### Title

Sound and complete state estimation for linear dynamical systems under sensor attacks using Satisfiability Modulo Theory solving




### ISBN

9781479986842



### Authors

Shoukry, Y
Puggelli, A
Nuzzo, P
et al.



### Publication Date

2015-07-28

### DOI

10.1109/ACC.2015.7171925

Peer reviewed

# Sound and Complete State Estimation for Linear Dynamical Systems Under Sensor Attacks Using Satisfiability Modulo Theory Solving

Yasser Shoukry, Alberto Puggelli, Pierluigi Nuzzo,
Alberto L. Sangiovanni-Vincentelli, Sanjit A. Seshia, and Paulo Tabuada

***Abstract*— We address the problem of detecting and mitigating the effect of malicious attacks on the sensors of a linear dynamical system. We develop a novel, efficient algorithm that uses a Satisfiability Modulo Theory approach to isolate the compromised sensors and estimate the system state despite the presence of the attack, thus harnessing the intrinsic combinatorial complexity of the problem. Simulation results show that our algorithm compares favorably with alternative techniques, with respect to both runtime and estimation error.**

## I. Introduction

This paper addresses the problem of detecting and mitigating the effects of an adversarial corruption of sensory data in a dynamical system. Such situation can occur, for instance, whenever an adversarial attacker has access to the software that processes sensor information [1], is able to spoof data packets holding sensor data exchanged over a network [2], or can directly tamper with the sensor environment [3]. In particular, we distinguish between two coupled challenges: (i) the ability to detect and isolate the sensors under attack, and (ii) the ability to estimate the state of the physical system from corrupted measurements.

In recent years, multiple solutions have been reported to the secure state estimation problem for linear dynamical systems. In addition to fast execution time, a key requirement of an estimation algorithm is to provide formal guarantees of *soundness* (i.e., if the algorithm returns a state estimate, then the system lies indeed in that state) and *completeness* (i.e., if the system state can be estimated, then the algorithm is indeed able to find such an estimate). One approach to secure state estimation is to formulate the problem as a non-convex $l_0$ minimization problem when sensor measurements are noiseless [4], or when they are affected by noise [5]. To improve the efficiency of the estimation algorithm, the $l_0$ minimization problem is then relaxed into a convex $l_r/l_1$ problem, which can be solved in polynomial time. Nonetheless, due to the convex relaxation step, this approach is not sound, i.e., it may return an incorrect estimate. To avoid the relaxation step altogether, an alternative formulation of the state estimation problem, which can be solved by using time-efficient projected gradient techniques, has also been proposed [6], [7]. However, restrictive conditions must be satisfied by the system structure to guarantee the soundness and completeness of this algorithm, thus limiting its applicability.

A technique that relies on an on-line learning mechanism based on approximate envelopes of collected data has also been recently reported [8]. The envelopes are used to detect any abnormal behavior without assuming any knowledge of the dynamical system model. However, no formal guarantee of completeness is provided regarding the ability to detect and mitigate attacks. Finally, the work reported in [9], [10] provide a suite of sound and complete algorithms to generate fault-monitor filters and observers, which can be used to detect the existence of an attack. However, if only an upper bound on the cardinality of the attacked sensors is available, the number of needed monitors is combinatorial in the size of the attacked sensors, which might hinder the scalability of the approach.

In this work, we resort to techniques from formal methods to develop a *sound and complete* algorithm that can *efficiently* handle the combinatorial complexity of the state estimation problem. We show that the state estimation problem can be cast as a satisfiability problem for a formula, including logic and pseudo-Boolean constraints on Boolean variables as well as convex constraints on real variables. The Boolean variables model the presence (or absence) of an attack, while the convex constraints capture properties of the system state. We then show how this satisfiability problem can be efficiently solved using the *Satisfiability Modulo Theories* (SMT) paradigm [11], specifically adapted to convex constraint solving [12], to provide both the attacked sensors and the state estimate. To improve the execution time of our decision procedure, we equip the convex constraint solver of our SMT-based algorithm with heuristics that can exploit the specific geometry of the state estimation problem. Finally, we compare the performance of our approach against other algorithms via numerical experiments.

The rest of this paper is organized as follows. Section II introduces the formal setup for the problem under consideration. The main contributions of this paper – the introduction of an efficient SMT-based detector and the characterization

Y. Shoukry and P. Tabuada with Electrical Engineering Department, UCLA, {yshoukry, tabuada}@ucla.edu. A. Puggelli, P. Nuzzo, A. L. Sangiovanni-Vincentelli, and S. A. Seshia are with Electrical Engineering and Computer Science Department, UC Berkeley, {puggelli,nuzzo,alberto,sseshia}@eecs.berkeley.edu.

This work was partially sponsored by the NSF award 1136174, by DARPA under agreement number FA8750-12-2-0247, by TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA, and by the NSF project ExCAPE: Expeditions in Computer Augmented Program Engineering (award 1138996). 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 NSF, DARPA or the U.S. Government.

of its soundness and completeness – are presented in Section III. Numerical results are then shown in Section IV. Finally, Section V concludes the paper and discusses new research directions. For the sake of brevity, we focus on the ideal case when the all sensors are noise-free and we omit all the proofs in this paper. For further details, we refer the reader to our extended report [13], where we discuss the more general case of noisy sensors along with all the proofs of our results.

## II. The Secure State Estimation Problem

We provide a mathematical formulation of the state estimation problem considered in this paper, and discuss the conditions for the existence and uniqueness of the solution.

### A. Notation

The symbols $\mathbb{N}, \mathbb{R}$ and $\mathbb{B}$ denote the sets of natural, real, and Boolean numbers, respectively. The symbols $\wedge$ and $\neg$ denote the logical AND and logical NOT operators, respectively. The support of a vector $x \in \mathbb{R}^n$, denoted by $\text{supp}(x)$, is the set of indices of the non-zero elements of $x$. Similarly, the complement of the support of a vector $x$ is denoted by $\overline{\text{supp}(x)} = \{1, \dots, n\} \setminus \text{supp}(x)$. If $S$ is a set, $|S|$ is the cardinality of $S$. We call a vector $x \in \mathbb{R}^n$ $s$-sparse, if $x$ has at most $s$ nonzero elements, i.e., if $|\text{supp}(x)| \leq s$. For a vector $x \in \mathbb{R}^n$, we denote by $\|x\|_2$ the 2-norm of $x$ and by $\|M\|_2$ the induced 2-norm of a matrix $M \in \mathbb{R}^{m \times n}$. We also denote by $M_i \in \mathbb{R}^{1 \times n}$ the $i$th row of $M$. For the set $\Gamma \subseteq \{1, \dots, m\}$, 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 $\Gamma$. Then, $M_{\overline{\Gamma}} \in \mathbb{R}^{(m-|\Gamma|) \times n}$ is the matrix obtained from $M$ by removing the rows indexed by the set $\Gamma$, $\overline{\Gamma}$ representing the complement of $\Gamma$. For example, if $m = 4$, and $\Gamma = \{1, 2\}$, we have:

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

### B. System and Attack Model

We consider a system under sensor attacks of the form:

$$\Sigma_a \quad \begin{cases} x^{(t+1)} &= Ax^{(t)} + Bu^{(t)}, \\ y^{(t)} &= Cx^{(t)} + a^{(t)} \end{cases} \tag{II.1}$$

where $x^{(t)} \in \mathbb{R}^n$ is the system state at time $t \in \mathbb{N}$, $u^{(t)} \in \mathbb{R}^m$ is the system input, and $y^{(t)} \in \mathbb{R}^p$ is the observed output. Matrices $A, B$, and $C$ represent the system dynamics and have appropriate dimensions. The attack vector $a^{(t)} \in \mathbb{R}^p$ is an $s$-sparse vector modeling how an attacker corrupted the sensor measurements at time $t$. If sensor $i \in \{1, \dots, p\}$ is attacked then the $i$th element in $a^{(t)}$ is non-zero; otherwise the $i$th sensor is not attacked. Hence, $s$ describes the number of attacked sensors. Note that we make no assumptions on the vector $a^{(t)}$, apart from 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, although we assume the knowledge of an upper bound $\overline{s}$ on the maximum number of sensors that can be attacked.

### C. Problem Formulation

To formulate the state estimation problem, we assume the state is reconstructed from a set of $\tau \in \mathbb{N}$ measurements, where $\tau \leq n$ is selected to guarantee that the system observability matrix, as defined below, has full rank. Therefore, we can arrange the outputs from the $i$th sensor at different time instants as follows:

$$\widetilde{Y}_i^{(t)} = \mathcal{O}_i x^{(t-\tau+1)} + E_i^{(t)} + F_i U^{(t)},$$

where:

$$\widetilde{Y}_i^{(t)} = \begin{bmatrix} y_i^{(t-\tau+1)} \\ y_i^{(t-\tau)} \\ \vdots \\ y_i^{(t)} \end{bmatrix}, E_i^{(t)} = \begin{bmatrix} a_i^{(t-\tau+1)} \\ a_i^{(t-\tau)} \\ \vdots \\ a_i^{(t)} \end{bmatrix},$$

$$U^{(t)} = \begin{bmatrix} u^{(t-\tau+1)} \\ u^{(t-\tau+2)} \\ \vdots \\ u^{(t)} \end{bmatrix}, \mathcal{O}_i = \begin{bmatrix} C_i \\ C_i A \\ \vdots \\ C_i A^{\tau-1} \end{bmatrix},$$

$$F_i = \begin{bmatrix} 0 & 0 & \dots & 0 & 0 \\ C_i B & 0 & \dots & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ C_i A^{\tau-2} B & C_i A^{\tau-3} B & \dots & C_i B & 0 \end{bmatrix}.$$

Since all the inputs in $U^{(t)}$ are known, we can simplify the output equation as:

$$Y_i^{(t)} = \mathcal{O}_i x^{(t-\tau+1)} + E_i^{(t)}, \tag{II.2}$$

where $Y_i^{(t)} = \widetilde{Y}_i^{(t)} - F_i U^{(t)}$. We also define:

$$Y^{(t)} = \begin{bmatrix} Y_1^{(t)} \\ \vdots \\ Y_p^{(t)} \end{bmatrix}, \quad E^{(t)} = \begin{bmatrix} E_1^{(t)} \\ \vdots \\ E_p^{(t)} \end{bmatrix}, \quad \mathcal{O} = \begin{bmatrix} \mathcal{O}_1 \\ \vdots \\ \mathcal{O}_p \end{bmatrix} \tag{II.3}$$

to denote, respectively, the vector of outputs, attacks and observability matrices related to all sensors over the same time window of length $\tau$. Here, with some abuse of notation, $Y_i, E_i$ and $\mathcal{O}_i$ are used to denote the $i$th block of $Y, E$ and $\mathcal{O}$. We also denote by $Y_\Gamma, E_\Gamma$, and $\mathcal{O}_\Gamma$ the blocks indexed by the elements in the set $\Gamma$.

### D. Problem Statement

For each sensor, we define a binary indicator variable $b_i \in \mathbb{B}$ such that $b_i = 0$ when the $i$th sensor is attack-free and $b_i = 1$ otherwise. Based on the formulation in Sec. II-C, our goal is to find $x^{(t-\tau+1)}$ in (II.2), knowing that:

1) if a sensor is attack-free (i.e., $b_i = 0$), then (II.2) reduces to $Y_i^{(t)} - \mathcal{O}_i x^{(t-\tau+1)} = 0$;
2) the maximum number of attacked sensors is $\overline{s}$.

Therefore, using the binary variables $b_i$, we can pose the problem of secure state estimation as follows.

*Problem 2.1:* **(Secure State Estimation)** For the linear control system under attack $\Sigma_a$ (defined by (II.1)), construct

an estimate $\eta = (x, b) \in \mathbb{R}^n \times \mathbb{B}^p$ such that $\eta \models \phi$, i.e., $\eta$ satisfies the formula $\phi$, where $\phi$ is defined as:

$$\phi ::= \bigwedge_{i=1}^{p} \left( \neg b_i \Rightarrow \|Y_i - \mathcal{O}_i x\|_2 = 0 \right) \wedge \left( \sum_{i=1}^{p} b_i \leq \overline{s} \right).$$

In Problem 2.1, $Y_i$ and $\mathcal{O}_i$ are the vectors of outputs and the observability matrix related to sensor $i$ as defined in Sec. II-C. The first conjunction of constraints requires $(Y_i - \mathcal{O}_i x)$ is 0 if sensor $i$ is attack-free. The last constraint enforces an upper bound on the number of attacked sensors. We drop the time $t$ argument in Problem 2.1 since the satisfiability problem is to be solved at every time instance. Although we reconstruct a delayed version of the state $x^{(t-\tau+1)}$, we can always reconstruct the current state $x^{(t)}$ from $x^{(t-\tau+1)}$ by recursively rolling the dynamics forward in time.

The secure state estimation problem 2.1 does not ask for the minimal number of attacked sensors for which the estimated state matches the measured output. That is, if $b^*$ is the vector of indicator variables characterizing the actual attack, any assignment $\eta = (x, b) \models \phi$ with $\text{supp}(b^*) \subseteq \text{supp}(b)$ is a valid solution for Problem 2.1. Therefore, it is useful to modify Problem 2.1 to ask for the minimal number of attacked sensors that explains the collected measurements as follows.

*Problem 2.2:* **(Minimal Attack Support)** For the linear control system under attack $\Sigma_a$ (defined by (II.1)), construct the estimate $\eta = (x, b) \in \mathbb{R}^n \times \mathbb{B}^p$ obtained as the solution of the optimization problem:

$$\min_{(x,b)\in\mathbb{R}^n\times\mathbb{B}^p} \sum_{i=1}^{p} b_i \quad \text{s.t.} \bigwedge_{i=1}^{p} \left( \neg b_i \Rightarrow \|Y_i - \mathcal{O}_i x\|_2 = 0 \right).$$

It is straightforward to show that the solution to Problem 2.2 can be obtained by performing a binary search over $s$ and invoking a solver for Problem 2.1 at each step, starting with $s = \overline{s}$ and then decreasing $\overline{s}$ until Problem 2.1 becomes unfeasible or $\overline{s} = 0$. Since any solution of (II.2) must necessarily satisfy the constraints of Problem 2.1, such a procedure will terminate by returning the solution with the minimal attack support. We denote this solution as *minimal support solution*. In the reminder of the paper, we will focus on the analysis of the feasibility problem 2.1, since a solution to the optimization problem 2.2 can be obtained by solving a sequence of instances of Problem 2.1.

In Sec. II-E, we discuss the conditions for the uniqueness of the minimal support solution of Problem 2.2. However, we first recall that the satisfiability problem over real numbers, and specifically over $\mathbb{R}^n$, is inherently intractable, i.e., decision algorithms for formulas with non-linear polynomials already suffer from high complexity [14], [15]. Moreover, linear programming and convex programming solvers usually perform floating point (hence inexact) calculations, which may be inadequate for some applications. Therefore, to provide formal guarantees about correctness of Problem 2.1, we resort to the notion of $\delta$-completeness previously introduced in [16].

*Definition 2.3:* **Soundness and Completeness of Decision Algorithms for Problem 2.1** Let a minimal solution $\eta^* = (x^*, b^*)$ (the true state and indicator variables) exist for Problem 2.1. Then, a solution $\eta = (x, b) \models \phi$ is said to $\delta$-satisfy $\phi$ (or $\delta$-SAT for short) if $\text{supp}(b^*) \subseteq \text{supp}(b)$ and $\|x^* - x\|_2^2 \leq \delta$ for some $\delta \in \mathbb{R}$. Moreover, an algorithm that solves Problem 2.1 is said to be $\delta$-*complete* if it returns a $\delta$-SAT solution.

Definition 2.3 asks for an algorithm which terminates and returns a solution $\eta = (x, b)$ that is correct (up to the tolerance $\delta$). Hence, a $\delta$-*complete* decision algorithm in the sense of Definition 2.3 is also ($\delta$-)sound since, if it returns a solution $\eta$, $\eta$ is actually a $\delta$-SAT solution.

### *E. Uniqueness of Minimal Support Solutions*

To characterize the existence and uniqueness of solutions to Problem 2.2, we recall the notion of $s$-sparse observability [7].

*Definition 2.4:* **($s$-Sparse Observable System)** The linear control system $\Sigma_a$, defined by (II.1), is said to be $s$-sparse observable if for every set $\Gamma \subseteq \{1, \ldots, p\}$ with $|\Gamma| = s$, the system $\Sigma_{\overline{\Gamma}}$ is observable, where $\Sigma_{\overline{\Gamma}}$ is defined as:

$$\Sigma_{\overline{\Gamma}} \quad \begin{cases} x^{(t+1)} &= Ax^{(t)} + Bu^{(t)}, \qquad t \in \mathbb{N} \\ y^{(t)} &= C_{\overline{\Gamma}} x^{(t)} \end{cases}. \tag{II.4}$$

In other words, a system is $s$-sparse observable if it is observable from any choice of $p - s$ sensors. For $2\overline{s}$-sparse observable systems, the following result holds.

*Theorem 2.5:* **(Existence and Uniqueness of the Solution)**[Theorem III.2 in [7]] Problem 2.2 admits a unique solution $\eta^* = (x^*, b^*)$ if and only if the dynamical system $\Sigma_a$ defined by (II.1) is $2\overline{s}$-sparse observable.

Problem 2.2 can be solved by transforming it into a Mixed Integer-Quadratic Program (MIQP) as follows:

$$\min_{(x,b)\in\mathbb{R}^n\times\mathbb{B}^p} \sum_{i=1}^{p} b_i \tag{II.5}$$
$$\text{s.t. } \|Y_i - \mathcal{O}_i x\|_2 \leq M b_i \qquad 1 \leq i \leq p,$$

where $M \in \mathbb{R}$ is a constant that should be "big" enough to make each constraint not active when $b_i = 1$. The relaxation in (II.5) is typically used to express constraints including logical implications [17]; however, in this case, the choice of $M$ affects the completeness of the approach. For example, since $\|Y_i - \mathcal{O}_i x\|_2$ is ultimately bounded by the power of the attack $\|E_i\|_2$, a value of $M < \|E_i\|_2 = \|Y_i - \mathcal{O}_i x\|_2$, can produce an incorrect result. While a physical sensor has a bounded dynamic range in practice, such a bound is not known *a priori* in our formulation, which makes no assumptions on $\|E_i\|_2$. Therefore, completeness of the MIQP formulation (II.5) cannot be guaranteed in general.

In the sequel, we detail an algorithm which exploits the geometry of the state estimation problem and the convexity

of the quadratic constraints to generate a provably correct solution using the SMT paradigm. We then compare the SMT-based solution with the MIQP formulation in (II.5) using a commercial MIQP solver.

## III. SMT-Based Detector

To decide whether a combination of Boolean and convex constraints is satisfiable, we construct the detection algorithm Imhotep[1]-SMT using the *lazy* SMT paradigm [11]. As in the CalCS solver [12], our decision procedure combines a SAT solver (SAT-Solve) and a theory solver ($\mathcal{T}$-Solve) for convex constraints on real numbers. The SAT solver efficiently reasons about combinations of Boolean and pseudo-Boolean constraints, using the David-Putnam-Logemann-Loveland (DPLL) algorithm [18], to suggest possible assignments for the convex constraints. The theory solver checks the consistency of the given assignments, and provide the reason for the conflict, a *certificate*, or a counterexample, whenever inconsistencies are found. Each certificate results in learning new constraints which will be used by the SAT solver to prune the search space. The complex detection and mitigation decision task is thus broken into two simpler tasks, respectively, over the Boolean and convex domains. We denote the approach as lazy, because it checks and learns about consistency of convex constraints only when necessary, as detailed below.

### A. Overall Architecture

As illustrated in Algorithm 1, we start by mapping each convex constraint to an auxiliary Boolean variable $c_i$ to obtain the following (pseudo-)Boolean satisfiability problem:

$$\phi_B ::= \left( \bigwedge_{i\in\{1,\dots,p\}} \neg b_i \Rightarrow c_i \right) \wedge \left( \sum_{i\in\{1,\dots,p\}} b_i \leq \overline{s} \right)$$

where $c_i = 1$ if $\|Y_i - \mathcal{O}_i x\|_2 \leq 0$ is satisfied, and zero otherwise. By only relying on the Boolean structure of the problem, SAT-Solve returns an assignment for the variables $b_i$ and $c_i$ (for $i = 1, \dots, p$), thus hypothesizing which sensors are attack-free, hence which convex constraints should be jointly satisfied.

This Boolean assignment is then used by $\mathcal{T}$-Solve to determine whether there exists a state $x \in \mathbb{R}^n$ which satisfies all the convex constraints related to the unattacked sensors, i.e., $\|Y_i - \mathcal{O}_i x\|_2 \leq 0$ for $i \in \overline{\text{supp}}(b)$. If $x$ is found, Imhotep-SMT terminates with `SAT` and provides the solution $(x, b)$. Otherwise, the `UNSAT` certificate $\phi_{\text{cert}}$ is generated in terms of new Boolean constraints, explaining which sensor measurements are conflicting and may be under attack. This augmented Boolean problem is then fed back to SAT-Solve to produce a new assignment. The sequence of new SAT queries is then repeated until $\mathcal{T}$-Solve terminates with `SAT`.

[1]Imhotep (pronounced as "emmo-tepp") was an ancient Egyptian polymath who is considered to be the earliest known architect, engineer and physician of the early history. He is famous for the design of the oldest pyramid in Egypt, the Pyramid of Djoser (the Step Pyramid) at Saqqara, Egypt, 2630 – 2611 BC.

**Algorithm 1** Imhotep-SMT

status := `UNSAT`;
$\phi_B := \left(\bigwedge_{i\in\{1,\dots,p\}} \neg b_i \Rightarrow c_i\right) \wedge \left(\sum_{i\in\{1,\dots,p\}} b_i \leq \overline{s}\right)$;
**while** status == `UNSAT` **do**
  $(b, c)$ := SAT-Solve($\phi_B$);
  (status, $x$) := $\mathcal{T}$-Solve.Check($\overline{\text{supp}}(b)$);
  **if** status == `UNSAT` **then**
    $\phi_{\text{cert}}$ := $\mathcal{T}$-Solve.Certificate($b, x$);
    $\phi_B := \phi_B \wedge \phi_{\text{cert}}$;
**return** $\eta = (x, b)$;

**Algorithm 2** $\mathcal{T}$-Solve.Check($\mathcal{I}$)

**Solve:** $x := \arg\min_{x\in\mathbb{R}^n} \|Y_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}} x\|_2^2$
**if** $\|Y_{\mathcal{I}} - \mathcal{O}_{\mathcal{I}} x\|_2^2 = 0$ **then**
  status = `SAT`;
**else**
  status = `UNSAT`;
**return** (status, $x$);

By the $2\overline{s}$-sparse observability condition (Theorem 2.5), the existence and uniqueness of a solution to Problem 2.2 is guaranteed, hence Algorithm 1 will always terminate. However, to help the SAT solver quickly converge towards the correct assignment, a central problem in lazy SMT solving is to generate succinct explanations whenever conjunctions of convex constraints are unfeasible, possibly highlighting the minimum set of conflicting assignments. The rest of this section will then focus on the implementation of the two main tasks of $\mathcal{T}$-Solve, namely, (i) checking the satisfiability of a given assignment ($\mathcal{T}$-Solve.Check), and (ii) generating succinct UNSAT certificates ($\mathcal{T}$-Solve.Certificate).

### B. Satisfiability Checking

Given an assignment of the Boolean variables $b$, with $|\text{supp}(b)| \leq \overline{s}$, the following condition holds:

$$\min_{x\in\mathbb{R}^n} \left\| Y_{\overline{\text{supp}}(b)} - \mathcal{O}_{\overline{\text{supp}}(b)} x \right\|_2^2 = 0 \qquad \text{(III.1)}$$

if and only if $(x, b)$ is a solution of Problem 2.1. This is a direct consequence of the $2\overline{s}$-sparse observability property discussed in Section II. The preceding *unconstrained least-squares optimization* problem can be solved very efficiently, thus leading to Algorithm 2. In practical implementations, (III.1) should actually be replaced with:

$$\min_{x\in\mathbb{R}^n} \left\| Y_{\overline{\text{supp}}(b)} - \mathcal{O}_{\overline{\text{supp}}(b)} x \right\|_2^2 \leq \epsilon,$$

where $\epsilon > 0$ is the solver tolerance, accounting for numerical errors. However, for the sake of clarity, we focus here on the case when $\epsilon$ is zero. For a full treatment of correctness in the presence of numerical errors, we refer the reader to [13].

### C. Generating UNSAT Certificates

Whenever $\mathcal{T}$-Solve.Check provides `UNSAT`, a certificate could be easily generated as follows:

$$\phi_{\text{triv-cert}} = \sum_{i\in\overline{\text{supp}}(b)} b_i \geq 1, \qquad \text{(III.2)}$$

indicating that at least one of the sensors, which was initially assumed as attack-free (i.e. for which $b_i = 0$), is actually under attack; one of the $b_i$ variables should then be set to one in the next assignment of the SAT solver. However, such *trivial certificate* $\phi_{\text{triv-cert}}$ does not provide much information, since it only excludes the current assignment from the search space, and can lead to exponential execution time [13].

### D. Enhancing the Execution Time

To generate a compact Boolean constraint that explains a conflict, we aim to find a small set of sensors that cannot all be attack-free. We first compute the (normalized) residuals $r_i$ for all $i \in \mathcal{I}$, as defined in Algorithm 3, and sort them in ascending order. We then pick the $p - 2\bar{s}$ minimum (normalized) residuals indexed by $\mathcal{I}\_min\_r$, and search for one more affine subspace that leads to a conflict with the affine subspaces indexed by $\mathcal{I}\_min\_r$. To do this, we start by solving the same optimization problem as in Algorithm 2, but on the reduced set of affine subspaces indexed by $\mathcal{I}_{temp} = \mathcal{I}\_min\_r \cup \mathcal{I}\_max\_r$, where $\mathcal{I}\_max\_r$ is the index associated with the affine subspace having the maximal (normalized) residual. If this set of affine subspaces intersect in one point, they are labelled as "non-conflicting", and we repeat the same process by replacing the affine subspace indexed by $\mathcal{I}\_max\_r$ with the affine subspace associated with the second maximal (normalized) residual from the sorted list, till we reach a conflicting set of affine subspaces. Once the set is discovered, we stop by generating the following, more compact, certificate:

$$\phi_{\text{conf-cert}} := \sum_{i \in \mathcal{I}_{temp}} b_i \geq 1.$$

The cardinality of $\mathcal{I}_{temp}$ heavily affects the overall execution time of Algorithm 1: the smaller $|\mathcal{I}_{temp}|$, the more information is learnt and the faster is the convergence of the SAT solver to the correct assignment. For example, a certificate with $|\mathcal{I}_{temp}| = 1$ would identify exactly one attacked sensor at each step, a substantial improvement with respect to the exponential worst-case complexity of the plain SAT problem, which is NP-complete. On the other hand, to generate $\phi_{\text{conf-cert}}$, we only pay the cost of a linear search over $\mathcal{I}$ and, for each step, a least-square optimization problem, which amounts to an overall complexity that is polynomial.

Finally, as a post-processing step, we can further reduce the cardinality of $\mathcal{I}_{temp}$ by exploiting the dimension of the affine subspaces corresponding to the index list. Intuitively, the lower the dimension, the more information is provided by the corresponding sensor. For example, a sensor $i$, for which the dimension of the affine subspace $\mathbb{H}_i$ is $\dim(\mathbb{H}_i) = \dim(\ker \mathcal{O}_i) = 0$ corresponds to only one point $\mathcal{O}_i^{-1} Y_i$. This restricts the search space to the unique point and makes it easier to generate a conflict formula. Therefore, to converge faster towards a conflict, we iterate through the indexes in $\mathcal{I}_{temp}$ and remove at each step the one which corresponds to the affine subspace with the highest dimension until we are left with a reduced index set that is still conflicting.

**Algorithm 3** $\mathcal{T}$-SOLVE.CERTIFICATE-CONFLICT$(\mathcal{I}, x)$

**Compute normalized residuals**
  $r := \bigcup_{i \in \mathcal{I}} \{r_i\}, \quad r_i := \|Y_i - \mathcal{O}_i x\|_2^2 / \|\mathcal{O}_i\|_2^2, \ i \in \mathcal{I}$;
**Sort the residual variables**
  $r\_sorted :=$ sortAscendingly$(r)$;
**Pick the index corresponding to the maximum residual**
  $\mathcal{I}\_max\_r :=$ Index$(r_sorted_{\{|\mathcal{I}|, |\mathcal{I}|-1, \ldots, p-2\bar{s}+1\}})$;
  $\mathcal{I}\_min\_r :=$ Index$(r_sorted_{\{1, \ldots, p-2\bar{s}\}})$;
**Search linearly for the UNSAT certificate**
  status = `SAT`; counter $= 1$;
  $\mathcal{I}\_temp := \mathcal{I}\_min\_r \cup \mathcal{I}\_max\_r_{counter}$;
**while** status == `SAT` **do**
  (status, $x$) := $\mathcal{T}$-SOLVE.CHECK$(\mathcal{I}_temp)$;
  **if** status == `UNSAT` **then**
    $\phi_{\text{conf-cert}} := \sum_{i \in \mathcal{I}\_temp} b_i \geq 1$;
  **else**
    counter := counter + 1;
    $\mathcal{I}\_temp := \mathcal{I}\_min\_r \cup \mathcal{I}\_max\_r_{counter}$;
**[Optional] Sort the rest according to dim**$(ker\{\mathcal{O}\})$
  $\mathcal{I}\_temp2 =$ sortAscendingly$(dim(ker\{\mathcal{O}_{\mathcal{I}_temp}\}))$;
  status = `UNSAT`; counter2 $= |\mathcal{I}\_temp2| - 1$;
  $\mathcal{I}\_temp2 := \mathcal{I}\_temp2_{\{1, \ldots, counter2\}}$;
**while** status == `UNSAT` **do**
  (status, $x$) := $\mathcal{T}$-SOLVE.CHECK$(\mathcal{I}_{temp})$;
  **if** status == `SAT` **then**
    $\phi_{\text{conf-cert}} := \sum_{i \in \mathcal{I}\_temp2_{\{1, \ldots, counter2+1\}}} b_i \geq 1$;
  **else**
    counter2 := counter2 - 1;
    $\mathcal{I}\_temp2 := \mathcal{I}\_temp2_{\{1, \ldots, counter2\}}$;
**return** $\phi_{\text{conf-cert}}$

### E. Soundness and Completeness of Algorithm 1

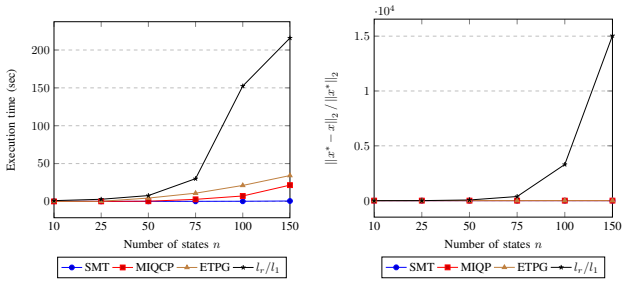
We are now ready to state the main result of this section, which is a direct consequence of our previous results.

*Theorem 3.1:* Let the linear dynamical system $\Sigma_a$ defined in (II.1) be $2\bar{s}$-sparse observable. Let $\epsilon = 0$ be the numerical solver tolerance for Algorithm 2. Algorithm 1 is $\delta$-complete (in the sense of Definition 2.3) with $\delta = 0$. Moreover, the upper bound on the number of iterations of Algorithm 1 is $\binom{p}{p-2\bar{s}+1}$.

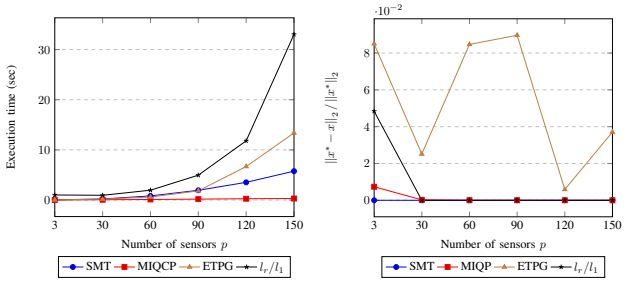
## IV. RESULTS

We developed our theory solver in MATLAB, and interfaced it with the pseudo-Boolean SAT solver SAT4J [19]. All the experiments were executed on an Intel Core i7 3.4-GHz processor with 8 GB of memory. We compared the performance of IMHOTEP-SMT against the MIQP formulation (II.5), the ETPG algorithm [7], and the $l_r/l_1$ decoder [4], with respect to both execution time and estimation error.

The MIQP is solved using the commercial solver GUROBI [20], the ETPG algorithm is implemented in MATLAB, while the $l_r/l_1$ decoder is implemented using the convex solver CVX [21]. Figure 1 reports the numerical results in two test cases. In Figure 1(a), we fix the number




(a) Execution time (left) and estimation error (right) versus number of states $n$ for different algorithms ($p = 20$, $\overline{s} = 5$).




(b) Execution time (left) and estimation error (right) versus number of sensors $p$ for different algorithms ($n = 50$, $\overline{s} = p/2 - 1$).

Fig. 1. Simulation results showing number of iterations, execution time, and estimation error with respect to number of states and number of sensors.

of sensors $p = 20$ and increase the number of system states from $n = 10$ to $n = 150$. In Figure 1(b), we fix the number of states $n = 50$ and increase the number of sensors from $p = 3$ to $p = 150$. In both cases, half of the sensors are attacked. Our algorithm always outperforms both the ETPG and the $l_r/l_1$ approaches and scales nicely with respect to both $n$ and $p$. In particular, as evident from Figure 1(a), increasing $n$ has a small effect on the overall execution time, which reflects the fact that the number of constraints to be satisfied does not depend on $n$. Conversely, as shown in Figure 1(b), as the number of sensors increases, the number of constraints, hence the execution time of our algorithm, also increases. The runtime of the MIQP formulation in (II.5) scales worse than our algorithm with $n$, but better with $p$, because GUROBI can efficiently process many conic constraints (whose number scales with $p$) but is more sensitive to the size of each conic constraint (which scales with $n$). Finally, Figure 1(a) (right) shows that the $l_r/l_1$ decoder reports incorrect results in multiple test cases, because of its lack of soundness, as discussed in Section I.

## V. CONCLUSIONS

We proposed a sound and complete algorithm which adopts the Satisfiability Modulo Theories paradigm to tackle the intrinsic combinatorial complexity of the secure state estimation problem for linear dynamical systems under sensor attacks. Our approach was validated via numerical simulations, and compares favorably with alternative techniques. Future directions include the extension and the characterization of the proposed algorithm for nonlinear and hybrid dynamical systems.

## REFERENCES


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

[2] 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.

[3] 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.

[4] 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.

[5] 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)*, April 2014, pp. 163–174.

[6] Y. Shoukry and P. Tabuada, "Event-triggered projected luenberger observer for linear systems under sensor attacks," in *IEEE 53rd Annual Conference on Decision and Control (CDC)*, Dec. 2014, pp. 3548 – 3553.

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

[8] A. Tiwari, B. Dutertre, D. Jovanović, T. de Candia, P. D. Lincoln, J. Rushby, D. Sadigh, and S. Seshia, "Safety envelope for security," in *Proceedings of the 3rd International Conference on High Confidence Networked Systems*, ser. HiCoNS '14. New York, NY, USA: ACM, 2014, pp. 85–94.

[9] F. Pasqualetti, F. Dorfler, and F. Bullo, "Attack detection and identification in cyber-physical systems," *IEEE Transactions on Automatic Control*, vol. 58, no. 11, pp. 2715–2729, Nov 2013.

[10] M. S. Chong, M. Wakaiki, and J. P. Hespanha, "Observability of linear systems under adversarial attacks," in *The proceedings of the 2015 IEEE American Control conference (ACC)*, 2015.

[11] C. Barrett, R. Sebastiani, S. A. Seshia, and C. Tinelli, *Satisfiability Modulo Theories, Chapter in Handbook of Satisfiability*. IOS Press, 2009.

[12] P. Nuzzo, A. Puggelli, S. A. Seshia, and A. Sangiovanni-Vincentelli, "CalCS: SMT solving for non-linear convex constraints," in *Formal Methods in Computer-Aided Design (FMCAD), 2010*, Oct 2010, pp. 71–79.

[13] Y. Shoukry, P. Nuzzo, A. Puggelli, A. L. Sangiovanni-Vincentelli, S. A. Seshia, and P. Tabuada, "Secure State Estimation For Cyber Physical Systems Under Sensor Attacks: A Satisfiability Modulo Theory Approach," *ArXiv e-prints*, Dec. 2014, [online] http://arxiv.org/abs/1412.4324.

[14] C. W. Brown and J. H. Davenport, "The complexity of quantifier elimination and cylindrical algebraic decomposition," in *Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation*, ser. ISSAC '07. New York, NY, USA: ACM, 2007, pp. 54–60.

[15] G. E. Collins, "Quantifier elimination for real closed fields by cylindrical algebraic decomposition: A synopsis," *SIGSAM Bull.*, vol. 10, no. 1, pp. 10–12, Feb. 1976.

[16] S. Gao, M. Ganai, F. Ivancic, A. Gupta, S. Sankaranarayanan, and E. Clarke, "Integrating icp and lra solvers for deciding nonlinear real arithmetic problems," in *Formal Methods in Computer-Aided Design (FMCAD), 2010*, Oct 2010, pp. 81–89.

[17] W. L. Winston, *Operations Research: Applications & Algorithms*. Thomson Business Press, 2008.

[18] R. Nieuwenhuis, A. Oliveras, and C. Tinelli, "Solving SAT and SAT Modulo Theories: From an abstract Davis–Putnam–Logemann–Loveland procedure to DPLL(T)," *J. ACM*, vol. 53, no. 6, pp. 937–977, Nov. 2006.

[19] D. L. Berre and A. Parrain, "The Sat4j library, release 2.2," *Journal on Satisfiability, Boolean Modeling and Computation*, vol. 7, pp. 59–64, 2010.

[20] "Gurobi Optimizer." [Online]: http://www.gurobi.com/.

[21] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 1.21," http://cvxr.com/cvx, May 2010.