Deep Learning of a Communication Policy for an Event-Triggered Observer for Linear Systems

Mathieu Marchand, Vincent Andrieu, Sylvain Bertrand, Steeven Janny, Hélène Piet-Lahanier

Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 56-2 (2023) 5261–5268

Deep Learning of a Communication Policy for an Event-Triggered Observer for Linear Systems

Mathieu Marchand * **Vincent Andrieu** ** **Sylvain Bertrand** *
Steeven Janny *** **Hélène Piet-Lahanier** *

* *Université Paris-Saclay, ONERA, Traitement de l'information et systèmes, 91123, Palaiseau, France (e-mail: {mathieu.marchand; sylvain.bertrand; helene.piet-lahanier}@onera.fr).*
** *Université Lyon, CNRS, LAGEPP, Villeurbanne, France (e-mail: vincent.andrieu@gmail.com).*
*** *INSA Lyon, LIRIS Villeurbanne, France (e-mail: steeven.janny@insa-lyon.fr).*

Abstract: The problem of learning a communication policy is investigated in this paper for the design of an event-triggered observer for discrete-time LTI systems. Firstly, the event-triggered observer problem is formulated as an optimisation problem. The existence of a solution to this problem (communication policy) is investigated and it is verified if this solution still preserves the stability of the estimation error dynamics. Secondly, an algorithm is provided to approximate this optimal solution using neural networks and deep learning. Simulation examples are provided to illustrate the effectiveness of the learned communication policies.

Keywords: Observer design; Event-triggered communications; Neural networks; Deep learning.

1. INTRODUCTION

For many applications it is mandatory to have access to the value of state of a system, eg. for control, supervision, decision making, etc. Nevertheless, there exist many situations where the state, or some of its components, can not be directly accessible from available information provided by the system or measurements generated by some sensors. In that case, one solution consists in designing an observer to compute an estimate of the state of the system from available information/measurements. For linear systems, Luenberger observers are one well-known class of such algorithms.

When the measurements are not directly accessible to the observer, they have to be transmitted from the system to the observer. However, many applications do not benefit from transparent and unlimited access to measurement. In this case, a parsimonious use of the sensors must be implemented, e.g. to limit energy consumption, or interferences on communication channels. However, this strategy of communication may have an impact on the performance, the robustness and the stability of the observer. Straightforward periodic policy are very limited, as measurements occur on a regular time basis regardless of the accuracy of the observer.

In event-triggered communication (Tabuada [2007]), exchange of information is decided by evaluating a Communication Triggering Condition (CTC) usually defined from a function depending on measured output of the system and/or state estimate from an observer. Such a communication policy usually leads to reduced number of transmissions while preserving good performances.

Literature on event-triggered mechanisms for control and estimation is vast, see e.g. Ge et al. [2021]. Event-triggered schemes have been developed for distributed control law of multi-agent systems, see e.g. Garcia et al. [2014], Liu et al. [2017, 2019], Seyboth et al. [2013], Wang et al. [2019], Xie et al. [2015]. Or to know when the model of a system needs to be learned, see e.g. Schlüter et al. [2020], Solowjow et al. [2018], Solowjow and Trimpe [2020]. In these applications, the state of the system is assumed to be fully accessible and the CTC is a function of the error between the current state of the system and its estimate. The event-triggered paradigm has also been applied to decide when communicate information from a sensor to an observer (Li et al. [2010]). Various works propose to implement the dynamics of the observer in the sensor, see e.g. Scheres et al. [2021], Trimpe [2014], but when the sensor has limited computation capabilities, an event-trigger scheme which uses only the measured output and the past transmitted values has to be implemented, see e.g. Etienne and Di Gennaro [2016], Petri et al. [2021]. In theses papers, the triggering structures are obtained from Lyapunov analysis which allows to obtain stability and robustness certificates. However, these approaches implies strong conservatism on the design parameters of the triggering law, which yields intensive communication flows.

Here, another approach is considered. Indeed, the communication policy is obtained as an approximation of the

[1] *This work was funded by French grant ANR Delicio (ANR-19-CE23-0006).

10.1016/j.ifacol.2023.10.166

solution of an optimal control problem. More precisely, we introduce the formulation of an optimisation problem involving a cost function with a discount factor which quantifies the quality of the estimation and the number of communications required. Our approach allows us to approximate optimal communication policy in a way that minimises communication while maintaining state estimation low. As a preliminary step, we show that a solution to this problem exists and guarantees a certain level of estimation of the unknown state. In a second step, the solution to this optimal control problem is approximated by training a neural network to predict a binary communication triggering signal from the current measured and estimated output of the system corresponding to either a communication needs to be done or not.

The remainder of the paper is organised as follows. The problem statement is provided in Section 2. In Section 3, the existence of an optimal communication policy for the system is proved and its property is analysed. The considered learning method is explained in Section 4 and some examples are studied in Section 5. Conclusions are finally given in Section 6.

Notations: The notation $\mathbb{R}$ stands for the set of real numbers. We denote by $\mathbb{R}_+$ the set of real positive numbers and $\mathbb{R}^*$ the set of non null numbers. The set $\mathbb{N}_{\leq k}$ corresponds to the set of integers inferior to k. We also denote by C^1, the set of differentiable functions with continuous derivative. The notation $\mathcal{K}$ refers to the continuous functions defined on $\mathbb{R}_+$ to $\mathbb{R}_+$ which are strictly increasing, and vanishing at zero. A continuous function $\beta : [0,\ a) \times [0,\ \infty) \to [0,\ \infty)$, $a \in \mathbb{R}^+$, is said to belong to class $\mathcal{KL}$ if for each fixed s the mapping $\beta(s,\, r)$ is class $\mathcal{K}$ with respect to r, and for fixed r the mapping $\beta(s,\, r)$ is decreasing with respect to s and $\beta(s,\, r) \to 0$ as $s \to \infty$. The standard Euclidean norm is denoted by $|\cdot|$.

2. PROBLEM STATEMENT

Consider the following Linear Time-Invariant discrete-time system

$$\begin{cases} x_{k+1} = A\, x_k + B\, u_k, & \text{(1a)} \\ y_k = C\, x_k, & \text{(1b)} \end{cases}$$

where $x_k \in \mathbb{R}^n$ is the state of the system, $u_k \in \mathbb{R}^m$ is a known control input of the system and $y_k \in \mathbb{R}^p$ is the measured output. The pair $(A,\, C)$ is assumed to be observable. Hence, a standard Luenberger observer can be designed in the following form

$$\begin{cases} \hat{x}^s_{k+1} = A\, \hat{x}^s_k + B\, u_k + L\left(y_k - \hat{y}^s_k\right), & \text{(2a)} \\ \hat{y}^s_k = C\, \hat{x}^s_k, & \text{(2b)} \end{cases}$$

where $\hat{x}^s_k \in \mathbb{R}^n$ is the state estimate, $\hat{y}^s_k \in \mathbb{R}^p$ is the estimated output, $L \in \mathbb{R}^{n\times p}$ a matrix. If $A - L\,C$ is Schur, i.e. the eigenvalues of $A - L\,C$ are contained within the unity circle, this observer guarantees that the estimation error between x_k and $\hat{x}^s_k$ exponentially converges to zero. In our case, we are considering a scenario where it is expensive for the observer to access the measurement y_k. Thus we want to design a mechanism to sporadically transmit it, while maintaining the estimation error low. An illustration of the whole considered setup is presented in Figure 1.

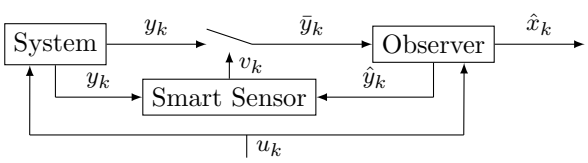

Fig. 1. Illustration of the considered event-triggered observer

A smart sensor is placed on the system to decide when to transmit the measurement y_k to the observer. Communications from the observer to system are not considered expensive and therefore the observer can "continuously" transmit its estimated measurements $\hat{y}_k$. An example of such a structure could be an experiment, represented by the system, which is performed far from where its measurements are going to be processed, i.e. the observer. And the place where the experience takes place is limited in upstream bandwidth but not in downstream bandwidth. Another example could be a mobile robot with very restricted embedded energy resource and that must therefore reduce its energy consumption by limiting emission of communication signals as much as possible.

Assuming that the control input u is accessible from the observer at each time instant k, we suggest the following observer structure

$$\begin{cases} \hat{x}_{k+1} = A\, \hat{x}_k + B\, u_k + L\left(\bar{y}_k - \bar{\hat{y}}_k\right), & \text{(3a)} \\ \hat{y}_k = C\, \hat{x}_k, & \text{(3b)} \end{cases}$$

where $\hat{x} \in \mathbb{R}^n$ is the state estimate, L is the matrix defined in (2), $\bar{y}_k \in \mathbb{R}^p$ the last value of y_k transmitted, and $\bar{\hat{y}}_k$ corresponding to the value of $\hat{y}_k$ at the last instant of transmission of y_k hence

$$\bar{y}_{k+1} = v_k\, y_{k+1} + (1 - v_k)\, \bar{y}_k, \tag{4}$$

and

$$\bar{\hat{y}}_{k+1} = v_k\, \hat{y}_{k+1} + (1 - v_k)\, \bar{\hat{y}}_k, \tag{5}$$

with the communication triggering signal $v_k \in \{0,\, 1\}$ representing whether the measured output is transmitted $(v_k = 1)$ or not $(v_k = 0)$.

Let us define the estimation error as $\tilde{x}_k = x_k - \hat{x}_k \in \mathbb{R}^n$. Its dynamics are given by the following relation

$$\tilde{x}_{k+1} = A\, \tilde{x}_k - L\left(\bar{y}_k - \bar{\hat{y}}_k\right). \tag{6}$$

One can notice that

$$\bar{y}_{k+1} - \bar{\hat{y}}_{k+1} = v_k\, C\, \tilde{x}_{k+1} + (1 - v_k)\left(\bar{y}_k - \bar{\hat{y}}_k\right). \tag{7}$$

Introduce $\mathcal{W} = \mathbb{R}^{n+p} \times \{0,\, 1\}$. From the relations (6) and (7), an extended system can be designed as

$$\xi_{k+1} = f(\xi_k,\, v_k) \tag{8}$$

where $\xi_k = \left[\xi_k^{1\top},\, \xi_k^{2\top}\right]^\top = \left[\tilde{x}_k^\top,\, \left(\bar{y}_k - \bar{\hat{y}}_k\right)^\top\right]^\top \in \mathbb{R}^{n+p}$, and the function $f : \mathcal{W} \to \mathbb{R}^{n+p}$ such that,

$$f(\xi_k,\, v_k) = \begin{bmatrix} A\, \xi_k^1 - L\, \xi_k^2 \\ v_k\, C\left(A\, \xi_k^1 - L\, \xi_k^2\right) + (1 - v_k)\, \xi_k^2 \end{bmatrix}, \tag{9}$$

representing the dynamics of ξ_k. Now, we will consider v_k as the input of the extended system (8).

We denote the solution to (8) at the k-th step from the initial condition $\xi_0 \in \mathbb{R}^{n+p}$ by $\Xi\left(k,\, \xi_0,\, \{v_l\}_{l\in\mathbb{N}_{<k}}\right)$. We are considering the following cost function

$$J_\gamma(\{v_k\}_{k\in\mathbb{N}},\,\xi_0)=\sum_{k=0}^{\infty}\gamma^k\Big(\Big|\Xi\left(k,\xi_0,\{v_l\}_{l\in\mathbb{N}_{\leq k}}\right)\Big|+\lambda\,v_k\Big), \quad (10)$$

with $\gamma\in(0,\,1)$ a discount factor, and $\lambda\in\mathbb{R}_+$ a parameter, which represents the compromise between communication and estimation performances.

Subject to the existence of an optimal solution, we denote by $V_\gamma(\xi_0)$ the optimal cost obtained from the initial state ξ_0 by minimising the cost function (10), i.e.

$$V_\gamma(\xi_0)=\min_{\{v_k\}_{k\in\mathbb{N}}} J_\gamma(\{v_k\}_{k\in\mathbb{N}},\,\xi_0). \quad (11)$$

In this work, we will first investigate the existence of an infinite length sequence $\{v^*_{\gamma,\,k}\}_{k\in\mathbb{N}}$ such that the cost function (10) is minimal and such that it guarantees a bounded error estimation of the state x_k of the system (1) by the the observer (3), i.e. there exists a $\delta\in\mathbb{R}_+$ such that $\lim_{k\to\infty}|x_k-\hat{x}_k|\leq\delta$. Secondly we will propose a Deep Learning (DL) approach to obtain an approximation of $v^*_{\gamma,\,k}$ using only the measured and estimated outputs, i.e. y_k and $\hat{y}_k$.

3. EXISTENCE AND PROPERTIES OF THE OPTIMAL COMMUNICATION POLICY

In this section, it is proven that there exists an optimal communication policy which minimises the cost function (10) and that this solution ensures an asymptotic bound on the estimation error which is related to the discount factor γ and the parameter λ. Contrary to the results in Postoyan et al. [2016], it is not possible to obtain a positive definite bound on the value function due to the particular structure of the cost function in (10). Before stating the theorem, we introduce the following lemma.

Lemma 1. There exists $\bar{\alpha}_v\in\mathbb{R}^*_+$ such that for any $\gamma\in(0,\,1)$ and $\xi\in\mathbb{R}^{n+p}$,

$$V_\gamma(\xi)\leq\bar{\alpha}_v\,|\xi|+\frac{\lambda}{1-\gamma}. \quad (12)$$

Proof. Assume that $\forall\,k\in\mathbb{N}$, $v_k=1$. In this case, for any $\xi_0\in\mathbb{R}^{n+p}$ it yields

$$V_\gamma(\xi_0)\leq J_\gamma(\{v_k\}_{k\in\mathbb{N}},\,\xi_0)=\sum_{k=0}^{\infty}\gamma^k\Big(\Big|\Xi(k,\,\xi_0,\,\{v_l\}_{l\in\mathbb{N}_{\leq k}})\Big|+\lambda\Big).$$

Since $\forall\,k$, $v_k=1$ then $\forall\,k\geq1$, $\xi^2_k=C\,\xi^1_k$. It implies

$$\begin{aligned}\xi^1_k&=(A-L\,C)^{k-1}\,\xi^1_1,\\ \xi^1_k&=(A-L\,C)^{k-1}\left(A\,\xi^1_0-L\,\xi^2_0\right).\end{aligned} \quad (13)$$

Therefore

$$\begin{aligned}\Big|\Xi(k,\,\xi_0\,,\,\{v_l\}_{l\in\mathbb{N}_{\leq k}})\Big|&\leq\Big|\xi^1_k\Big|+\Big|\xi^2_k\Big|,\\ &\leq(1+|C|)\,\Big|(A-L\,C)^{k-1}\Big|\left(|A|\,\Big|\xi^1_0\Big|+|L|\,\Big|\xi^2_0\Big|\right),\\ &\leq(1+|C|)\,\Big|(A-L\,C)^{k-1}\Big|\,(|A|+|L|)\left(\Big|\xi^1_0\Big|+\Big|\xi^2_0\Big|\right). \quad (14)\end{aligned}$$

Since $A-L\,C$ is Schur stable, there exist two positive real numbers κ_1, and c with $0<\kappa_1<1$ and $c>1$ such that

$$\Big|(A-L\,C)^k\Big|\leq c\,\kappa_1^k\ ,\forall k\geq0\,. \quad (15)$$

Consequently,

$$\Big|\ \Xi(k,\,\xi_0\,,\,\{v_l\}_{l\in\mathbb{N}_{\leq k}})\Big|\leq\frac{1}{\sqrt{2}}\kappa_2\,\kappa_1^k\left(\Big|\xi^1_0\Big|+\Big|\xi^2_0\Big|\right), \quad (16)$$

with $\kappa_2=\sqrt{2}\,\frac{c}{\kappa_1}\,(1+|C|)\,(|A|+|L|)$. Moreover, since the considered norm $|\cdot|$ is the Euclidean second-order norm, we have

$$\left(\Big|\xi^1_0\Big|+\Big|\xi^2_0\Big|\right)\leq\sqrt{2}\,|\xi_0|\,. \quad (17)$$

Thus,

$$V_\gamma(\xi_0)\leq\sum_{k=0}^{\infty}\gamma^k\left(\kappa_2\,\kappa_1^k\,|\xi_0|+\lambda\right)\ . \quad (18)$$

Hence,

$$V_\gamma(\xi_0)\leq\sum_{k=0}^{\infty}\kappa_2(\gamma\,\kappa_1)^k\,|\xi_0|+\lambda\sum_{k=0}^{\infty}\gamma^k\ . \quad (19)$$

Therefore using the fact that $\gamma<1$ and $\kappa_1<1$, we obtain

$$V_\gamma(\xi_0)\leq\frac{\kappa_2}{1-\gamma\,\kappa_1}\,|\xi_0|+\frac{\lambda}{1-\gamma}, \quad (20)$$

which concludes the proof by choosing $\bar{\alpha}_v=\kappa_2/(1-\gamma\kappa_1)$. ■

Remark 2. Lemma 1 gives an upper bound on the value function V_γ in which a constant term $\frac{\lambda}{1-\gamma}$ appears. Note that this is different from the bound obtained in Postoyan et al. [2016] and prevents us from obtaining asymptotic convergence of the estimation error to zero. This is directly linked to the structure of the cost function.

Theorem 3. Consider the system (8). Then, there exists an optimal input sequence $\{v^*_{\gamma,\,k}\}_{k\in\mathbb{N}}$ that minimises the cost function (10), and there exist $\beta\in\mathcal{KL}$ and $\gamma^*\in(0,\,1)$ such that for any $\gamma\in(\gamma^*,\,1)$ and $\xi_0\in\mathbb{R}^{n+p}$, any solution $\Xi(k,\,\xi_0,\,\{v^*_{\gamma,\,l}\}_{l\in\mathbb{N}_{\leq k}})$ to the system (8) satisfies, $\forall\,k$

$$\Big|\Xi\left(k,\,\xi_0,\,\{v^*_{\gamma,\,l}\}_{l\in\mathbb{N}_{\leq k}}\right)\Big|\leq\max\left\{\beta\left(\bar{\alpha}_v\,|\xi_0|+\frac{\lambda}{1-\gamma},\,k\right),\delta\right\}, \quad (21)$$

where $\delta=\left(\frac{\lambda}{\bar{\alpha}_v(1-\gamma)}\right)\left(\frac{1}{\alpha_v}-\frac{1-\gamma}{\gamma}\right)^{-1}$.

This theorem suggests that there exists an optimal infinite length sequence of v_k such that the estimation error converges to a ball of radius δ. One can notice that this bound δ can be made as small as desired by decreasing the parameter λ. However, as λ decreases, the penalisation of communications in the cost function (10) decreases as well. This implies that for smaller parameter λ, the amount of communications for the optimal policy will be higher.

Proof. The proof in inspired by the results developed in Postoyan et al. [2016]. By applying [Keerthi and Gilbert, 1985, Theorem 1], for each γ in $(0,1)$, and each ξ_0 in $\mathbb{R}^{n+p}$ there exists an infinite length sequence $\{v^*_{\gamma,\,k}\}_{k\in\mathbb{N}}$ in $\{0,1\}^{\mathbb{N}}$ such that

$$V_\gamma(\xi_0)=J_\gamma(\{v^*_{\gamma,\,k}\}_{k\in\mathbb{N}},\,\xi_0). \quad (22)$$

Define $\xi^*_1=f(\xi_0,\,v^*_{\gamma,\,0})$, where $v^*_{\gamma,\,0}$ is the first element of the optimal sequence $\{v^*_{\gamma,\,k}\}_{k\in\mathbb{N}}$. One has

$$|\xi_0|+\lambda\,v^*_{\gamma,0}\leq V_\gamma(\xi_0)\leq\bar{\alpha}_v\,|\xi_0|+\frac{\lambda}{1-\gamma}. \quad (23)$$

Moreover according to the Bellman equation,

$$V_\gamma(\xi_0)=|\xi_0|+\lambda\,v^*_{\gamma,0}+\gamma\,V_\gamma(\xi^*_1). \quad (24)$$

Therefore

$$V_\gamma(\xi^*_1)-V_\gamma(\xi_0)=-|\xi_0|-\lambda\,v^*_{\gamma,0}+(1-\gamma)\,V_\gamma(\xi^*_1). \quad (25)$$

Since $\gamma\,V_\gamma(\xi^*_1)\leq V_\gamma(\xi_0)$, thus

$$\begin{aligned}V_\gamma(\xi^*_1)-V_\gamma(\xi_0)&\leq-|\xi_0|-\lambda\,v^*_{\gamma,0}+(1-\gamma)\,\gamma^{-1}V_\gamma(\xi_0),\\ &\leq-|\xi_0|+(1-\gamma)\,\gamma^{-1}V_\gamma(\xi_0). \quad (26)\end{aligned}$$

On another hand, with (23), we have

$$-|\xi_0| \leq \frac{1}{\bar{\alpha}_v}\left(-V_\gamma(\xi_0) + \frac{\lambda}{1-\gamma}\right). \tag{27}$$

Injecting the relation (27) into (26), we have

$$\begin{aligned} V_\gamma(\xi_1^*) - V_\gamma(\xi_0) &\leq -\frac{1}{\bar{\alpha}_v}\left(V_\gamma(\xi_0) - \frac{\lambda}{1-\gamma}\right) + \frac{1-\gamma}{\gamma} V_\gamma(\xi_0), \\ &\leq -\left(\frac{1}{\bar{\alpha}_v} - \frac{1-\gamma}{\gamma}\right) V_\gamma(\xi_0) + \frac{\lambda}{\bar{\alpha}_v (1-\gamma)}. \end{aligned} \tag{28}$$

Since $\frac{1-\gamma}{\gamma}$ converges to 0 as γ tends to 1, there exists a $\gamma^* \in (0,\, 1)$ such that $\forall \gamma > \gamma^*$, $\frac{1}{\bar{\alpha}_v} - \frac{1-\gamma}{\gamma} > 0$. By proceeding by iteration and using Theorem 8 in Nešić et al. [1999], we deduce that there exists a function $\beta \in \mathcal{KL}$ such that for any solution Ξ of (8), initialised at ξ_0 and with the optimal control input $\{v^*_{\gamma,\,k}\}_{k\in\mathbb{N}}$, we have

$$V_\gamma\left(\Xi\left(k,\, \xi_0,\, \{v^*_{\gamma,\,l}\}_{l\in\mathbb{N}_{\leq k}}\right)\right) \leq \max\left\{\beta(V_\gamma(\xi_0), k), \delta\right\}, \tag{29}$$

with $\delta = \left(\frac{\lambda}{\bar{\alpha}_v(1-\gamma)}\right)\left(\frac{1}{\bar{\alpha}_v} - \frac{1-\gamma}{\gamma}\right)^{-1}$. Using Lemma 1, we have

$$\left|\Xi\left(k,\, \xi_0,\, \{v^*_{\gamma,\,l}\}_{l\in\mathbb{N}_{\leq k}}\right)\right| \leq \max\left\{\beta\left(\bar{\alpha}_v\,|\xi_0| + \frac{\lambda}{1-\gamma},\, k\right), \delta\right\}, \tag{30}$$

which completes the proof. ■

According to Theorem 3, if the cost function is well designed, there exists an infinite-length sequence of $\{v_k\}_{k\in\mathbb{N}}$ that minimises the cost function and this sequence guarantees an asymptotic bound on the estimation error. In the next section, we will investigate how this sequence can be approximated.

4. APPROXIMATION OF THE OPTIMAL COMMUNICATION POLICY VIA DEEP LEARNING

In Section 3, we have shown that the optimisation problem associated to system (8) and cost function (10) has a solution which guarantees an asymptotically bounded estimation error. In order to approximate this optimal solution, we propose to investigate the use of a learning method.

Reinforcement learning (RL) has been explored in Baumann et al. [2018] to learn an optimal event-triggered state feedback controller for a linear system. This approach relies on "trial and error", in the sense that the RL agent interacts with its environment and tries to maximise a user-defined reward $R = \sum_{k=0}^{\infty} \gamma^k\, r_k$, where r_k is the reward at instant k. Derivation to our problem follows readily but considering the maximisation of $-J_\gamma$ defined in (10) Reinforcement learning is however hinder by many limitations making the training difficult and time-consuming since (i) the agent have to learn a communication policy from a poorly informative reward signal that is weakly linked to its action in the environment, and (ii) RL methods requires carefully designed exploration policy which may prevent from finding optimal solution and leads to unwanted behaviour.

Another approach consists in relying on (potentially approximated) models of the system of interest, which allows to use standard deep learning algorithms. In that case, the communication policy have direct access to the physical equations of the environment and can learn by directly back-propagating the error through the dynamical model. Moreover, exploration issues can be alleviated by manual design.

For these reasons, an DL approach have been implemented in our study. It could be interesting to investigate RL approaches with safe exploration methods. This is left for future research studies. In this section, we present the considered learning algorithm.

4.1 Feedback nature of the communication policy

One can notice that the minimisation of the loss (10) can be written as a Markov Decision Process, and thus according to the principle of Optimality of Bellman (Bellman [1954]), the optimal decision v_k^* at the instant k depends only on the state resulting from the previous decision, i.e. ξ_k. Nevertheless, in the common case, the state of the system (1) is not completely known, so is ξ_k.

If the control input u_k is equal to 0 at each time step k, and if the system (1) is observable then, according to Kalman theory (Kalman [1960]), the state of the system can be reconstructed from $n-p$ data of the measurement. Therefore a function $\psi : (\mathbb{R}^p)^{n-p} \times (\mathbb{R}^p)^{n-p} \to \{0,\, 1\}$ can be designed such that $v_k^* = \psi(y_k,\, \ldots,\, y_{k-(n-p)},\, \hat{y}_k,\, \ldots,\, \hat{y}_{k-(n-p)})$.

If the control input u_k is not equal to 0 at each time step k, then the sequences $\{y_k\}_{k\in\mathbb{N}}$ and $\{\hat{y}_k\}_{k\in\mathbb{N}}$ depend on the sequence of inputs and ξ_k can not be obtained anymore through the sole data of y_k and $\hat{y}_k$.

Note that the measurement y_k can be written as

$$y_k = C\,A^k\,x_0 + \sum_{i=0}^{k-1} C\,A^i\,B\,u_i\,. \tag{31}$$

In order to make the measurements no longer dependent on the control input, the variable $\breve{y}_k$ defined as

$$\breve{y}_k = y_k - \sum_{i=0}^{k-1} C\,A^i\,B\,u_i \tag{32}$$

is introduced. In order to evaluate $\breve{y}_k$, the previous control inputs, i.e. u_i, $\forall\, i \leq k$, are required. To avoid this drawback, we introduce the following dynamic system

$$z_{k+1} = A\,z_k + B\,u_i, \tag{33}$$

initialised at $z_0 = 0$. One can notice that the solution of (33) is

$$z_k = \sum_{i=0}^{k-1} A^i\,B\,u_i\,, \tag{34}$$

and thus $\breve{y}_k = y_k - C\,z_k$. Similarly, we introduce $\breve{\hat{y}}_k = \hat{y}_k - C\,z_k$. Doing so, the state x_k and its estimate $\hat{x}_k$ can be reconstructed through data of $\breve{y}_k$ and $\breve{\hat{y}}_k$.

4.2 Recurrent Neural Network

We propose to model the function $v_k = \sigma(\breve{y}_k, \breve{\hat{y}}_k)$ that maps the known measurements $\breve{y}_k$ and $\breve{\hat{y}}_k$ to the triggering decision v_k as deep neural network. However, unique measurements at a given time k do not yield enough information to find the optimal v_k^*. In what follows, we leverages the results in section 4.1 showing that the dynamics of $\breve{y}$ and $\breve{\hat{y}}$ can be used to recover the state ξ_k, and therefore to evaluate v_k^*.

Dynamical information are embedded in the model through a recurrent neural network (RNN). Such class of deep learning models are particularly suited to handle time-dependent signals. A RNN leverages a latent memory vector h_k storing temporal information across time. This memory is recursively updates based on the current value of a time-varying input i_k via a learned non-linear function: $h_k = \text{RNN}(i_k, h_{k-1})$. There is multiple possibility to define the transition function. We used Gated Recurrent Unit introduced in (Cho et al. [2014]) usually leading to better performances on small datasets.

4.3 Deep Learning in a semi-supervised framework

We proposed a semi-supervised network to train our communication trigger function σ, in the sense that, conversely to standard supervised learning, we do not require ground truth optimal solution v_k^* on a set of training trajectories to learn our approximated solution. Instead, we train our model to minimise the following cost:

$$\mathcal{L} = \sum_{k=0}^{N} \gamma^k \left(|x_k - \hat{x}_k| + \lambda\, v_k \right). \tag{35}$$

where $v_k = \sigma(\breve{y}_k, \breve{\hat{y}}_k)$ and x_k and $\hat{x}_k$ are respectively defined in (1) and (3). This loss (35) is an approximation of the cost function (10) since $N \neq +\infty$. Note also that only the first component of Ξ is considered in the formulation of this loss function, since it is the one only necessary from a practical point of view (the second component being directly related to $x_k - \hat{x}_k$). If N is *high enough*, so that γ^N is *small*, then the loss (35) will *nearly be equal* to the cost function (10). And according to Theorem 3, there exists $\gamma^* \in (0,\, 1)$ such that for any $\gamma \in (\gamma^*,\, 1)$, the optimal infinite-length sequence v_k makes the observer state (3) converge to a ball around the state of the system (1) with radius depending on λ and γ. Therefore, optimising the parameter of $\sigma(\cdot)$ to minimise $\mathcal{L}$ will lead the neural network to approximate the provably existing optimal solution.

The resulting proposed algorithm to learn the communication policy associated to the considered event-triggered observer problem is summarised in Algorithm 1.

At first, a training database is generated by simulating the system from different initial conditions. This database must be big enough to cover the set of values of x_k and $\hat{x}_k$ where the trained network will then operate. In a second step, a trajectory of the observer is generated thanks to the outputs of the neural network. Of course, at the beginning of the training, i.e. when ε is low, the neural network will poorly perform. Finally, losses (35) is computed for each trajectories, and the parameters of the neural network are optimised to minimise the mean of losses (35).

4.4 Learning a binary number

One of the key challenges for designing the structure of σ is to constrain the output of the neural network, canonically lying in $\mathbb{R}$, to binary numbers. During test time (that is, when the neural network is trained), a straightforward method consists in comparison of the output with a threshold. Yet, this implementation introduces discontinuity which is not compatible with the gradient descent algorithm used to optimise the problem. Thus,

Algorithm 1: Communication policy learning algorithm

inputs: number of training epochs N_e, number N_τ of training trajectories and associated time duration N.
Generate a training data set $\mathcal{S}_\tau$ of N_τ trajectories $\tau = \{(x_k, y_k)\}_{k \in \mathbb{N}_{\leq N}}$ from multiple random samples of initial condition x_0.
Initialise the parameters of the network $\sigma(\cdot)$.
for $\varepsilon = 1$ *to* N_e **do**
  Initialise the hidden vector h_0 to 0.
  for *each trajectory* τ *in* $\mathcal{S}_\tau$ **do**
    for $k = 1$ *to* N **do**
      Compute $(v_k,\, h_k) = \sigma\left(\left[\breve{y}_k, \breve{\hat{y}}_k\right], h_{k-1}\right)$.
      Update the variable $\bar{y}$ according to (4).
      Update the variable $\bar{\hat{y}}$ according to (5).
      Compute $\hat{x}_{k+1}$.
      Evaluate the loss $\mathcal{L}_\tau$ using (35).
    end
  end
  Update the parameters of the network $\sigma(\cdot)$ by minimising $\frac{1}{N_\tau} \sum_{\tau \in \mathcal{S}_\tau} \mathcal{L}_\tau$ using backward propagation.
end

during training, we proposed another approach. We relaxed the binary constraint by using a sigmoid function to bound the output of σ to $[0, 1]$. The interest we have in this implementation is that the sigmoid function remains differentiable. However, to enforce the model to outputs nearly binary numbers, we progressively increase the slope of the sigmoid during training (the setup is illustrated in figure 2)

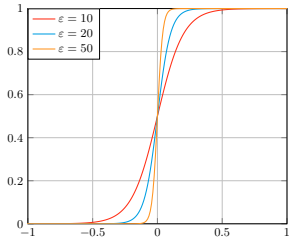

Fig. 2. Illustrations of a sigmoid function for different values of ε.

As we can notice, the higher ε the closer the sigmoid to a step function. Therefore, the training will benefit of a smooth function at the beginning of the training phase to ease learning, and the output will progressively become closer and closer to binary numbers when the training progress.

This is this method based on the introduction of a simgoid function that has been considered in our numerical implementations.

4.5 Structure of the Neural Network

The structure of the neural network is presented in Figure 3. We first project the inputs $\breve{y}_k, \breve{\hat{y}}_k$ to a higher dimensional space via a one-layer multilayer perceptron (MLP). The resulting vector is then used as input to a GRU to update

the latent memory h_k. Finally, a second MLP decode the latent memory into the binary communication trigger v_k. The model is then used in an auto-regressive manner to estimates future values of v_k.

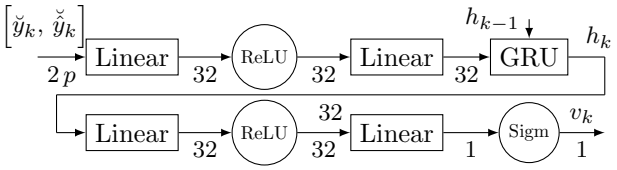

Fig. 3. Structure of the considered Neural Network

5. SIMULATION EXAMPLES

We demonstrate the performances of our method on two numerical simulations : a discrete time oscillator with the observer presented in figure 1, and another observer structure on which theoretical results presented in section 3 ca not be used directly. However, we show that even in that case, our model gives good estimation performances with a reduced amount of transmitted data. For this second example, a comparison is made between the proposed observer obtained through Deep Learning and the observer designed in (Petri et al. [2021]).

5.1 First example: an oscillator system

Consider an oscillator with LTI dynamics described as in (1) with

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix}. \tag{36}$$

An observer described by (3) is designed with

$$L = [-0.18,\ 0.0107,\ 0.099742,\ 0.982]^\top. \tag{37}$$

The neural network presented in Figure 3 has been trained on a training database containing $N_\tau = 10\,000$ trajectories starting from different initial conditions sampled following a Normal distribution with mean equal to 0 and standard deviation equal to 20 independently for each state component. The chosen optimiser is the Adam algorithm with a learning rate of 10^{-3}. The control input u_k, assumed to be known, is generated as a random signal following a normal distribution $\mathcal{N}(0, 20)$. The hyperparameter λ balancing between accuracy of the estimation and number of communications is set to $\lambda = 3$. The discount factor γ is equal to $\gamma = 0.995$. During the training, the trajectories length N is equal to $N = 400$. One can notice that $\gamma^N = 0.13$. Define $R_v = \frac{1}{N}\sum_{k=0}^{N} v_k$ representing the triggering rate of the observer. R_v equals to 1 corresponds to a communication triggered at every time step.

We evaluate our neural communication policy on a testing set of 10 000 new trajecories unseen during training starting from different initial states x_0 with components sampled following the normal distribution $\mathcal{N}(0, 10)$ and considering sequences of independent control input values u_k randomly generated following the normal distribution $\mathcal{N}(0, 1)$. The mean values of the obtained loss $\mathcal{L}$ and of the triggering rate R_v computed over all the trajectories of this test data set are exposed in Table 1. A zoom for k ranging from 0 to 200 of one trajectory of this database from the initial state $x_0 = [5.3492, 1.9880, 6.5921, 6.5689]^\top$ and $\hat{x}_0 = [0, 0, 0, 0]^\top$ with the control input sequence plotted in Figure 4, is presented in Figure 5.

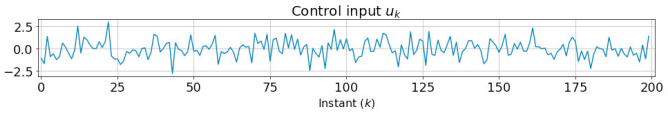

Fig. 4. Control input sequence corresponding to one trajectory of the test data set and used for illustration

At the top of Figure 5, the trajectories of the second component of the state of the system x_k and of the state of the observer $\hat{x}_k$ are plotted. At the centre, the norm of the estimation error is plotted. At the bottom of the figure, the communication policy v_k is presented.

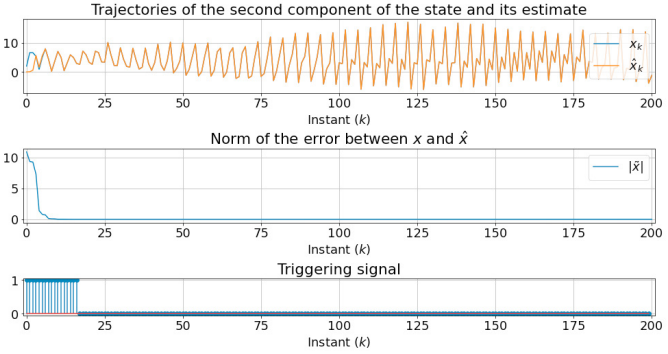

Fig. 5. Illustration of a trajectory without any noise.

The observer driven by our communication policy converged well to the state of the system, and no communication are triggered once the estimation error $\tilde{x}_k$ is close enough to 0. To assess robustness wrt. noise of the obtained event-triggered observer with learned communication policy, further tests have been performed by considering another test data set of 10 000 trajectories generated by considering additive state noise and measurement noise to system (1), i.e. a dynamic system modeled by

$$\begin{cases} x_{k+1} = A\,x_k + B\,u_k + w_k^x, & \text{(38a)} \\ y_k = C\,x_k + w_k^y, & \text{(38b)} \end{cases}$$

where w_k^x and w_k^y follow respectively the normal distribution $\mathcal{N}(0, \Sigma_k^x)$ and $\mathcal{N}(0, \Sigma_k^y)$, with $\Sigma_k^x = 0.05\,\mathrm{diag}(x_k)$ and $\Sigma_k^y = 0.1\,\mathrm{diag}(y_k)$ where $\mathrm{diag}(\cdot)$ generate a diagonal matrix from the components of a vector. One can notice that the noise-to-signal ratio is around 5% for the state, and 10% for the measurement. Figure 6 presents the results obtained with the observer for one trajectory of this test data-set "with noise" with the same initial state x_0 and $\hat{x}_0$.

In presence of noise, asymptotic bounded convergence of the estimation error can not be guaranteed anymore. Nevertheless, one can notice that when the estimation error becomes too large, the neural network will consider that it is worth to trigger. Of course the observer does not know directly the estimation error, but from the knowledge on $\breve{y}_k$ and $\breve{\breve{y}}_k$ and the GRU layer, it can estimate it.

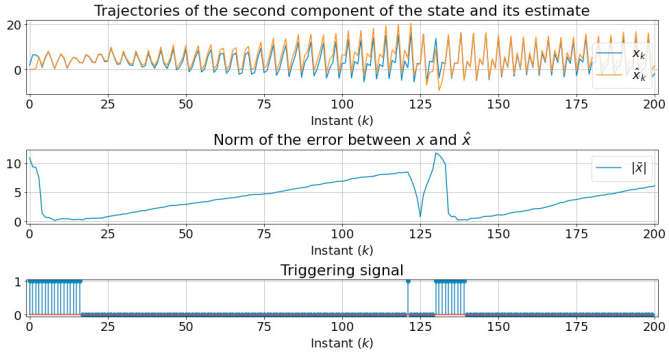

Fig. 6. Illustration of a trajectory with noises.

Table 1. Performances of the event-triggered observer with learned communication policy for the first example

$N = 200$			
Σ_k^x	Σ_k^y	Mean of $\mathcal{L}$	Mean of R_v
$0\,\mathrm{diag}(x_k)$	$0\,\mathrm{diag}(y_k)$	121	0.07
$0.1\,\mathrm{diag}(x_k)$	$0.2\,\mathrm{diag}(y_k)$	395	0.13

5.2 Second example: a passive sensor

In this subsection, a comparison is made between the event-trigger observer obtained through deep learning and the one described in (Petri et al. [2021]). In that article, authors are considering a different structure than the one presented in Figure 1. An illustration of this structure is provided in Figure 7.

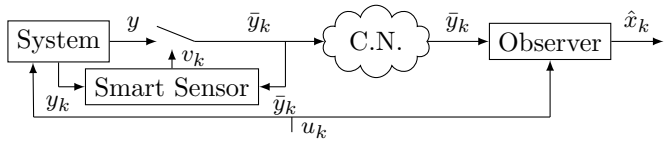

Fig. 7. Illustration of the event-triggered observer considered in (Petri et al. [2021]). C.N. stands for Communication Network.

The smart sensor will decide when to trigger a communication only based on the measured output y_k and the transmitted one $\bar{y}_k$. Thus it has no information on the state estimate computed by the observer.

In this section, our neural network is trained by using the proposed Algorithm 1, but for the structure illustrated in Figure 7. One can notice that in this case, the neural network will not have enough information to reconstruct the estimation error, and thus ξ_k, since it will not be able to reconstruct $\hat{x}_k$. The obtained v_k will not be a feedback function of ξ_k, thus Theorem 3 can not be applied to provide guarantees on the existence of an optimal communication policy leading to bounded convergence of the estimation error. Nevertheless, we observed that good performances are obtained in practice when applying the proposed approach to this case.

To be consistent with (Petri et al. [2021]), continuous-time dynamics are first considered for the description of the system, defined as

$$\begin{cases} \dot{x}(t) = A\,x(t) + B\,u(t) + w^x(t), & \text{(39a)} \\ y(t) = C\,x(t) + w^y(t), & \text{(39b)} \end{cases}$$

and the dynamics of the observer:

$$\begin{cases} \dot{\hat{x}}(t) = A\,\hat{x}(t) + B\,u(t) + L\,(\bar{y}(t) - \hat{y}(t))\,, & \text{(40a)} \\ \hat{y}(t) = C\,\hat{x}(t), & \text{(40b)} \end{cases}$$

with $\bar{y}(t)$ defined as in (4), and A, B, C, L defined as in Petri et al. [2021].

Numerical simulations are done by considering Euler discretisation method, with sampling period $T_s = 10^{-1}\,s$. In the following, the term *Petri et al.'s observer* will refer to the event-triggered observer developed in (Petri et al. [2021]), whereas the term *neural network observer* will refer to the event-triggered observer developed based on the learning approach proposed in Algorithm 1.

The Petri et al.'s observer is simulated with the same hyper parameters as in (Petri et al. [2021]), from the initial state $x_0 = [1,\,3]^\top$ and estimate $\hat{x}_0 = [0,\,0]^\top$, and with the control input presented in Figure 8.

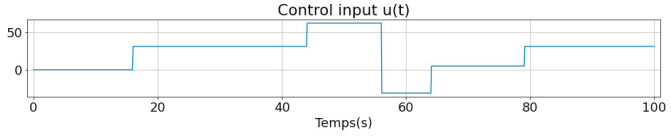

Fig. 8. Illustration of the control input $u(t)$.

The discrete-time realisations of the noise signals $w^x(t)$ and $w^y(t)$ are generated randomly following a normal distribution $\mathcal{N}(0,\,0.1)$ for each of their components. Note that these noise characteristics result in noise-to-signal ratios of 10% for the first component of the initial state x_0 and 3% for its second component.

The trajectory of the norm of the estimation error, from Petri et al.'s observer, is plotted at the top of Figure 9. The triggering signal v_k for this trajectory are illustrated at the bottom of Figure 9.

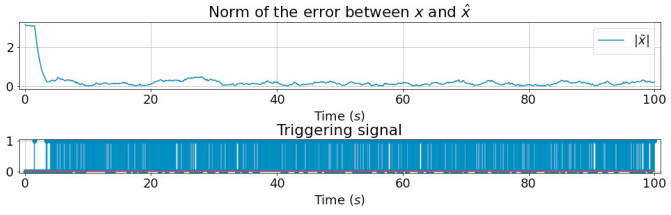

Fig. 9. Illustration of a trajectory obtained with the Petri et al.'s observer in a noisy environment.

The convergence of the estimation error to a neighbourhood of 0 can be observed while not having the communication policy v_k equal to 1 at each time step.

Since the control input $u(t)$ presented in Figure 8 is piecewise constant, we claim that the neural network can be trained on a dataset of trajectories generated by considering different constant inputs. The neural network observer has been trained on a dataset of 20 000 trajectories. These trajectories are generated from 1 000 different initial states x_0 with components randomly sampled following $\mathcal{N}(0, 1)$ and with 20 different values of a constant signal for $u(t)$ randomly generated according to a uniform distribution between $[-250,\,250]$. We chose $\lambda = 0.5$, $N = 400$, and $\gamma = 0.995$. We selected the same optimiser as in Section 5.1.

The trained neural network observer has been tested on the same trajectory as for the Petri et al.'s observer. A zoom for k ranging from 0 to 100 of the trajectory is given in Figure 10.

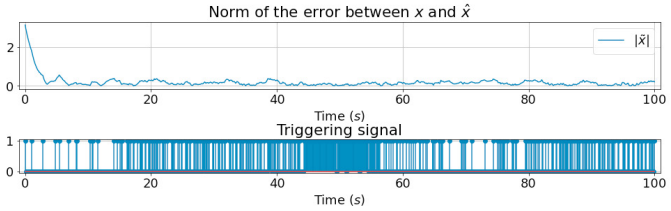

Fig. 10. Illustration of a trajectory obtained with the neural network observer.

As for the Petri et al.'s observer case, the converge of the estimation error to a neighbourhood of 0 can be noted. However, fewer triggering instants can be observed when using the neural network observer.

In order to quantify and compare the performances between the two observers, the loss $\mathcal{L}$ defined in (35) and the triggering rate R_v are calculated for this trajectory. For the Petri et al.'s observer case, we obtained $\mathcal{L} = 168$ and $R_v = 0.83$ whereas for the neural network observer we obtained $\mathcal{L} = 87.7$ and $R_v = 0.41$. As can be noticed, the neural network observer leads to better performances than the Petri et al.'s one. Indeed, with the neural network, the evaluated loss $\mathcal{L}$ is half as large as with the Petri et al.'s observer. A similar result is obtained regarding the triggering ratio R_v. Note that these results are consistent, since the parameters of the neural network have been optimised to minimise the loss $\mathcal{L}$ while it is not the case for the observer from (Petri et al. [2021]). However, in our context and in contrast with (Petri et al. [2021]) no stability guarantee is obtained.

6. CONCLUSION

In this paper, the design of a communication policy for an event-triggered observer for linear systems has been formulated as an optimisation problem. A proof of the existence of an optimal solution to this problem has been provided. It has also been shown that this optimal communication policy results in a bounded convergence of the estimation error provided by the event-triggered observer. Afterwards, a method based on Deep Learning has been proposed to approximate this solution and learn a communication policy based on neural networks. Two simulation examples have been finally proposed to illustrate the good performance obtained by the proposed approach.

REFERENCES

Baumann, D., Zhu, J.J., Martius, G., and Trimpe, S. (2018). Deep reinforcement learning for event-triggered control. In *2018 IEEE Conference on Decision and Control (CDC)*, 943–950. IEEE.

Bellman, R. (1954). The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6), 503–515.

Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. URL `https://arxiv.org/abs/1409.1259`.

Etienne, L. and Di Gennaro, S. (2016). Event–triggered observation of nonlinear lipschitz systems via impulsive observers. *IFAC-PapersOnLine*, 49(18), 666–671.

Garcia, E., Cao, Y., and Casbeer, D.W. (2014). Decentralized event-triggered consensus with general linear dynamics. *Automatica*, 50(10), 2633–2640.

Ge, X., Han, Q.L., Zhang, X.M., and Ding, D. (2021). Dynamic event-triggered control and estimation: A survey. *International Journal of Automation and Computing*, 18(6), 857–886.

Kalman, R.E. (1960). On the general theory of control systems. In *Proceedings First International Conference on Automatic Control, Moscow, USSR*, 481–492.

Keerthi, S. and Gilbert, E. (1985). An existence theorem for discrete-time infinite-horizon optimal control problems. *IEEE Transactions on Automatic Control*, 30(9), 907–909.

Li, L., Lemmon, M., and Wang, X. (2010). Event-triggered state estimation in vector linear processes. In *Proceedings ACC*, 2138–2143. IEEE.

Liu, J., Yu, Y., Wang, Q., and Sun, C. (2017). Fixed-time event-triggered consensus control for multi-agent systems with nonlinear uncertainties. *Neurocomput.*, 260.

Liu, J., Zhang, Y., Yu, Y., and Sun, C. (2019). Fixed-time event-triggered consensus for nonlinear multiagent systems without continuous communications. *IEEE Tran. on Sys., Man, and Cyber.: Sys.*, 49(11), 2221–2229.

Nešić, D., Teel, A.R., and Sontag, E.D. (1999). Formulas relating KL stability estimates of discrete-time and sampled-data nonlinear systems. *Sys. & Con. Let.*

Petri, E., Postoyan, R., Astolfi, D., Nešić, D., and Heemels, W.M.H. (2021). Event-triggered observer design for linear systems. In *60th IEEE CDC*, 546–551. IEEE.

Postoyan, R., Buşoniu, L., Nešić, D., and Daafouz, J. (2016). Stability analysis of discrete-time infinite-horizon optimal control with discounted cost. *IEEE TAC*.

Scheres, K.J., Chong, M., Postoyan, R., and Heemels, W.M.H. (2021). Event-triggered state estimation with multiple noisy sensor nodes. In *2021 60th IEEE CDC*.

Schlüter, H., Solowjow, F., and Trimpe, S. (2020). Event-triggered learning for linear quadratic control. *IEEE Transactions on Automatic Control*, 66(10), 4485–4498.

Seyboth, G.S., Dimarogonas, D.V., and Johansson, K.H. (2013). Event-based broadcasting for multi-agent average consensus. *Automatica*, 49(1), 245–252.

Solowjow, F., Baumann, D., Garcke, J., and Trimpe, S. (2018). Event-triggered learning for resource-efficient networked control. In *2018 ACC*, 6506–6512. IEEE.

Solowjow, F. and Trimpe, S. (2020). Event-triggered learning. *Automatica*, 117, 109009.

Tabuada, P. (2007). Event-triggered real-time scheduling of stabilizing control tasks. *IEEE TAC*.

Trimpe, S. (2014). Stability analysis of distributed event-based state estimation. In *53rd IEEE CDC*, 2013–2019.

Wang, Y.W., Lei, Y., Bian, T., and Guan, Z.H. (2019). Distributed control of nonlinear multiagent systems with unknown and nonidentical control directions via event-triggered communication. *IEEE Trans. on Cyber.*, 50(5).

Xie, D., Xu, S., Chu, Y., and Zou, Y. (2015). Event-triggered average consensus for multi-agent systems with nonlinear dynamics and switching topology. *Journal of the Franklin Institute*, 352(3), 1080–1098.