Event-triggered Learning

Friedrich Solowjow, Sebastian Trimpe

Max Planck Institute for Intelligent Systems, Intelligent Control Systems Group, Stuttgart, Germany

Abstract

Efficient information exchange over wireless networks is an essential part of achieving intelligent behavior in groups of autonomous agents. Deciding when to communicate and utilizing model-based predictions in the absence of sensor data reduces communication significantly. In this article, we propose event-triggered learning as a novel concept to reduce communication even further. We derive different classes of learning triggers solely based on a statistical analysis of inter-communication times and proof the effectiveness of the proposed learning triggers.

Key words: Networked Control Systems; Statistical Learning Theory.

1 Introduction

Wireless communication in networked control systems (NCSs) copes with many challenges such as limited bandwidth, delays, and package drops. Applications are among others mobile sensor networks, distributed sensor fusion, and coordination control (cf. [1] and references therein for an overview). Many proposed methods rely on accurate dynamics models. For instance, model-based predictions can replace periodic communication up to a certain extent. However, if unexpected behavior occurs communication of sensor measurements is indispensable. To detect a change in the system’s behavior, we propose to extend the paradigm of event-triggering to model learning and introduce the novel idea of event-triggered learning (ETL), as depicted in Fig. 1.

Classical approaches rely on a physical understanding of the world and use first principles to model dynamical systems. Further, they require human expert knowledge and are therefore expensive and lack the flexibility to be adjusted online. The ability to learn is a fundamental aspect of future intelligent systems that are facing uncertain environments. However, the process of learning a new model or behavior often does not come for free but involves a certain cost. For example, gathering informative data can be challenging due to physical limitations, or updating models can require extensive computation. Moreover, learning for autonomous agents often requires exploring new behavior and thus typically means deviating from nominal or desired behavior. Hence, the question when to learn is essential for the efficient and intelligent operation of autonomous systems.

Main Idea: Event-triggered Learning

We explain the main idea of event-triggered learning using the schematic in Fig. 1. The figure depicts a canonical problem, where one agent (‘Sending agent’ on the left) has information that is relevant for another agent at a different location (‘Receiving agent’). For instance, this setting is representative for remote monitoring scenarios, distributed sensor fusion, or two agents of a multi-agent network. For resource-efficient communication, a standard event-triggered state estimation (ETSE) architecture is used (shown in blue). The main contribution of this work is to incorporate learning into the ETSE architecture. By designing an event trigger also for model learning (in green), learning tasks are performed only when necessary. We next explain the core components of the proposed framework.

The sending agent in Fig. 1 monitors the state of a dynamic process (either directly measured or obtained via state estimation) and can transmit this state information to the remote agent via a network link. In order to save network resources, an event-triggered protocol is used. The receiving agent makes model-based predictions to anticipate the state at times of no communication.
The key idea of this work is to trigger model learning whenever communication rates deviate significantly from what is expected. We show that the probability distribution of inter-communication times is fully parametrized with expected inter-communication times.

The sending agent implements a copy of the same prediction parameters. Since communication itself is triggered by model-based state predictions we obtain a tractable feature to quantify model accuracy by analyzing the communication pattern. Thus, we obtain improved models from data whenever there is demand and further, achieve superior communication rates. In addition, we avoid analyzing raw output data, which is possibly multidimensional and highly correlated.

The updated model is then shared with the remote agent to improve its predictions. Because performing a learning task is costly itself (e.g., involving computation and communication resources, as well as possibly causing deviation from the control objective), we propose event-triggering rules also for model learning.

While the idea of using event triggering to save communication in estimation or control is quite common by now, this work proposes event triggering also on a higher level. Triggering of learning tasks yields improved prediction models, which are the basis for ETSE at the lower level.

**Related Work**

Various event-triggered control [5, 6] and state estimation [2, 4, 7] algorithms have been proposed for improving resource usage in NCSs. Approaches differ, among others, in the type of models that are used for predictions. The send-on-delta protocol [8] assumes a trivial constant model by triggering when the difference between the current and lastly communicated value passes a threshold. This protocol is extended to linear predictions in [9], which are obtained from approximating the signal derivative from data. More elaborate protocols use dynamics models of the observed process, which typically leads to more effective triggering [10–12]. None of these works considers model learning to improve prediction accuracy, as we do herein.

In order to obtain effective learning triggers, we take a probabilistic view on inter-communication times (i.e., the time between two communication events) and trigger learning experiments whenever the expected communication differs from the empirical. A similar interpretation of inter-communication times is considered in [13], where NCSs are modeled as jump diffusion processes, and the probabilistic point of view is used to infer stability results. We extend this idea with statistical convergence results to design the event trigger for learning.

Adaptive control [14] aims to improve control performance by adapting model parameters continuously. Under certain persistent excitation conditions, convergence to a reference trajectory and even parameter convergence of the system can be guaranteed. However, divergence is a serious concern during nominal operation, where the persistent excitation conditions are not necessarily satisfied. By separating learning from nominal behavior, we ensure resource efficiency and informative data, which is collected in dedicated learning experiments.

Adaptive filtering, change, fault, and anomaly detection have been developed to identify defective and malfunctioning systems for a variety of applications. See [15–18] for an overview. In this article, we develop a method that is naturally related to these research areas, however, closely tailored to networked systems. Event-triggered learning combines tools from stochastical analysis and statistics to predict and analyze inter-communication times. Due to their advantageous statistical properties such as being one-dimensional, independent, and identically distributed, we can provide guarantees for the developed learning triggers.

Comparing expected behavior with observed realizations is also commonly done in cognitive science to model human learning [19]. This effect is often quantified with the aid of internal models and anticipation.
along surprise boundaries [20]. On an abstract point of view, the ideas are very similar to ETL. However, the concrete implementation and considered systems differ significantly. Albeit, there exists a natural connection to learning in biological systems.

Contributions

In detail, this article makes the following contributions:

- introducing the novel idea of event-triggered learning;
- derivation of a data-driven learning trigger based on statistical properties of inter-communication times;
- probabilistic guarantees ensuring the effectiveness of the proposed learning triggers;
- derivation of observation-based learning triggers, which are in particular covering the Kalman filter setting;
- demonstration of improved prediction accuracy in numerical simulations.

Preliminary results were presented in the conference paper [21] and significantly extended herein. The conference paper [21] solely considers learning triggers based on the expected value and for perfect state knowledge. Furthermore, the theoretical properties of the sample-based triggers are improved in this article.

2 Problem Formulation

In this section, we make the problem of event-triggered learning precise for linear Gaussian time-invariant systems. The framework is developed in the context of NCSs and primarily focuses on limited bandwidth. Information exchange over networks is abstracted to be ideal in the sense that there are no package drops or delays. First, we state the problem formulation for continuous time systems and afterward, address the discrete time case separately since the technical details differ slightly. In Sec. 7, the problem is extended to output measurements and in particular, the Kalman filter setting.

2.1 Continuous Time Formulation

Let \((S, \mathcal{F}, (\mathcal{F}_t)_{t \geq 0}, \mathbb{P})\) be a filtered probability space and \(X(t) \in \mathbb{R}^n\) a stochastic process, indexed by time \(t \geq 0\). Furthermore, assume \(X(t)\) (cf. 'Process' block in Fig. 1) is a solution to the following linear stochastic differential equation (SDE)

\[
dX(t) = AX(t)dt + QdW(t), \quad X(0) = x_0. \tag{1}
\]

Solutions to SDE (1) are well investigated and also known as Ornstein-Uhlenbeck processes [22, 23]. Further, let \(A \in \mathbb{R}^{n \times n}\) be a negative definite matrix, which can be obtained by applying a local feedback controller and considering the stable closed loop dynamics. Assume \(Q \in \mathbb{R}^{n \times n}\) is a positive definite matrix and \(W(t) \in \mathbb{R}^n\) a standard Wiener process that models process noise. We denote the system parameters as \(\theta = (A, Q)\) and models as \(\hat{\theta} = (\hat{A}, \hat{Q})\).

For the model-based predictions (cf. 'Model-based Predictions' block in Fig. 1), we use the expected value of system (1), which coincides with the open loop predictions of the deterministic system

\[
d\hat{X}(t) = \hat{A}\hat{X}(t)dt, \quad \hat{X}(0) = x_0. \quad (2)
\]

Due to the stochasticity of the system, the prediction error will almost surely leave any predefined domain after sufficient time. Event-triggered communication (cf. 'State Trigger' block in Fig. 1) bounds the prediction error by resetting the open loop predictions \(\hat{X}(t)\) to the current state \(X(t)\). Further, the binary event trigger

\[
\gamma_{\text{state}} = 1 \iff \|X(t) - \hat{X}(t)\|_2 \geq \delta, \tag{3}
\]

is only activated, when the error threshold \(\delta > 0\) is crossed and hence, limits communication to necessary instances. The corresponding inter-communication time is defined as

\[
\tau := \inf\{t \in \mathbb{R} : \|X(t) - \hat{X}(t)\|_2 \geq \delta\}, \tag{4}
\]

and realizations of this random variable are denoted as \(\tau_1, \ldots, \tau_n\).

Assumption 1 We assume \(\tau \leq \tau_{\text{max}} < \infty\). Therefore, communication is enforced after \(\tau_{\text{max}}\).

Bounded communication times are usually implemented in real-world applications to detect defect agents, which never communicate. For the design of the final learning trigger, the assumption could be omitted, however, it is useful for intermediate results such as the expectation-based learning trigger.

In this article, we address the problem of designing learning triggers (cf. 'Learning Trigger' block in Fig. 1) based on inter-communication time analysis. Since \(\tau\) can be fully parametrized by \(\theta\), we can derive an expected distribution based on the model \(\theta\) and test if empirically observed inter-communication times are drawn from this distribution. Further, this statistical analysis yields theoretical guarantees, which are obtained from concentration inequalities and ensure that the derived learning triggers are effective. Therefore, we design a method to perform dedicated learning experiments on necessity and update models \(\theta\) in an event-triggered fashion.

2.2 Discrete Time Formulation

Micro controllers and sensors are crucial elements for real world applications. Since digital data processing mostly
works with discrete time data, we provide an additional discrete time formulation of the considered problem. In principle, the problem formulation does not change. However, some technical details differ such as that the inter-communication times from (4) need to be treated differently due to discontinuities in the states. We obtain

\[ x(k + 1) = Ax(k) + \varepsilon(k), \quad x(0) = x_0, \]  
with discrete time index \( k \in \mathbb{N} \) and state \( x(k) \in \mathbb{R}^n \). Furthermore, we assume \( A \in \mathbb{R}^{n \times n} \) has all eigenvalues within the unit sphere and \( \varepsilon(k) \sim N(0, Q) \) with \( Q \in \mathbb{R}^{n \times n} \) being symmetric and positive definite. The model-based predictions (cf. 'Model-based Predictions' block in Fig. 1) are obtained through

\[ \hat{x}(k + 1) = \hat{A}\hat{x}(k), \]  
which yields the trigger

\[ \gamma_{\text{state}} = 1 \iff \|x(k) - \hat{x}(k)\|_2 \geq \delta. \]  

We define the system parameters and model as \( \theta = (A, Q) \) and \( \hat{\theta} = (\hat{A}, \hat{Q}) \). Hence, we obtain the inter-communication times

\[ \tau^d := \min\{k \in \mathbb{N} : \|x(k) - \hat{x}(k)\|_2 \geq \delta\}, \]  
and denote realizations of \( \tau^d \) as \( \tau^{d_1}, \ldots, \tau^{d_n} \).

3 Communication as Stopping Times

In this section, we characterize inter-communication times (Eq. (4)) as stopping times of the prediction error process. The inter-communication time \( \tau \) is a random variable and depends on the stochastic system (1). We seek to compare model-based expectations to observed data in order to detect significant inconsistencies between \( \theta \) and \( \hat{\theta} \). The core idea of the learning triggers comes down to deriving expected stopping time distributions based on the model \( \hat{\theta} \) and then analyzing how likely it is that observed stopping times \( \tau_1, \ldots, \tau_n \) are drawn from this distribution.

**Assumption 2** In the following, we want to investigate how the system parameters \( \theta \) are parametrizing the probability distribution of \( \tau \). Therefore, we assume \( \hat{\theta} = \theta \) in order to derive model-based statistical properties of \( \tau \). Later on, we will test the hypothesis that empirical inter-communication times are drawn from the model-induced probability distribution.

We define the error process between true state \( X(t) \) and prediction \( \hat{X}(t) \) as

\[ Z(t) := X(t) - \hat{X}(t) \]  
and show next that \( Z(t) \) is an Ornstein-Uhlenbeck process.

**Lemma 3** The process \( Z(t) \) is an Ornstein-Uhlenbeck process and subject to the SDE

\[ dZ(t) = AZ(t)dt + QdW(t), \quad Z(0) = 0. \]  

**Proof.** We can write the solutions of the SDE (1) as

\[ X(t) = e^{At}x_0 + \int_0^t e^{A(t-s)}QdW(s), \]  
as shown in [24, Sec. 3.7]. For the prediction part, we obtain

\[ \hat{X}(t) = e^{At}x_0. \]  

Thus, the error process becomes

\[ Z(t) = \int_0^t e^{A(t-s)}QdW(s), \]  
which yields the desired result. \( \square \)

Next, we introduce inter-communication times rigorously with respect to the stochastic process \( Z(t) \). Assume \( \mathcal{F}_t = \sigma(Z_s : s \leq t) \) is the natural filtration on the given probability space and \( \tau \) a stopping time with respect to \( \mathcal{F}_t \). In particular, we consider the first exit time of the stochastic process \( Z(t) \) from a sphere with radius \( \delta \)

\[ \tau := \inf\{t \in \mathbb{R} : \|Z(t)\|_2 > \delta\}, \]  
which precisely coincides with (4). Hence, we use the terms stopping times and inter-communication times synonymously in this article.

After each communication instance, we reset the process \( Z(t) \) and set it to zero again by correcting \( X(t) \) to \( \hat{X}(t) \). The sample paths of the process \( Z(t) \) are (almost surely) continuous between two inter-communication times, which follows from the existence and uniqueness theorem of solutions to SDEs (cf. [23]). Therefore, we can precisely quantify the moment when the error threshold (3) is crossed. We will make this statement precise in the following section.

3.1 Stopping Times and Boundary Value Problems

The following result connects the distribution of \( \tau \) with system parameters \( \theta \) through nonlinear boundary value problems.

**Theorem 4 (First Exit Probabilities)** Consider the stopping time \( \tau = \inf\{t \in \mathbb{R} : \|Z(t)\|_2 > \delta\} \) and the
boundary value problem

\[ G_v(z,t) = \frac{\partial v(z,t)}{\partial t} \text{ in } \Omega \times \mathbb{R}^+, \]

\[ v(z,0) = 1 \text{ in } \Omega, \]

\[ v(z,t) = 0 \text{ on } \partial\Omega \times \mathbb{R}^+, \]

with gradient \( \frac{\partial v(z)}{\partial z} \), Hessian \( \frac{\partial^2 v(z)}{\partial z^2} \), and \( \Omega \) a \( n \)-dimensional sphere with radius \( \delta \). Furthermore, we define the differential operator \( G_v(z) = \frac{\partial v(z)}{\partial z} A_z + \frac{1}{2} \text{trace} \left[ Q^T \frac{\partial^2 v(z)}{\partial z^2} Q \right] \). Then, there exists a smooth unique solution with regularity \( v(z,t) \in C^0((t_0, \tau_{\text{max}})) \) in time for any \( t_0 > 0 \) to the boundary value problem (15). Furthermore, the solution coincides with the exit probabilities \( v(z_0,t) = 1 - \mathbb{P}[t < \tau|Z(t) = z_0] \) of the stochastic process \( Z(t) \).

**PROOF.** The proof of this theorem can be found for general Itô diffusions with smooth coefficients in [25]. Here, we use the result for an Ornstein-Uhlenbeck process (1) and a \( n \)-dimensional sphere as the target domain. The differential operator (or generator) \( G \) corresponding to Ornstein-Uhlenbeck processes is well investigated and known to satisfy the here required smoothness assumptions, see e.g., [22, 23].

The above theorem yields an existence and uniqueness result for the boundary value problem (15) and, therefore, connects system parameters \( \theta \) to intercommunication times \( \tau \). In particular, we conclude that the design of learning triggers based on stopping times is well posed, since there is a continuous connection between model parameters and stopping times due to the above regularity results. Hence, small changes in \( \theta \), which is \( A \) and \( Q \), result in small changes in \( \tau \).

**Remark 5** The expected value [24, Sec. 7.2] and, therefore, also higher moments of \( \tau \) can also be stated as solutions to similar nonlinear boundary value problems with respect to the differential operator \( G \).

The domain \( \Omega \), on which the partial differential equation (PDE) (15) is solved, is induced by the specific choice of the state trigger (3). It is in general not possible to obtain analytical solutions, and hence, numerical approximations are required. There are sophisticated methods such as the \( hp \)-discontinuous Galerkin methods discussed in [25], which yield accurate solutions for all starting points \( Z(0) \in \Omega \). Due to the design of the state trigger, we only require the solution for one specific starting point, which is zero. Hence, we will use Monte Carlo simulations herein in order to approximate the statistical properties of \( \tau \).

### 3.2 Monte Carlo Approximations

Next, we describe how we obtain statistical properties of \( \tau \) such as expected value \( \mathbb{E}[\tau] \), variance \( \mathbb{V}[\tau] \), and cumulative distribution function (CDF) \( F(t) \) with the aid of sample-based methods and hence, without solving nonlinear PDEs such as (15). Given the system parameters \( \theta \), we can simulate trajectories of the stochastic process (1) with the aid of numerical sample methods such as the Euler-Maruyama scheme (cf. [26, Sec. 10.2] for an introduction to numerical solutions of SDEs).

In order to obtain independent and identically distributed (i.i.d.) samples \( \tau_1, \ldots, \tau_n \), we sample the process \( Z(t) \) and restart from zero after reaching the threshold \( \delta \). Alternatively, we could also simulate \( X(t) \) and \( \tilde{X}(t) \) and set the predictions \( \tilde{X}(\tau) \) to the true value \( X(\tau) \) when communication is triggered. The statistical properties of the corresponding stopping times do not differ, because the processes \( Z(t) \) and \( X(t) - \tilde{X}(t) \) are indistinguishable. Furthermore, stable Ornstein-Uhlenbeck processes \( X(t) \) are stationary and satisfy the strong Markov property, which generalizes the Markov property to stopping times.

For given i.i.d. random variables, we can approximate the expected value with \( \frac{1}{n} \sum_{i=1}^{n} \tau_i \) and the CDF with \( \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{[\tau_i \leq t]} \), where \( \mathbb{1} \) is the indicator function. Quantifying the convergence speed of the above approximation will be vital in designing learning triggers, which is done in the next section.

### 4 Learning Trigger Design for Continuous Time

In this section, we design the learning trigger \( \gamma_{\text{learn}} \) to detect a change in a dynamical system based on the inter-communication time \( \tau \).

#### 4.1 Concentration Inequalities

The following results will form the backbone of the later derived learning triggers. Concentration inequalities quantify the convergence speed of empirical distributions to their analytical counterparts. In particular, Hoeffding’s inequality bounds the expected deviation between mean and expected value. Further, we also consider the Dvoretzky-Kiefer-Wolfowitz (DKW) inequality, which compares empirical and analytical CDF functions and the bounds the error between them uniformly. Essentially we test if observed data fits the distribution, which is induced by the model \( \theta \) (cf. Sec. 3).

If the distributions do not match, we conclude an unfit model and update \( \theta \) through model learning.

**Lemma 6 (Hoeffding’s Inequality [27])** Let \( \tau_1, \ldots, \tau_n \) be i.i.d. bounded random variables, s. t. \( \tau_i \in \mathbb{R} \).
We will first design learning triggers around the Hoeffding’s inequality and later move on to richer statistical information. Therefore, we also want to analyze the convergence speed of the empirical CDF function.

**Lemma 7 (DKW Inequality [28])** Assume \( \tau_1, \ldots, \tau_n \) are i.i.d. random variables with CDF \( F(t) \) and empirical CDF \( F_n(t) \). Then

\[
P \left[ \sup_{t \in \mathbb{R}} |F_n(t) - F(t)| > \kappa \right] \leq 2 \exp(-2n \kappa^2),
\]

(17)

### 4.2 Expectation-based Learning Trigger

Next, we construct a learning trigger \( \gamma_{\text{learn}} \) based on the expected value \( E[\tau] \).

#### 4.2.1 Exact Learning Trigger

Based on the foregoing discussion, we propose the following learning trigger:

\[
\gamma_{\text{learn}} = 1 \iff \left| \frac{1}{n} \sum_{i=1}^{n} \tau_i - E[\tau] \right| \geq \kappa_{\text{exact}},
\]

(18)

where \( \gamma_{\text{learn}} = 1 \) indicates that a new model shall be learned; \( E[\tau] \) is the analytical expected value, which is based on the model \( \hat{\theta} \); and \( \tau_1, \tau_2, \ldots, \tau_n \) are the last \( n \) empirically observed inter-communication times (\( \tau_i \) the duration between two state triggers (3)). The horizon \( n \) is chosen to yield robust triggers. The threshold parameter \( \kappa_{\text{exact}} \) quantifies the error we are willing to tolerate. There are some examples, where it is possible to compute \( E[\tau] \) analytically, however, it is in general intractable. Hence, we also propose the approximated learning trigger, which takes the approximations for the statistical analysis into account. We denote (18) as the **exact learning trigger** because it involves the exact expected value \( E[\tau] \), as opposed to the trigger derived in the next subsection, which is based on a Monte Carlo approximation of the expected value.

Even though the trigger (18) is meant to detect inaccurate models, there is always a chance that the trigger fires not due to an inaccurate model, but instead due to the randomness of the process (and thus randomness of inter-communication times \( \tau_i \)). Even for a perfect model, (18) may trigger at some point. False positives are inevitable due to the stochastic nature of the problem. However, we obtain a confidence interval, which contains the empirical mean with high confidence. If observations violate the derived confidence interval, we conclude that distributions do not match and learning is beneficial. Therefore, we propose to choose \( \kappa_{\text{exact}} \) to yield effective triggering with a user-defined confidence level. For this, we make use of Hoeffding’s inequality. We then have the following result for the trigger (18):

**Theorem 8 (Exact learning trigger)** Assume \( \tau_1, \ldots, \tau_n \) are i.i.d. random variables and the parameters \( \alpha, n, \) and \( \tau_{\text{max}} \) are given. If the trigger (18) gets activated (\( \gamma_{\text{learn}} = 1 \)) with

\[
\kappa_{\text{exact}} = \tau_{\text{max}} \sqrt{\frac{1}{2n} \ln \frac{2}{\alpha}},
\]

(19)

then

\[
P \left[ \left| \frac{1}{n} \sum_{i=1}^{n} \tau_i - E[\tau] \right| \geq \kappa_{\text{exact}} \right] \leq \alpha.
\]

(20)

**PROOF.** Substituting (19) for \( \kappa_{\text{exact}} \) into the right-hand side of Hoeffding’s inequality yields the desired result.

The theorem quantifies the expected convergence rate of the empirical mean to the expected value for a perfect model. This result can be used as follows: the user specifies the desired confidence level \( \alpha \), the number \( n \) of inter-communication times considered in the empirical average, and the maximum inter-communication time \( \tau_{\text{max}} \). By choosing \( \kappa_{\text{exact}} \) as in (19), the exact learning trigger (18) is guaranteed to make incorrect triggering decisions (false positives) with a probability of less than \( \alpha \).

#### 4.2.2 Approximated Learning Trigger

As discussed in Sec. 3, obtaining \( E[\tau] \) can be difficult and computationally expensive. Instead, we propose to approximate \( E[\tau] \) by sampling \( \tau \). For this, we simulate the Ornstein-Uhlenbeck process \( Z(t) \) (10) until it reaches a sphere with radius \( \delta \) for \( m \) times, and average the obtained stopping times \( \hat{\tau}_1, \ldots, \hat{\tau}_m \). This yields the **approximated learning trigger**

\[
\gamma_{\text{learn}} = 1 \iff \left| \frac{1}{n} \sum_{i=1}^{n} \tau_i - \frac{1}{m} \sum_{i=1}^{m} \hat{\tau}_i \right| \geq \kappa_{\text{approx}}.
\]

(21)

The Monte Carlo approximation leads to a choice of \( \kappa_{\text{approx}} \), which is different from \( \kappa_{\text{exact}} \).

**Theorem 9 (Approximated Learning Trigger)** Assume \( \tau_1, \ldots, \tau_n \) and \( \hat{\tau}_1, \ldots, \hat{\tau}_m \) are i.i.d. random
variables. If the trigger (21) gets activated with
\[ \kappa_{\text{approx}} = \sqrt{\frac{n + m}{2nm} \tau^2_{\text{max}} \ln \frac{2}{\alpha}} \] (22)
then
\[ P \left[ \left| \frac{1}{n} \sum_{i=1}^{n} \tau_i - \frac{1}{m} \sum_{i=1}^{m} \hat{\tau}_i \right| \geq \kappa_{\text{approx}} \right] \leq \alpha. \] (23)

**PROOF.** First, we introduce an alternative formulation of Hoeffding’s inequality (16)
\[ P \left[ \left| \frac{1}{n} \sum_{i=1}^{n} \tau_i - \frac{1}{m} \sum_{i=1}^{m} \hat{\tau}_i - (E[\tau] - E[\hat{\tau}]) \right| > \kappa_{\text{approx}} \right] \leq 2 \exp \left( - \frac{2\kappa^2_{\text{approx}}}{(m^{-1} + n^{-1})\tau^2_{\text{max}}} \right), \]
which was already stated in the original article by Hoeffding [27] as a corollary (Eq. 2.6). Here, we assume that \( \tau \) and \( \hat{\tau} \) are identically distributed and, therefore, the analytical expected values cancel out. Rearranging for \( \kappa_{\text{approx}} \) yields the desired result. \( \Box \)

**Remark 10** It is possible to extend the presented results to higher moments of order \( l \) by substituting \( \tau^l \) with a new variable and then applying Hoeffding’s inequality.

### 4.3 Density-based Learning Trigger

Analyzing the expected values is in general not enough to distinguish random variables since higher moments such as variance can differ. Therefore, we propose to look at the CDF and build learning triggers around the DKW inequality (17) and use richer statistical information. We propose the following learning trigger:

\[ \gamma_{\text{learn}} = 1 \iff \sup_{t \in \mathbb{R}} |\hat{F}_m(t) - F_n(t)| > \kappa_{\text{approx}}, \] (24)

Next, we analyze the theoretical properties of the density-based learning trigger.

**Theorem 11 (Exact Density Learning Trigger)**

Assume \( \tau_1, \ldots, \tau_n \) are i.i.d. random variables with CDF \( F(t) \) and empirical CDF \( F_n(t) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\tau_i \leq t} \). If the learning trigger (24) gets activated (\( \gamma_{\text{learn}} = 1 \)) with

\[ \kappa_{\text{exact}} = \sqrt{\frac{1}{2\alpha} \ln \frac{2}{\alpha}}, \] (25)

then
\[ P \left[ \sup_{t \in \mathbb{R}} |F(t) - F_n(t)| > \kappa_{\text{exact}} \right] \leq \alpha. \] (26)

**PROOF.** Follows directly from the DKW Inequality (17). \( \Box \)

Further, we can follow the reasoning from before and obtain the sample-based trigger
\[ \gamma_{\text{learn}} = 1 \iff \sup_{t \in \mathbb{R}} |\hat{F}_m(t) - F_n(t)| > \kappa_{\text{approx}}, \] (27)

where \( \hat{F}_m(t) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{\hat{\tau}_i \leq t} \) and \( \hat{\tau}_i \) are obtained by creating samples based on the model \( \hat{\theta} \). This trigger is essentially the well established two-sample Kolmogorov-Smirnov (KS) test, which was first proposed in [29] and is for instance summarized in [30].

**Theorem 12 (Two-sample KS Learning Trigger)**

Assume \( \tau_1, \ldots, \tau_n \) and \( \hat{\tau}_1, \ldots, \hat{\tau}_m \) are i.i.d. random variables with empirical CDFs \( F_n(t) \) and \( \hat{F}_m(t) \). If the trigger (27) gets activated with

\[ \kappa_{\text{approx}} = \sqrt{\frac{n + m}{2nm} \ln \left( \frac{2}{\alpha} \right)}, \] (28)

then
\[ P \left[ \sup_{t \in \mathbb{R}} |\hat{F}_m(t) - F_n(t)| > \kappa_{\text{exact}} \right] \leq \alpha. \] (29)

**PROOF.** Follows directly from the two-sample KS test, which is discussed in [29] and [30]. \( \Box \)

The asymptotic behavior of the expectation-based trigger (22) and density-based trigger (28) are the same. However, the density-based learning triggers do not depend on \( \tau_{\text{max}} \) and consider richer statistical information, which can be an advantage and will be discussed in detail in the experimental sections.

### 5 Learning Trigger Design for Discrete Time

Based on the previous discussion, we will now highlight how to apply the previously derived learning triggers to discrete time systems. The random variables \( \tau \) and \( \tau^d \) can differ significantly due to discretization effects. Intuitively, this effect can be thought of as the continuous time process crossing the \( \delta \)-threshold and returning within the discretization time. Therefore, the discrete-time process has no possibility of observing the crossing and hence, stopping times tend to be larger for discrete time systems. For small time steps the difference tends to be negligible, and \( \tau^d \) converges to \( \tau \) in the limit. Next, we show that the approximated learning triggers transfer without any modification to the discrete time system.
Theorem 13 (Discrete Time Learning Trigger)
Assume \( \hat{\theta} \) corresponds to the discrete time system (5). Then, the previously derived approximated learning triggers (21) and (27) are applicable without any further modification.

PROOF. The derived learning triggers test if given observations of inter-communication times \( \tau_1, \ldots, \tau_n \) are drawn from a distribution, which is induced by \( \theta \). For the approximated learning triggers, the induced distribution is estimated via Monte Carlo approximations. The actual shape of the distribution is irrelevant for the test statistic. It is important to adjust the model \( \theta = (A, Q) \) (cf. 2.2) to discrete time in order to sample from the correct distribution. Otherwise, the trigger does not change at all.

Remark 14 To the best of the authors’ knowledge, it is intractable to derive statistical properties (e.g., \( E[\tau^d] \)) of \( \tau^d \) analytically. Given these values, it would be straightforward to apply the exact learning triggers (18) and (24) to the discrete time system as well.

Remark 15 The presented ideas can also be extended to systems with constant load disturbances and more complex noise such as colored noise instead of white noise.

6 Simulation – Reduced Communication

In this section, we illustrate the main ideas of the developed learning triggers based on a one dimensional stable linear Gaussian system.

Setup

First, we introduce the system, and afterward, we apply the learning trigger in order to demonstrate how to detect an inaccurate model. We consider the dynamical system

\[
x(k + 1) = 0.9x(k) + \epsilon(k)
\]

with \( \epsilon(k) \sim N(0, 1) \). Further, we assume the disturbed model \( \hat{\theta} = (0.8, 1) \) and hence, we obtain the predictions

\[
\hat{x}(k + 1) = 0.8\hat{x}(k).
\]

To bound the prediction error we deploy the state trigger (7) with \( \delta = 3 \). In Fig. 2, we can see a trajectory of states subject to system (30) as a black dashed line and the model-based predictions \( \hat{x}(k) \) in blue. Whenever \( \gamma_{state} = 1 \) we set \( \hat{x}(\tau) \) to \( x(\tau) \) and denote the inter-communication times \( \tau_1, \ldots, \tau_n \), which are highlighted with a vertical dotted black line. The error signal and the communication instances are depicted in the second and third subplot. The last subplot shows the average intercommunication rate before and after updating the model.

buffer and afterward, the learning trigger (21) compares if the mean computed on this buffer lies outside the expected confidence interval, which is the case here. After updating the model, we can see that the model-based expectation coincides with the mean on the next buffer. More details on the expectation-based learning trigger and the difference to the density-based learning trigger are presented in the next section.

Expectation and Density Triggers

Next, we will demonstrate how the learning trigger detects the inaccurate model. Based on the model, we per-
form $m = 100,000$ Monte Carlo simulations and obtain the expected stopping times $\hat{\tau}$ with mean $E[\hat{\tau}] \approx 28.6$. Further, let $n = 1000$ observations of actual inter-communication times $\tau$ be given, which yields $E[\tau] \approx 15.9$. Based on Equation (22), together with $\tau_{\text{max}} = 100$ and $\alpha = 0.05$, we can derive a confidence bound of 7.9. Indeed, the observed deviation is 12.6 and hence, the learning trigger (21) gets active. The probability of observing this deviation through random fluctuations is less than 0.0002. The expected values and the confidence interval are depicted in Fig. 3 with dashed lines. Further, we can also see empirical CDF functions and the corresponding confidence interval around the model-based CDF. For the density-based learning trigger (27) it is enough if the CDFs deviate in a single point by more than the critical threshold, which is 0.079 and can be obtained with Equation (28). The actual deviation is also depicted in Fig. 3 and clearly outside of the confidence interval. Both learning triggers detect the inaccurate model successfully and after setting $\hat{\theta} = \theta$, we can see in Fig. 4 that expected and empirical distributions coincide. Further, the communication in the system reduces due to the better model to mean inter-communication times of 19. Clearly, the model has also be communicated at some point. However, this happens very rarely and in particular, when there is a significant change in the system. A higher dimensional example will be considered in Sec. 8, where we also consider output measurements. Further, event-triggered learning has been applied in [21] to a cart-pole system, where communication was successfully reduced.

Fig. 3. Statistical properties of expected communication $\hat{\tau}$ (in blue) and observed inter-communication times $\tau$ (in red). The dashed lines capture the expected values, and the shaded blue region is a confidence interval which should contain the dashed red line with a 95% probability. As there is a significant deviation between observation and expectation, the learning trigger initiates to relearn the model. The solid lines represent the CDF functions and also here, the empirical distribution is not contained within the confidence bounds, which triggers learning.

Fig. 4. After updating the model $\hat{\theta}$ to the true system parameters $\theta$, the estimated stopping times $\hat{\tau}$ coincide with empirical stopping times $\tau$. As a direct consequence, the communication behavior is improved as the stopping times increase.

Fig. 5. Proposed event-triggered learning architecture for a networked control problem between two agents with output measurements and an intermediate Kalman filtering block. The proposed extension (in green) analyzes stopping times of the Kalman filter process.

7 Output Measurements

So far we assumed that we are able to measure the full state. In the following, we will drop this assumption and consider systems, where only part of the state can be measured. This yields the system

$$x(k + 1) = Ax(k) + \epsilon(k)$$
$$y(k) = Cx(k) + \nu(k),$$

with output measurements $y(k) \in \mathbb{R}^m$. Further, let $A \in \mathbb{R}^{n \times n}$ and $C \in \mathbb{R}^{n \times m}$. The system is again assumed to be stochastic with process noise $\epsilon(k) \sim \mathcal{N}(0, Q)$ and observation noise $\nu(k) \sim \mathcal{N}(0, R)$, which are independent of each other. We also assume that $A$ is stable and the pair $(A, C)$ is observable. Hence, the system is parametrized by $\theta = (A, C, Q, R)$ and modeled by $\hat{\theta} = (\hat{A}, \hat{C}, \hat{Q}, \hat{R})$.

To reconstruct the full state, we use a Kalman filter (cf. ‘Kalman Filter’ block in Fig. 5), which is the optimal filter for linear Gaussian systems [31]. Here, we consider
the steady state Kalman filter and obtain
\[ \dot{x}(k + 1) = \dot{A} \dot{x}(k) + K \left( y(k + 1) - \hat{C} \hat{A} \dot{x}(k) \right), \quad (33) \]
where \( K \in \mathbb{R}^{m \times n} \) is the Kalman filter gain.

Ideally, we want to use the Kalman filter states \( \hat{x}(k) \) on the receiving agent’s site. However, this would require periodic communication of the Kalman filter states \( \hat{x}(k) \), or measurements \( y(k) \), which we try to avoid. Exactly as in Sec. 2, we run a model-based prediction step in the absence of data and obtain
\[ \dot{x}(k + 1) = \dot{A} \dot{x}(k), \quad (34) \]
which yields the state trigger
\[ \gamma_{\text{state}} = 1 \iff \| \dot{x}(k) - \hat{x}(k) \|_2 \geq \delta. \quad (35) \]
This event trigger ensures a bound between \( \dot{x}(k) \) and \( \hat{x}(k) \). Hence, the inter-communication time is defined as
\[ \tau^o := \min\{k \in \mathbb{N} : \| \dot{x}(k) - \hat{x}(k) \|_2 \geq \delta \} \quad (36) \]
and we denote realizations of this random variable as \( \tau^o_1, \ldots, \tau^o_n \). Next, we show how to design a learning trigger with output measurements by reducing the problem to the previous setting with the aid of the innovation sequence.

### 7.1 Innovation Emulation

The key idea of this approach is based on treating the Kalman filter sequence as a stochastic process on its own right and investigating the distribution of \( \dot{x}(k) \) in a weak sense, instead of pathwise realizations, which depend on \( y(k) \) and \( x(k) \). With the aid of the innovation sequence, we can derive an auto-regressive structure. The innovation of a filter is defined as
\[ I(k) = y(k) - C \hat{x}(k). \quad (37) \]
Furthermore, it is well known that \( I(1), \ldots, I(n) \) are independent normal distributed random variables with \( I(k) \sim \mathcal{N}(0, S) \). For a stationary Kalman filter, the covariance \( S \) is given by
\[ S = CPC^\top + R, \quad (38) \]
where \( R \) is the measurement noise and \( P \) the error covariance matrix, which can be obtained by solving the corresponding Riccati equation \([31, \text{Equation (4.4)}]\). Hence, we can formulate the Kalman filter as
\[ \dot{x}(k + 1) = \dot{A} \dot{x}(k) + K I(k), \quad (39) \]
and regard \( I(k) \sim \mathcal{N}(0, S) \) as a random variable. By regarding \( I(k) \) as process noise, we are back to the previously discussed problem and can apply the derived tools and learning triggers. Hence, we can effectively analyze the distribution of the corresponding stopping time with the previously derived tools. More precisely, we sample Equation (39) and obtain model-based stopping times \( \hat{\tau}^o \).

#### 7.2 The Threshold \( \delta \) and Stochastic Exit Problems

The intuition behind the threshold parameter \( \delta \) for the trigger \( \| X(t) - X(t) \|_2 \geq \delta \) is straightforward—it directly controls the error between state and open loop prediction. However, it is less clear how to chose the parameter \( \delta \) when there is a Kalman filter in the loop. The trigger \( \| X(t) - X(t) \|_2 \geq \delta \) bounds the error between Kalman filter and open loop prediction, which is slightly different and requires a careful choice of \( \delta \). Based on the proposed stopping time analysis, we can modify the threshold \( \delta \) in a structured way to obtain a new value \( \delta' \) as follows:

When resetting the open loop prediction \( \hat{X}(t) \) to \( X(t) \), we may also consider this with respect to the true state \( X(t) \). It is well known that \( \hat{X}(t) \sim \mathcal{N}(X(t), P) \) and therefore, we propose to reset the error process \( Z(t) \) to \( z_0 \sim \mathcal{N}(0, P) \) instead of \( z_0 = 0 \). Hence, it is possible to define \( \delta \) with respect to the process \( X(t) \) and analyze the inter-communication time for random initial points, which yields the stopping time \( \xi \). Afterward, we suggest tuning \( \delta' \) in order to make the stopping time \( \tau^o \) coincide in expectation with the reference stopping time \( \xi \). Indeed, this does not guarantee that we have a bound with respect to the true state, but that the trigger is well behaved in distribution.

### 8 Limitations and Insights

This section is divided into two parts. First, we illustrate that the CDF-based learning trigger has significant advantages over the learning trigger based on the expected value. For once, it is not dependend on \( \tau_{\text{max}} \) (cf. (22) and (28)) and, therefore, more sample efficient. Furthermore, there is a class of examples, where \( \theta \neq \hat{\theta} \), despite of, \( \mathbb{E}[\tau] = \mathbb{E}[\hat{\tau}] \). Intuitively, process noise is pushing the error out of the domain \( \Omega \), while the stable system dynamics are pulling the error back to zero. We construct the counterexamples by creating a hypersurface of models \( \hat{\theta} \), where the noise and stability effects cancel out. Afterward, we demonstrate unexpected behavior of the state trigger with output measurements (35), which results in interesting behavior of the learning trigger.

#### 8.1 Expected Value is not Enough

Consider the toy example (30) with \( \theta = (0.9, 1) \). In Fig. 4 we depict the optimal communication behavior
the Kalman filter states $\hat{x}$ coincides with the true system parameters. We consider $\hat{I} \sim I$ for the considered system with the model $\hat{\theta} = (0.5, 1.7)$. Expected and empirical expected values of inter-communication times are almost identical – bad performance is expected and also realized due to the bad prediction model. Nonetheless, they are only half of the optimal value with a perfect model and there is the possibility for improvement when the model is updated. Better models increase the possibility for improvement when the model is updated.

8.2 More Communication can be Better

More accurate models result in better predictions. However, we also demonstrate an rather unexpected effect - better models may lead to more communication in the Kalman filter setting. Better models increase the Kalman filter performance and thus, it is possible to track the unobserved states better. Therefore, it is possible to construct examples where communication increases, which is desirable for performance but unintuitive.

Consider System (32) with the matrices

$$
A = \begin{pmatrix}
1.000 & 0.010 & -0.005 & 0.000 \\
0.017 & 1.027 & -0.301 & -0.061 \\
0.000 & 0.000 & 0.997 & 0.009 \\
0.046 & 0.067 & -0.507 & 0.850
\end{pmatrix},
C^T = \begin{pmatrix}
1 & 0 \\
0 & 0 \\
0 & 1 \\
0 & 0
\end{pmatrix}
$$

that models the closed loop dynamics of a stabilized inverted pendulum. We assume process noise $\epsilon \sim \mathcal{N}(0, 0.1I_n)$ and observation $\nu \sim \mathcal{N}(0, 0.1I_2)$, where $I_n$ is the identity matrix of dimension $n$. Further, we assume that $\nu \sim \mathcal{N}(0, 0.5I_2)$ and that our model otherwise coincides with the true system parameters. We consider the Kalman filter states $\hat{x}(k)$ (cf. (33)), the predictions $\hat{x}(k)$ (cf. Equation (34)) and the state trigger

$$
\gamma_{\text{state}} = 1 \iff \|\hat{x}(k) - \bar{x}(k)\|_2 \geq 1. 
$$

We initialize $x(0) = \hat{x}(0) = \bar{x}(0) = 0$ and obtain the in Fig. 7 depicted distribution over stopping times. The expected model-based communication is derived via Monte Carlo simulation of the innovation process (39) and are lower than the empirical inter-communication times. Updating the model actually increases communication, which is because the Kalman filter improves and tracks the states $x(k)$ better, which is depicted in Fig. 8.

The first plot shows the tracking performance when a perfect model $\hat{\theta} = \theta$ in green. For the second plot, we changed the covariance of $\nu$ to $\mathcal{N}(0, 0.5I_2)$ and are lower than the empirical inter-communication times. Updating the model actually increases communication, which is because the Kalman filter improves and tracks the states $x(k)$ better, which is depicted in Fig. 8.

9 Discussion and Future Work

Event-triggered learning is proposed in this article as a novel concept to trigger model learning when needed. While this article mainly focuses on the rigorous design of learning triggers, the concept has already been applied in simulation and physical experiments and shown...
Fig. 8. State trajectories of the first dimension of a four dimensional system with output measurements. Worsening the model results in less communication because the Kalman filter performance decreases.

to lead to reduced communication. For this setting, we obtained (provably) effective learning triggers utilizing statistical stopping time analysis.

This article is the first to develop the concept of event-triggered learning. Extending the method to nonlinear dynamic systems, e.g., through Gaussian process model learning [32, 33], is an obvious next step we plan as future work. Instead of using predefined communication strategies it is possible to jointly learn them as discussed in [54]. Combining the presented ideas with model-based reinforcement learning is also a natural extension. First steps in this direction were taken in [35], where ETL is combined with control. Further, it should also be possible to generalize ETL to other signals such as control performance. While event-triggered learning has been motivated as an extension to already existing methods to reduce communication in NCSs, the concept generally addresses the fundamental question of when to learn, and thus potentially has much broader relevance.

Acknowledgements

The authors thank Dominik Baumann, René Geist, Alexander von Rohr, and Steve Heim for helpful comments and fruitful discussions.

References


