NETWORKED AND DISTRIBUTED CONTROL SYSTEMS

(SC42100)

Assignment 2

Student

Daniel Varela - 5321263

Responsible Instructors

Manuel Mazo

Tamás Keviczky

Gabriel Gleizer

May 25, 2021

Contents

1 Introduction

This assignment covers the topics in lectures 3 and 4 of the Networked and Distributed Control Systems Course.

Concepts related to packet losses, both deterministic and stochastic, are approached. Stability of deterministic models is explored by analysing the switched system that results from the range of possible consecutive packet losses. Stochastic packet losses are analysed through the lens of Mean Square Stability and Almost Sure Stability.

On the topic of event-triggered control, conditions for guaranteeing exponential stability for both discrete and continuous time systems are derived and the effects of implementing this type of controller are explored.

All code related to this Assingment can be found in this Github Repository.

2 Mechanisms for Modelling and Handling with Packet Losses

The basis for this section is the model of packet losses with recourse to an auxiliary variable m_k as:

$$m_k = \begin{cases} 0, & \text{if no packet loss at time } k \\ 1, & \text{if packet loss at time } k \end{cases} \tag{1}$$

and the subsequent model of packet-drops as extensions of sampling interval in a model:

$$x_{l+1} = \tilde{F}^{cl}(h, hl)x_l \tag{2}$$

2.1 To-Hold Mechanism

The to-hold mechanism is a way to deal with packet losses that consists on holding the last received input until the next valid input is received.

Describing this type of system can be done an auxiliary variable m_k to denote whether or not a packet was lost at time step k, with $m_k = 0$ meaning there were no packet losses, and $m_k = 1$ signifying packet was lost. Using this variable, the inputs can be described as follows:

$$u_k = \begin{cases} -\bar{K}x_k, & \text{if } \ m_k = 0 \\ u_{k-1}, & \text{if } \ m_k = 1 \end{cases} \tag{3}$$

Closing the loop here proves rather simple. If there is no packet being lost at time step k, the closed loop is the standard:

$$x_{k+1} = F_0^{cl} x_k = (F(h) - G(h)\bar{K})x_k \tag{4}$$

A packet loss, on the other hand, will utilize a past input as the current input to the system. This requires the use of an extended system description to store the previous input to the systems. The closed loop for the extended system $x_k^e = [x_k \;\; u_{k-1}]^\intercal$ then becomes:

$$\begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix} = F_1^{cl} \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix} = \begin{bmatrix} F(h) & G(h) \\ 0 & I \end{bmatrix} \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix} \tag{5}$$

Even if multiple packet losses happen sequentially, the model is able to hold the previous input and transfer it to the current input. The result is the following switched model:

$$x_{k+1}^e = \begin{cases} F_0^{cl} x_k^e, & \text{if} \;\; m_k = 0 \\ F_1^{cl} x_k^e & \text{if} \;\; m_k = 1 \end{cases} \tag{6}$$

Taking into account that a maximum of 2 possible consecutive packet losses is possible in this system, the full switched system can also be derived. If packet losses are modelled as extensions of sampling interval, and the closed loop is of the type $x_{l+1} = \tilde{F}^{cl}(h_l)x_l$, and h_l is the time between two packets. If a maximum of $\delta = 2$ packet losses are taken into account, then:

$$h_l \in \{h, 2h, 3h\} \tag{7}$$

If the packet loss behaviour is modelled as:

$$m \models \left(0^* \circ (10)^* \circ (110)^* \circ (1110)^* \circ \cdots \circ (\overbrace{1 \cdots 1}^{\delta} 0)^* \right)^\omega \tag{8}$$

the graphic representation of the To-Hold mechanism on the interval $[x_l, x_{l+1}]$ is in Figure 1. The closed loop with the to-hold mechanism will be:

$$x_{l+1}^e = \begin{cases} F_0^{cl} x_l^e, & \text{if} \;\; m_k = 0 \\ F_{(10)}^{cl} x_l^e = F_0^{cl} F_1^{cl} x_l^e, & \text{if} \;\; m_k = (10) \\ F_{(110)}^{cl} x_l^e = F_0^{cl} F_1^{cl} F_1^{cl} x_l^e, & \text{if} \;\; m_k = (110) \end{cases} \tag{9}$$

And the respective matrices:

$$F_0^{cl} = \begin{bmatrix} F(h) - G(h)\bar{K} & 0 \\ -\bar{K} & 0 \end{bmatrix}, \quad F_{(10)}^{cl} = \begin{bmatrix} (F(h) - G(h)\bar{K})F(h) & (F(h) - G(h)\bar{K})G(h) \\ -\bar{K}F(h) & -\bar{K}G(h) \end{bmatrix}, \tag{10}$$

$$F_{(110)}^{cl} = \begin{bmatrix} (F(h) - G(h)\bar{K})F^2(h) & (F(h) - G(h)\bar{K})(F(h)G(h) + G(h)) \\ -\bar{K}F^2(h) & -(\bar{K}F(h) + \bar{K})G(h) \end{bmatrix}$$

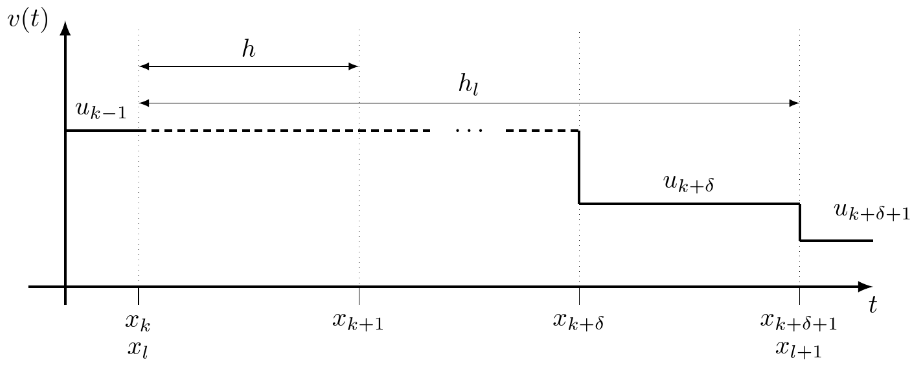


Figure 1: To-Hold method with dropped packets at the beginning of the time-interval.

If, on the other hand, the model considers packet losses at the end:

$$m \models \left(0^* \circ (01)^* \circ (011)^* \circ (0111)^* \circ \cdots \circ (0\overbrace{1 \cdots 1}^{\delta})^* \right)^{\omega} \tag{11}$$

with corresponding graphical representation as in Figure 2, then the closed loop with the to-hold mechanism will be:

$$x^e_{l+1} = \begin{cases} F^{cl}_0 x^e_l, \quad \text{if} \;\; m_k = 0 \\ F^{cl}_{(01)} x^e_l = F^{cl}_1 F^{cl}_0 x^e_l, \quad \text{if} \;\; m_k = (01) \\ F^{cl}_{(011)} x^e_l = F^{cl}_1 F^{cl}_1 F^{cl}_0 x^e_l, \quad \text{if} \;\; m_k = (011) \end{cases} \tag{12}$$

With matrices:

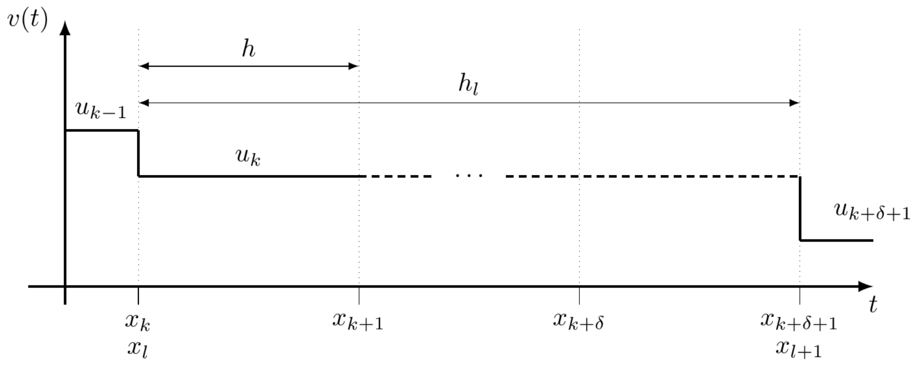


Figure 2: To-Hold method with dropped packets at the end of the time-interval.

$$F^{cl}_0 = \begin{bmatrix} F(h) - G(h)\bar{K} & 0 \\ -\bar{K} & 0 \end{bmatrix}, \quad F^{cl}_{(01)} = \begin{bmatrix} F(h)(F(h) - G(h)\bar{K}) - G(h)\bar{K} & 0 \\ -\bar{K} & 0 \end{bmatrix},$$

$$F^{cl}_{(011)} = \begin{bmatrix} F^2(h)(F(h) - G(h)\bar{K}) - (G(h) + F(h)G(h))\bar{K} & 0 \\ -\bar{K} & 0 \end{bmatrix} \tag{13}$$

Note that, if the packet losses are modelled at the end of the interval h_l, then the extended model is not even necessary. Note that the component u_{l-1} never influences the dynamics x_{l+1}, as evidenced by the column of zeros in the matrix. This is because the information about the last input received is encoded in the state x_l, as $u_k = -\bar{K}x_l$.

If the model considers packet losses at the end of the interval h_l, then the closed loop with the to-hold mechanism will be:

$$x_{l+1} = \begin{cases} \left\{F(h) - G(h)\bar{K}\right\} x_l, & \text{if } \ m_k = 0 \\ \left\{F(h)(F(h) - G(h)\bar{K}) - G(h)\bar{K}\right\} x_l, & \text{if } \ m_k = (01) \\ \left\{F^2(h)(F(h) - G(h)\bar{K}) - (G(h) + F(h)G(h))\bar{K}\right\} x_l, & \text{if } \ m_k = (011) \end{cases} \tag{14}$$

2.2 To-Zero Mechanism Closed Loop Derivation

The to-zero mechanism is an alternative to the to-hold mechanism which handles packet losses at time step k by setting the corresponding input u_k to zero. Otherwise, state feedback is assumed.

$$u_k = \begin{cases} -\bar{K}x_k, & \text{if } \ m_k = 0 \\ 0, & \text{if } \ m_k = 1 \end{cases} \tag{15}$$

If no packet is lost at time step k, the closed loop dynamics will be dictated by the state feedback.

$$x_{k+1} = F_0^{cl} x_k = (F(h) - G(h)\bar{K})x_k \tag{16}$$

On the other hand, the state simply evolves autonomously if a packet is dropped at time step k.

$$x_{k+1} = F_1^{cl} x_k = F(h)x_k \tag{17}$$

The framework for this question has been laid out in the previous section. Considering a bounded number of consecutive inputs, the system can be modelled in a deterministic fashion by considering the time between received packages h_l instead of a fixed clock time h, and modelling the behaviour including packet drops as a Linear Time Varying switched system. For a finite number of δ packet losses in a row:

$$h_l \in \{h, 2h, \ldots, (\delta + 1)h\} \tag{18}$$

there will be $\delta + 1$ switched systems to fully capture the behaviour of the system.

Under the to-zero behaviour, if a system doesn't receive a packet at a time step k, the input is considered to be zero and the systems evolution is autonomous until the next packet is received, at time step $k + \delta$. The closed loop model will, once again, change depending on whether the packet drops are clumped at the beginning or at the end of the interval h_l.

If one considers that all the packet drops are at the end of the time interval h_l, as in (12), the derivation of the closed loop is rather direct.

The input $v(t)$ in the interval $x_k \in [x_l, x_{l+1}]$ is shown in Figure 3. Starting from the general expression for the evolution of the system:

$$x_{l+1} = e^{Ah_l}x_l + \int_0^{h_l} e^{A(h_l-\tau)}Bv(\tau)d\tau \tag{19}$$

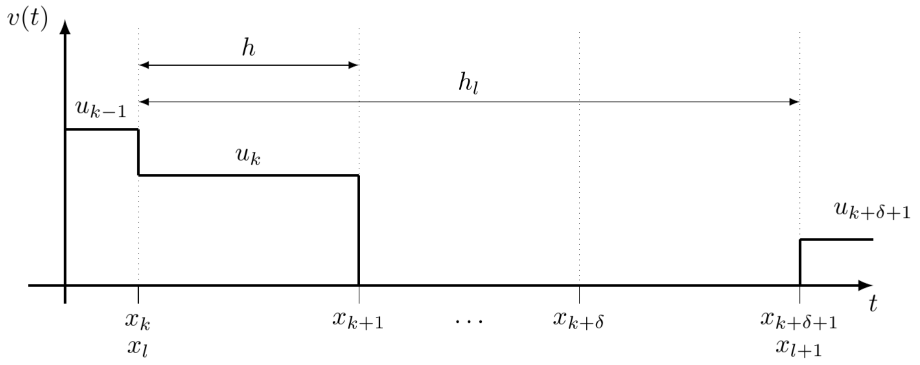


Figure 3: To-Zero method with dropped packets at the end of the time-interval.

Note first that the evolution is autonomous in the interval $t \in [h, h_l]$. As such:

$$x_{l+1} = e^{A(h_l-h)}x_{k+1} \tag{20}$$

and x_{k+1} is easily obtained considering the static feedback case $u_k = \bar{K}x_k = \bar{K}x_l$:

$$x_{l+1} = e^{A(h_l-h)}\left(e^{Ah}x_l + \int_0^h e^{A(h-\tau)}B\bar{K}d\tau x_l\right) = \left(e^{Ah_l}x_l + e^{A(h_l-h)}\int_0^h e^{A(h-\tau)}B\bar{K}d\tau\right)x_l \tag{21}$$

Under a simple change of variable $s = h - \tau$, the result presented in the slides can be obtained:

$$x_{l+1} = \left(e^{Ah_l}x_l + e^{A(h_l-h)}\int_0^h e^{As}B\bar{K}ds\right)x_l \tag{22}$$

The difference is subtle but important. Since we're dealing with matrix multiplications, the order does matter in this case, and one should take measures to ensure the closed loop formula used matches the formulation used to model the packet losses.

2.3 Formal Stability Analysis of the To-Hold Mechanism

In this section, the to-hold mechanism is again considered. Recall Section 2.1, where a switched system was derived for the case of a maximum of 2 packet drops in a row. Depending on where the packet drops are placed, the closed loop system will be described differently. Nevertheless, both representations result in switched system representation consisting of a set of three autonomous closed loops that switch between each other depending on the sequence of lost packets, $m_k = \{0, 01, 011\}$ (or equivalently, $m_k = \{0, 10, 110\}$). To simplify this section, the focus will be on the representation of delays at the end of the interval h_l, since this doesn't require the use of an extended system. The system is first derived in equation 14, and reproduced here for simplicity.

$$x_{l+1} = \begin{cases} F_0^{cl} x_l = \left\{F(h) - G(h)\bar{K}\right\} x_l, \ \text{if} \ \ m_k = 0 \\ F_{(01)}^{cl} x_l = \left\{F(h)(F(h) - G(h)\bar{K}) - G(h)\bar{K}\right\} x_l, \ \text{if} \ \ m_k = 01 \\ F_{(011)}^{cl} x_l = \left\{F^2(h)(F(h) - G(h)\bar{K}) - (G(h) + F(h)G(h))\bar{K}\right\} x_l, \ \text{if} \ \ m_k = 011 \end{cases} \tag{23}$$

Analysis of stability for linear time varying switched systems has been previously seen, in the context of a finite set of time varying delays. The approach used also resulted in a switched system depending on the delay at time step k. Though the underlying process resulting in a switched system representation is different here and in the previous case, mathematically, the representation is fundamentally the same, and therefore, the same approach for stability can be used.

Once again, eigenvalue-based analysis is not enough to ensure stability of the switched system. The alternative is to follow a Lyapunov function approach. If it's possible to find a common Lyapunov function for all the sub-systems in the switched system, then stability can be guaranteed.

For an autonomous system $x_{l+1} = F^{cl} x_l$, if the Lyapunov function is further assumed quadratic in nature, $V(x) = x^\intercal P x$, then finding a valid matrix $P \succ 0$ yields exponential stability if it's possible to also find a suitable matrix $Q \succ 0$ such that:

$$F^{cl\intercal} P F^{cl} - P \preceq -Q \tag{24}$$

Given the homogeneity of the problem, solving it is equivalent to:

$$F^{cl\intercal} P F^{cl} - P \preceq -I \tag{25}$$

Extending this approach to switched systems involves adding another equation per mode in the system. For the case under consideration, this means finding a common Lyapunov function such that:

$$\begin{cases} F_0^{cl\intercal} P F^{cl} - P \preceq -I \\ {F_{(01)}^{cl}}^\intercal P F^{cl} - P \preceq -I \\ {F_{(011)}^{cl}}^\intercal P F^{cl} - P \preceq -I \end{cases} \tag{26}$$

This is the set of LMIs that require solving to guarantee stability of the switched system under a finite, deterministic set of packet drops. Finding a solution to this set of LMIs can be done easily in `Matlab` with recourse to the `CVX` solver.

2.4 LMI conditions for Exponential Mean Square Stability

Mean Square Stability is a concept related to stochastic systems, and the goal is to provide a measure of stability for systems, independent of initial state and conditions of the system. A system is said to be Mean Square Stable if:

$$\lim_{k\to\infty} \mathbb{E}\left[\|x\|^2\right] = 0 \tag{27}$$

A Markov Jump Linear Systems is a set of autonomous linear systems that switch between each other according to a Markov Process. This is essentially a switched system, but governed by the probabilities associated to a Markov Process. For this type of systems, there's and equivalence between Mean Square Stability and Exponential MSS.

Shifting focus to the system under analysis, it's possible to arrive at a graphical description of the system via a Discrete-Time Markov Decision Process, as seen in Figure 4. Given $\mathbb{P}[m_k = 1 \mid m_{k-1} = 0] = p_1^a$, and the impossibility of going from zero to two packet drops directly, it's possible to find $\mathbb{P}[m_k = 0 \mid m_{k-1} = 0] = 1 - p_1^a$. Similarly, one packet drop is either followed by no packet drop, or by a second packet drop, therefore, form $\mathbb{P}[m_k = 1 \mid m_{k-1} = 1, m_{k-2} = 0] = p_2^a$ it's easy to retrieve $\mathbb{P}[m_k = 0 \mid m_{k-1} = 1, m_{k-2} = 0] = 1 - p_2^a$. Given that no more than two packets can be lost at a time, $\mathbb{P}[m_k = 0 \mid m_{k-1} = 1, m_{k-2} = 1] = 1$.

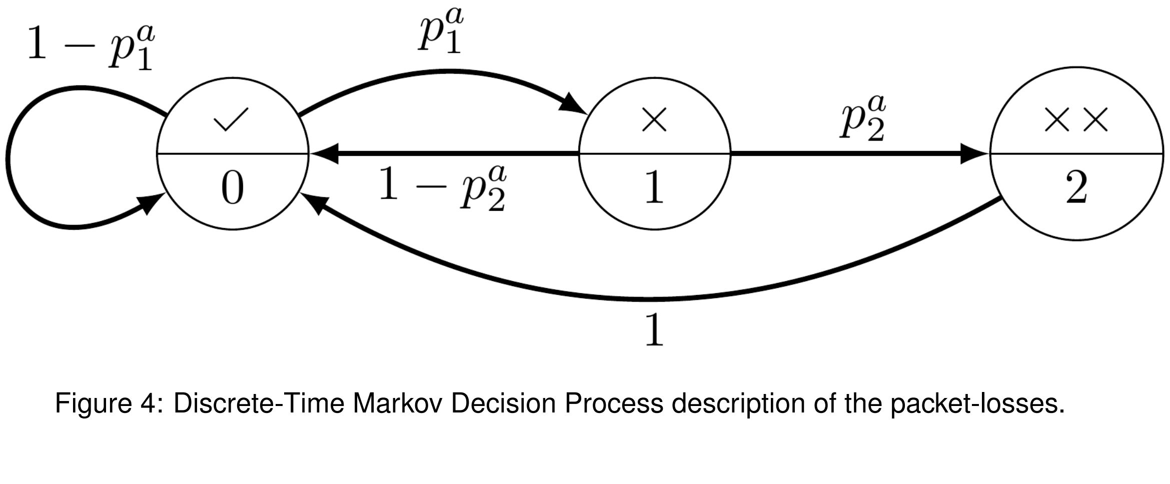


Figure 4: Discrete-Time Markov Decision Process description of the packet-losses.

If the framework is assumed to be similar to the one in the lectures, each state of the Markov process will have a linear system associated to it. State 0 will have it's dynamics evolve as usual, via a the closed loop with state feedback $x_{k+1} = F_0^{cl} x_k = (F(h) - G(h)\bar{K})x_k$. States $\{1, 2\}$ will have experienced packet losses, and will evolve as $x_{k+1} = F_1^{cl} x_k$ according to equation 5 if a to-hold method is used. The system, as a whole, can be described as a set of linear systems which evolve according to a Markov Chain: a Markov Jump Linear System (MJLS).

As such, the tools used to prove MSS will equivalently show Exponential MSS as well, and can be applied directly. MJLS are MSS if there exist $P \succ 0, i = 0, \ldots, N$ that satisfy the following condition:

$$P_i - \sum_{j=0}^{N} p_{ij} A_i^{cl\intercal} P_j A_i^{cl} \succ 0, \forall i = 0, \ldots, N \tag{28}$$

The result is a set of linked Lyapunov equations which can be solved by a set of linear matrix inequalities (LMIs).

Looking back at the given system, with a total number of states N = 2, the LMIs are:

$$\begin{cases} P_0 - p_{00} F_0^{cl\intercal} P_0 F_0^{cl} - p_{01} F_1^{cl\intercal} P_1 F_1^{cl} - \overset{=0}{\cancel{p_{02}}} F_1^{cl\intercal} P_2 F_1^{cl} \succ 0 \\ P_1 - p_{10} F_0^{cl\intercal} P_0 F_0^{cl} - \overset{=0}{\cancel{p_{11}}} F_1^{cl\intercal} P_1 F_1^{cl} - p_{12} F_1^{cl\intercal} P_2 F_1^{cl} \succ 0 \\ P_2 - p_{20} F_0^{cl\intercal} P_0 F_0^{cl} - \overset{=0}{\cancel{p_{21}}} F_1^{cl\intercal} P_1 F_1^{cl} - \overset{=0}{\cancel{p_{22}}} F_1^{cl\intercal} P_2 F_1^{cl} \succ 0 \end{cases} \tag{29}$$

Replacing in the known values for the remaining probabilities, the set of LMIs that establish Exponential Mean Square Stability of this system are:

$$\begin{cases} P_0 - (1 - p_1^a) F_0^{cl\intercal} P_0 F_0^{cl} - p_1^a F_1^{cl\intercal} P_1 F_1^{cl} \succ 0 \\ P_1 - (1 - p_2^a) F_0^{cl\intercal} P_0 F_0^{cl} - p_2^a F_1^{cl\intercal} P_2 F_1^{cl} \succ 0 \\ P_2 - F_0^{cl\intercal} P_0 F_0^{cl} \succ 0 \end{cases} \tag{30}$$

3 GES of an Event-Triggered Discrete-Time System

A discrete-time system can be described by the evolution:

$$x_{k+1} = Ax_k + Bu_k \tag{31}$$

Performance is guaranteed for discrete time systems if:

$$V(x_{k+1}) - V(x_k) \leq x_k^\intercal Q x_k \tag{32}$$

Since the aim is to hold a sample for longer than just the sampling period h, performance will be lost, since control will not be optimal, $u_k \neq -\bar{K} x_k$. Further, it's to be expected that the longer a sample is held, the worse the system will perform. As such, a desired performance level can be set for some $\sigma \in (0, 1)$.

$$V(x_{k+1}) - V(x_k) \leq \sigma x_k^\intercal Q x_k \tag{33}$$

Assume a sample is held at time k, and denote this sample as x_s. The evolution of the system while this sample is held will be:

$$x_{k+1} = Ax_k - B\bar{K}x_s \tag{34}$$

If the shape of Lyapunov function is further assumed to be quadratic, $v(x_k) = x_k^\intercal P x_k$, then the performance is:

$$\begin{aligned} (Ax_k - B\bar{K}x_s)^\intercal P(Ax_k - B\bar{K}x_s) - x_k^\intercal P x_k \leq \sigma x_k^\intercal Q x_k \Leftrightarrow \\ x_k^\intercal (A^\intercal PA - P + \sigma Q)x_k - x_k^\intercal A^\intercal PB\bar{K}x_s - x_s^\intercal \bar{K}^\intercal B^\intercal PAx_k + x_s^\intercal \bar{K}^\intercal B^\intercal PB\bar{K}x_s \leq 0 \end{aligned} \tag{35}$$

Performance can be imposed by ensuring that for $k \in [s, s^+)$:

$$\phi(x_k, x_s) := \begin{bmatrix} x_k^\intercal & x_s^\intercal \end{bmatrix} \begin{bmatrix} A^\intercal PA - P + \sigma Q & -A^\intercal PB\bar{K} \\ -\bar{K}^\intercal B^\intercal PA & \bar{K}^\intercal B^\intercal PB\bar{K} \end{bmatrix} \begin{bmatrix} x_k \\ x_s \end{bmatrix} \leq 0 \tag{36}$$

Where s^+ is the next sample to be held, that is, the first sample after s for which $\phi(x_k, x_s) \leq 0$ no longer holds.

$$s^+ = \inf\{k > s \mid \phi(x_k, x_s) \geq 0\} =: \Phi_{\text{ETC}}(x_k, x_s) \tag{37}$$

This triggering condition can be rewritten in terms of an error term $\varepsilon_k = x_s - x_k$. The dynamics of an extended system $x_k^e = [x_k \ \ \varepsilon_k]^\intercal$ are:

$$\begin{bmatrix} x_{k+1} \\ \varepsilon_{k+1} \end{bmatrix} = \begin{bmatrix} A - B\bar{K} & -B\bar{K} \\ -A + B\bar{K} & B\bar{K} \end{bmatrix} \begin{bmatrix} x_k \\ \varepsilon_k \end{bmatrix} \tag{38}$$

Similarly, the performance equation can be expressed in terms of the current state and this error term.

$$\begin{aligned} ((A - B\bar{K})x_k - B\bar{K}\varepsilon_K)^\intercal P((A - B\bar{K})x_k - B\bar{K}\varepsilon_K) - x_k^\intercal P x_k \leq \sigma x_k^\intercal Q x_k \Leftrightarrow \\ x_k^\intercal \left(\overbrace{(A - B\bar{K})^\intercal P((A - B\bar{K}) - P}^{=-Q} + \sigma Q \right) x_k - x_k^\intercal (A - B\bar{K})^\intercal PB\bar{K}\varepsilon_k - \\ \varepsilon_k^\intercal K^\intercal B^\intercal P(A - B\bar{K})x_k + \varepsilon_k^\intercal K^\intercal B^\intercal PBK\varepsilon_k \leq 0 \Leftrightarrow \\ x_k^\intercal ((1 - \sigma)Q) x_k + x_k^\intercal (A - B\bar{K})^\intercal PB\bar{K}\varepsilon_k + \varepsilon_k^\intercal K^\intercal B^\intercal P(A - B\bar{K})x_k - \varepsilon_k^\intercal K^\intercal B^\intercal PB\bar{K}\varepsilon_k \geq 0 \end{aligned} \tag{39}$$

Performance can be imposed by ensuring that for $k \in [s, s^+)$:

$$\phi^e(x_k, x_s) := \begin{bmatrix} x_k^\intercal & \varepsilon_k^\intercal \end{bmatrix} \begin{bmatrix} (1 - \sigma)Q & (A - B\bar{K})^\intercal PB\bar{K} \\ \bar{K}^\intercal B^\intercal P(A - B\bar{K}) & -\bar{K}^\intercal B^\intercal PB\bar{K} \end{bmatrix} \begin{bmatrix} x_k \\ \varepsilon_k \end{bmatrix} \geq 0 \tag{40}$$

The next step to be held, s^+ will be the first time step larger than s that no longer guarantees $\phi^e(x_k, x_s) \geq 0$.

$$s^+ = \inf\{k > s \mid \phi^e(x_k, \varepsilon_k) \leq 0\} =: \Phi_{\text{ETC}}(x_k, \varepsilon_k) \tag{41}$$

The condition derived guarantees that the a minimum time between events is always lower bounded by a positive quantity (the time between two samples), since the system is discrete, and therefore won't run into the problem of Zeno Behaviour, where the minimum inter-event time $\tau_{\min}$ becomes infinitely small.

4 Stability of Systems Under Packet Losses

4.1 Global Asymptotic Stability with Packet-Losses

In this section, the second order system is analysed for robustness to a maximum number of $\bar{\delta}$ consecutive packet losses. For parameters $a = 5, b = 2$ and $c = 3$, the following system is obtained:

$$A = \begin{bmatrix} 5 & 2.5 \\ 0 & -3 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{42}$$

A static state feedback controller was previously derived to place the poles of the closed loop continuous system at $s = \{-2, -3\}$.

$$u(t) = -\bar{K}x(t) \Leftrightarrow u(t) = -\begin{bmatrix} 22.4 & 7 \end{bmatrix} x(t) \tag{43}$$

Discretization of for a sampling time h results in the discrete time system with closed loop:

$$x_{k+1} = (F(h) - G(h)\bar{K})x_k \tag{44}$$

Matrices $F(h)$ and $G(h)$ have been derived in a previous report, and are therefore merely reproduced here.

$$F(h) = \begin{bmatrix} e^{5h} & \frac{5}{16}\left(e^{5h} - e^{-3h}\right) \\ 0 & e^{-3h} \end{bmatrix}, \quad M(h) = \begin{bmatrix} \frac{e^{5h}}{16} + \frac{5e^{-3h}}{48} - \frac{1}{6} \\ \frac{1}{3} - \frac{e^{-3h}}{3} \end{bmatrix} \tag{45}$$

Picking up the work done in Sections 2.1 and 2.2, a system with a to-zero mechanism to handle packet losses will behave as a regular closed loop state feedback system if no packet is lost, and as an autonomous system otherwise.

$$x_{k+1} = \begin{cases} F_0^{cl}x_k = (F(h) - G(h)\bar{K})x_k, & \text{if } \; m_k = 0 \\ F_1^{cl}x_k = F(h)x_k, \quad \text{if } \; m_k = 1 \end{cases} \tag{46}$$

Under a deterministic setting, this system can be represented as a switched system dependent on the maximum number of possible packet losses in a row. For a system with:

$$h_l \in \{h, 2h, \ldots, (\delta + 1)h\} \tag{47}$$

The closed loop was previously derived to be (considering packet losses at the end of the interval h_l):

$$x_{l+1} = \left(e^{Ah_l} x_l + e^{A(h_l - h)} \int_0^h e^{As} B\bar{K} ds \right) x_l \tag{48}$$

This type of system can alternatively be represented in terms of matrices F_0^{cl} and F_1^{cl} and the total number of delays δ in the interval $[x_l, x_{l+1}]$. This is done by applying the equations recursively:

$$\begin{gathered} x_{l+1} = x_{k+\delta+1} = F_1^{cl} x_{k+\delta} = \left(F_1^{cl}\right)^2 x_{k+\delta-1} = \dots = \left(F_1^{cl}\right)^\delta F_0^{cl} x_{k+1} = \left(F_1^{cl}\right)^\delta F_0^{cl} x_k \Leftrightarrow \\ x_{l+1} = \left(F_1^{cl}\right)^\delta F_0^{cl} x_l \end{gathered} \tag{49}$$

This representation is useful when trying to understand how many delays a system can handle, since the problem is reduced to finding a common Lyapunov function for the switched system with as many equations as possible delays on the system, $\delta = 0, \dots, \bar{\delta}$.

$$x_{l+1} = \begin{cases} F_0^{cl} x_l, \quad \text{if} \;\; m_k = 0 \\ F_{(01)}^{cl} x_l = F_1^{cl} F_0^{cl} x_l, \quad \text{if} \;\; m_k = (01) \\ \dots \\ F_{(01\dots1)}^{cl} x_l = \left(F_1^{cl}\right)^{\bar{\delta}} F_0^{cl} x_l, \quad \text{if} \;\; m_k = (0\overbrace{1\dots1}^{\bar{\delta}}) \end{cases} \tag{50}$$

The procedure is simple from here on. Starting from $\delta = 0$ the LMI is solved. If a Lyapunov Function can be found, then the equation for $\delta = 1$ is added to the set of LMIs is solved to find a Lyapunov Function for both equations. The procedure is repeated until a value $delta = \bar{\delta} + 1$, at which point it'll be impossible to find a common Lyapunov Function. The maximum number of delays for which the system is guaranteed stability is concluded to be $\delta = \bar{\delta}$. Assuming a quadratic Lyapunov Function $V(x_k) = x_k^\intercal P x_k$, the homogeneous set of LMIs to be solved is:

$$\begin{cases} F_0^{cl\intercal} P F_0^{cl} - P \preceq -I \\ \left(F_1^{cl} F_0^{cl}\right)^\intercal P F_1^{cl} F_0^{cl} - P \preceq -I \\ \dots \\ \left(F_1^{cl^{\bar{\delta}}} F_0^{cl}\right)^\intercal P F_1^{cl^{\bar{\delta}}} F_0^{cl} - P \preceq -I \end{cases} \tag{51}$$

Applying this procedure for multiple values of sampling time h, the plot in Figure 5 is obtained. The majority of sampling times, as is visible, cannot handle a single packet loss while maintaining assurances of stability. This does not mean the system will become unstable if a packet is lost, just that no Lyapunov function could be found to prove stability. Of interest here, however, is the interval

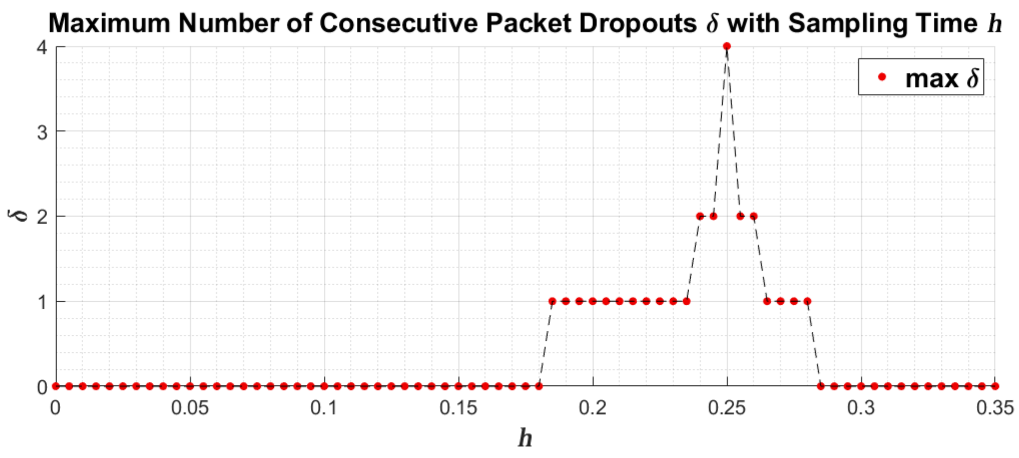


Figure 5: Maximum number of consecutive packet losses $\bar{\delta}$ admissible as a function of the sampling time h.

$h \in [0.185, 0.28]s$. In this interval, the system can handle at least one packet loss. If $h = 0.25s$, the system can handle the maximum of four packet losses in a row.

It's interesting to notice the overlap between this plot and a plot of the eigenvalues of the closed loop $x_{k+1} = (F(h) - G(h)\bar{K})x_k$ obtained for Assignment 1 and reproduced here in Figure 6. Note that the higher values of packet losses line up perfectly with lower absolute values of the eigenvalue λ_2, corresponding to the unstable eigenvalue $s = 5$.

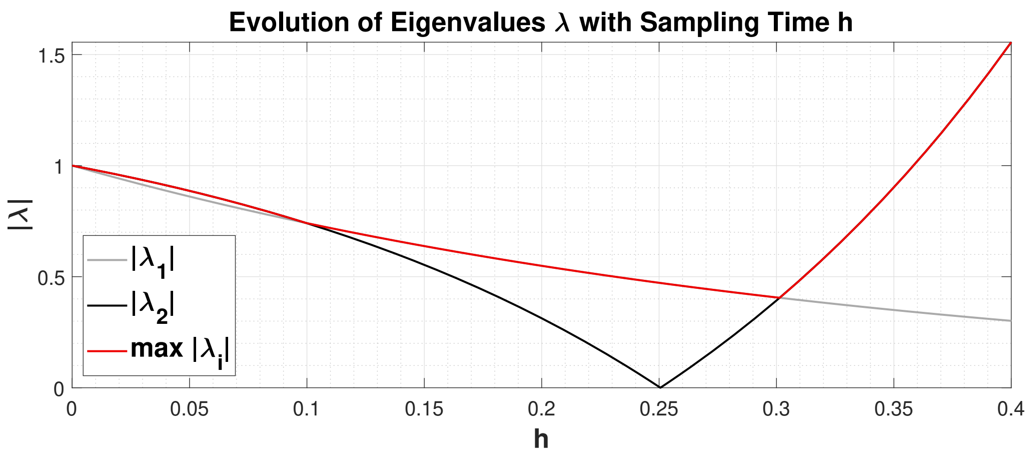


Figure 6: Eigenvalues of the closed loop discrete system as a function of the sampling time.

As a case study on how conservative this method (and the assumptions made, such as the quadratic nature of the Lyapunov Function), one can consider a single sampling interval $h = 0.1s$. Figure 7 shows 10000 simulations where three consecutive packets out of every nine are lost. Note that stability is maintained throughout, despite the LMI approach implying that no packets can be dropped for this sampling interval. This is, by no means, a rigorous proof of stability, but it goes to show how conservative the application of this method can be.

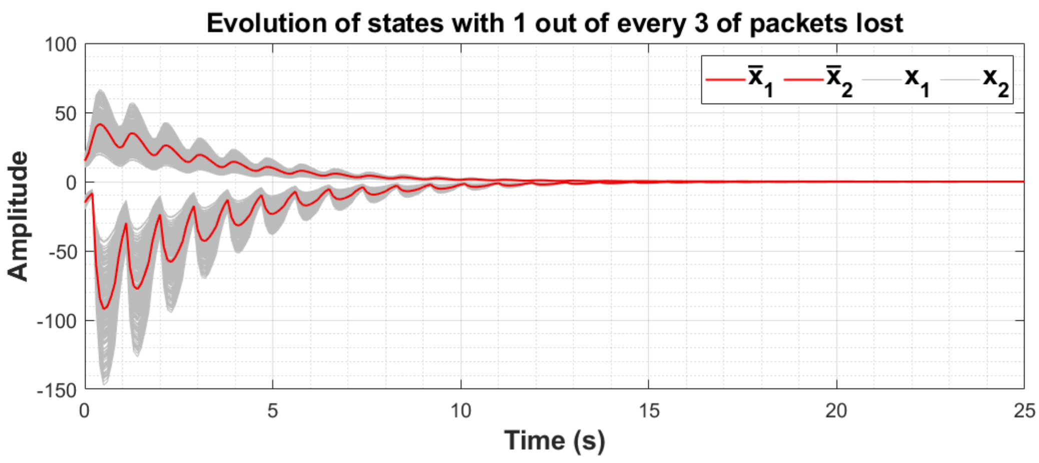


Figure 7: Evolution of states x_k for 250 different initial conditions for sampling time $h = 0.1s$.

4.2 Mean Square Stability of a Bernoulli Process

This section looks at the effects of packet losses governed by a Bernoulli process. This process can be simply described by the probability of a packet being lost, $\mathbb{P}[m_k = 1] = p$. Figure 8 shows a graphical representation of the process. Considering a to-zero mechanism, for state 0, there is no packet loss and the closed loop system evolves according to $x_{k+1} = F_0^{cl} x_k$. State 1 implies a packet loss occured and the system evolves autonomously as $x_{k+1} = F_1^{cl} x_k$.

Therefore, the system is a MJLS, and MSS is reduced to evaluating a set of as many Lyapunov equations as states in the Markov Chain. In this particular case, both Laypunov equations will be the same, and the problem is simplified to the analysis of a single Lyapunov equation.

$$P - (1-p){F_0^{cl}}^\intercal P F_0^{cl} - p{F_1^{cl}}^\intercal P F_1^{cl} \succ 0 \tag{52}$$

The system will remain stable as long as it's possible to find a matrix $P \succ 0$ that satisfies the equation LMI above. As such, a simple procedure arises to find the maximum value of p, p^*. This probability can simply be increased incrementally until a solution P can no longer be found to the LMI. The value p^* will be the maximum p for which a valid solution $P \succ 0$ exists.

Application of this procedure to the system in equation 46, for a range of sampling times h, results in the plot in Figure 9. Note how differently the system appears to behave here, when compared to the results from Figure 5. Roughly speaking, here a package will be lost with probability p. As such, this plot conveys that lower sampling intervals can lose packets more frequently than

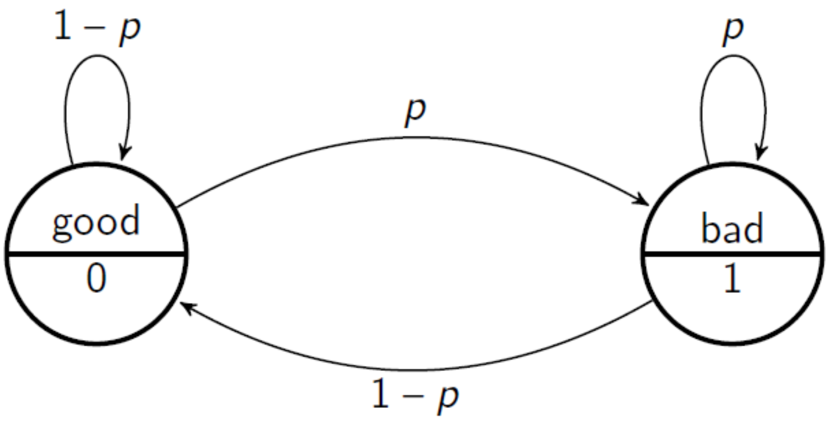


Figure 8: Stochastic packet losses model (Bernoulli). [3]

higher sampling intervals, and that this evolution seems linear. A study on how conservative this method is will be had in the next section.

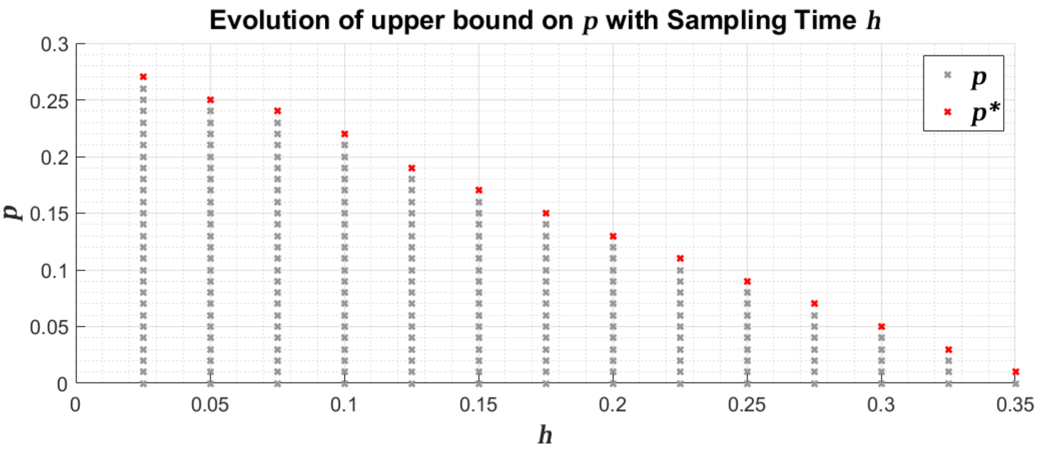


Figure 9: Evolution of p^* with sampling time h

4.3 Numerical Simulations of MSS for a Bernoulli Process

This section goes over the numerical simulations of mean square stability for a Bernoulli Process.

Simulations are done for sampling period $h = 0.1s$. At every time step, the system evolves according to equation 46. The choice of whether a packet is lost or not is done randomly, as well as the initial state of the system. A random number r is generated at every time step, in the interval $r \in [0, 1]$. If $r \leq p^*$, then a packet is dropped and $m_k = 1$. Otherwise, $m_k = 1$ and the system evolves according to the state feedback law.

Starting with $h = 0.1s$, Figure 9 shows $p^* = 0.22$. The plot in Figure 10 shows 10000 simulations starting from random initial conditions, for this value p^*. As expected, the systems remains stable throughout. In general, for all sampling intervals h, testing a probability of packet drops $p \leq p^*$ results in similarly stable behaviour, thus confirming the values for p^* shown found in the last Section are reasonable.

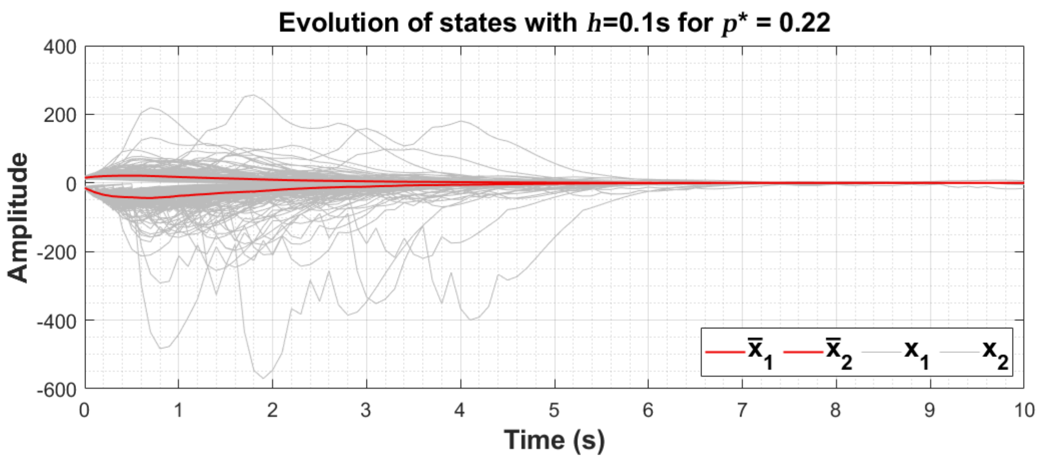


Figure 10: Results of 10000 simulations for $h = 0.1s$ and $p = p^* = 0.22$.

Analysing directly the $\mathbb{E}(\|x_k\|^2)$ can give us some more insights into these results. For $h = 0.1$, proceeding similarly to before, but with a probability of packet losses $p = 0.25 > p^*$ still results in stable behaviour, and convergence to zero as $k \to \infty$, as seen in Figure 11. Increasing this value further results in larger offshoots (borderline unreasonably), but convergence still happens for larger values of k, as seen in Figure 12 for $p = 0.3$. The system only starts experiencing constant blow-ups at around $p = 0.35$.

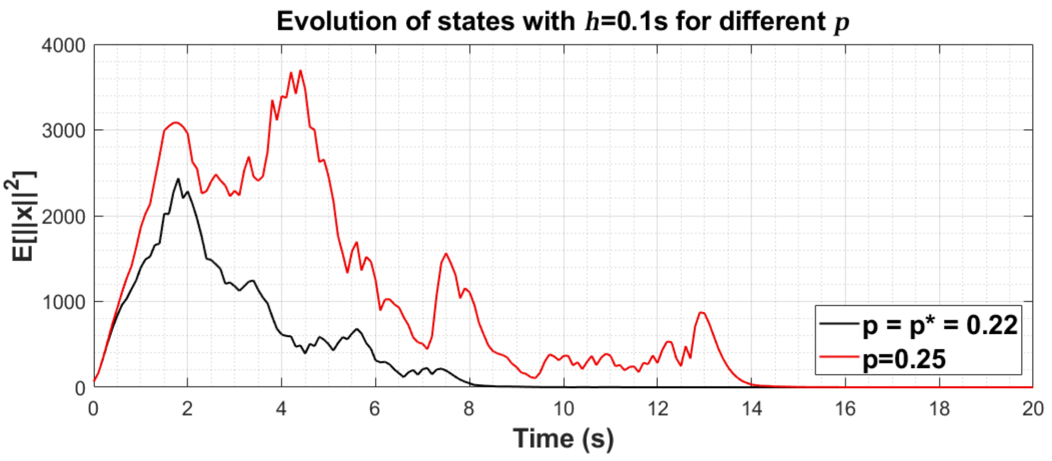


Figure 11: Results of 10000 simulations for $h = 0.1s$ with $p = 0.22$ and $p = 0.25$.

These numerical simulations once again show that the method used to go about proving Mean

Evolution of states with h=0.1s for different p

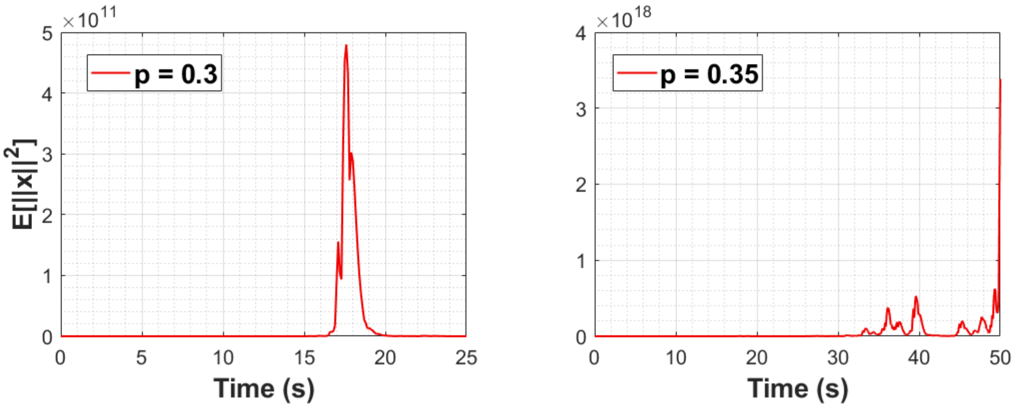


Figure 12: Results of 10000 simulations for $h = 0.1s$ with $p = 0.3$ and $p = 0.35$.

Square Stability produces stable results, and therefore seems to be working as expected. Naturally, however, the restrictions imposed on the shape of Lyapunov Function and on the method used to prove MSS via a set of linked LMIs are somewhat conservative, and it's possible to find (without proof, however), systems that are still stable even if they lie outside what the tool at our disposal would consider stable.

Further, note that while finding a valid matrix P implies MSS, the implication does not go the other way around, that is, not being able to find a valid matrix P doesn't mean the system isn't MSS. Further still, note that the framework used to prove MSS, as given in equation 28, is only one of four sets of possible linked Lyapunov equations one can use to go about proving MSS, as seen in Slide 34 of Lecture 3 of [3]. The formulations shown in the aforementioned slide are not equivalent, so it could happen that, using another formulation, MSS could have been shown for higher values of probability p.

4.4 Almost Sure Stability of a Bernoulli Process

Almost Sure Stability (ASS) is a condition for stability of a MJLS governed by a Markov Chain which is less conservative than Mean Square Stability, since the only requirement is that the system's states converge to zero asymptotically.

$$\mathbb{P}\left[\lim_{k\to\infty} \|x_k\| = 0\right] = 1 \tag{53}$$

For a Bernoulli process, the main idea is that, if no packet is lost, then the Lyapunov Function will decrease by some value $\lambda \in [0, 1)$.

$$V(F_0^{cl} x_k) \leq \lambda V(x_k) \tag{54}$$

On the other hand if a packet is lost, a decrease in the Lyapunov function is not guaranteed. The value of the Lyapunov function at the next stage will be scaled by a factor $L \geq 0$.

$$V(F_1^{cl} x_k) \leq L V(x_k) \tag{55}$$

What ASS stability tells us is that, as long as L is bounded, $L < \infty$, then the net result between the decreases in the Lyapunov function if no packet is lost, and the increases/decreases if a packet is lost is a decrease on the whole (is smaller than one), then the net result will be a decay in the Lyapunov function. Since a packet will be lost with probability p, this net result can be summarized as follows:

$$\lambda^{(1-p)} L^p < 1 \tag{56}$$

Note that the amount of parameters to vary here can increase the problem complexity quite a bit. The probability $p \in [0, 1]$, the bound $\lambda \in [0, 1)$ and the bound $L \in [0, \infty)$ can all vary. Further, even if the shape of the Lyapunov function is assumed quadratic, $V(x_k) = x_k^\intercal P x_k$, there's still the problem with finding a valid matrix $P \succ 0$.

As such, it can be useful to fix some parameters in order to reduce search space complexity. The major issue here is the bound L. This sets an upper bound on how much the Lyapunov Function can increase if a packet is lost. If the objective is to be as encompassing as possible, then the upper bound on equation 55 should be as large as possible while meeting equation 56. This will allow for the largest possible increase in the Lyapunov function value, while still ensuring ASS. As such, for a fixed p and λ, L is fixed the largest value for which equation 56 is fulfilled.

$$L = \left(\frac{1}{\lambda^{(1-p)}} \right)^{\frac{1}{p}} - \varepsilon \tag{57}$$

Having reduced the search space, the procedures is as follows: for a fixed value of p, the possible values for λ (with corresponding L) are searched for a solution to the set of LMIs described by equations 54 and 55.

$$\begin{cases} F_0^{cl\intercal} P F_0^{cl} - \lambda P \preceq 0 \\ F_1^{cl\intercal} P F_1^{cl} - L P \preceq 0 \end{cases} \tag{58}$$

If a value of λ exists for which a valid solution $P \succ 0$ can be found, then ASS is guaranteed for probability p. The probability value is then increased and the process repeated until it's no longer possible to find a valid solution $P \succ 0$ for any $\lambda \in [0, 1)$. The value of p just before this last one is the maximum for which ASS is guaranteed, p^*_{ASS}.

This is a tedious procedure, even when considering L as a strict function of other parameters since it requires searching a whole range possible parameters for the probability p and the bound λ.

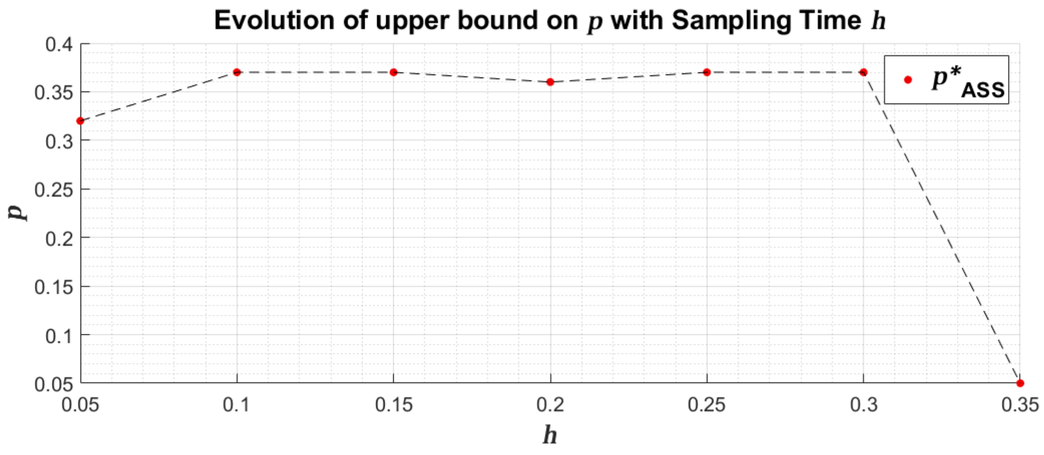


Figure 13: Evolution of p^*_{ASS} with sampling time h

Figure 13 shows the results obtained for p^*_{ASS} for a range of sampling times $h \in [0.05, 0.35]$. Note that the values p^*_{ASS} obtained here are larger across the board than the ones in Figure 9. This is in line with the theory stating that the ASS approach is less conservative than MSS. Interestingly, for most values of h attempted, p^*_{ASS} seems to range around the same value, $p^*_{ASS} = 0.37$. For $h = 0.35s$ the closed loop itself is very close to instability, and as such, the sharp drop in p^*_{ASS} makes sense.

4.5 Mean Square Stability of a Gilbert-Elliot Process

A Gilbert-Elliot process is more complex than the simple Bernoulli model, but nevertheless, is a rather simple model that can described by two conditional probabilities. These are the probabilities that the system will stick to the last behaviour it exhibited.

$$\begin{cases} \mathbb{P}[m_k = 0 \mid m_{k-1} = 0] = p_{00} \\ \mathbb{P}[m_k = 1 \mid m_{k-1} = 1] = p_{11} \end{cases} \tag{59}$$

Figure 14 shows a graphical representation of the process. Considering a to-zero mechanism, for state 0, there is no packet loss and the closed loop system evolves according to $x_{k+1} = F_0^{cl} x_k$. State 1 implies a packet loss occured and the system evolves autonomously as $x_{k+1} = F_1^{cl} x_k$.

Therefore, the system is a MJLS, and MSS is reduced to evaluating a set of as many Lyapunov equations as states in the Markov Chain. In this particular case, both Laypunov equations will be the same, and the problem is simplified to the analysis of a single Lyapunov equation.

$$\begin{cases} P_0 - p_{00} F_0^{cl\intercal} P_0 F_0^{cl} - (1 - p_{00}) F_1^{cl\intercal} P_1 F_1^{cl} \succ 0 \\ P_1 - (1 - p_{11}) F_0^{cl\intercal} P_0 F_0^{cl} - p_{11} F_1^{cl\intercal} P_1 F_1^{cl} \succ 0 \end{cases} \tag{60}$$

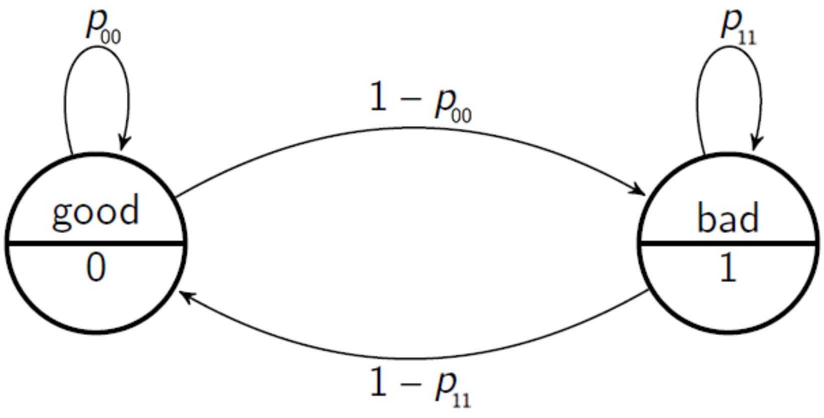


Figure 14: Stochastic packet losses model (Gilbert-Elliot). [3]

The system will remain stable as long as it's possible to find a matrix $P \succ 0$ that satisfies the LMIs above. For a fixed sampling times, it's possible to scan a set of probabilities p_{00} and p_{11} and check to see which pairs yield feasible solutions for P. Application of this procedure to the system in equation 46, for a fixed sampling times $h = 0.1s$, results in the plot in Figure 15.

Reflecting on what the plots show us, it's possible to reason about the results. If the probability of staying consecutive packet drops is any higher than $p_{11} = 0.33$, then no combination will be stable, since the probability of packets lost is too high. Conversely, if the probability of not loosing packets consecutively is low, $p_{00} <= 0.54$, then the system will be too likely to lose packets, and the system will be unstable. In the range where the probability of not loosing packets consecutively is high, and the probability of dropping lots of packets in a row is low, the system will have a chance to remain stable. The higher p_{00} is, the more tolerance the system will have for dropping packets consecutively, p_{11}.

Similar simulations were conducted for other sampling intervals. Figure 16 shows these results.

4.6 Comparison of Bernoulli and Gilbert-Elliot Processes

This section goes over the similarities and differences found between the Bernoulli and Gilbert-Elliot models in the metrics of MSS and ASS.

It's rather easy to view the Bernoulli process as a subset of the Gilbert-Elliot Model. In the Gilbert-Elliot model, simply denote $p_{11} = p$ and set $p00 = 1 - p_{11} = 1 - p$. The result is a function $p00 = 1 - p_{11}$ describing an line where the Gilbert Elliot and Bernoulli models are equivalent.

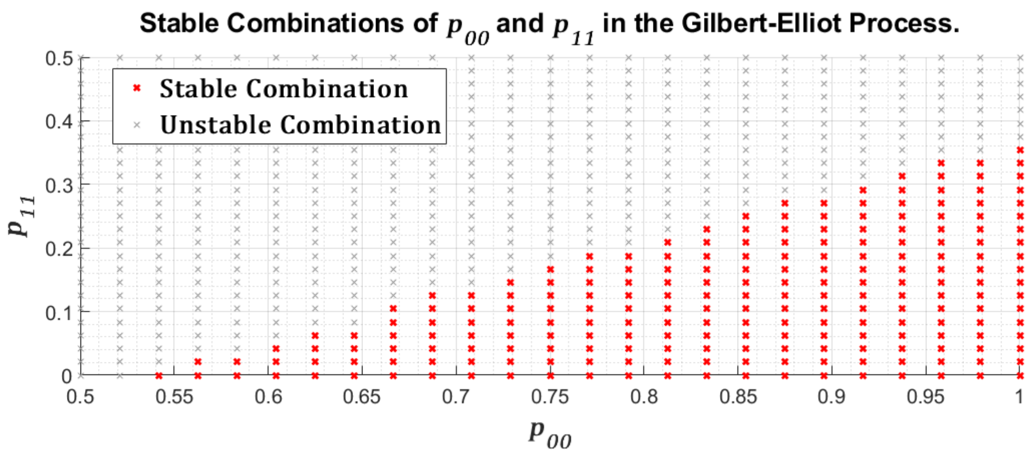


Figure 15: Stable and Unstable combinations of probabilities p_{00} and p_{11}.

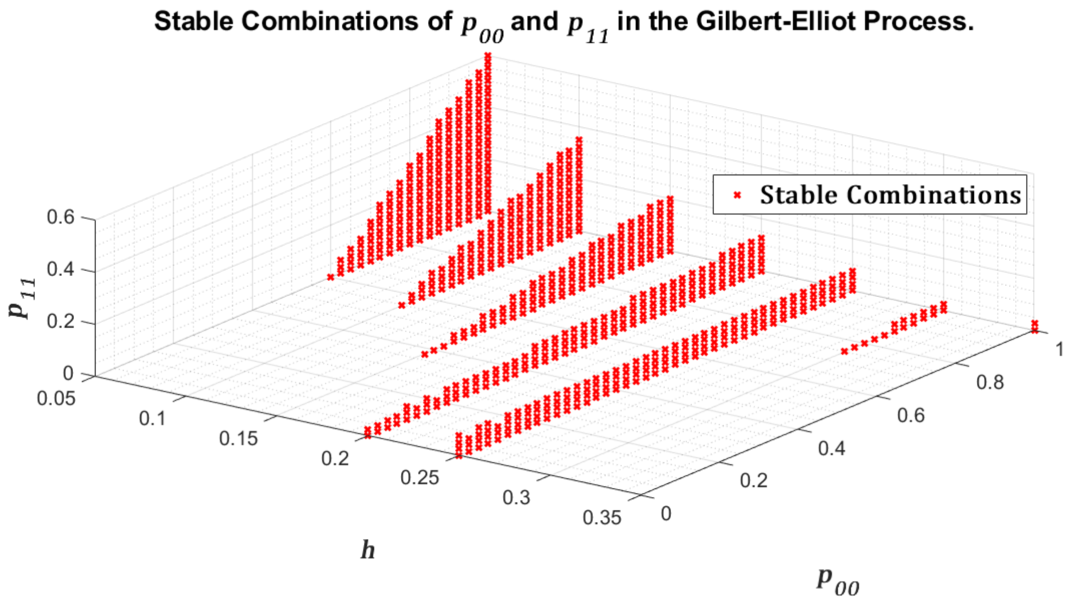


Figure 16: Stable combinations of probabilities p_{00} and p_{11} for multiple sampling intervals.

This can be seen in Figure 17 plotted over the results from Figure 15 For a Gilbert-Elliot process, solving the set of LMIs needed to prove MSS in the subset of probabilities $p11 = 1 - p_{00}$ is an easy accomplishable task. Figure 18 shows a plot of these findings. Note the heavy similarities between this and the plot found in Figure 9.

Though quite similar, they are, in fact, not the same. Note that the plot shown here is a bit more conservative than the one in Section 4.2. This may be due to the non-equivalence between

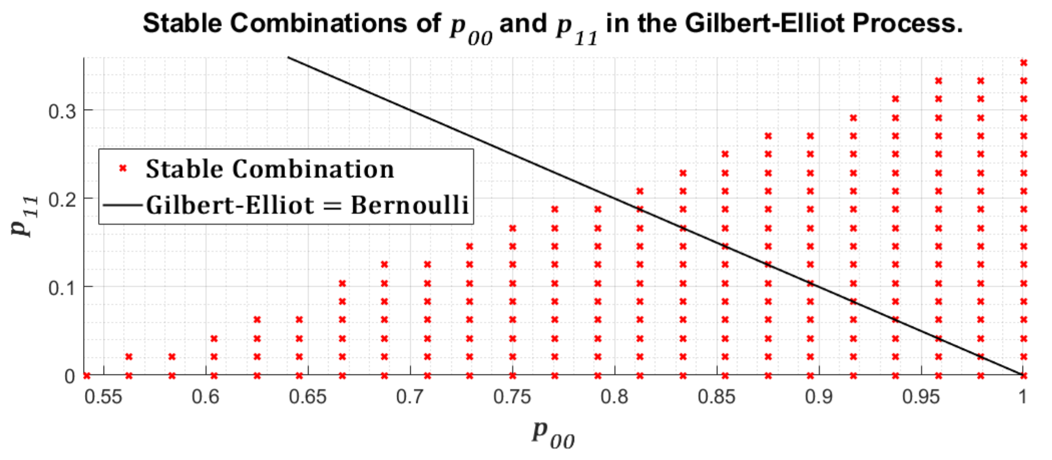


Figure 17: Stable combinations of probabilities p_{00} and p_{11} for $h = 0.1s$.

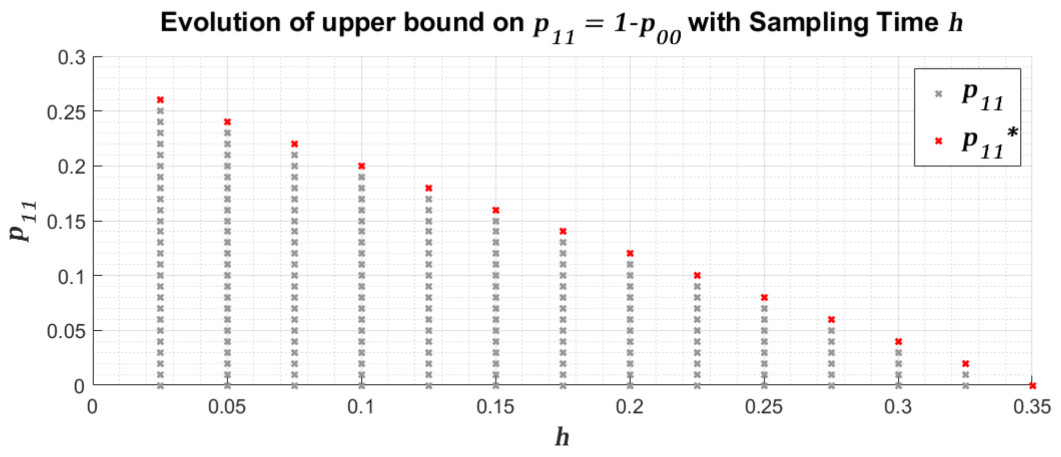


Figure 18: Stable combinations of probabilities $p_{11} = 1 - p_{00}$ for multiple sampling intervals.

the methods used. In fact, if $p = p_{11} = 1 - p_{00}$, the following implication holds.

$$\begin{cases} P_0 - p_{00} F_0^{cl\top} P_0 F_0^{cl} - (1-p_{00}) F_1^{cl\top} P_1 F_1^{cl} \succ 0 \\ P_1 - (1-p_{11}) F_0^{cl\top} P_0 F_0^{cl} - p_{11} F_1^{cl\top} P_1 F_1^{cl} \succ 0 \end{cases} \Leftrightarrow$$
$$\begin{cases} P_0 - (1-p) F_0^{cl\top} P_0 F_0^{cl} - p F_1^{cl\top} P_1 F_1^{cl} \succ 0 \\ P_1 - (1-p) F_0^{cl\top} P_0 F_0^{cl} - p F_1^{cl\top} P_1 F_1^{cl} \succ 0 \end{cases} \implies P - (1-p) F_0^{cl\top} P F_0^{cl} - p F_1^{cl\top} P F_1^{cl} \succ 0 \tag{61}$$

Note that the implication is just that, an implication. This means that the simplified LMI solving for a common Lyapunov matrix $P = P_0 = P_1$ is more general than solving the set of two LMIs for P_0 and P_1, implying the second method may be more restrictive, since finding P_0 and P_1, then it's possible to find a matrix P for the simplified LMI, but the same isn't true the other way around, and finding a matrix P for the simplified LMI doesn't mean it's possible to find matrices P_0 and P_1 for the full set. This is mirrored exactly in the differences of the plots in Figures 9 and 18.

Comments comparing ASS and MSS for the Bernoulli case were made more extensively in Section 4.4. The values found make sense there as well. Once again, the implication that if a system is MSS then it's also ASS only holds in one direction, and as such, the values of p^* are expected to be larger in the ASS case. This was corroborated by the experimental results.

5 Stability conditions via Over-Approximation

This section will concern itself with the stability of systems under small uncertain delays, $\tau \in [0, h)$. Since the delay is uncertain, there's an essentially infinite set of different delays that the system can have. Under the previously discussed approaches to prove stability of systems under time-delays, this would imply solving an LMI for every single delay the system can exhibit, resulting in and infinite set of LMIs, since every different delay will correspond to a different set of matrices describing the evolution of the system.

Fortunately, the bounded nature of the delays means it's possible to construct an over- approximation of the system. This is simply a way to bound the uncertain (and possibly non-convex) set of matrices describing the evolution of the system by a convex polytopic set. This is helpfully illustrated in the lectures relative to Lecture 4 in [3], and reproduced in Figure 19 for simplicity.

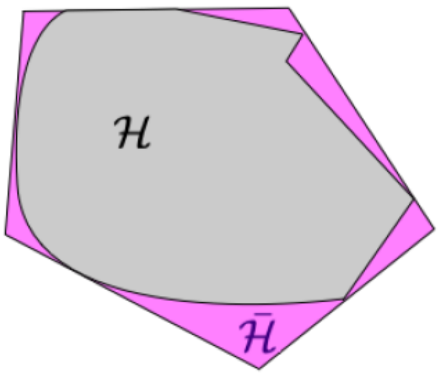


Figure 19: Illustration of an over-approximation of set $\mathcal{H}$ by a set $\bar{\mathcal{H}}$.

The theory of why over-approximations are useful is simple. Since the polytope is convex, proving stability for the matrices corresponding to the vertices of the polytope also guarantees stability of the set of matrices inside the polytope. Since the over-approximation fully encloses the set of matrices describing all possible dynamics of the system, stability of the system under any and all delays follows. This allows us to reduce an infinite set of LMIs to a finite set of N LMIs, where N is the number of vertices of the over-approximating polytope.

5.1 Over-approximation with a Jordan form approach

The Jordan Form is one way to go about obtaining a polytopic over-approximation of the system under uncertain delays. The initial step is to re-write the system matrices $F^e(h)$ and $G^e(h)$ as a decomposition depending on scalar factors α_i.

Before proceeding, it's useful to indicate what the matrices $F^e(h)$ and $G^e(h)$ are in this context. Simulating small delays requires storing past inputs as system variables, using an extended system $x_k^e = [x_k, u_{k-1}]$. The corresponding system matrices are as follows:

$$\begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix} = \underbrace{\begin{bmatrix} F(h) & M_1(h,\tau) \\ 0 & 0 \end{bmatrix}}_{F^e(h,\tau)} \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix} + \underbrace{\begin{bmatrix} M_0(h,\tau) \\ I \end{bmatrix}}_{G^e(h,\tau)} u_k \tag{62}$$

With matrices $F(h)$, $M_0(h)$ and $M_1(h)$, derived in a previous assignment and simply indicated here without proof.

$$F(h) = \begin{bmatrix} e^{5h} & \frac{5}{16}\left(e^{5h} - e^{-3h}\right) \\ 0 & e^{-3h} \end{bmatrix} \tag{63}$$

$$M_0(h,\tau) = \begin{bmatrix} \frac{5\,\mathrm{e}^{-3\,(h-\tau)}}{48} + \frac{\mathrm{e}^{5\,(h-\tau)}}{16} - \frac{1}{6} \\ \frac{1}{3} - \frac{\mathrm{e}^{3\,(h-\tau)}}{3} \end{bmatrix} \qquad M_1(h,\tau) = \begin{bmatrix} \frac{5\,\mathrm{e}^{-3\,h}}{48} + \frac{\mathrm{e}^{5\,h}}{16} - \frac{5\,\mathrm{e}^{-3\,(h-\tau)}}{48} - \frac{\mathrm{e}^{5\,(h-\tau)}}{16} \\ \frac{\mathrm{e}^{-3\,(h-\tau)}}{3} - \frac{\mathrm{e}^{-3\,h}}{3} \end{bmatrix} \tag{64}$$

Using the Jordan form approach, it's possible to rewrite the system in 62 as:

$$x_{k+1}^e = \left(F_0 + \sum_{i=1}^{r} \alpha_i\left(\tau_k\right) F_i\right) x_k^e + \left(G_0 + \sum_{i=1}^{r} \alpha_i\left(\tau_k\right) G_i\right) u_k \tag{65}$$

Where the parameters $\alpha_i\left(\tau_k\right)$ are given as follows for the eigenvalues of the original system, $\lambda_i = \{-3, 5\}$.

$$\alpha_i\left(\tau_k\right) = \frac{(h-\tau_k)^j}{j!} \exp\left(\lambda_i(h-\tau_k)\right) \tag{66}$$

Since the original system has two eigenvalues, each with multiplicity 1, then $i \in \{1,2\}$ and $j = 0$, meaning there will be two separate α values for the system. Simple inspection of 62 and the matrices in 63 and 64 allows us to extract both.

$$\alpha_1\left(\tau_k\right) = \exp\left(-3(h-\tau_k)\right) \qquad \alpha_2\left(\tau_k\right) = \exp\left(5(h-\tau_k)\right) \tag{67}$$

This means the system in 62 can be rewritten as:

$$x_{k+1}^e = \left(F_0 + \alpha_1\left(\tau_k\right) F_1 + \alpha_2\left(\tau_k\right) F_2\right) x_k^e + \left(G_0 + \alpha_1\left(\tau_k\right) G_1 + \alpha_2\left(\tau_k\right) G_2\right) u_k \tag{68}$$

Separating $F^e(h,\tau)$ and $G^e(h,\tau)$ accordingly:

$$F_0 = \begin{pmatrix} \mathrm{e}^{5\,h} & \frac{5\,\mathrm{e}^{5\,h}}{16} - \frac{5\,\mathrm{e}^{-3\,h}}{16} & \frac{5\,\mathrm{e}^{-3\,h}}{48} + \frac{\mathrm{e}^{5\,h}}{16} \\ 0 & \mathrm{e}^{-3\,h} & -\frac{\mathrm{e}^{-3\,h}}{3} \\ 0 & 0 & 0 \end{pmatrix} F_1 = \begin{pmatrix} 0 & 0 & -\frac{5}{48} \\ 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 \end{pmatrix} F_2 = \begin{pmatrix} 0 & 0 & -\frac{1}{16} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$
$$G_0 = \begin{pmatrix} -\frac{1}{6} \\ \frac{1}{3} \\ 1 \end{pmatrix} \quad G_1 = \begin{pmatrix} \frac{5}{48} \\ -\frac{1}{3} \\ 0 \end{pmatrix} \quad G_2 = \begin{pmatrix} \frac{1}{16} \\ 0 \\ 0 \end{pmatrix} \tag{69}$$

From this framework, the construction of a polytopic over-approximation is done by taking the maximum and minimum values of α_1 and α_2. The matrices corresponding to all combinations of these edge cases will be the vertices of the polytope.

$$\begin{aligned} \overline{F^e(h,\tau)} &= \left\{ F_0 + \sum_{i=1}^{r} \delta_i F_i \mid \delta_i \in [\underline{\alpha}_i, \bar{\alpha}_i], i = 1,2,\dots,r \right\} \\ \overline{G^e(h,\tau)} &= \left\{ G_0 + \sum_{i=1}^{r} \delta_i G_i \mid \delta_i \in [\underline{\alpha}_i, \bar{\alpha}_i], i = 1,2,\dots,r \right\} \end{aligned} \tag{70}$$

The final result are four sets of matrices describing the over-approximation:

$$\begin{cases} x^e_{k+1} = (F_0 + \overline{\alpha}_1 F_1 + \bar{\alpha}_2 F_2)\, x^e_k + (G_0 + \overline{\alpha}_1 G_1 + \overline{\alpha}_2 G_2)\, u_k \\ x^e_{k+1} = (F_0 + \overline{\alpha}_1 F_1 + \underline{\alpha}_2 F_2)\, x^e_k + (G_0 + \overline{\alpha}_1 G_1 + \underline{\alpha}_2 G_2)\, u_k \\ x^e_{k+1} = (F_0 + \underline{\alpha}_1 F_1 + \bar{\alpha}_2 F_2)\, x^e_k + (G_0 + \underline{\alpha}_1 G_1 + \overline{\alpha}_2 G_2)\, u_k \\ x^e_{k+1} = (F_0 + \underline{\alpha}_1 F_1 + \underline{\alpha}_2 F_2)\, x^e_k + (G_0 + \underline{\alpha}_1 G_1 + \underline{\alpha}_2 G_2)\, u_k \end{cases} \tag{71}$$

Since $\alpha_1 \in \left[e^{-3\,(h-\tau_{\min})}, e^{-3\,(h-\tau_{\max})}\right]$ and $\alpha_2 \in \left[e^{5\,(h-\tau_{\max})}, e^{5\,(h-\tau_{\min})}\right]$, the maximum and minimum values of α_1 and α_2 will naturally be:

$$\begin{cases} \overline{\alpha}_1 = e^{-3\,(h-\tau_{\max})}, & \underline{\alpha}_1 = e^{-3\,(h-\tau_{\min})} \\ \overline{\alpha}_2 = e^{5\,(h-\tau_{\min})}, & \underline{\alpha}_2 = e^{5\,(h-\tau_{\max})} \end{cases} \tag{72}$$

For a certain sampling interval, varying $\tau_{\min}$ and $\tau_{\max}$ such that $0 \leq \tau_{\min} \leq \tau_{\max} < h$ and solving the four corresponding LMIs based on the set of equations in 71 will yield the maximum and minimum values of delay the polytopic over-approximation can handle without losing stability. This will provide a somewhat conservative approximation of the values of delay for which the true system remains stable.

The approximation will be conservative since the polytopic approximation represents a larger set of dynamics than the real system. As such, all the dynamics of the system will be assured to remain stable, but so will other matrices whose dynamics are not reachable by the system. This method guarantees stability for a set of matrices larger than strictly necessary, and is therefore conservative in it's results.

5.2 Stability Analysis via Polytopic Over-Approximation

In the last subsection, the framework was introduced to analyse the stability of the system under a varying uncertain delay, known to be in the interval $\tau \in [0, h)$. The final result is a set of four systems whose dynamics encompass all infinite possible dynamics the real system can take. Guaranteeing stability for those 4 systems will also guarantee stability of the real system.

It was hinted that, to analyse the stability of the four vertices of the polytopic over-approximation, the strategy was to solve a set of LMIs in search of a common Lyapunov equation, a strategy that has been pursued before with satisfactory results, when a finite set of matrices needs stability guaranteed for.

The initial step is to understand what those LMIs look like. The four sets of dynamics will have corresponding closed loops of the form:

$$x^e_{k+1} = F_{cl}(h,\tau,\alpha_1,\alpha_2) = x^e_k = \left\{ \left(F_0 + \sum_{i=1}^{r} \alpha_i\left(\tau_k\right) F_i \right) - \left(G_0 + \sum_{i=1}^{r} \alpha_i\left(\tau_k\right) G_i \right) \bar{K} \right\} x^e_k \tag{73}$$

The problem of finding a Lyapunov Function for this system, under the assumption of a quadratic form, is reduced to solving the following LMI:

$$F_{cl}(h,\tau,\alpha_1,\alpha_2)^{\intercal} P F_{cl}(h,\tau,\alpha_1,\alpha_2) - P \preceq -I \tag{74}$$

As such, the four LMIs that need to be solved to prove stability of the vertices of the polytopic over-approximation are:

$$\begin{cases} F_{cl}(h,\tau,\overline{\alpha}_1,\overline{\alpha}_2)^{\intercal} P F_{cl}(h,\tau,\overline{\alpha}_1,\overline{\alpha}_2) - P \preceq -I \\ F_{cl}(h,\tau,\overline{\alpha}_1,\underline{\alpha}_2)^{\intercal} P F_{cl}(h,\tau,\overline{\alpha}_1,\underline{\alpha}_2) - P \preceq -I \\ F_{cl}(h,\tau,\underline{\alpha}_1,\overline{\alpha}_2)^{\intercal} P F_{cl}(h,\tau,\underline{\alpha}_1,\overline{\alpha}_2) - P \preceq -I \\ F_{cl}(h,\tau,\underline{\alpha}_1,\underline{\alpha}_2)^{\intercal} P F_{cl}(h,\tau,\underline{\alpha}_1,\underline{\alpha}_2) - P \preceq -I \end{cases} \tag{75}$$

Naturally the set $\{\underline{\alpha}_1, \underline{\alpha}_2, \overline{\alpha}_1, \overline{\alpha}_2\}$ will vary depending on $\{\tau_{\min}, \tau_{\max}\}$. As such, a strategy to solve for stability of the system is to vary the values of $\{\tau_{\min}, \tau_{\max}\}$, calculate $\{\underline{\alpha}_1, \underline{\alpha}_2, \overline{\alpha}_1, \overline{\alpha}_2\}$ and check for stability of the four LMIs. If stability is guaranteed for these, then it will also be guaranteed for the underlying system.

Assuming, for simplicity, $\tau_{\min} = 0s$ (zero delay), it's simple to find the maximum range of values for which the polytopic approximation is stable by simple increasing the value of $\tau_{\max}$ until a common Lyapunov function can no longer be found. As long as the delay affecting the true system stays inside the bounds found for the polytopic approximation, stability for will be guaranteed independent of the actual time delay inside the range $\tau \in [\tau_{\min}, \tau_{\max}]$.

Following this strategy it's possible to find values of $\{\tau_{\min}, \tau_{\max}\}$ for a range of sampling times. The results are in Figure 20. The resulting plot is similar to the one obtained in Assignment 1.

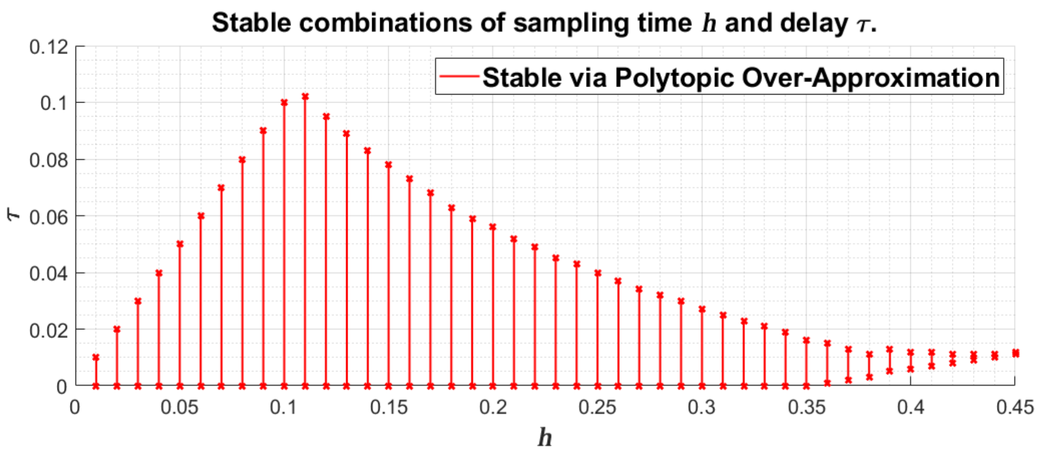


Figure 20: Range of uncertain delays resulting in stable closed loop dynamics, employing an over-approximating polytope.

Note that, up to $h = 0.1s$, the remains stable under any an all delays $\tau \in [0, h)$. The largest range of delays the system can support without becoming unstable is found for $h = 0.11s$, where $\tau_{\max} = 0.102s$. For sampling times higher than that, the range of delays progressively diminishes. After $h = 0.35s$, simply having zero delay no longer stabilizes the system, but having some delay added to the system can make it return to stability. No combination of sampling time and delay is found that can make the system stable for $h > 0.45s$.

5.3 Refining the Polytopic Over-Approximation

In refining the polytopic over-approximation, it's perhaps useful to understand what the initial polytope looks like. For a specific sampling time interval, $h = 0.1s$, it's possible to plot the evolution of α_1 and α_2 with the evolution of$\{\tau_{\min}, \tau_{\max}\}$. The result is a line evolving from $\{\underline{\alpha}_1, \underline{\alpha}_2\}$ to $\{\overline{\alpha}_1, \overline{\alpha}_2\}$. Naturally, the over-approximation will be a rectangle fully encompassing this line. Figure 21 shows an illustration of this concept. For the aforementioned sampling time, $h = 0.1s$, the the system is stable for the full range of delays $\tau \in [0, h)$.

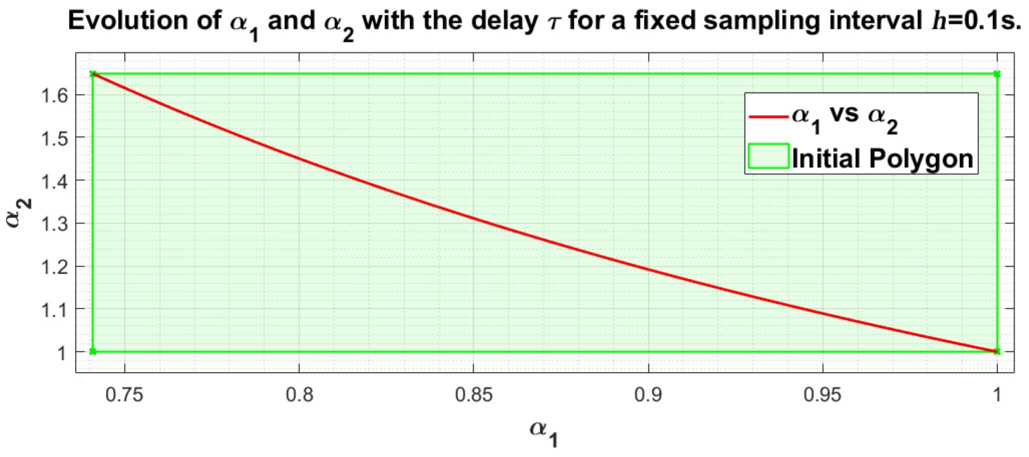


Figure 21: Over-approximating polytope of the evolution α_1 vs. α_2.

Under the polytopic over-approximation analysis, stability is guaranteed for all combinations of sampling times in the area shaded in green. This is a much larger area than necessary. As such, it's possible to add more points in order to refine the stability results obtained in the last subsection. For instance, interpolating between α_1 and α_2, it's possible to arrive at an hexagonal (convex) polygon which is a much better fit for the initial curve. For this particular example, 6 points are employed at coordinates in (76). The resulting shape is found in Figure 22.

$$\begin{cases} (\underline{\alpha}_1, \underline{\alpha}_2)\,, & (0.99\,\underline{\alpha}_1 + 0.01\,\overline{\alpha}_1, \underline{\alpha}_2)\,, & (\overline{\alpha}_1, 0.01\,\overline{\alpha}_2 + 0.99\,\underline{\alpha}_2)\,, \\ (\overline{\alpha}_1, \overline{\alpha}_2)\,, & (0.10\,\underline{\alpha}_1 + 0.90\,\overline{\alpha}_1, \overline{\alpha}_2)\,, & (\underline{\alpha}_1, 0.90\,\overline{\alpha}_2 + 0.10\,\underline{\alpha}_2) \end{cases} \tag{76}$$

Naturally, the shape of the curve describe the evolution of α_1 and α_2 will change depending on the

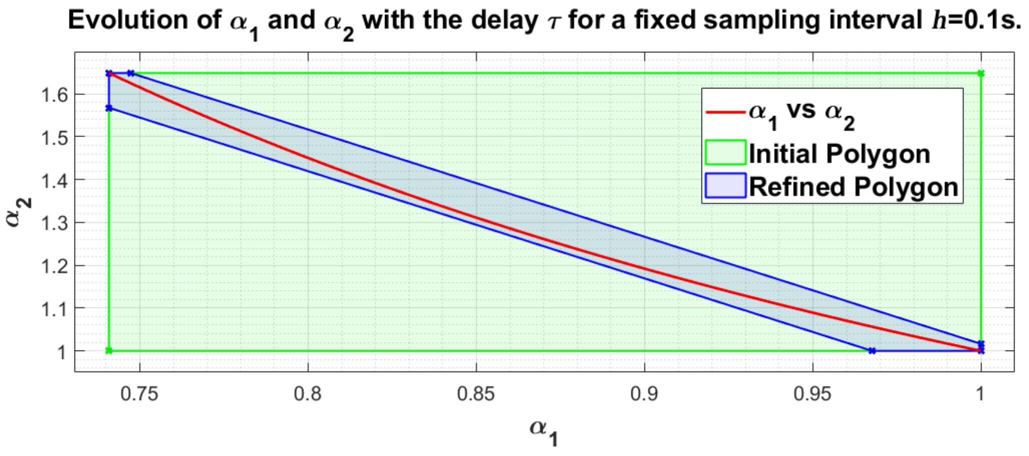


Figure 22: Over-approximating polytope and refined polytope of the evolution α_1 vs. α_2.

sampling time h and the delays $\{\tau_{\min}, \tau_{\max}\}$. The refined polytopes can, however, be adjusted to the different curvatures by changing the location of the points of the hexagon.

This was attempted for the same range of sampling times as seen previously. The now 6 LMIs were solved using the same strategy described above. The results are found in Figure Unfortunately,

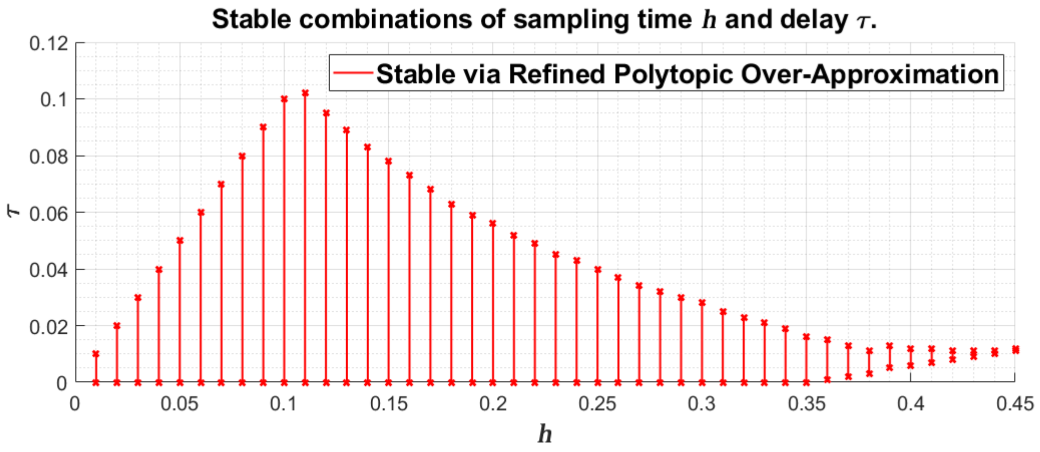


Figure 23: Range of uncertain delays resulting in stable closed loop dynamics, employing a refined over-approximating polytope.

the results don't seem to differ, at all, from the ones obtained previously, despite what would be expected from such an approach since much of the area covered by the original polytope is not covered by the refined over-approximation. Possible explanations, aside from the obvious coding and implementation errors that could have occurred, are restricted to the possibility that, somehow, the relatively small range of delays which is, at most, $1/10th$ of a second imply that the original polytope is already a good fit for the system.

5.4 Comparison to previously obtained results

Assignment 1 included a question about stability of the system under a varying number of time delays and sampling intervals. For simplicity, the obtained plot is reproduced in Figure 24. For simplicity in the comparison, the Figure shows an overlay of the results obtained in Assignment 1 and the results obtained by (both) over-approximating polytopes. The portion in red is the same as in Figures 20 and 23. The grey portion shows the extra area covered by the plot in Figure 24. Note that the results obtained by the over-approximating polytope yield the same range up to $h = 0.1s$, where the range of delays supported by the system is the full $\tau \in [0, h)$. For sampling times higher than this, however, the polytope results in a more conservative range than the cloud of points used in Assignment 1.

This is to be expected. The obvious reason here is the conservative nature of the polytope approach, which covers a larger range of matrices than strictly necessary, and may therefore yield more

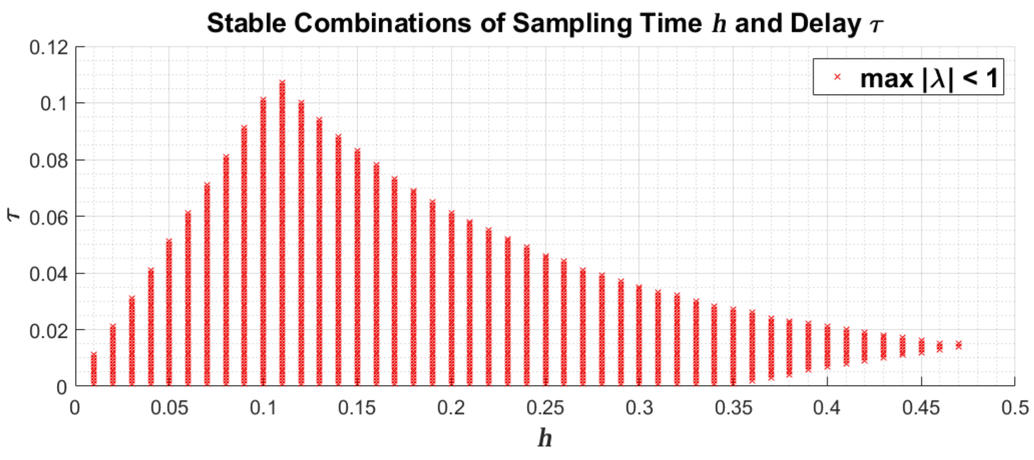


Figure 24: Pairs of sampling time and delay resulting in stable closed loop dynamics.

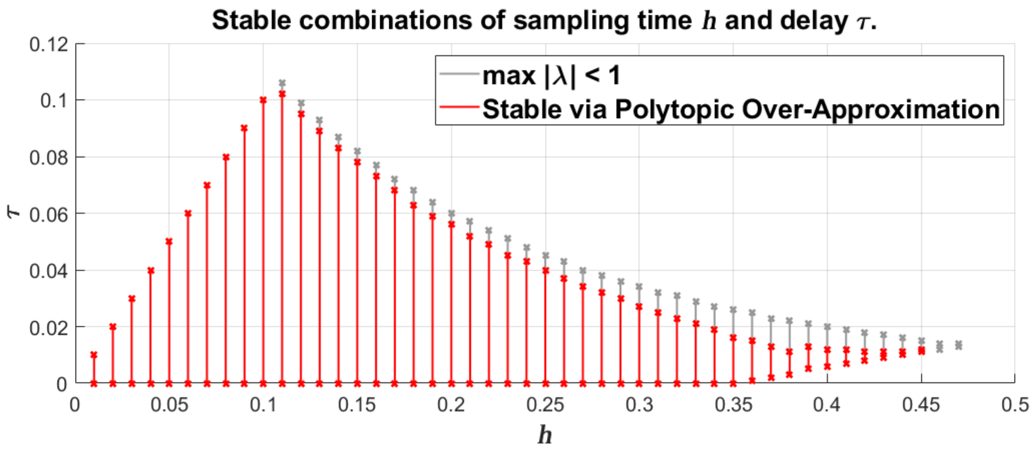


Figure 25: Overlay of results obtained in Assignments 1 and 2.

conservative results, as seen. Another possibility here is that the interactions between some sets of delays which are independently stable may result in an unstable system if they happen together. Note that the cloud of points used to obtain the plot in Figure 24 considered individual constant delays, and not the interactions between them. The plot in Figures 20 and 23 gives assurance for varying time delays, implying any and all delays in the red range will result in a stable system.

As a final note, it would be expected that the refined polytope would result in slightly larger ranges of delays for an individual sampling interval, when compared to the original four point over-approximation. The reasoning for this is that the range of matrices whose stability is guaranteed for is a small subset of the original polytope. This, however, proved not to be the case, for reasons discussed in the previous section. Further, it's not expected that the range obtained by the refined polytope would exceed that of Figure 24, since, even though the subset of matrices is much smaller than in the original polytope, it would still be larger than strictly necessary. Additionally, the interactions between sampling times could still prove a limiting factor in this scenario.

6 Event-Triggered Control of Continuous-Time Systems

An event-triggered controller will sample the system not according to regular time intervals, but rather depending on a performance metric. For continuous time systems, the system will trigger when the decrease in the Lyapunov function is no longer deemed sufficient, according to a performance value $\sigma \in (0, 1)$.

$$\frac{d}{dt}V(x(t)) \leq -\sigma x(t)^{\intercal} Q x(t) \tag{77}$$

Naturally, $\sigma \neq 1$, since, using this strategy, some performance will always be sacrificed. The reason for this is that the input will be held for some time. At the first moment the input is held, performance will be optimal. However, as the system evolves, that same input may no longer be optimal (or desirable), hence, performance will deteriorate. The crux of the matter here is in how to choose when to take the next sample, such that performance doesn't deteriorate beyond an acceptable level.

6.1 Event-Triggered control guaranteeing GES

Starting from the initial performance metric in (77), it's possible to arrive at a more concrete triggering condition if the Lyapunov function is assumed quadratic, $V(x(t)) = x(t)^{\intercal} P x(t)$. Note additionally that, for a certain held time sample $t = s_k$, the system will evolve according to:

$$\dot{x}(t) = A\,x(t) - B\bar{K}\,x(s_k) \tag{78}$$

The error at time t, between the optimal input $u(t) = -\bar{K}x(t)$ and the actual input $u(t) = -\bar{K}x(s_k)$ can be written as $\tilde{u}(t) = -\bar{K}\varepsilon(t)$, where $\varepsilon(t) = x(s_k) - x(t)$. The system's evolution can be seen can be captured in the extended system with $x^e(t) = [x(t) \;\; \varepsilon(t)]$:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{\varepsilon}(t) \end{bmatrix} = \begin{bmatrix} A - B\bar{K} & -B\bar{K} \\ -A + B\bar{K} & B\bar{K} \end{bmatrix} \begin{bmatrix} x(t) \\ \varepsilon(t) \end{bmatrix} \tag{79}$$

This system representation, together with the quadratic assumption on the Lyapunov function allows for the manipulation of the performance metric in (77) into the following quadratic form:

$$\phi^e\left(x(t), \varepsilon(t)\right) := \begin{bmatrix} x(t)^{\intercal} & \varepsilon(t)^{\intercal} \end{bmatrix} \begin{bmatrix} (1-\sigma)Q & PB\bar{K} \\ \bar{K}^{\intercal}B^{\intercal}P & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ \varepsilon(t) \end{bmatrix} \geq 0 \tag{80}$$

As such, performance will be guaranteed, and the system will remain globally exponentially stable, as long as equation 80 is true. This provides a simple way to know when to take the next sample. As soon as the inequality is no longer true, the next sample should be taken, as to guarantee continued performance and GES of the system. Note that as per the slides on Lecture 4 in [3], this triggering condition is bounded by a minimum time interval, implying any problems relating to Zeno behaviour will not be an issue here.

6.2 Simulation of the Event Triggered System

The initial problem to overcome when trying to simulate the event triggered system is finding matrices P and Q in order to actually implement the condition in (80). One way to do this is by solving the LMI:

$$(A - B\bar{K})^\intercal P + P(A - B\bar{K}) = -Q \tag{81}$$

Solving this LMI, or, equivalently,

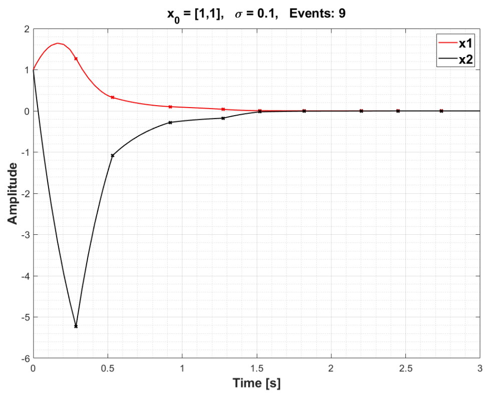
$$\begin{cases} (A - B\bar{K})^\intercal P + P(A - B\bar{K}) \geq -Q \\ (A - B\bar{K})^\intercal P + P(A - B\bar{K}) \leq -Q \end{cases} \tag{82}$$

allows for the recovery of matrices P and Q.

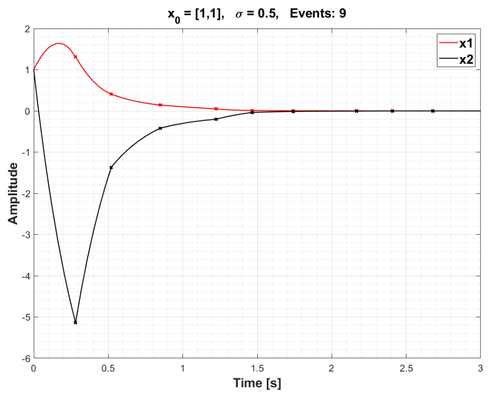
Simulating the system in continuous time with triggering conditions can now be done easily in `Matlab` using the `ode45()` function to the simulate system in (79), and the `'Events'` option to continuously monitor the triggering condition in (80). The system was simulated for a total of 3 seconds.

Figure 26 shows differences in the system's response for the same initial condition and different values of σ. The expected behaviour is to have the number of events increase as σ increases, since as sigma approaches $\sigma = 1$, the performance is expected to be closer to the optimal continuous time performance. Unfortunately, the system being analysed doesn't show many differences in the majority of the range, $\sigma \in (0, 0.9]$. The number of events does start increasing, as expected, but for higher values of $\sigma \geq 0.9$.

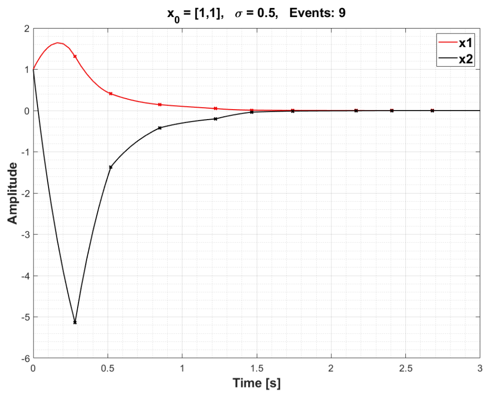
Another test for the triggering system is to fix σ and vary the initial conditions, verifying how the behaviour of the system changes. Here, the number of events is expected to vary slightly between different conditions, but not as dramatically as the changes in the number of events cause by varying σ. Once again, however, it seems that the particular system under analysis is quite resistant to changes in the number of triggers despite changes. In fact despite varied initial conditions, the system always responded with 9 events within the 3 seconds of simulation, a fact visible in Figure 27.



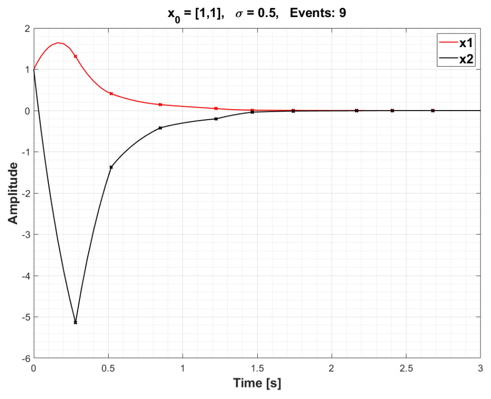
(a) $\sigma = 0.1$ - 9 Events



(b) $\sigma = 0.5$ - 9 Events



(c) $\sigma = 0.9$ - 10 Events



(d) $\sigma = 0.99$ - 17 Events

Figure 26: State Evolution and Events for different σ value and $x_0 = [1,\ 1]$.

Finally, the three different triggering conditions seen in class were implemented, in order to explore the differences between them. The three conditions are summarized here for convenience:

- Condition 1:

$$\phi^e\left(x(t), x(s_k)\right) := \begin{bmatrix} x(t)^\intercal & x(s_k)^\intercal \end{bmatrix} \begin{bmatrix} A^\intercal P + PA + \sigma Q & -PB\bar{K} \\ \bar{-}K^\intercal B^\intercal P & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ x(s_k) \end{bmatrix} \geq 0 \quad (83)$$

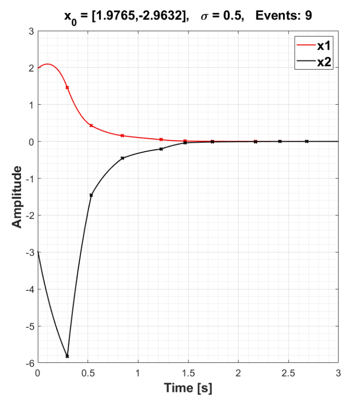
- Condition 2:

$$\phi^e\left(x(t), \varepsilon(t)\right) := \begin{bmatrix} x(t)^\intercal & \varepsilon(t)^\intercal \end{bmatrix} \begin{bmatrix} (1-\sigma)Q & PB\bar{K} \\ \bar{K}^\intercal B^\intercal P & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ \varepsilon(t) \end{bmatrix} \leq 0 \quad (84)$$

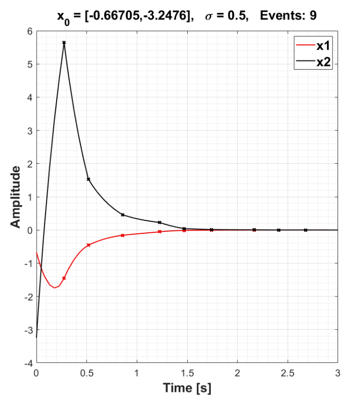
- Condition 3:

$$\frac{\|\varepsilon(t)\|}{\|x(t)\|} \geq \frac{(1-\sigma)\lambda_{\min}(Q)}{2\|PB\bar{K}\|} \quad (85)$$

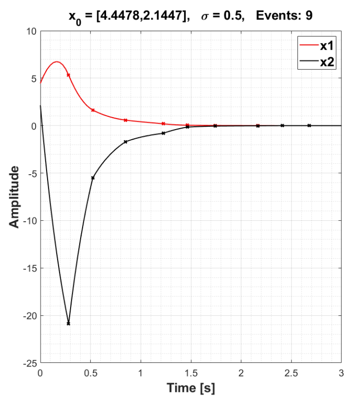
For the same initial condition x_0 and fixed σ, Figure 28 shows the results. As expected, the first two conditions give the same results. This is because both conditions are derived from the same



(a) $x_0 = [1.98,\ -2.96]$ - 9 Events



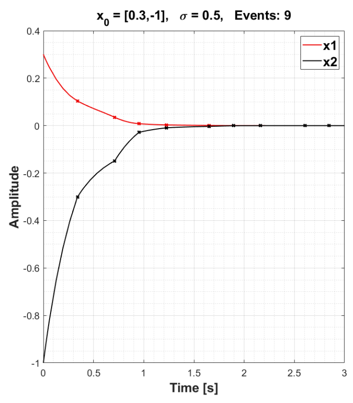
(b) $x_0 = [-0.67,\ -3.25]$- 9 Events



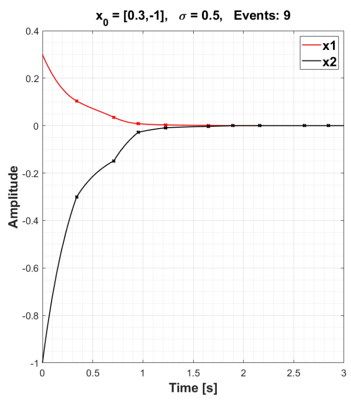
(c) $x_0 = [4.45,\ 2.14]$ - 9 Events

Figure 27: State Evolution and Events for different x_0 value and $\sigma = 0.5$.

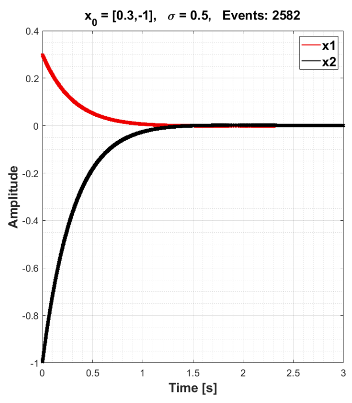
expression, and are just a transformation of each other. The last condition, as expected from the theory, is much more conservative than the first two. Somewhat unexpected is how much more conservative this final condition is than the other two, since it triggers about 287 times more often.



(a) Condition 1 - 9 Events



(b) Condition 2 - 9 Events



(c) Condition 3 - 2582 Events

Figure 28: State Evolution and Events for $x_0 = [0.3,\ -1]$, $\sigma = 0.5$ and different triggering conditions.

As a final note, it's likely that the last digit of the student number used in obtaining matrix A had a large influence in the behaviour of the system. Attempts were made with different (larger) final digits, and it was observed that the behaviour was closer to what would be expected by the theory.

6.3 Average inter-sample time guaranteeing 90% decrease of the Lyapunov function

Fixing $\sigma = 0.5$, it's possible to use the `Events` option in the $ode45()$ solver to further monitor for the Lyapunov decrease condition, $V(T) \leq 0.1V(0)$. Doing so, one can register the time T at which the condition is fulfilled.

The average inter-sample time can then be calculated by dividing the number of triggers in the interval $t \in [0, T]$ by the time T at which the Lyapunov decrease is guaranteed. If all triggers of the system are registered in a vector of times $s = [s_1, \ldots, s_k, \ldots, s_n]$, then this corresponds to:

$$h_i = \frac{T_i}{\sup\{k \mid s_k < T_i\}}, \qquad h^{avg} = \frac{\sum_{i=1}^{N} h_i}{N} \tag{86}$$

This is simple procedure in theory. However, in practise, the system being analysed resulted in some issues. The first issue was that, for some initial conditions, the Lyapunov decrease condition is met before any triggers happen. An example of this is $x_0 = [-0.25 \;\; 1.25]$. Having found matrices P and Q by solving the LMI indicated in the last section, at $T = 0.1856s$, the following holds:

$$\begin{aligned} x^e(t) &= [0.032 \;\; 0.1623 \;\; -0.2842 \;\; 1.3377]^\intercal \\ \phi^e\left(x(t), \varepsilon(t)\right) &= 10.6933 \geq 0 \\ V(T) = x(t)^\intercal P x(t) &= 0.4733 = 0.1 * x_0^\intercal P x_0 \end{aligned} \tag{87}$$

Figure shows an illustration of the evolution of this system. Naturally, inter-sample time in such systems, if calculated as in (**??**), will result in an infinite sampling time.

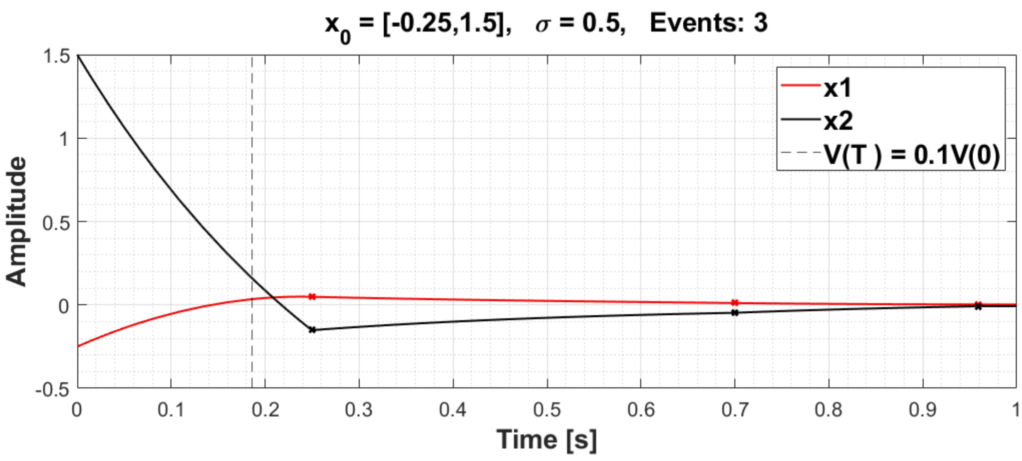


Figure 29: System evolution for $\sigma = 0.5$ and $x_0 = [-0.25 \;\; 1.25]$

Most initial conditions didn't have this kind of problem. Still, the maximum number of samples found, independent of initial condition, was $k = 1$. This is to be expected, since the system tends to have its first trigger relatively late as seen in Figure 27.

Starting from $N = 200$ random initial conditions, and removing the effect of initial conditions for which the Lyapunov condition is met before a trigger happens, the result was always $h = T_i$, and the average over the 200 initial conditions was $h^{avg} = 0.45s$.

Though not strictly necessary for this section, it was noted that simulating a discrete time system with sampling time h = h^{avg} and no delay results in an unstable system. This is easily confirmed by looking at Figure 25, for instance.

As such, a different strategy for calculating h^{avg} was attempted. The first trigger after time $t = T$ was picked as the first trigger at which the condition $V(T) \leq 0.1V(0)$ is fulfilled. The sampling time was then calculated using the index and time of this trigger. If all triggers of the system are registered in a vector of times $s = [s_1, \dots, s_k, \dots, s_n]$, then this corresponds to:

$$h_i = \frac{\inf\{s_k \mid s_k > T_i\}}{k}, \qquad h^{avg} = \frac{\sum_{i=1}^{N} h_i}{N} \tag{88}$$

For this way of going about calculating the average inter-sampling time, $N = 200$ runs gives an $h^{avg} = 0.32s$, which will result in a stable discrete system.

6.4 Comparison against an equivalent discrete system with constant sampling time

Simulating the equivalent discrete-time system for both h^{avg} found in the previous section, it's possible to come to some conclusions about the performance of event-triggered systems.

As previously stated, the first method used to find the average inter-sampling time results in $h^{avg} = 0.45s$. Figure 30 shows the difference in response of the continuous system when controller with the triggered controller, or when controller with an underlying discrete controller with zero delay and no packet losses. As far as conclusions go, it's interesting to see that the event triggered system is still stable even if the average time between triggers is larger than the maximum inter-sampling time for which the equivalent discrete time system is stable.

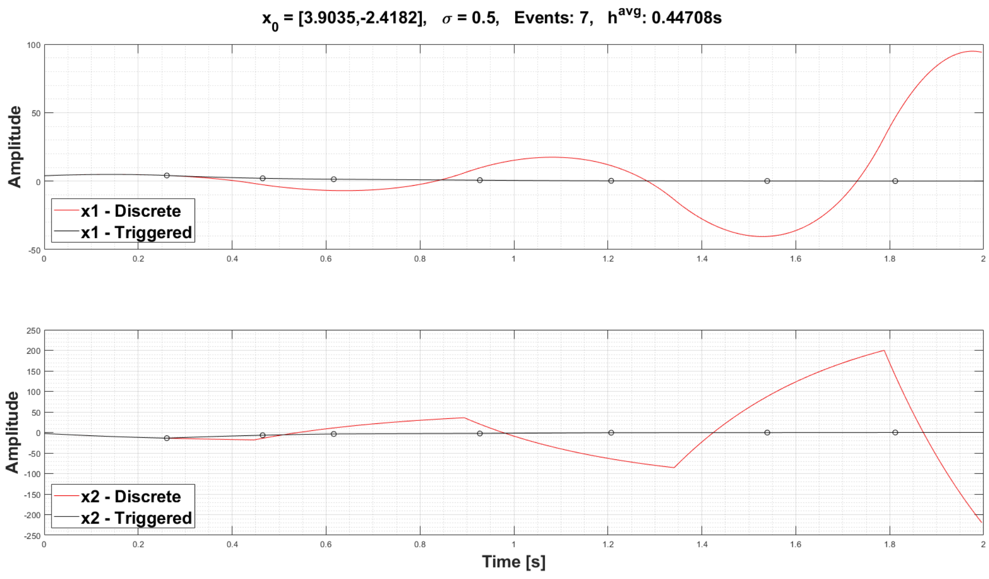


Figure 30: Response of the Continuous system when controlled by a Discrete vs. Triggered controller.

Using the second approach, with resulting $h^{avg} = 0.32s$, the equivalent discrete-time system does remain stable. However, the sampling time is quite large, and the system controlled by the discrete controller shows considerably more oscillatory behaviour, higher overshoot and takes longer to settle. Results visible in figure 31.

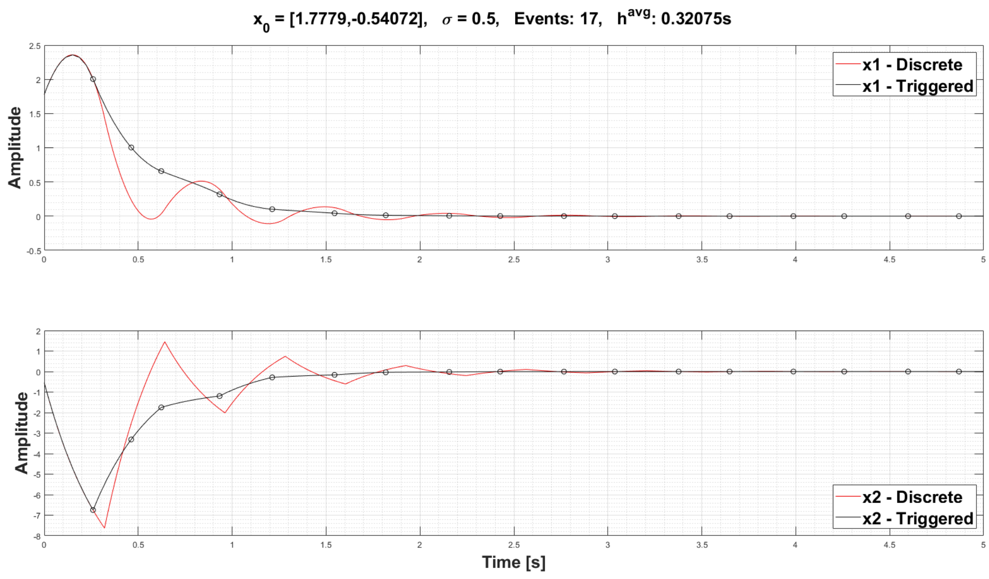


Figure 31: Response of the Continuous system when controlled by a Discrete vs. Triggered controller.

References

[1] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

[2] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

[3] Tamás Keviczky and Manuel Mazo Jr. Lecture slides on networked and distributed control systems [sc42100], 2021.

[4] Li Li and Fucheng Liao. Design of a preview controller for discrete-time systems based on lmi. *Mathematical Problems in Engineering*, 2015:1–12, 12 2015.