trivial kernel, Q^TQ has a non-zero minimum eigenvalue when operating on a vector $z = (x, E)$ with cyclic $2s$-sparse E, i.e., the following inequality holds if $z = (x, E) \in \mathbb{R}^n \times \mathbb{S}_{2s}$:

$$0 < \delta_{2s} z^T z \leq z^T Q^T Q z. \tag{III.3}$$

The calculation of δ_{2s} is combinatorial in nature. The reader is referred to [18] for the details on the computation of δ_{2s}.

IV. An Event-Triggered Projected Luenberger Observer

The problem of reconstructing a sparse signal from measurements is well studied in the compressive sensing literature [20], [21]. In this section we propose an observer for state reconstruction that can be seen as an extension of the iterative hard thresholding algorithm reported in [22] to the case where part of the signal to be reconstructed is sparse and part is governed by linear dynamics.

A. The Algorithm

The Event-Triggered Projected Luenberger (ETPL) Observer consist of the iteration of the following two steps:

*a) **Event-Triggered Projected Luenberger (Measurement) Update**:* When a new measurement is available, we apply the following Luenberger update step multiple times:

$$\hat{z}^{(m+1)}(t) = \hat{z}^{(m)}(t) + L\left(Y(t) - Q\hat{z}^{(m)}(t)\right),$$

where the Luenberger gain matrix L is chosen as $L = Q^T\Sigma$ with Σ a positive definite matrix. This Luenberger update forces a decrease in the Lyapunov candidate function V defined as:

$$\begin{aligned} V(\hat{z}(t)) &= \frac{1}{2}\left\|Y(t) - Q\hat{z}(t)\right\|_2^2 = \frac{1}{2}\left\|Qz^*(t) - Q\hat{z}(t)\right\|_2^2 \\ &= \frac{1}{2}\left\|Qe(t)\right\|_2^2, \end{aligned}$$

where $z^*(t)$ is defined in Problem 2.3 and $e(t)$ is the error vector defined as $e(t) = z^*(t) - \hat{z}(t)$.

Since $\hat{z}(t)$ will not, in general, satisfy the desired sparsity constraints, Luenberger updates are followed by a projection step which uses the projection operator:

$$\Pi : \mathbb{R}^n \times \mathbb{R}^{p\tau} \to \mathbb{R}^n \times \mathbb{S}_s$$

that takes $\hat{z}(t) \in \mathbb{R}^n \times \mathbb{R}^{p\tau}$ to the closest point in $\mathbb{R}^n \times \mathbb{S}_s$. In other words, we apply the following step at the end of the Luenberger update loop:

$$\hat{z}_\Pi(t) = \Pi\left(\hat{z}^{(m)}(t)\right).$$

The analysis reported in [18] shows that the projection step may lead to an increase in the value of the Lyapunov candidate function V. Therefore, the use of the projection operator Π requires us to perform multiple Luenberger update steps in order to guarantee convergence of the observer. This contrasts with the standard Luenberger observer where only one Luenberger update step is necessary. Although we can compute a priori a worst-case upper bound on the number of measurement update steps, an implementation based on such bound would result in a waste of computational resources. Instead, we draw inspiration from the event-triggered control paradigm by performing the measurement update step while the triggering condition $V_{temp} < V(\hat{z}_\Pi(t))$ is satisfied (see Algorithm 1). We will show in Section V how this leads to an efficient usage of resources and in Section IV-C how this choice of triggering condition ensures convergence of the observer.

*b) **Time Update**:* In this step, we use the dynamics to update the previous estimate:

$$\hat{z}(t) = \overline{A}\hat{z}(t-1) + \overline{B}\overline{u}(t-1).$$

This time update may result also in an increase in the value of the Lyapunov candidate function V. It follows from from our analysis that alternating between multiple Luenberger updates and projection steps (in the projected luenberger update loop) forces a decrease of the Lyapunov candidate function V. In order to compensate for the increase introduced by the time update step, multiple projected luenberger update loops are performed for each time-update step. Again, by monitoring the evolution of V, the algorithm can determine when the decrease in V caused by the projected luenberger update loop compensates the increase caused by the time update. This is, again, akin to triggering an event in event-triggered control [3].

By driving V to zero while forcing $\hat{z}(t)$ to be cyclic s-sparse, $\hat{z}(t)$ is guaranteed to converge to $z^*(t)$ since, as shown in [18], the intersection of the cyclic s-sparse set $\mathbb{S}_s$ with the kernel of Q is only one point, $z^*(t)$. This sequence of steps results in Algorithm 1.

To make the connection with the standard Luenberger observer clearer, assume that only one Luenberger/projection update is required per time update. In this case, the ETPL observer can be written as:

$$\begin{aligned}\hat{z}(t) &= \overline{A}\hat{z}_\Pi(t-1) + \overline{B}\overline{u}(t-1) \\ &\quad + L'\left(Y(t-1) - Q\hat{z}_\Pi(t-1)\right) \\ \hat{z}_\Pi(t) &= \Pi(\hat{z}(t))\end{aligned}$$

Algorithm 1 Event Triggered Projected Luenberger Observer

a) Time Update:
$\hat{z}(t) = \overline{A}\hat{z}(t-1) + \overline{B}\overline{u}(t-1)$;
b) Event-Triggered Projected Luenberger (Measurement) Update:
while $V(\hat{z}_\Pi(t)) \geq V(\hat{z}_\Pi(t-1))$ **do**
  reset $m := 0$, $\hat{z}^{(0)}(t) = \hat{z}_\Pi(t) = \Pi(\hat{z}(t))$;
  $V_{temp} := V(\hat{z}_\Pi(t))$;
  while $V_{temp} \geq V(\hat{z}_\Pi(t))$ **do**
    $\hat{z}^{(m+1)}(t) := \hat{z}^{(m)}(t) + L\left(Y(t) - Q\hat{z}^{(m)}(t)\right)$;
    $V_{temp} := V(\Pi\left(\hat{z}^{(m+1)}(t)\right))$;
    $m := m + 1$;
  end while
  $\hat{z}(t) := \hat{z}^{(m)}(t)$;
end while

which has the form of a standard Luenberger observer with gain $L' = A\ L$ along with the projection Π step.

B. The Projection Operator

Before discussing the convergence of the proposed algorithm, it is important to show how to compute the projection operator Π.

Definition 4.1: Given a vector $z = (x, E) \in \mathbb{R}^n \times \mathbb{R}^{p\tau}$, we denote by $\Pi(z)$ the element of $\mathbb{R}^n \times \mathbb{S}_s$ closest to z in the 2-norm sense, i.e.,

$$\|\Pi(z) - z\|_2 \leq \|z' - z\|_2, \tag{IV.1}$$

for any $z' \in \mathbb{R}^n \times \mathbb{S}_s$.

We first note that $\Pi(z) = \Pi(x, E) = (x, \Pi'(E))$. To explain how Π' is computed, recall from Example 2.1 the matrix $\widetilde{E}$ obtained by formatting $E \in \mathbb{R}^{p\tau}$ so that the ith column of $\widetilde{E}$ is given by $a(t-i+1)$. Now, define $\overline{E} \in \mathbb{R}^p$ by $\overline{E}_i = \left\|\widetilde{E}_i\right\|_2^2$. By noting that $\|E\|_2^2 = \left\|\overline{E}\right\|_2^2$ and that E is cyclic s-sparse if and only if $\overline{E}$ is s-sparse, it immediately follows that $\Pi'(z)$ can be computed by setting to zero the elements of E corresponding to the smallest $p - s$ entries of $\overline{E}$.

Example 4.2: Let us consider the following example with $p = 3, \tau = 2$ and $s = 1$:

$$E = \begin{bmatrix}1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9\end{bmatrix}^T \in \mathbb{R}^{p\tau}$$

Hence,

$$\widetilde{E} = \begin{bmatrix}1 & 4 & 7\\ 2 & 5 & 8\\ 3 & 6 & 9\end{bmatrix} \in \mathbb{R}^{p\times\tau}, \qquad \overline{E} = \begin{bmatrix}66\\ 93\\ 126\end{bmatrix} \in \mathbb{R}^p,$$

$$\Pi'(\overline{E}) = \begin{bmatrix}0\\ 0\\ 126\end{bmatrix} \in \mathbb{R}^p,$$

which leads to:

$$\Pi'(E) = \begin{bmatrix}0 & 0 & 3 & 0 & 0 & 6 & 0 & 0 & 9\end{bmatrix}^T.$$

C. Convergence of the ETPL Observer

The main result of this work is then stated in the following Theorem.

Theorem 4.3: **(Convergence of the ETPL observer)** Let the linear control system defined by (II.1) and (II.2) be 2s-observable with $2s$-restricted eigenvalue δ_{2s}. If the following condition holds:

$$\delta_{2s} > \frac{4}{9}\lambda_{\max}\{Q^T Q\}$$

then the dynamical system defined by the ETPL observer is a solution to Problem 2.3 whenever $L = Q^T\Sigma$ for any positive definite weighting matrix Σ satisfying $\lambda_{\max}\{\Sigma\} < \lambda^{-1}_{\max}\{Q^T Q\}$.

V. Simulation Results

A. Computational Performance

We compare the efficiency of the proposed ETPL algorithm against the L_1/L_r decoder introduced in [15]. To perform this comparison, we randomly generated 100 systems with $n = 20$ and $p = 25$ and simulated them against an increasing number of attacked sensors ranging from 0 to 12. For each test case we generated a random support set for the attack vector, random attack signal and random initial conditions. Averaged results for the different numbers of attacked sensors are shown in Fig.1.

Although we claim no statistical significance, the results in Figure 1 are characteristic of the many simulations performed by the authors. The L_1/L_r decoder is implemented using CVX while the ETPL algorithms are direct implementations of Algorithm 1 in Matlab. The tests were performed on a desktop equipped with an Intel Core i7 processor operating at 3.4 GHz and 8 GBytes of memory.

In Fig. 1 we can appreciate how the ETPL algorithm outperform the L_1/L_r decoder by an order of magnitude in execution time.

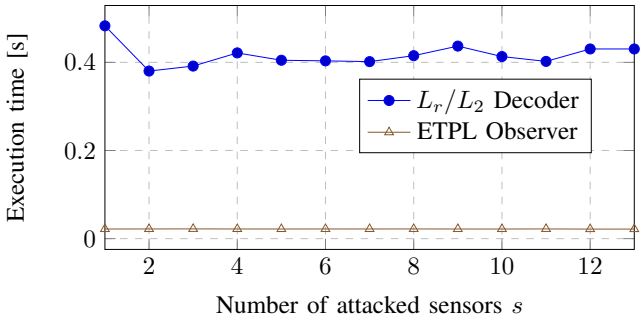


Fig. 1. Execution timing analysis of the L_1/L_2 decoder versus the ETPL algorithm for different number of attacked sensors.

B. Multiple Attacking Sequences

In this example we consider an Unmanned Ground Vehicle (UGV) under different types of sensor attacks. We assume that the UGV moves along straight lines and completely stops before rotating. Under these assumptions, we can describe the dynamics of the UGV by:

$$\begin{bmatrix}\dot{x}\\ \dot{v}\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & \frac{-B}{M}\end{bmatrix}\begin{bmatrix}x\\ v\end{bmatrix} + \begin{bmatrix}0\\ \frac{1}{M}\end{bmatrix}F,$$
$$\begin{bmatrix}\dot{\theta}\\ \dot{\omega}\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & \frac{-B_r}{J}\end{bmatrix}\begin{bmatrix}\theta\\ \omega\end{bmatrix} + \begin{bmatrix}0\\ \frac{1}{J}\end{bmatrix}T,$$

where x, v, θ, ω are the states of the UGV corresponding to position, linear velocity, angular position and angular velocity, respectively. The parameters M, J, B, B_r denote denote the mechanical mass, inertia, translational friction coefficient and the rotational friction coefficient. The inputs to the UGV are the force F and the torque T. The UGV is equipped with a GPS sensor which measures the UGV position, two motor encoders which measure the translational velocity and an IMU which measures both rotational velocity and rotational position. The resulting output equation is:

$$y = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\ v\\ \theta\\ \omega\end{bmatrix} + \begin{bmatrix}\psi_1\\ \psi_2\\ \psi_3\\ \psi_4\\ \psi_5\end{bmatrix},$$

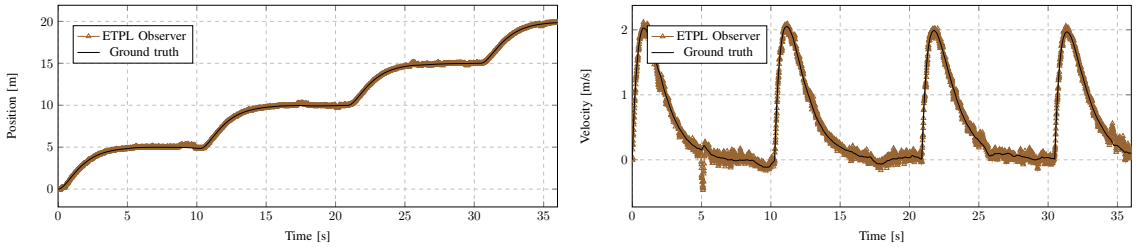
where ψ_i is the measurement noise of the $i$the sensor which is assumed to be gaussian with zero norm and finite covariance.

By applying Proposition 3.3 for different values of s, we conclude that the UGV can be resilient only to one attack on any of the two encoders. Attacking any other sensor precludes the system from being $2s$-sparse observable.

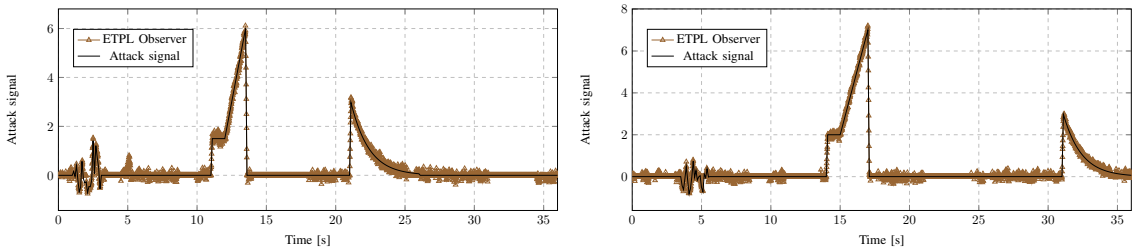
Fig. 2 shows the performance of the proposed algorithms under different attacks on the UGV motor encoders. The attacker alternates between corrupting the left and the right encoder measurements as shown in Fig. 2(b). Three different types of attacks are considered. First, the attacker corrupts the sensor signal with random noise. The next attack consists of a step function followed by a ramp. Finally a replay-attack is mounted by replaying the previously measured UGV velocity.

The UGV vehicle goal is to move 5m along a straight line, stop and perform a 90^o rotation and repeat this pattern 3 times until it traces a square and returns to its original position and orientation. In Figures 2(a) and 2(b) we show the result of using the ETPL observer to reconstruct the state under sensor attacks. The reconstructed state is used by a linear feedback tracking controller forcing the UGV to track the desired square trajectory. These figures show that both algorithms are able to successfully reconstruct the state and hence the UGV is able to reach its goal despite the attacks.

Recall that the attack model in Section II requires the set of attacked sensors to remain constant over time. However, Figure 2 shows the proposed algorithms correctly constructing the state even though this assumption is violated. This is due to the fact that the period during which only one sensor is attacked is sufficiently long compared with the time it takes for the algorithms to converge.



(a) Reconstructed states (position and velocity) versus ground truth.



(b) Reconstructed attack on left and right encoders versus ground truth.

Fig. 2. Performance of the UGV controller in the cases where no attack takes place versus the case where the attack signal is applied to the UGV encoders. The objective is to move 5 m, stop and perform a 90^o rotation and repeat this pattern to follow a square path. The controller uses ETPG algorithm and ETPL Observer to reconstruct the UGV states. We show the linear position, linear velocity (top), and the reconstruction of the attack signal (bottom).

VI. Conclusions

In this paper we considered the problem of designing a Luenberger-like observer to reconstruct the state of a system whose sensor measurements are attacked by an adversary. We introduced the notion of s-sparse observability and characterized the solvability of the observer design problem using this notion. By adopting an event-triggered approach we proposed an observer that is an order of magnitude computationally more efficient than the existing solutions to the considered problem reported in [14], [15], [16].

References

[1] K. Astrom and B. Bernhardsson, "Comparison of riemann and lebesgue sampling for first order stochastic systems," in *Proceedings of the 41st IEEE Conference on Decision and Control*, vol. 2, Dec 2002, pp. 2011–2016 vol.2.

[2] K.-E. Årzén, "A simple event-based pid controller," in *Proceedings of the 14th World Congress of IFAC*, March 1999, pp. 423–428.

[3] P. Tabuada, "Event-triggered real-time scheduling of stabilizing control tasks," *IEEE Transactions on Automatic Control*, vol. 52, no. 9, pp. 1680–1685, Sept 2007.

[4] W. P. M. H. Heemels, J. H. Sandee, and P. P. J. Van Den Bosch, "Analysis of event-driven controllers for linear systems," *International Journal of Control*, vol. 81, no. 4, pp. 571–590, 2008.

[5] W. Heemels, K. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control," in *Proceedings of the 51st IEEE Conference on Decision and Control (CDC)*, Dec 2012, pp. 3270–3285.

[6] D. Han, Y. Mo, J. Wu, S. Weerakkody, B. Sinopoli, and L. Shi, "Stochastic Event-triggered Sensor Schedule for Remote State Estimation," *ArXiv e-prints*, Feb. 2014.

[7] L. Li and M. Lemmon, "Performance and average sampling period of sub-optimal triggering event in event triggered state estimation," in *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, Dec 2011, pp. 1656–1661.

[8] L. Li, M. Lemmon, and X. Wang, "Event-triggered state estimation in vector linear processes," in *American Control Conference (ACC)*, June 2010, pp. 2138–2143.

[9] M. Rabi, G. V. Moustakides, and J. S. Baras, "Adaptive sampling for linear state estimation," *SIAM Journal of Control and Optimization*, vol. 50, no. 2, pp. 672–702, Mar. 2012.

[10] M. Rabi, K. Johansson, and M. Johansson, "Optimal stopping for event-triggered sensing and actuation," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, Dec 2008, pp. 3607–3612.

[11] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *IEEE Security and Privacy Magazine*, vol. 9, no. 3, pp. 49–51, 2011.

[12] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," in *Proceedings of the 16th ACM conference on Computer and communications security*, ser. CCS '09. New York, NY, USA: ACM, 2009, pp. 21–32.

[13] Y. Shoukry, P. D. Martin, P. Tabuada, and M. B. Srivastava, "Non-invasive spoofing attacks for anti-lock braking systems," in *Workshop on Cryptographic Hardware and Embedded Systems*, ser. G. Bertoni and J.-S. Coron (Eds.): CHES 2013, LNCS 8086. International Association for Cryptologic Research, 2013, pp. 55–72.

[14] H. Fawzi, P. Tabuada, and S. Diggavi, "Secure estimation and control for cyber-physical systems under adversarial attacks," *IEEE Transactions on Automatic Control*, vol. 59, no. 6, pp. 1454–1467, June 2014.

[15] ——, "Secure state-estimation for dynamical systems under active adversaries," in *49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, sept. 2011, pp. 337–344.

[16] ——, "Security for control systems under sensor and actuator attacks," in *IEEE 5st Annual Conference on Decision and Control (CDC)*, Nov. 2012, pp. 3412–3417.

[17] M. Pajic, J. Weimer, N. Bezzo, P. Tabuada, O. Sokolsky, I. Lee, and G. Pappas, "Robustness of attack-resilient state estimators," in *ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, Apr 2014.

[18] Y. Shoukry and P. Tabuada, "Event-Triggered State Observers for Sparse Sensor Noise/Attacks," *ArXiv e-prints*, Sep. 2013. [Online]. Available: http://arxiv.org/abs/1309.3511

[19] G. Raskutti, M. J. Wainwright, and B. Yu, "Restricted eigenvalue properties for correlated gaussian designs," *J. Mach. Learn. Res.*, vol. 99, pp. 2241–2259, August 2010.

[20] E. Candes and M. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[21] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[22] T. Blumensath, "Accelerated iterative hard thresholding," *Signal Processing*, vol. 92, no. 3, pp. 752–756, 2012.