Systems & Control Letters 98 (2016) 44–48

Contents lists available at ScienceDirect

Systems & Control Letters

journal homepage: www.elsevier.com/locate/sysconle

Antagonistic control

Thomas Lipp [a,*], Stephen Boyd [b]

[a] Department of Mechanical Engineering, Stanford University, Stanford, CA 94305, USA
[b] Department of Electrical Engineering, Stanford University, Stanford, CA 94305, USA

HIGHLIGHTS

- Presented a framework for considering an aggressor attacking a system.
- Explained how this could be used to defend a system.
- Presented a convex–concave procedure method for attaining a lower bound on damage.
- Presented an S-procedure method for attaining an upper bound on damage.
- Demonstrated these methods with a simple example.

ARTICLE INFO

Article history:
Received 9 June 2015
Received in revised form 14 August 2016
Accepted 10 October 2016
Available online 4 November 2016

Keywords:
Optimization
Control
Defending against attacks
Convex–concave procedure
S-procedure

ABSTRACT

In antagonistic control we find an input sequence that *maximizes* (or at least makes large) an objective that is minimized in typical control. Applications include designing inputs to attack a control system, worst-case analysis of a control system, and security assessment of a control system. The antagonistic control problem is not convex, and so cannot be efficiently solved. We present here a powerful convex-optimization-based heuristic for antagonistic control, based on the convex–concave procedure, which can be used to find bad, if not the global worst-case, inputs. We also give an S-procedure-based upper bound for antagonistic control, applicable in cases when the objective and constraints can be described by quadratic inequalities, and use this to verify on examples that our method yields inputs very close to the (global) worst-case.

1. Introduction

More and more of the world's control systems, from irrigation systems to the power grid, are transitioning to control by supervisory control and data acquisition (SCADA) systems that transmit their measurements from remote locations over networks. This vast increase in points of entry, and sometimes public nature of these networks, has increased the risk of attack on these systems. Much research has gone into how to protect against various types of attacks including stealthy deception [1,2], false data injection [3,4], denial of service [5], and replay data attacks [6]. It has been demonstrated how these attacks can occur and proposals have been made for how to protect against these attacks [7–10]. While the approaches above are typically concerned with monitoring and detecting attacks, there is an alternative approach which attempts to constrain the system so that a catastrophic event cannot occur [11,12]. Another branch of inquiry models the problem from a game theoretic perspective, often asking how best to invest resources to defend a system [13,14].

We concern ourselves here with a different set of questions. We simply assume that an aggressor has taken control of (parts of) a system, and ask how much damage are they capable of inflicting, or the related question of how quickly can they inflict this damage. Knowing the answers to these questions allows for the design of better protections for the system by showing which vulnerabilities are the most dangerous and therefore deserving of the most protection. Knowing how quickly a catastrophic event can occur post-intrusion gives information about how often monitoring systems need to run and how quickly a response must be taken.

2. Antagonistic control

We consider a discrete-time time-varying linear dynamical system with state $x_t \in \mathbf{R}^n$, input $u_t \in \mathbf{R}^m$, and affine dynamics

$$x_{t+1} = A_t x_t + B_t u_t + c_t, \quad t = 1, \ldots, T-1,$$

where t denotes (discrete) time, and $A_t \in \mathbf{R}^{n\times n}$, $B_t \in \mathbf{R}^{n\times m}$, $c_t \in \mathbf{R}^n$. Here c_t can be known disturbances to the system, or simply model

* Corresponding author.
E-mail address: tlipp@stanford.edu (T. Lipp).

http://dx.doi.org/10.1016/j.sysconle.2016.10.002
0167-6911/

affine dynamics perhaps created by some bias in the system. The states and inputs are constrained, $(x_t, u_t) \in \mathcal{C}_t$, $t = 1, \ldots, T$, where $\mathcal{C}_t \subseteq \mathbf{R}^n \times \mathbf{R}^m$ is convex. The objective has the traditional time-separable form

$$J = \ell_1(x_1, u_1) + \cdots + \ell_T(x_T, u_T),$$

where ℓ_t, $t = 1, \ldots, T$ are the stage cost functions, which we assume are convex. If J were only a function of x_T it would be the familiar terminal cost from model predictive control. This leads to the standard control problem

$$\begin{array}{ll} \text{minimize} & J \\ \text{subject to} & x_{t+1} = A_t x_t + B_t u_t + c_t, \quad t = 1, \ldots, T-1 \\ & (x_t, u_t) \in \mathcal{C}_t, \quad t = 1, \ldots, T, \end{array} \tag{1}$$

where x_t and u_t are optimization variables. (A known or fixed initial state x_1 can be incorporated into $\mathcal{C}_1$.) This is a convex optimization problem that is easily solved, indeed with a complexity that grows only linearly in T. This gives us the best input and state trajectory.

The *antagonistic control problem* is simply the problem of maximizing rather than minimizing J,

$$\begin{array}{ll} \text{maximize} & J \\ \text{subject to} & x_{t+1} = A_t x_t + B_t u_t + c_t, \quad t = 1, \ldots, T-1 \\ & (x_t, u_t) \in \mathcal{C}_t, \quad t = 1, \ldots, T, \end{array} \tag{2}$$

with optimization variables x_t and u_t. This gives us the worst input and state trajectory. We let $p^\star$ be the optimal value of (2), i.e., the worst possible objective value. This problem is evidently not convex, and simple versions of it can be shown to be NP-hard.

In this paper we present a heuristic for approximately solving the antagonistic control problem (2). Our goal is to efficiently find bad, if not necessarily worst case, feasible input and state trajectories.

We will approximately solve the antagonistic control problem using the convex–concave procedure, which we describe in Section 4. This is a standard method for approximately solving a problem in which the objective is a sum of a convex and a concave function. This gives us a bad sequence of inputs, if not necessarily the worst, i.e., a lower bound on $p^\star$. This is useful even if it is not optimal; for example, when it is large, it tells us that an attacker can indeed do grave damage.

For cases in which the objective is quadratic and the constraints are described by quadratic inequalities, we develop an upper bound on $p^\star$ using the S-procedure, which we present in Section 5. Numerical examples show that our two dual methods – the convex–concave procedure for lower bounds and the S-procedure for upper bounds – often yield bounds that are close to each other, which implies that they are each (nearly) globally optimal.

3. Applications

In this section we elaborate on more specific applications of antagonistic control.

The antagonistic control problem arises in a variety of situations. In the simplest case, it can be used by an aggressor who has taken control of (parts of) a control system and wishes to do maximum (or at least very much) damage. In this case, what we call the input u_t is not necessarily the actual control system input, but rather the signal injection points that the aggressor has access to, e.g., a sensor signal that can be manipulated. The dynamics are then not the open-loop dynamics of the control system, but rather the closed-loop dynamics. The constraints $\mathcal{C}_t$ can include not only actual constraints, like actuator limits, but also constraints that an alarm not be triggered, or that the intrusion is unlikely to be detected. (This idea of adding stealth constraints will be addressed in more detail later, when we discuss ambush control.) Thus the problem (2) asks us to find a sequence of actions (which can include modified sensor measurements) that do the most damage (in terms of the objective), while respecting constraints that can include maintaining stealth (to the extent possible).

3.1. Vulnerability monitoring

From the defender's point of view, it is very useful to have a method that can (approximately) solve the antagonistic control problem (2). This can be used to do (approximate) worst-case analysis, or to analyze or improve defenses. For example, we can solve the antagonistic control problem, starting from the current state (with some specific set of inputs taken over by the attacker) and use $p^\star$ as a measure of the current system vulnerability. This can be displayed in real-time, with a warning issued if the value of $p^\star$ gets too large. Antagonistic control can be used to monitor the current safety or vulnerability of the system. This analysis can be carried out for different values of T (the horizon), and different assumptions about which subsystems have been taken over by the attacker. All of the values can be monitored in real-time.

In one variation on this, we can take $\ell_t = 0$ for $t = 1, \ldots, T-1$, and ℓ_T is such that $\ell_T(x_T, u_T) \geq 1$ (say) corresponds to system failure or destruction. By solving the antagonistic control problem for different values of T, we can find the smallest value $T^\star$ for which $p^\star \geq 1$, and this tells us the minimum time it would take an attacker to destroy the system. This is of course a function of the current state. If $T^\star$ is large, we have time (to react) if an attack occurs; if $T^\star$ is small, an attack could destroy the system quickly. We can interpret $T^\star$ as the vulnerability time of the current state.

3.2. Security assessment

Rather than actively monitoring the system, antagonistic control can be used a priori to detect the vulnerabilities in a system and drive the focus of defenses to the appropriate subsystems. A typical system will consist of many sensors and actuators many of which are on subsystems that are isolated from each other. The cost to the aggressor of controlling additional sensors and actuators increases as more and more subsystems are involved. Therefore the aggressor would like to gain control of as few sensors and actuators as possible. We can help secure our systems by identifying subsets of critical systems from which an aggressor can easily do much damage, and defending them robustly or isolating them to force the aggressor to gain control of several subsystems.

To carry out this assessment, we simply use antagonistic control using an appropriate model for each specific subset of subsystems that are taken over by an attacker, or for different configurations of alarms or warning systems that we might install. We derive the dynamics for the system with various actuators and sensors made available to the aggressor (that is represented in u) and determine the relative values of J that can be achieved under different configurations. Knowing which configurations allow for small and large values of J reveals which subsets of injection points it would be detrimental for an aggressor to take control of, and which subsets of points give the aggressor limited control authority. Resources can then be allocated to defending those systems (or subsets of systems) which are critical, freeing resources from those systems whose loss would be less harmful.

3.3. Ambush control

We now take on the role of the aggressor and consider a specific instance of the antagonistic control problem (2), which we call ambush control. In ambush control an aggressor manipulates the system by choosing the inputs u_t, with the requirement that these manipulations remain undetected (or probably undetected) until time $T^{\text{det}} < T$ (the detection time). We include these stealth constraints in the constraints $\mathcal{C}_1, \ldots, \mathcal{C}_{T^{\text{det}}}$.

The inputs designed in this case can be interpreted as an ambush that occurs at time $t = T^{\text{det}}$. Actions taken before that time are required to be (probably) undetectable; they are used to set the system state up so that once the attack is detected, at time $t = T^{\text{det}}$, much damage can be done quickly.

4. Convex–concave procedure

The convex–concave procedure is a powerful heuristic that can be used to find bad controls (u_t that give J near $p^\star$), although not necessarily worst case control. CCP is an iterative procedure which addresses difference of convex problems [15]. Much more about the algorithm and its variations and the related difference of convex algorithms can be found in [16,17].

To simplify the notation we introduce the variable

$$z = (x_1, u_1, \ldots, x_T, u_T) \in \mathbf{R}^{T(n+m)},$$

and write the problem (2) as

$$\begin{array}{ll} \text{maximize} & J(z) \\ \text{subject to} & Fz = g, \quad z \in \mathcal{C}, \end{array} \tag{3}$$

where $F \in \mathbf{R}^{n(T-1)\times T(n+m)}$, $g \in \mathbf{R}^{n(T-1)}$, and $\mathcal{C} = \mathcal{C}_1 \times \cdots \times \mathcal{C}_T$. (The matrix F has block banded structure, allowing for efficient solving, but this will not be needed in the following description of the method.)

To apply the convex–concave procedure to problem (3) we solve a series of convex optimization problems created by linearizing the objective function J at the current value of z.

Algorithm 4.1 *CCP algorithm.*

given an initial point $z^{(0)}$.
$k := 0$.
repeat
1. *Convexify.* Form $\hat{J}(z; z^{(k)}) \triangleq J(z^{(k)}) + \nabla J(z^{(k)})^T (z - z^{(k)})$.
2. *Solve.* Set $z^{(k+1)}$ to be a solution of the (convex) problem, with variable z,
$$\begin{array}{ll} \text{maximize} & \hat{J}(z; z^{(k)}) \\ \text{subject to} & Fz = g, \quad z \in \mathcal{C}. \end{array}$$
3. *Update iteration.* $k := k + 1$.
until stopping criterion is satisfied.

The convex–concave procedure does not guarantee convergence to a global maximum, but it is an ascent algorithm, i.e., we have $J(z^{(k+1)}) \geq J(z^{(k)})$. This is readily derived from the inequality $\hat{J}(z; z^{(k)}) \leq J(z)$ which holds for all z and all $z^{(k)}$, which follows from convexity of J. The stopping criterion can be as simple as $J(x^{(k)}) - J(x^{(k-1)}) \leq \epsilon$, where $\epsilon > 0$ is a tolerance. The final result can depend on the initial choice of $z^{(k)}$. It is typical to run the CCP algorithm several times with different initial conditions, and take the value of z with the largest final value of J as our approximate solution. For our examples a single initiation was sufficient, although, as mentioned, the problem can be NP-hard. One can determine if the solution found is sufficiently bad, or if alternative initializations should be tried, by looking at the S-procedure upper bound detailed in the next section.

5. S-procedure upper bounds

In this section we assume that J is convex quadratic,

$$J(z) = \begin{bmatrix} z & 1 \end{bmatrix} Q_0 \begin{bmatrix} z \\ 1 \end{bmatrix},$$

where Q_0 is positive semidefinite. (The bounds we describe here can also be derived for problems when J is not convex quadratic, but we can find a convex quadratic upper bound on J.)

We will also assume that the constraints are described by (or covered by) quadratic inequalities. Let $f_1, \ldots, f_k$ be a set of quadratic functions for which

$$z \in \mathcal{C} \Longrightarrow f_i(z) = \begin{bmatrix} z & 1 \end{bmatrix} Q_i \begin{bmatrix} z \\ 1 \end{bmatrix} \leq 0, \quad i = 1, \ldots, k. \tag{4}$$

In other words, $f_i(z) \leq 0$ are valid quadratic inequalities over $\mathcal{C}$. The quadratic functions f_i do not need to be convex.

The S-procedure is a well known method that provides a sufficient condition under which nonnegativity (nonpositivity) of a set of quadratic functions implies nonnegativity (nonpositivity) of another quadratic function [18, §2.6.3]. In the current application, it has the following form. If there exist $\tau_1, \ldots, \tau_k \geq 0$ and $W = W^T$ for which

$$J(z) - \gamma \leq \tau_1 f_1(z) + \cdots + \tau_k f_k(z) + (Fz - g)^T W (Fz - g), \tag{5}$$

holds for all z, then $\gamma \in \mathbf{R}$ is an upper bound on $p^\star$. Here we have applied the S-procedure to the inequalities

$$f_1(z) \leq 0, \ldots, f_k(z) \leq 0, \quad (Fz - g)^T W (Fz - g) \leq 0,$$

in order to imply $J(z) - \gamma \leq 0$. This assertion that γ is an upper bound on $J(z)$ is easily verified, noting that the last term on the right hand side of (5) is zero for any z that satisfies $Fz = g$, and the first k terms are nonpositive for any $z \in \mathcal{C}$ by construction. Thus for any feasible z for (3) all of the $f_i \leq 0$ and $(Fz - g)^T W (Fz - g) \leq 0$. Therefore if z is feasible and the S-procedure conditions are satisfied $J(z) \leq \gamma$. The condition (5) states that a quadratic inequality holds, and is equivalent to the linear matrix inequality (LMI)

$$Q_0 - \begin{bmatrix} 0 & 0 \\ 0 & \gamma \end{bmatrix} \preceq \sum_{i=1}^{k} \tau_i Q_i + \begin{bmatrix} F^T \\ -g^T \end{bmatrix} W \begin{bmatrix} F & -g \end{bmatrix}$$

with variables τ_i, W.

Since γ is an upper bound on $p^\star$, we can find the best such upper bound by minimizing γ subject to the LMI above. This leads to the (readily solved) semidefinite program (SDP) [19,20, § 4.6.2]

$$\begin{array}{ll} \text{minimize} & \gamma \\ \text{subject to} & Q_0 - \begin{bmatrix} 0 & 0 \\ 0 & \gamma \end{bmatrix} \preceq \sum_{i=1}^{k} \tau_i Q_i + \begin{bmatrix} F^T \\ -g^T \end{bmatrix} W \begin{bmatrix} F & -g \end{bmatrix} \\ & \tau_i \geq 0, \quad i = 1, \ldots, k \\ & W = W^T, \end{array}$$

where γ, τ_i, and W are optimization variables. The optimal value of this SDP is an upper bound on $p^\star$. It depends on the choice of the valid quadratic inequalities $f_i(z) \leq 0$. This bound may be tightened by adding additional inequality constraints.

6. Examples

6.1. Ambush control

Here we consider an instance of the ambush control problem

$$\begin{array}{ll} \text{maximize} & x_T^T Q_0 x_T \\ \text{subject to} & x_{t+1} = A_t x_t + B_t u_t, \quad t = 1, \ldots, T-1 \\ & x_1 = x_{\text{init}} \\ & \|u_t\|_\infty \leq 1, \quad t = 1, \ldots, T-1 \\ & x_t^T Q_0 x_t \leq q, \quad t = 1, \ldots, T^{\text{det}}, \end{array} \tag{6}$$

with optimization variables x_t and u_t, where $n = 4$, $m = 2$, $T = 26$, $T^{\text{det}} = 20$, $q = 0.1$, and Q_0 is a randomly generated positive definite matrix. We generate an A matrix by perturbing the entries of the identity by values drawn from a Gaussian normal distribution with mean 0 and variance 0.015. A B matrix is drawn from the same distribution. The A_i and B_i matrices are formed by further perturbing the entries of A and B with values drawn from a Gaussian distribution with mean 0 and variance 0.0015.

For the first 30 time steps (the time before $t = 1$) the system runs model predictive control feedback to respond to random disturbances. This is solely to show the behavior of the system under normal circumstances and is not incorporated into the antagonistic

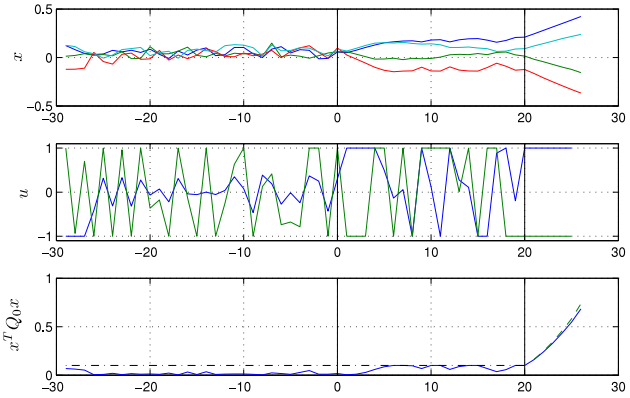

Fig. 1. Ambush control example where the system is taken over at $t = 1$, and the ambush reveals itself at $t = 20$. The dash dotted line is $q = 0.1$. The dashed line after $t = 20$ is the S-procedure upper bound.

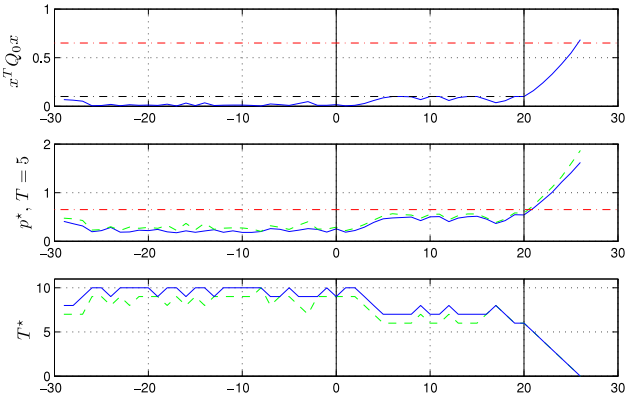

Fig. 2. Two monitors for the system in Fig. 1. The first plot depicts the system undergoing ambush control. The second plot is a monitor of that system that determines how large it is possible for $x^T Q_0 x$ to become in five time steps. The third plot is a monitor that calculates how many time steps are required until $x^T Q_0 x > p_{\text{failure}}$ can be achieved. The dash dotted lines are $q = 0.1$ and $p_{\text{failure}} = 0.65$. On the monitor plots, the solid line is the CCP bound and the dashed line is the S-procedure bound.

control problem. We use the state after 30 time steps (at $t = 1$) as x_{init}. We apply the convex–concave procedure to problem (6) and plot the results in Fig. 1. The dotted line shows the value q that $x^T Q_0 x$ stays beneath until T^{det}. The dashed line in the $x^T Q_0 x$ plot is an S-procedure bound using all of the constraints except $x_1 = x_{\text{init}}$. Observe that the constraints on u are easily represented as quadratic inequalities where each infinity norm becomes $m = 2$ quadratic inequalities (one for each control input). If we include the initial condition constraint, the resulting S-procedure bound shows that the CCP solution is optimal.

6.2. Ambush control with monitoring

On the same system presented in Section 6.1 we run two different monitors. The first monitor depicted in Fig. 2 ($p^\star,\ T = 5$) finds bad controls by applying CCP to (6) with detection constraints removed and $T = 5$, and reports how large it is possible for $p^\star$ (in this case $x_{k+T}^T Q_0 x_{k+T}$) to become. So at time $t = k$ with current state x_{current} the problem formulation is

$$\begin{array}{ll} \text{maximize} & x_{k+T}^T Q_0 x_{k+T} \\ \text{subject to} & x_{t+1} = A_t x_t + B_t u_t, \quad t = k, \ldots, k+T-1 \\ & x_k = x_{\text{current}} \\ & \|u_t\|_\infty \le 1, \quad t = k, \ldots, k+T-1, \end{array} \tag{7}$$

where the x_t and u_t are the optimization variables. The dashed green line is the S-procedure upper bound on this value. This monitor tracks how large it is possible for the objective to grow in five time steps, providing an early warning of when it will become possible to cross some dangerous threshold.

The second monitor solves a series of antagonistic control problems and reports $T^\star$ (the minimum T required for $x_{k+T}^T Q_0 x_{k+T} \ge p_{\text{failure}}$, represented by the dash dotted line). In other words we use the convex–concave procedure on the antagonistic control problem (7) for $T = 1, 2, \ldots, 10$. And report the minimum $T^\star$. In our example $p_{\text{failure}} = 0.65$. The dashed green line is the S-procedure lower bound on this value.

In both instances, our S-procedure bounds do not include a constraint $x_k = x_{\text{init}}$, but rather $x_k^T Q_0 x_k \le x_{\text{current}}^T Q_0 x_{\text{current}}$. This gives looser bounds, but means our monitor only needs $x^T Q_0 x$, not the full state. If this were a time invariant system, then all of the bounds could be computed offline once, and the monitor could be implemented as a look up table.

In the first monitor, T could be determined by the known response time of the system once the aggressor is detected. The slower the system is to respond, the larger T would be. Similarly, the second monitor could sound an alarm when $T^\star$ reaches a threshold near the response time.

Acknowledgments

We would like to thank Patrick Lincoln and Ashish Tiwari for conversations that helped us refine and improve this paper, as well as our reviewers. This research was made possible by the National Science Foundation Graduate Research Fellowship, grant DGE-1147470 and by the Cleve B. Moler Stanford Graduate Fellowship.

References

[1] R.S. Smith, A decoupled feedback structure for covertly appropriating networked control systems, in: Proc. of the IFAC World Congr., 2011, pp. 90–95.
[2] S. Amin, X. Litrico, S.S. Sastry, A.M. Bayen, Stealthy deception attacks on water scada systems, in: Proc. of the 13th ACM Int. Conf. on Hybrid Syst., 2012, pp. 161–170.
[3] Y. Mo, B. Sinopoli, False data injection attacks in control systems, in: 1st Workshop on Secur. Control, 2010, pp. 1–6.
[4] L. Xie, Y. Mo, B. Sinopoli, False data injection attacks in electricity markets, in: 1st IEEE Int. Conf. on Smart Grid Commun., 2010, pp. 226–231.
[5] M. Krotofil, A.A. Cárdenas, J. Larsen, D. Gollmann, Vulnerabilities of cyber-physical systems to stale data—determining the optimal time to launch attacks, Int. J. Crit. Infrastruct. Prot. 7 (4) (2014) 213–232.
[6] Y. Mo, B. Sinopoli, Secure control against replay attacks, in: Allerton Conf. on Commun., Control, and Comput., 2009, pp. 911–919.
[7] H. Sandberg, A. Teixeira, K.H. Johansson, On security indices for state estimators in power networks, in: Proc. of the 1st Workshop on Secur. Control Syst., 2010, pp. 1–6.
[8] A.A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, S.S. Sastry, Attacks against process control systems: Risk assessment, detection, and response, in: Proc. of the 6th ACM Symp. on Inf., Comput and Commun. Secur., 2011, pp. 355–366.
[9] Y. Mo, T.-H. Kim, K. Brancik, D. Dickinson, H. Lee, A. Perrig, B. Sinopoli, Cyber-physical security of a smart grid infrastructure, Proc. of the IEEE 100 (1) (2012) 195–209.
[10] F. Pasqualetti, F. Dörfler, F. Bullo, Attack detection and identification in cyber-physical systems, IEEE Trans. Automat. Control 58 (11) (2013) 2715–2729.
[11] J. Ding, J. Sprinkle, C.J. Tomlin, S.S. Sastry, J.L. Paunicka, Reachability calculations for vehicle safety during manned unmaned vehicle interaction, J. Guid. Control Dyn. 35 (2012) 138–152.
[12] A. Alam, A. Gattami, K.H. Johansson, C.J. Tomlin, Guaranteeing safety for heavy duty vehicle platooning: safe set computation and experimental evaluations, Control Eng. Pract. 24 (2014) 33–41.
[13] S. Amin, G.A. Schwartz, A. Hussain, In quest of benchmarking security risks to cyber-physical systems, IEEE Netw. (2013) 19–24.
[14] S. Amin, G.A. Schwartz, S.S. Sastry, Security of interdependent and identical networked control systems, Autom. 49 (1) (2013) 186–192.
[15] A.L. Yuille, A. Rangarajan, The concave–convex procedure, Neural Comput. 15 (4) (2003) 915–936.
[16] T. Lipp, S. Boyd, Variations and extensions on the convex–concave procedure, July 2014. Available at http://www.stanford.edu/~boyd/papers/cvx_ccv.

[17] T. Pham Dinh, H.A. Le Thi, Recent advances in dc programming and DCA, in: Transact. Comput. Intell. XIII, 2014, pp. 1–37.
[18] S. Boyd, L. El Ghaoui, E. Feron, V. Balakrishnan, Linear Matrix Inequalities in System and Control Theory, SIAM, 1994.
[19] S. Boyd, L. Vandenberghe, Semidefinite programming, SIAM Rev. 38 (1) (1996) 49–95.
[20] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, Cambridge, 2004.