Event-triggered Learning

Friedrich Solowjow, Sebastian Trimpe

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

Abstract

Efficient exchange of information is an essential aspect of intelligent collective behavior. Event-triggered control and estimation achieve some efficiency by replacing continuous data exchange between agents with intermittent, or event-triggered communication. Typically, model-based predictions are used at times of no data transmission, and updates are sent only when the prediction error grows too large. The effectiveness in reducing communication thus strongly depends on the quality of the prediction model. In this article, we propose event-triggered learning as a novel concept to reduce communication even further and to also adapt to changing dynamics. By monitoring the actual communication rate and comparing it to the one that is induced by the model, we detect mismatch between model and reality and trigger learning of a new model when needed. Specifically for linear Gaussian dynamics, we derive different classes of learning triggers solely based on a statistical analysis of inter-communication times and formally prove their effectiveness with the aid of concentration inequalities.

Key words: Networked Control Systems; Statistical Analysis.

1 Introduction

Modern communication technology allows for connecting many devices and systems with each other and for sharing data and information in unprecedented ways. This enables applications such as mobile sensor networks, distributed robotics, and multi-vehicle systems, often subsumed as networked control systems (NCSs) [1]. Alongside the potential of NCSs come significant challenges for control design such as delayed transmission, packet drops, or limited bandwidth, which originate from the fact that a shared network is used for feedback. Many methods that have been proposed for addressing these challenges rely on accurate dynamics models. For instance, model-based predictions in event-triggered state estimation and control [2–5] can replace periodic communication up to a certain extent. Only if predictions become inaccurate or unmodeled behavior occurs, communication of sensor measurements is indispensable and thus triggered when necessary.

In this article, we propose to extend the paradigm of event triggering to model learning and introduce the novel idea of event-triggered learning (ETL). With ETL, we detect mismatch between model and true dynamics, and trigger identification of a new model whenever needed. We built ETL on top of a typical event-triggered state estimation architecture (see Fig. 1) and show that the new architecture can cope with changing dynamics and yields further communication savings in a NCS.

Classical approaches use first principles to model dynamical systems, which relies on human expert knowledge and lacks the flexibility to be adjusted online. The ability to learn is thus a fundamental aspect of future autonomous systems that are facing uncertain and changing environments. However, the process of learning a new model or behavior typically does not come for free, but involves a certain cost. For example, gathering informative data can be challenging due to physical constraints, 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 operation of autonomous systems. ETL addresses this question, and this article develops the concept specifically in the context of NCSs.
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. The true parameters $\theta$ of the process are unknown to both agents. In order to save network resources, an event-triggered protocol is used. The receiving agent uses a model (with parameters $\hat{\theta}$) of the process for predicting the state at times of no communication. The sending agent implements a copy of the same prediction and compares it to the current state in the ‘State Trigger’, which triggers a state communication whenever the prediction deviates too much from the actual state. This general scheme is standard in ETSE literature (see [4, 6, 7] and references therein). The effectiveness of this scheme in reducing communication will generally depend on the accuracy of the prediction, and, thus, the quality of the model $\hat{\theta}$.

The key idea of this work is to trigger model learning when communication rates deviate significantly from what is expected. 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 (‘Learning Trigger’). Newly learned models are then shared with the remote agent to improve its predictions. We show that the probability distribution of inter-communication times is fully parametrized by the system parameters $\theta$. Since communication itself is triggered by model-based state predictions, we obtain a tractable feature to quantify model accuracy by analyzing the communication pattern. Further, we avoid analyzing raw output data, which is possibly multidimensional and highly correlated. Thus, we propose a method to obtain improved models from data when needed, which leads to superior communication rates.

While the idea of using event triggering to save communication in estimation or control is quite common by now,
mance by adapting model parameters continuously. Under certain persistent excitation conditions and for certain system classes, convergence to a reference trajectory and even parameter convergence can be guaranteed. However, divergence is a serious concern during nominal operation, where the persistent excitation conditions are not necessarily satisfied. This is a major issue in practice where the persistent excitation conditions are not necessarily satisfied. This is a major issue in practice.

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 packet drops or delays. First, we state the problem formulation for continuous time systems. We then 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 \((S, \mathcal{F}, (\mathcal{F}_t)_{t \in \mathbb{R}_+}, \mathbb{P})\) 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)

\[
\begin{align*}
    dX(t) &= AX(t)dt + QdW(t), \quad X(0) = x_0. \\
    \end{align*}
\]

Solutions to the SDE (1) are well investigated and also known as Ornstein-Uhlenbeck processes [27, 28]. Further, let \(A \in \mathbb{R}^{n \times n}\) be a negative definite matrix, which may in practice be obtained by applying local feedback control and considering the stable closed-loop dynamics. Assume \(Q \in \mathbb{R}^{n \times n}\) is a positive definite matrix, \(W(t) \in \mathbb{R}^n\) a standard Wiener process that models process noise, and the initial point \(x_0 \in \mathbb{R}^n\) is known. We denote the system parameters as \(\theta = (A, Q)\) and models as \(\hat{\theta} = (\hat{A}, \hat{Q})\).

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

\[
\begin{align*}
    d\hat{X}(t) &= \hat{A}\hat{X}(t)dt, \quad \hat{X}(0) = x_0. \\
    \end{align*}
\]

Due to the stochasticity of the system, the prediction error will almost surely leave any predefined domain after sufficient time. Event-triggered communication (‘State Trigger’ 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

$$\gamma_{\text{state}} = 1 \iff \|X(t) - \bar{X}(t)\|_2 \geq \delta,$$  

(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

$$\tau := \inf\{t \in \mathbb{R} : \|X(t) - \bar{X}(t)\|_2 \geq \delta\},$$  

(4)

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

**Assumption 1** We assume $\tau \leq \tau_{\text{max}} < \infty$.

Bounded communication times are usually implemented in real-world applications to detect defec t agents, which never communicate. Hence, communication is enforced after $\tau_{\text{max}}$. For the design of the final learning trigger to be derived in this article, the assumption can 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 ('Learning Trigger' in Fig. 1) based on inter-communication time analysis. Since the probability distribution of $\tau$ 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 $\hat{\theta}$ in an event-triggered fashion.

### 2.2 Discrete Time Formulation

Since processing on microcontrollers or sensors mostly happens on synchronously sampled data, we provide an alternative discrete time formulation of the considered problem. In principle, the problem formulation does not change. However, some essential details differ; for example, the inter-communication times from (4) need to be treated differently due to discontinuities in the states.

The discrete time analogon to (1) is

$$x(k + 1) = A x(k) + \epsilon(k), \quad x(0) = x_0,$$  

(5)

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 strictly within the unit sphere and $\epsilon(k) \sim \mathcal{N}(0, Q)$ with $Q \in \mathbb{R}^{n \times n}$ being symmetric and positive definite. The model-based predictions are obtained through

$$\dot{x}(k + 1) = \dot{A} \dot{x}(k),$$  

(6)

which yields the trigger

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

(7)

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

$$\tau^d := \min\{k \in \mathbb{N} : \|x(k) - \dot{x}(k)\|_2 \geq \delta\}$$  

(8)

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 $\theta$ and then analyzing how likely it is that observed stopping times $\tau_1, \ldots, \tau_n$ are drawn from this distribution.

Assuming $\hat{\theta} = \theta$, we derive model-based statistical properties of $\tau$. Later on, we will test the hypothesis that empirical inter-communication times are indeed drawn from the derived distribution of $\tau$—if not, this will indicate $\hat{\theta} \neq \theta$; that is, the model does not match reality.

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

$$Z(t) := X(t) - \bar{X}(t)$$  

(9)

and show next that $Z(t)$ is an Ornstein-Uhlenbeck process.

**Lemma 2** The process $Z(t)$ is an Ornstein-Uhlenbeck process and subject to the SDE

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

(10)

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

$$X(t) = e^{At}x_0 + \int_0^t e^{A(t-s)}QdW(s),$$  

(11)
as shown in [29, Sec. 3.7]. For the prediction part, we obtain
\[ X(t) = e^{At}x_0. \] (12)
Thus, the error process becomes
\[ Z(t) = \int_0^t e^{A(t-s)}Q dW(t), \] (13)
which yields the desired result.

With this, we can now rigorously introduce inter-communication times with respect to the stochastic process \( Z(t) \). Assume \( \mathcal{F}_s = \sigma(Z_s : s \leq t) \) 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 \( \delta \)
\[ \tau := \inf\{t \in \mathbb{R} : \|Z(t)\|_2 > \delta\}, \] (14)
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 \( \tilde{X}(t) \) to \( X(t) \). 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. [28]). 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 characterizes the distribution of \( \tau \) in terms of the system parameters \( \theta \) through nonlinear boundary value problems.

**Theorem 3 (First Exit Probabilities)** Consider the stopping time \( \tau = \inf\{t \in \mathbb{R} : \|Z(t)\|_2 > \delta\} \) and the boundary value problem
\[ \mathcal{G}v(z,t) = \frac{\partial v(z,t)}{\partial t} \quad \text{in } \Omega \times \mathbb{R}^+, \]
\[ v(z,0) = 1 \quad \text{in } \Omega, \] (15)
\[ v(z,t) = 0 \quad \text{on } \partial \Omega \times \mathbb{R}^+, \]
with gradient \( \frac{\partial v(z,t)}{\partial z} \), Hessian \( \frac{\partial^2 v(z)}{\partial z^2} \), \( \Omega \) an \( n \)-dimensional sphere with radius \( \delta \), and the differential operator \( \mathcal{G} \) defined as \( \mathcal{G}v(z) = \frac{\partial v(z)}{\partial z}Az + \frac{1}{2} \text{trace} \left[ Q^T \frac{\partial^2 v(z)}{\partial z^2} Q \right] \).

Then, there exists a smooth unique solution with regularity \( v(z,t) \in C^0((t_0, \tau_{\max})) \) in time for any \( t_0 > 0 \) to the boundary value problem (15). Furthermore, the solution coincides with the exit probabilities \( v(z_0,t) = 1 - \mathbb{P}[t < \tau|Z(0) = z_0] \) of the stochastic process \( Z(t) \).

**PROOF.** The proof of this theorem can be found for general Itô diffusions with smooth coefficients in [30]. Here, we use the result for an Ornstein-Uhlenbeck process (1) and an \( n \)-dimensional sphere as the target domain. The differential operator (or generator) \( \mathcal{G} \) corresponding to Ornstein-Uhlenbeck processes is well investigated and known to satisfy the here required smoothness assumptions, see e.g., [27, 28].

The above theorem yields an existence and uniqueness result for the boundary value problem (15) and, therefore, connects system parameters \( \theta \) to inter-communication 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 4** The expected value [29, 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 \( \mathcal{G} \). Analytical solutions to some special cases such as the one dimensional case and the radial symmetric Ornstein-Uhlenbeck process are provided in [31].

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 [30], 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. [32, Sec. 10.2] for an introduction to numerical solutions of SDEs).
Lemma 5 (Hoeffding’s Inequality [33]) Let \( \tau_1, \ldots, \tau_n \) be i.i.d. bounded random variables, s.t. \( \tau_i \in [0, \tau_{\text{max}}] \). Then
\[
P \left[ \left| \frac{1}{n} \sum_{i=1}^{n} \tau_i - \mathbb{E}[\tau] \right| > \kappa \right] \leq 2 \exp \left( -\frac{2n\kappa^2}{\tau_{\text{max}}^2} \right). \tag{16}
\]

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.

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 \( \hat{X}(t) \) and set the predictions \( \hat{X}(\tau) \) 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 \( X(t) - \hat{X}(t) \) 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 \( F_n(t) := \frac{1}{n} \sum_{i=1}^{n} 1_{\tau_i \leq t} \), where \( 1 \) 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 \( \gamma_{\text{learn}} \) (cf. Fig. 1) to detect a mismatch between model and true dynamics 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 bounds the error between them uniformly. Essentially, we test if observed data fits the distribution, which is induced by the model \( \theta \); that is, which was derived with \( \hat{\theta} = \theta \) (cf. Sec. 3). If the distributions do not match, we conclude an unfit model and update \( \hat{\theta} \) through model learning.

Lemma 6 (DKW Inequality [34]) Assume \( \tau_1, \ldots, \tau_n \) are i.i.d. random variables with CDF \( F(t) \) and empirical CDF \( F_n(t) \). Then
\[
P \left[ \sup_{t \in \mathbb{R}} |F_n(t) - F(t)| > \kappa \right] \leq 2 \exp(-2n\kappa^2). \tag{17}
\]

4.2 Expectation-based Learning Trigger

We propose a first learning trigger \( \gamma_{\text{learn}} \) based on the expected value \( \mathbb{E}[\tau] \).

4.2.1 Exact Learning Trigger

Based on the foregoing discussion, we propose the following learning trigger:
\[
\gamma_{\text{learn}} = 1 \iff \left| \frac{1}{n} \sum_{i=1}^{n} \tau_i - \mathbb{E}[\tau] \right| \geq \kappa_{\text{exact}}, \tag{18}
\]

where \( \gamma_{\text{learn}} = 1 \) indicates that a new model shall be learned; \( \mathbb{E}[\tau] \) 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 in the sense that a larger time horizon allows the detection of smaller changes, however, it also increases the delay until the \( n \) samples are actually observed. The threshold parameter \( \kappa_{\text{exact}} \) quantifies the error we are willing to tolerate. There are some examples, where it is possible to compute \( \mathbb{E}[\tau] \) 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 \( \mathbb{E}[\tau] \), 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. Such 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 7 (Exact learning trigger) Assume \( \tau \) and \( \tau_1, \ldots, \tau_n \) are i.i.d. random variables and the parameters \( \alpha, n, \) and \( \tau_{max} \) are given. If the trigger (18) gets activated \((\gamma_{\text{learn}} = 1)\) with
\[
\kappa_{\text{exact}} = \tau_{\text{max}} \sqrt{\frac{1}{2n} \ln \frac{2}{\alpha}},
\]
then
\[
P \left[ \left| \frac{1}{n} \sum_{i=1}^{n} \tau_i - \mathbb{E}[\tau] \right| \geq \kappa_{\text{exact}} \right] \leq \alpha. \tag{20}
\]

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

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 \( \tau_{\text{max}} \) (cf. Assumption 1). 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 \( \mathbb{E}[\tau] \) can be difficult and computationally expensive. Instead, we propose to approximate \( \mathbb{E}[\tau] \) 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 \( \tau_1, \ldots, \tau_m \). This yields the approximated learning trigger
\[
\gamma_{\text{learn}} = 1 \iff \left| \frac{1}{m} \sum_{i=1}^{m} \hat{\tau}_i - \frac{1}{m} \sum_{i=1}^{m} \tau_i \right| \geq \kappa_{\text{approx}}. \tag{21}
\]

The Monte Carlo approximation leads to a choice of \( \kappa_{\text{approx}} \), which is different from \( \kappa_{\text{exact}} \) for small \( m \). For \( m \to \infty \) we see that \( \kappa_{\text{approx}} \) converges to \( \kappa_{\text{exact}} \).

Theorem 8 (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 \((\gamma_{\text{learn}} = 1)\) with
\[
\kappa_{\text{approx}} = \tau_{\text{max}} \sqrt{\frac{n + m}{2nm} \ln \frac{2}{\alpha}}, \tag{22}
\]
then
\[
P \left[ \left| \frac{1}{n} \sum_{i=1}^{n} \tau_i - \frac{1}{m} \sum_{i=1}^{m} \hat{\tau}_i \right| \geq \kappa_{\text{approx}} \right] \leq \alpha. \tag{23}
\]

**PROOF.** First, we introduce an alternative formulation of Hoeffding’s inequality (16)
\[
P \left[ \left| \frac{1}{n} \sum_{i=1}^{n} \tau_i - \frac{1}{m} \sum_{i=1}^{m} \hat{\tau}_i - (\mathbb{E}[\tau] - \mathbb{E}[\hat{\tau}]) \right| > \kappa_{\text{approx}} \right] \leq 2 \exp \left( - \frac{2 \kappa_{\text{approx}}^2}{(m^{-1} + n^{-1}) \tau_{\text{max}}^2} \right),
\]
which was already stated in the original article by Hoeffding [33] 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. \( \square \)

Remark 9 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, build learning triggers around the DKW inequality (17), and thus use richer statistical information.

We propose the following learning trigger:
\[
\gamma_{\text{learn}} = 1 \iff \sup_{t \in \mathbb{R}} |F(t) - F_n(t)| > \kappa_{\text{exact}}. \tag{24}
\]

The density-based learning trigger has the following property:

Theorem 10 (Exact Density Learning Trigger) Assume \( \tau_1, \ldots, \tau_n \) are i.i.d. random variables with CDF \( F(t) \) and empirical CDF \( F_n(t) \). If the learning trigger (24) gets activated \((\gamma_{\text{learn}} = 1)\) with
\[
\kappa_{\text{exact}} = \sqrt{\frac{1}{2n} \ln \frac{2}{\alpha}}, \tag{25}
\]
then
\[
P \left[ \sup_{t \in \mathbb{R}} |F(t) - F_n(t)| > \kappa_{\text{exact}} \right] \leq \alpha. \tag{26}
\]

**PROOF.** Follows directly from the DKW Inequality in Lemma 6. \( \square \)
Finally, we can follow the reasoning as before and obtain the sample-based version of the trigger (24)

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

(27)

where \( \hat{F}_m(t) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{\tau_i \leq t} \) and \( \tau_i \) are obtained by creating samples based on the model \( \theta \). This trigger is essentially the well established two-sample Kolmogorov-Smirnov (KS) test, which was first proposed in [35] and is summarized in [36].

**Theorem 11 (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 [35] and [36].

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 derived learning triggers to discrete time systems (5). The random variables \( \tau \) (cf. Eq. (4)) and \( \tau^d \) (cf. Eq. (8)) 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.

In this section, we show that the approximated learning triggers transfer without any modification to the discrete time system. It is important to adjust the system parameters \( \theta = (A, Q) \) and the model \( \hat{\theta} = (\hat{A}, \hat{Q}) \) to discrete time (cf. Sec. 2.2) in order to sample from the correct distribution (i.e., sampling from the continuous time model, while the true dynamics are discrete, or vice versa). Only based on statistical tests, irrelevant of the actual shape, we decide if they coincide.

**Theorem 12 (Discrete Time Learning Trigger)**

Assume \( \theta \) and \( \hat{\theta} \) correspond 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 \( \theta \) under the assumption that \( \theta = \hat{\theta} \). If this is true, then the distributions coincide. However, if \( \theta \neq \hat{\theta} \), then the model-induced distribution does not match reality and learning is triggered to update \( \hat{\theta} \).

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.

**Remark 13** 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 is straightforward to apply the exact learning triggers (18) and (24) to the discrete time system as well.

**Remark 14** It is also possible to consider more complex noise models such as colored noise instead of white noise. The main challenge lies in accurately identifying the system and in particular, the noise model from data. Afterward, the inter-communication distribution can be obtained via Monte Carlo approximations.

6 Numerical Example – Reduced Communication

The learning triggers derived in the previous two sections are the core element in the proposed ETL architecture (Fig. 1, block ‘Learning Trigger’). For ‘Model Learning’ in the context of linear Gaussian systems considered herein, one can use standard techniques for linear systems identification [37], which we do not elaborate further. Thus, all components of the proposed ETL method in Fig. 1 are complete, and we present a first numerical example to illustrate the main ideas of the developed learning triggers.
Next, we introduce the system, and afterward, we apply the learning trigger in order to demonstrate how to detect an inaccurate model. We consider the first-order dynamical system

\[ x(k + 1) = 0.9x(k) + \epsilon(k) \]  
(30)

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

\[ \hat{x}(k + 1) = 0.8\hat{x}(k). \]  
(31)

To bound the prediction error, we deploy the state trigger (7) with \( \delta = 3 \). In Fig. 2, we can see in the first graph a trajectory of states subject to system (30) as a black dashed line and the model-based predictions \( \hat{x}(k) \) in blue. Whenever \( \gamma_{\text{state}} = 1 \), we set \( \hat{x}(\tau) \) to \( x(\tau) \). The error signal never crosses the \( \delta \)-threshold and is depicted in the second graph. The communication instances are shown in the third graph. The distances between two consecutive communication instances corresponds to the inter-communication times. Further, we set \( \alpha = 0.05 \), \( \tau_{\text{max}} = 100 \), \( n = 300 \), and \( m = 100\,000 \).

After updating the model, we empty the buffer, start collecting new stopping times, and reset the average inter-communication time accordingly. We can see that the model-based expectation coincides with the empirical mean. Further, the average inter-communication time was increased after updating the model, which results in less communication.

The test statistic is also depicted in Fig. 4 for the initial inaccurate model and in Fig. 5 for the exact model. The dashed blue line represents again the model-based
expected value, while the dashed red line depicts the empirical mean at the moment of triggering.

6.3 Density-based Learning Triggers

Next, we will discuss the density-based learning trigger (27), which is also illustrated in Fig. 4 and Fig. 5. The solid blue line represents the model-based CDF function $F_m(t)$, and the solid red line is the empirical CDF $F_m(t)$ based on observed inter-communication times. Here, both triggers detect the inaccurate model (cf. Fig. 4) and have high confidence in the true model since the model-based and empirical quantities coincide, which is depicted in Fig. 5. The confidence interval is derived with Theorem 11 and tighter than the expectation-based. Hence, inaccurate models can be detected faster and more reliably.

Clearly, the model has also to be communicated at some point. However, this happens very rarely and in particular, when there is a significant change in the system. We conclude that both learning triggers are effective in detecting mismatch between model and true dynamics. Also, average communication was successfully reduced after updating the model. A higher dimensional example will be discussed in Sec. 8, where we also consider output measurements.

7 Output Measurements

So far, we assumed that perfect measures of the full state $x(k)$ are available at the sending agent. In the following, we will drop this assumption and consider systems, where only part of the state can be measured (see Fig. 6).
IKEping (39) to obtain model-based stopping times ˆτ. Hence, we can efficiently analyze the distribution of the corresponding derived tools and learning triggers. Therefore, we can effectively discuss problem (cf. (5)) and can apply the previously derived tools – sampling (39) to obtain model-based stopping times ˆτ.

Imulate the KF as

\[ \dot{x}(k + 1) = A\dot{x}(k), \]

and employ the state trigger

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

This event trigger ensures a bound between \( \hat{x}(k) \) and \( \tilde{x}(k) \). Hence, the inter-communication time is defined as

\[ \tau^o := \min\{k \in \mathbb{N} : \|\hat{x}(k) - \tilde{x}(k)\|_2 \geq \delta\} \]

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 Approach

The key idea of this approach is based on treating the KF sequence as a stochastic process in its own right and investigating the distribution of \( \hat{x}(k) \). With the aid of the innovation sequence, we can derive an auto-regressive structure. The innovation of the KF is defined as

\[ I(k) = y(k) - C\hat{x}(k). \]  

Furthermore, it is well known that \( I(1), \ldots, I(n) \) are independent normal distributed random variables with \( I(k) \sim \mathcal{N}(0, S) \) [38]. For a stationary KF, the covariance \( S \) is given by

\[ S = PCP^\top + R, \]

where \( P \) is the stationary error covariance matrix of the KF, and can be obtained by solving the corresponding Riccati equation [38, Equation (4.4)]. Hence, we can formulate the KF as

\[ \hat{x}(k + 1) = A\hat{x}(k) + KI(k), \]

and regard \( I(k) \sim \mathcal{N}(0, S) \) as a random variable. By regarding \( KI(k) \) as process noise, we are back to the previously discussed problem (cf. (5)) 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 – sampling (39) to obtain model-based stopping times \( \tau^o \).

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

The intuition behind the threshold parameter \( \delta \) for the trigger \( \|x(k) - \hat{x}(k)\|_2 \geq \delta \) is straightforward – it directly controls the error between state and open loop prediction. However, it is less clear how to choose the parameter \( \delta \) when there is a KF in the loop. The trigger \( \|\hat{x}(k) - \tilde{x}(k)\|_2 \geq \delta \) bounds the error between KF 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 \( \delta' \). In particular, we show how to choose \( \delta' \) such that we obtain the same statistical communication behavior as if triggering on the underlying state \( x(k) \).

First, we define \( \delta \) as usual to control the deviation between \( x(k) \) and \( \hat{x}(k) \). Based on the discussed methods, we can approximate the statistical properties of the corresponding stopping time. For this, we ignore the observation equation in (32) and assume we can fully measure \( x(k) \). This gives us the desired communication behavior of the system in terms of a stopping time \( \tau^d \), which we want to imitate with \( \tau^o \).

However, the stopping times \( \tau^o \) and \( \tau^d \) will differ when the same \( \delta \) is used since the variances of (5) and (39) are different. Due to the continuous and monotone dependency between \( \delta \) and \( \mathbb{E}[\tau^o] \) we can iteratively adjust the threshold of \( \tau^o \). We keep iterating until we obtain a value \( \delta' \) for which \( \mathbb{E}[\tau^{o'}] = \mathbb{E}[\tau^d] \), where \( \tau^d \) is defined with respect to the original threshold \( \delta \). Indeed, this does not guarantee that we bound the KF state with respect to the true state, but that the trigger is well behaved in distribution.

When resetting the open loop prediction \( \hat{x}(k) \) to \( \tilde{x}(k) \), we may also consider the reset with respect to the true state \( x(k) \). It is well known that \( \hat{x}(k) \sim \mathcal{N}(x(k), P) \). Hence, we can model this as an error process, which starts in \( z_0 \sim \mathcal{N}(0, P) \) instead of \( z_0 = 0 \). Thus, we obtain an interesting way to compare the two stopping times \( \tau^d \) and \( \tau^o \) through an exit problem with stochastic initial conditions. However, this point of view has no computational advantage.

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. First, it is not dependent on \( \tau_{\max} \) (cf. (22) and (28)) and, therefore, more sample efficient. Furthermore, there is a class of examples, where \( \theta \neq \tilde{\theta} \), despite of, \( \mathbb{E}[\tau] = \mathbb{E}[\tau] \).

Afterward, we demonstrate unexpected behavior of the state trigger with output measurements (35). Usually,
better models result in less communication, however, the contrary is also possible when triggering on the KF state. In particular, better models result in better estimates for \( \hat{x} \) and \( \hat{\theta} \), hence, the effect on communication can be in both ways.

8.1 Expected Value is not Enough

Consider the same numerical example (30) as before. Here, we assume the model \( \hat{\theta} = (0.5, 1.7) \) with \( n = m = 10,000 \). 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. Figure 7 shows the expected and empirical expected values of inter-communication times, which are almost identical – bad performance is expected and also realized due to the bad prediction model. The CDF-based trigger is still able to detect the inaccuracy, which is a big advantage over the expectation-based learning triggers.

In Fig. 5, the communication behavior is shown for the case where the dynamics are known exactly and coincide with the used model (\( \hat{\theta} = \theta \)). After updating the model the empirical inter-communication times almost double, which is equivalent to reducing communication by a factor of 2.

8.2 Better Models may Result in more Communication

More accurate models result in better predictions. Thus, one may expect that improved models also lead to reduced communication of state information from sender to receiver (cf. Fig. 1). While this is indeed the case for perfect state measurements (as has been observed in the example of Sec. 6), it may actually be the opposite for the KF setting (Fig. 6). Here, we present an example that demonstrates this rather unexpected effect – better models may lead to more communication. The reason is as follows: better models increase the KF 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, though counterintuitive.

Consider system (32) with the matrices

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

(40)

which is obtained by linearizing the closed-loop dynamics of a stabilized inverted pendulum. We assume process noise \( \epsilon(k) \sim \mathcal{N}(0, 0.1I_1) \) and observation \( \nu(k) \sim \mathcal{N}(0, 0.1I_2) \), where \( I_n \) is the identity matrix of dimension \( n \). Further, we assume that \( \nu(k) \sim \mathcal{N}(0, 0.5I_2) \) and that the model otherwise coincides with the true system parameters. We consider the KF states \( \hat{x}(k) \) (cf. (33)), the predictions \( \hat{x}(k) \) (cf. (34)), and the state trigger

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

(41)

We initialize \( x(0) = \hat{x}(0) = \hat{x}(0) = 0 \) and obtain the distribution over stopping times depicted in Fig. 8. The expected model-based communication is derived via Monte Carlo simulation of the innovation process (39), where we set \( \tau_{\text{max}} = 100 \) and \( m = n = 5000 \).

In the first graph in Fig. 8, we can see that the empirical inter-communication times are higher than the model-based. Updating the model actually reduces the average inter-communication time (more communication), which is because the KF improves and tracks the states \( x(k) \) better, which is illustrated in Fig. 9. The first plot shows the tracking performance when a perfect model is used \( \hat{\theta} = \theta \) (KF states in yellow). For the second plot, we changed the covariance of \( \nu(k) \) to \( \mathcal{N}(0, 0.5I_2) \) as discussed above. In the third plot, we exaggerated this effect even further by assuming \( \nu(k) \sim \mathcal{N}(0, 10I_2) \) in the model. In all three plots, we consider the first \( k = 150 \) time steps and stop afterward. In the third graph, we can see that the KF states deviate a lot from the true underlying states. However, they are still close to the open loop predictions and hence, there is very little communication. Despite the counterintuitive link between model accuracy and average communication, the example shows that the derived learning triggers are effective in detecting model mismatch.

9 Discussion and Future Work

Event-triggered learning is proposed in this article as a novel concept to trigger model learning when needed.
Fig. 8. Communication behavior of a system with output measurements. In the first graph we see the test statistic for an inaccurate model and in the second for $\hat{\theta} = \theta$. Both learning triggers ((21) and (27)) are effective in detecting the model mismatch. Interestingly, updating the model results in more communication as can be seen by a decrease in the actual average inter-communication time $E[\tau]$ (dashed red) between top and bottom. With the improved model, the KF tracks the true states better and thus, we obtain more communication.

This article focuses on the rigorous design of learning triggers, and we obtained multiple (provably) effective learning triggers utilizing statistical stopping time analysis. The concept of ETL has also already been applied in hardware experiments [26] and shown to yield reduced communication.

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. Here, we ultimately care about communication and thus, it is a natural idea to analyze inter-communication times and trigger model learning based on these. Furthermore, they show advantageous statistical properties such as being i.i.d. and scalar-valued. Depending on the concrete problem at hand, the signal used for triggering learning should be chosen accordingly. Using control performance as such a triggering signal is a potential extension, and first steps in this direction were taken in [39].

Fig. 9. State trajectories of the first dimension of the four dimensional system with output measurements (40). The state trigger (41) is applied and therefore, communication triggered when $\hat{x}(k)$ and $\tilde{x}(k)$ deviate by $\delta$. Communication instances are depicted with dotted vertical lines. In the first graph, we can see how well the KF $\hat{x}(k)$ (yellow) tracks the true states $x(k)$ (dashed blue). The red line depicts the open loop predictions $\tilde{x}(k)$. By worsening the model from the first to the second graph, the KF performance gets worse, which results in less communication. From the second to the third graph, we worsen the model even more, which results in even less communication, because the KF is not able to track the states.

Acknowledgements

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


