NETWORKED AND DISTRIBUTED CONTROL SYSTEMS

(SC42100)

Assignment 1

Responsible Instructors

Manuel Mazo

Tamás Keviczky

Gabriel Gleizer

Student

Daniel Varela - 5321263

May 25, 2021

Contents

1 Introduction

This assignment covers the topics in lectures 1 and 2 of the Networked and Distributed Control Systems Course.

Starting from a linear system with a state feedback controller, the effects of various sampling times and both constant and varying time delays are analysed. For each of the proposed cases, the controller is redesigned in such a way as to improve the systems robustness to delays.

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

2 Analysis Under no Delay Conditions

2.1 System Matrices

The system under analysis is a second order system whose parameters are dependent on the student number of each student. With parameters $a = 5$, $b = 2$ and $c = 3$, the following system is obtained:

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

The poles of the system are directly extracted from A, as the values along the diagonal $\lambda = \{5, -3\}$, taking advantage of the upper triangular structure. The system is clearly unstable if not controlled, since one of its eigenvalues is in the right half plane.

Designing a controller to place the poles at $\lambda_d = \{-2, -3\}$ requires the unstable eigenvalue to be controllable, since one of the eigenvalues is already in the desired location. As it happens, our system is fully controllable, so this requirement is met. This can be simply checked by constructing the Controllability Matrix $\mathcal{C} = \begin{bmatrix} B & AB & \dots & A^{n-1}B \end{bmatrix} = [B \;\; AB]$ and checking its rank, which should be equal to the order of the system for controllability to hold.

$$\mathcal{C} = [B \;\; AB] = \begin{bmatrix} 0 & 2.5 \\ 1 & -3 \end{bmatrix}, \qquad \operatorname{rank}(\mathcal{C}) = 2.$$

This is clearly the case, and therefore the poles can be reassigned freely.

2.2 Feedback Control via Pole Placement

From the information above, a linear feedback controller of the from $u = -\bar{K}x$ can be constructed to place the poles of the closed loop in the desired locations. Using the `Matlab` command

`place(A,B,`λ_d`)`, the following controller is obtained.

$$\bar{K} = [22.4 \;\; 7]\,, \qquad A_{\bar{K}} = A - B\bar{K} = \begin{bmatrix} 5 & 2.5 \\ -22.4 & -10 \end{bmatrix} \tag{2}$$

2.3 Sampling Interval Dependent Discretization

The system is easily discretized with recourse to a zero order hold approach. The reasoning and resulting expressions are laid out in [3]. Taking those expressions, it's possible to compute the discrete time versions of matrices A $(F(h))$ and B $(M(h))$ as functions of the sampling interval h.

Before proceeding, it's useful to briefly present the eigendecompostion of matrix $A = S\Lambda S^{-1}$, where Λ contains the eigenvalues of A in its diagonal, and the columns of S the corresponding eigenvectors.

$$S = \begin{bmatrix} 1 & -5/16 \\ 0 & 1 \end{bmatrix}, \qquad \Lambda = \begin{bmatrix} 5 & 0 \\ 0 & -3 \end{bmatrix}, \qquad S^{-1} = \begin{bmatrix} 1 & 5/16 \\ 0 & 1 \end{bmatrix} \tag{3}$$

The discrete time matrices can now be easily calculated as follows.

$$F(h) = \exp(Ah) = S \begin{bmatrix} e^{5h} & 0 \\ 0 & e^{-3h} \end{bmatrix} S^{-1} = \begin{bmatrix} e^{5h} & \frac{5}{16}\left(e^{5h} - e^{-3h}\right) \\ 0 & e^{-3h} \end{bmatrix} \tag{4}$$

$$M(h) = \int_0^h \exp(As)Bds = (\exp(Ah) - I)\,A^{-1}B = \begin{bmatrix} \frac{e^{5h}}{16} + \frac{5e^{-3h}}{48} - \frac{1}{6} \\ \frac{1}{3} - \frac{e^{-3h}}{3} \end{bmatrix} \tag{5}$$

Using `Matlab`, it's possible to the loop with these matrices and the previous controller $\bar{K}$, leading to the following closed loop system with poles $\lambda = \{e^{-3h}, \frac{7-2e^{5h}}{5}\}$.

$$F_{\bar{K}}(h) = \begin{bmatrix} \frac{56}{15} - \frac{2e^{5h}}{5} - \frac{7e^{-3h}}{3} & \frac{7}{6} - \frac{e^{5h}}{8} - \frac{25e^{-3h}}{24} \\ \frac{112}{15}(e^{-3h} - 1) & \frac{1}{3}\left(10e^{-3h} - 7\right) \end{bmatrix} \tag{6}$$

2.4 Stability Analysis

The previous section expressions were obtained for the poles of the closed loop system depending on the sampling time. For LTI systems in discrete time, the system will be stable as long as $max(|\lambda_i|) < 1$. This will be the case for $\lambda_1 = e^{-3h}$ as long as $h > 0$, which will obviously always be the case. The second eigenvalue violates the condition imposed on the spectrum of the closed loop when $|7 - 2e^{5h}| = 5 \Leftrightarrow h = \ln(6)/5 \approx 0.358$. Plots of the evolution of eigenvalues with the sampling time are shown in Figure 1.

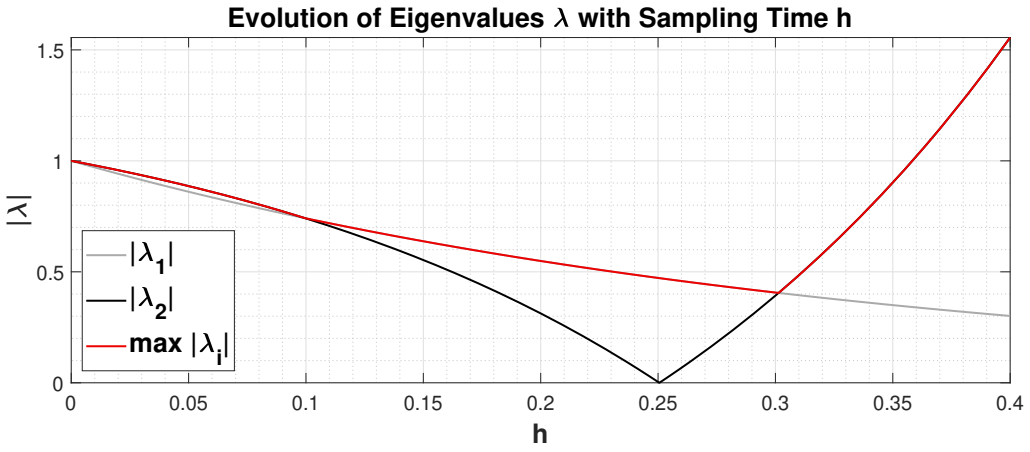

Figure 1: Eigenvalues of the closed loop discrete system as a function of the sampling time.

3 Analysis of the System Under Constant Small Time Delays

In this section, a constant small time delay is assumed, $\tau \in [0, h)$, the consequences arising from this change are studied and a controller is proposed to improve robustness to time delays, for a fixed sampling time.

3.1 Exact discrete-time model for the system with delays

Given the assumption on small time delays, the calculation of the evolution of the state from $t = s_k$ to $t = s_{k+1}$ will be dependent on both the current input u_k and the previous input u_{k-1}. This new compound input $v(t)$ can be visualised in Figure 2 and is defined as:

$$v(t) = \begin{cases} u_{k-1} \text{ for } t \in [s_k, s_k + \tau) \\ u_k \text{ for } t \in [s_k + \tau, s_{k+1}) \end{cases} \tag{7}$$

The evolution of the state equations is captured in the following set of equations:

$$x_{k+1} = e^{Ah} x_k + \int_{s_k}^{s_{k+1}} e^{A(h-t)} B v(t) dt \tag{8}$$

$$x_{k+1} = e^{Ah} x_k + \int_{s_k}^{s_k+\tau} e^{A(h-t)} B dt \; u_{k-1} + \int_{s_k+\tau}^{s_{k+1}} e^{A(h-t)} B dt \; u_k \tag{9}$$

For a constant sampling period, $s_{k+1} - s_k = h$, the state evolution is then given by

$$x_{k+1} = e^{Ah} x_k + \int_{0}^{\tau} e^{A(h-t)} B dt \; u_{k-1} + \int_{\tau}^{h} e^{A(h-t)} B dt \; u_k \tag{10}$$

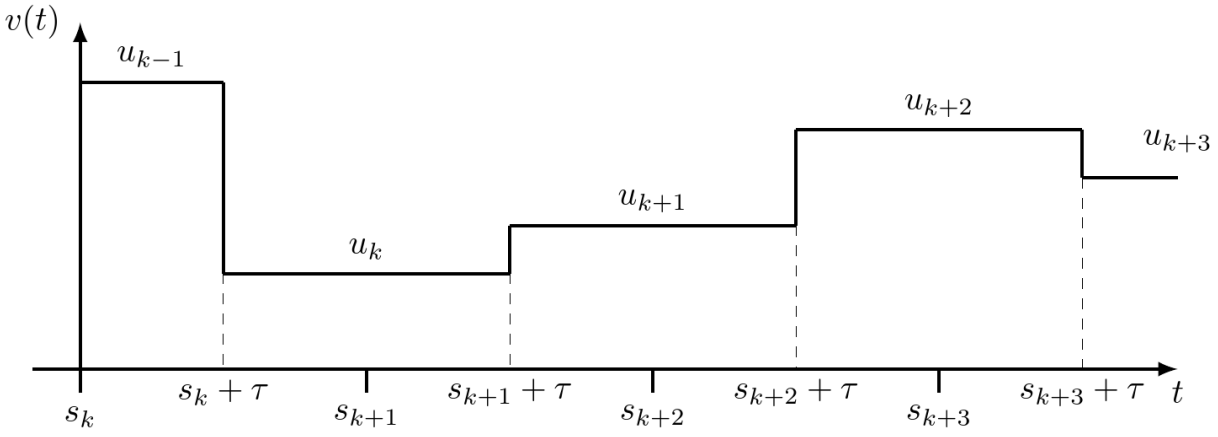

Figure 2: Evolution of input $v(t)$ when a small delay is introduced to the system

Note that u_k and u_{k-1} are constant over the respective integration intervals, and as such can be taken out of the integral. Changing the integration variable to $s = h - t$:

$$x_{k+1} = \underbrace{e^{Ah}}_{F(h)} x_k + \underbrace{\int_{h-\tau}^{h} e^{As} B ds}_{M_1(h,\tau)} u_{k-1} + \underbrace{\int_{0}^{h-\tau} e^{As} B ds}_{M_0(h,\tau)} u_k \tag{11}$$

$$x_{k+1} = F(h)x_k + M_0(h,\tau)u_k + M_1(h,\tau)u_{k-1} \tag{12}$$

The calculation of matrices $M_0(h,\tau)$ and $M_1(h,\tau)$ is straightforward and yields the following expressions.

$$M_0(h,\tau) = \begin{pmatrix} \frac{5\,\mathrm{e}^{3\,\tau-3\,h}}{48} + \frac{\mathrm{e}^{5\,h-5\,\tau}}{16} - \frac{1}{6} \\ \frac{1}{3} - \frac{\mathrm{e}^{3\,\tau-3\,h}}{3} \end{pmatrix} \qquad M_1(h,\tau) = \begin{pmatrix} \frac{5\,\mathrm{e}^{-3\,h}}{48} + \frac{\mathrm{e}^{5\,h}}{16} - \frac{5\,\mathrm{e}^{-3\,h}\,\mathrm{e}^{3\,\tau}}{48} - \frac{\mathrm{e}^{5\,h}\,\mathrm{e}^{-5\,\tau}}{16} \\ \frac{\mathrm{e}^{-3\,h}\left(\mathrm{e}^{3\,\tau}-1\right)}{3} \end{pmatrix} \tag{13}$$

The inclusion of past inputs in the current control input gives rise to the need to store these values in an extended system matrix. This matrix considers $x_e = [x \;\; u_{k-1}]$ and is given by:

$$\begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix} = \underbrace{\begin{bmatrix} F(h) & M_1(h,\tau) \\ 0 & 0 \end{bmatrix}}_{F_e(h,\tau)} \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix} + \underbrace{\begin{bmatrix} M_0(h,\tau) \\ I \end{bmatrix}}_{M_e(h,\tau)} u_k \tag{14}$$

Considering initially the completely static state feedback gain used so far, the input $u_k = -Kx_e \Leftrightarrow u_k = -\begin{bmatrix}\bar{K} & 0\end{bmatrix} x_e$, the closed loop system is given by:

$$x_e(k+1) = F_{e\bar{K}}(h,\tau)x_e(k) = \left(F_e(h,\tau) - M_e(h,\tau)K\right) x_e(k) = \begin{bmatrix} F_e(h,\tau) - M_0(h,\tau)\bar{K} & M_1(h,\tau) \\ -\bar{K} & 0 \end{bmatrix} x_e(k) \tag{15}$$

3.2 Combinations of sampling intervals and system delays

Using the symbolic calculus capabilities of `Matlab`, it's again possible to obtain expressions for the eigenvalues of $F_{e\bar{K}}(h,\tau)$ as a function of the time sample time h and the delay τ (closed form expressions become too long to reproduce here). Figure 3 is a plot of combinations of pairs h,τ. Red points correspond to matrices whose closed loop is stable. Values whose combination leads to eigenvalues larger than one are not plotted.

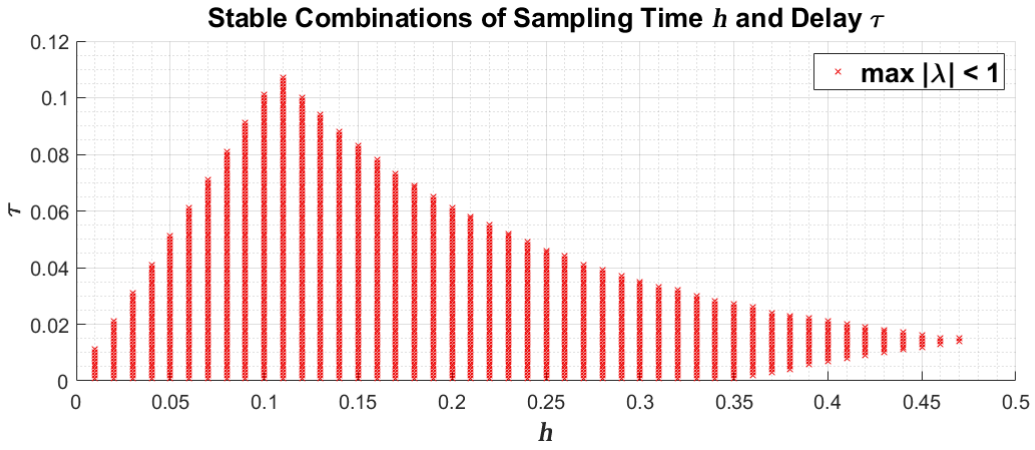

Figure 3: Pairs of sampling time and delay resulting in stable closed loop dynamics.

As is visible, for values of h up to 0.1 s, the system remains stable for any delay $\tau \in [0, h)$. For higher values of sampling time, however, stability isn't guaranteed for large values $\tau \approx h$. As h increases, the range of delays for which the closed loop remains stable reduces. After $h = 0.358$ s, stability is no longer guaranteed for small delays. This is to be expected given the analysis made in subsection 2.4, where the system becomes unstable for sampling times larger than this (if no delay is in place). Interestingly, adding some delay can actually keep the system stable for larger values of h. Sampling times larger than $h = 0.47$ s produce no stable matrices, independent of delay.

3.3 Controller design for robustness against delays

The first step in this section is to fix a delay for which a controller can be designed. Picking a value $h \in (0, 0.1]$ isn't useful, since the controller is already robust to all delays $\tau \in [0, h)$. Any value higher than this is a fair candidate for which the controller can be redesigned. The choice is made to pick $h = 0.2$s. For this sampling period, the system is stable with delays up to $\tau = 0.06$s.

Picking a controller structure here is important in improving the robustness to delays. As a first attempt, a static feedback LQR approach was attempted. Matrices $F(h,\tau)$ and $M(h,\tau)$ were obtained for the case with zero delay. The controller was then synthesised using the `Matlab` com-

mand `dlqr(Fu,G,Q,R)`. Tuning the weight matrices to $Q = 0.1I_2$ and $R = 10$ results in a controller $K = [K_{LQR}\ 0]$. Improvements are very shy over the initial controller $K = [\bar{K}\ 0]$, and were calculated to be 2.6% as seen in Figure 4.

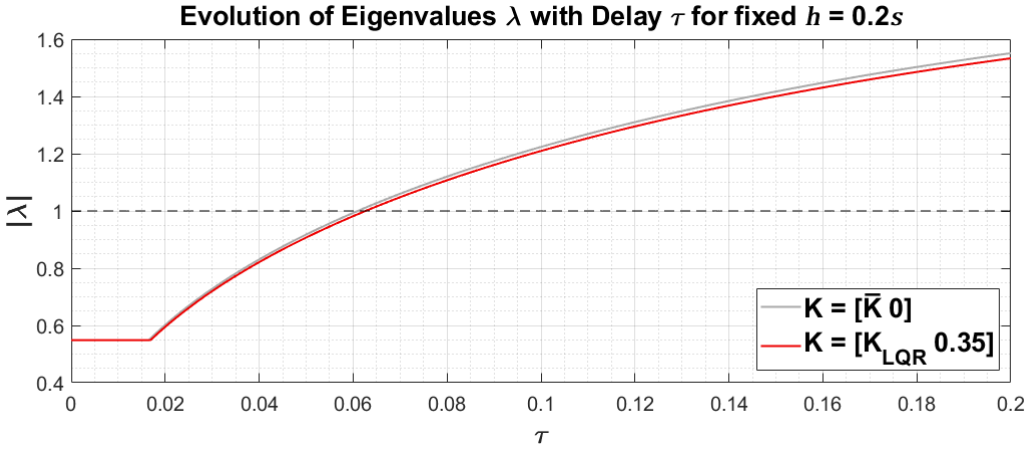

Figure 4: Evolution of closed loop eigenvalues with τ of both static controllers.

Improving the controller further requires a slight change in the input structure. Taking advantage of the entry in K currently set to 0, a dynamic controller that takes into account also the past inputs can be synthesised. The structure for the input becomes $u_k = K_{LQR}x_k + K_u u_{k-1} = [K_{LQR}\ \ K_u]x_{e_k}$. The parameter K_u was tuned and the best performance in terms of robustness to delays was found for $K_u = 0.35$. Figure 5 shows the results obtained with this controller in terms of stability. Stability is now guaranteed for up to $\tau = 0.149\text{s}$, an improvement of 143.9% over the original controller.

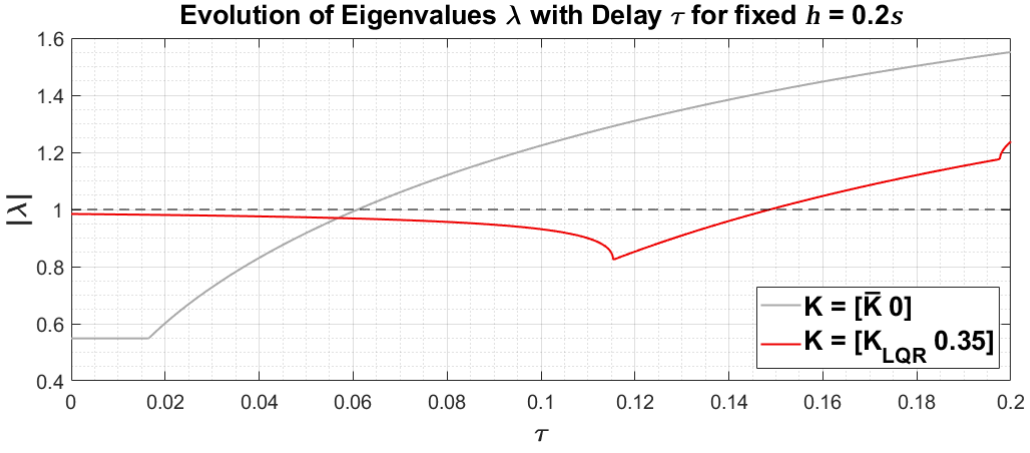

Figure 5: Evolution of closed loop eigenvalues with τ of the original static controller and the dynamic controller.

A quick note can be made about the performance of such a system. Despite this controller resulting in the best increase in robustness to delays, ti's visible that the eigenvalues of the closed

loop remain close to $|\lambda| = 1$ for most of these same delays. This means performance in this system will be much slower than the original controller for small delay values, despite being more robust overall.

With this in mind, and looking to better understand how the system responds to the dynamic component of the controller, a study was done of the stability of the system depending on the value of K_u. Results are found in Figure 6. Better robustness to delays than the static controller is obtained for any $K_u > 0$, and this robustness keeps increasing for larger values of K_u up to $K_u \approx 0.35$. Conversely, the maximum eigenvalues increase for increasing K_u, and the system becomes completely unstable $K_u > 0.4$. Values for $K_u \approx 0.15$ provide a nice balance between robustness to delays (improvement of 56% over the original controller) and performance of the system (since the eigenvalues aren't close to $|\lambda| = 1$ for most of the range where the closed loop is stable) and should be considered if the performance metric isn't restricted to just robustness against delays.

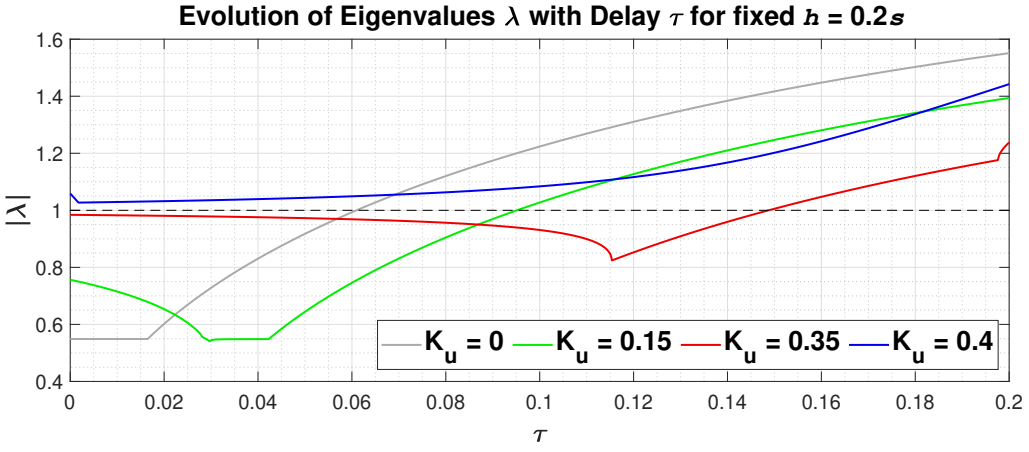

Figure 6: Evolution of closed loop eigenvalues with τ of the original static controller and multiple dynamic controllers.

4 Analysis of the System Under Constant Longer Time Delays

Having tackled small time delays in the previous section, this section focuses on longer delay, $\tau \in [h, 2h)$. Once again, a study is done on the consequences of this delay and a controller is proposed to improve robustness to time delays, for a fixed sampling time $\tau = h$.

4.1 Exact discrete-time model for the system with delays

To find the exact discrete time model, the procedure is essentially the same as in the last section. The first step is to obtain the control input that acts on the system. Assuming a fixed time

interval h, the input $v(t)$ that acts on the system in the time interval $[s_k, s_{k+1}]$ is comprised of the inputs u_{k-1} and u_{k-2}. This is due to the higher delay, and requires that two past inputs are stored in order to simulate this system properly.

$$v(t) = \begin{cases} u_{k-2} \text{ for } t \in [s_k, s_{k-1} + \tau) \\ u_{k-1} \text{ for } t \in [s_{k-1} + \tau, s_{k+1}) \end{cases} \tag{16}$$

The evolution of the state equations is captured in the following set of equations:

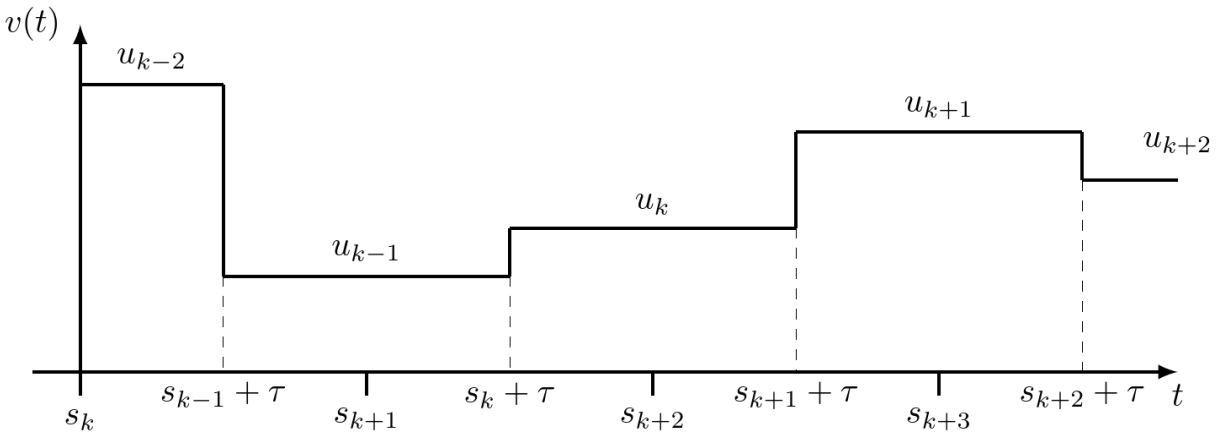

Figure 7: Evolution of input $v(t)$ when a larger delay is introduced to the system

$$x_{k+1} = e^{Ah} x_k + \int_{s_k}^{s_{k+1}} e^{A(h-t)} B v(t) dt \tag{17}$$

$$x_{k+1} = e^{Ah} x_k + \int_{s_k}^{s_{k-1}+\tau} e^{A(h-t)} B dt \; u_{k-2} + \int_{s_{k-1}+\tau}^{s_{k+1}} e^{A(h-t)} B dt \; u_{k-1} \tag{18}$$

For a constant sampling period, $s_{k+1} - s_k = h$, the state evolution is then given by

$$x_{k+1} = e^{Ah} x_k + \int_{0}^{-h+\tau} e^{A(h-t)} B dt \; u_{k-2} + \int_{-h+\tau}^{h} e^{A(h-t)} B dt \; u_{k-1} \tag{19}$$

Changing the integration variable to $s = h - t$:

$$x_{k+1} = e^{Ah} x_k + \int_{2h-\tau}^{h} e^{As} B ds u_{k-2} + \int_{0}^{2h-\tau} e^{As} B ds u_{k-1} \tag{20}$$

$$\tag{21}$$

The connection may not yet be apparent to the matrices derived in the last section. For that, consider that $\tau \in [0, h)$ can be expressed as $\tau = h + \tau'$, with $\tau' \in [0, h)$. Replacing this in the above integrals:

$$x_{k+1} = \underbrace{e^{Ah}}_{F(h)} x_k + \underbrace{\int_{h-\tau'}^{h} e^{As} B ds}_{M_1(h,\tau')} u_{k-2} + \underbrace{\int_{0}^{h-\tau'} e^{As} B ds}_{M_1(h,\tau')} u_{k-1} \tag{22}$$

$$x_{k+1} = F(h) x_k + M_0(h, \tau') u_{k-1} + M_1(h, \tau') u_{k-2} \tag{23}$$

The matrices obtained here are the exact same as the ones used for the small delay case in eq. (13), when considering τ', except the inputs are shifted by one due to the integer multiple of the sampling time in the delay τ. This is easily generalizable, and any delay $\tau \in [nh, (n+1)h)$ can be viewed as $\tau = nh + \tau'$ and expressed as $x_{k+1} = F(h)x_k + M_0(h, \tau')u_{k-n} + M_1(h, \tau')u_{k-n-1}$.

Including past inputs in the current control once again requires storing older inputs. This time, the need to store two past inputs gives rise to the extended state space $x_e = [x \;\; u_{k-1} \;\; u_{k-2}]$ and respective matrices:

$$\begin{bmatrix} x_{k+1} \\ u_k \\ u_{k-1} \end{bmatrix} = \underbrace{\begin{bmatrix} F(h) & M_0(h,\tau') & M_1(h,\tau') \\ 0 & 0 & 0 \\ 0 & I & 0 \end{bmatrix}}_{F_e(h,\tau')} \begin{bmatrix} x_k \\ u_{k-1} \\ u_{k-2} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix}}_{M_e} u_k \tag{24}$$

Considering initially the completely static state feedback gain used so far, the input $u_k = -Kx_e \Leftrightarrow u_k = -\begin{bmatrix} \bar{K} & 0 \end{bmatrix} x_e$, the closed loop system is given by:

$$x_e(k+1) = F_{e\bar{K}}(h,\tau')x_e(k) = \left(F_e(h,\tau') - M_e K\right) x_e(k) = \begin{bmatrix} F(h) & M_0(h,\tau') & M_1(h,\tau') \\ -\bar{K} & 0 & 0 \\ 0 & I & 0 \end{bmatrix} x_e(k) \tag{25}$$

4.2 Combinations of sampling intervals and system delays

Using the same procedure described in the last section, Figure 8 can be obtained. It is a plot of combinations of pairs h, τ. Red points are the same as in Figure 3 correspond to matrices whose closed loop is stable, for values of delay $\tau \in [h, 2h)$. Black points correspond to matrices whose closed loop is stable, for values of delay $\tau \in [h, 2h)$. Values whose combination leads to eigenvalues larger than one are not plotted.

The first, rather obvious, conclusion to take is that if a certain sampling time doesn't produce stable closed loop dynamics for a delay in the range $\tau \in [0, h)$, then combinations $\tau \in [h, 2h)$ will not be stable either. For this reason, the range of sampling times that are capable of producing stable closed loops for larger delays is contained to the interval $h \in (0, 0.1]$. Within this range, $h \in (0, 0.06]$ still result in stable dynamics for the full range of delays. The range $h \in (0.06, 0.1]$ see the range of stable delays shrink constantly, to the point where the maximum value of delay admissible for $h = 0.1\ s$ is $\tau = 1.15h$.

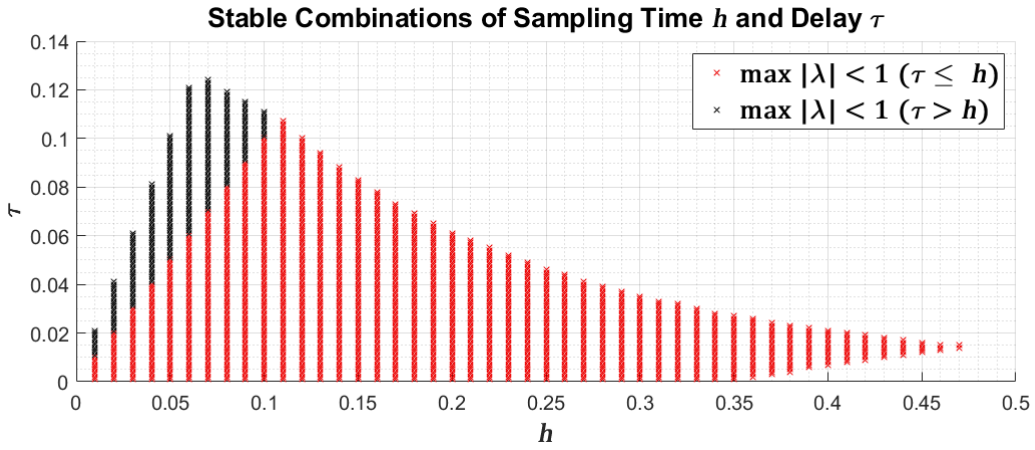

Figure 8: Pairs of sampling time and delay resulting in stable closed loop dynamics.

4.3 Controller design for robustness against delays

Designing a controller for a fixed time delay, in this case $\tau = h$, can be done in much the same way as attempted in the last section. Starting from the nominal delay case, an LQR approach is taken to place the first two poles of the controller, and the (now) two last entries, corresponding to the dynamical portion of the controller, can the be tuned to achieve the best performance. The choice of sampling time was $h = 0.1s$, the upper limit of sampling times that guarantee stability for a delay $\tau = h$, as seen in the previous subsection. Applying the procedure described above, with the LQR tuning matrices $Q = 0.1I$ and $R = 10$, a controller $K = [25.7087 \;\; 8.0337 \;\; 0.3200 \;\; 0.2500]$ is obtained. Using this controller, the system is able to maintain stability up to the maximum delay considered, $\tau = 2h$. As a benchmark, the purely static controller found in Question 1 is used, $\bar{K} = [22.4 \;\; 7.0 \;\; 0.0 \;\; 0.0]$. This static controller can keep stability up to $\tau = 0.1106s$. Given that the robust controller is able to achieve $\tau = 0.2s$, the improvement is of 80.83%. Corresponding plots are found in Figure

Once again, the question of the trade-off between performance of the system and robustness to delays can be posed. Though stable in the full range of delays, the maximum eigenvalue remains quite large throughout. To try and improve this situation, an LMI based approach was taken in this section. A more thorough explanation will be given in the next section about exactly how this process works. The process involved discretizing the range of possible delays $\tau \in [h, 2h)$ in steps of size $0.001s$. For each of these delays, the corresponding matrices $F_e(h, \tau)$ and $M_e(h, \tau)$ are found and stored. Then the set of LMIs in equation (26) can be solved for a common controller K. The resulting controller is $K = [37.4095 \;\; 11.7059 \;\; 0.6918 \;\; 0.4706]$. Figure 10 shows how this process maintains stability in the whole range of considered delays, while also lowering the maximum eigenvalues of

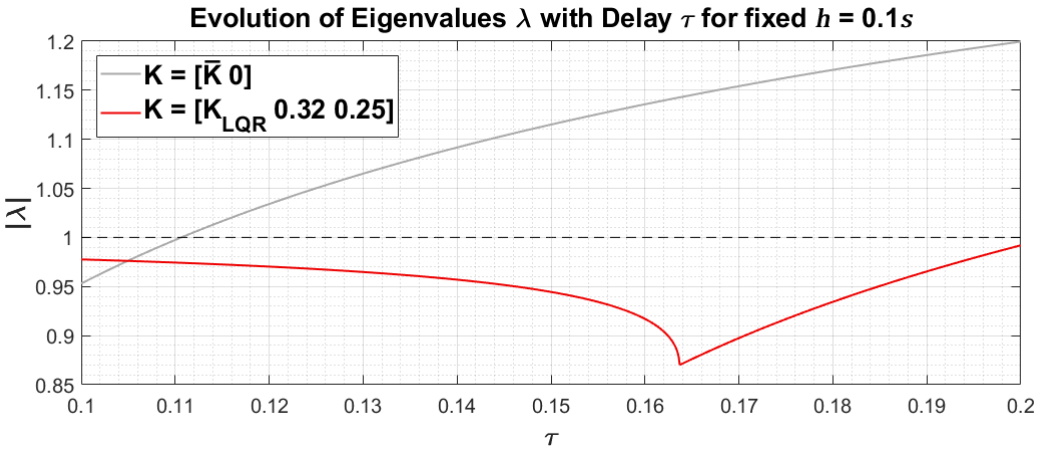

Figure 9: Evolution of closed loop eigenvalues with τ of the static and the dynamic controller.

the system across the whole range of delays, resulting in an overall better controller.

$$(F_e(h,\tau) - M_e(h,\tau)K)^\intercal P(F_e(h,\tau) - M_e(h,\tau)K) - P \prec 0 \tag{26}$$

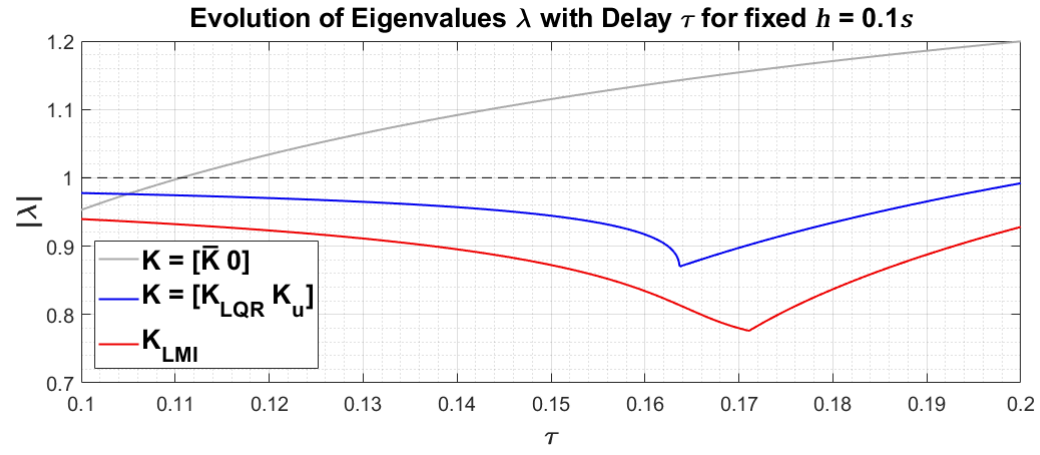

Figure 10: Evolution of closed loop eigenvalues with τ of the static and the dynamic controllers.

5 Time-varying delays

This section is concerned with the study of stability and controller design for a finite set of of time varying delays. Linear Matrix Inequalities (LMIs) are used throughout this section to both guarantee stability and design the controller for a set of delays $\mathcal{T} = \{0.2h, 0.5h, h, 1.5h\}$.

From a practical standpoint, all LMIs were solved using CVX (specifically, the SDPT3 solver), a package for specifying and solving convex programs [1], [2].

5.1 Stability Guarantees via LMIs

As indicated in the course lectures [3], guaranteeing stability of the closed loop under time varying delays isn't as trivial as checking if the the closed loop for each time delay is individually stable. For time-varying systems, Lyapunov-based techniques must be used instead.

Summarily, exponential stability of the system is guaranteed if one can find a Lyapunov function for the system such that this function is bounded between two exponential functions and the it's decrease is can also be bounded by a function of the same type. That is:

$$\alpha_1\|x\|^p \leq V(x_k) \leq \alpha_2\|x\|^p, \quad \forall k \geq k_0, \forall x \in \mathbb{R}^n \quad \alpha_1, \alpha_2 > 0, \quad p > 1 \tag{27}$$

$$V(x_{k+1}) - V(x_k) \leq -\alpha_3\|x\|^p, \quad \forall k \geq k_0, \forall x \in \mathbb{R}^n, \quad \alpha_3 > 0, \quad p > 1 \tag{28}$$

For autonomous systems, where $x_{k+1} = Ax_k$, choosing $V(x) = x^\intercal Px$ quadratic, and $P \succ 0$ guarantees the fulfilment of both conditions, as long as valid P can be found. The first condition is trivial:

$$\lambda_{\min}(P)\|x\|^2 \leq V(x) \leq \lambda_{\max}(P)\|x\|^2 \tag{29}$$

The second condition reduces to finding a matrix $Q \succ 0$ such that $A^\intercal PA - P \preceq -Q$.

$$\begin{aligned} V(x_{k+1}) - V(x_k) \leq -\alpha_3\|x\|^p \Leftrightarrow \\ V(x_{k+1}) - V(x_k) \leq -x_k^\intercal Q x_k \leq -\lambda_{\min}(Q)\|x\|^2 \Leftrightarrow \\ x_{k+1}^\intercal P x_{k+1} - x_k^\intercal P x_k \leq -x_k^\intercal Q x_k \Leftrightarrow \\ x_k^\intercal A^\intercal P A x_k - x_k^\intercal P x_k \leq -x_k^\intercal Q x_k \Leftrightarrow \\ x_k^\intercal (A^\intercal P A - P) x_k \leq -x_k^\intercal Q x_k \Leftrightarrow \\ A^\intercal P A - P \preceq -Q \end{aligned} \tag{30}$$

In the case of time varying systems, this same approach can be used, as long as a common Lyapunov function can be found, such that these properties hold for all time varying systems $x_{k+1} = A_i x_k$:

$$A_i^\intercal P A_i - P \preceq -Q \tag{31}$$

For the type of systems analysed in this report, where $x_{e_{k+1}} = F_e(h,\tau)x_{e_k} + M_e(h,\tau)u_k$, with state feedback, $u_k = -Kx_{e_k}$, the close loop is described as $x_{e_{k+1}} = (F_e(h,\tau) - M_e(h,\tau)K)x_{e_k}$. This is ultimately just an autonomous system, and therefore, exponential stability is guaranteed if

$$(F_e(h,\tau) - M_e(h,\tau)K)^\intercal P(F_e(h,\tau) - M_e(h,\tau)K) - P \preceq -Q \tag{32}$$

Note that, from equation (28) a value $\alpha_3 > 0$ need to be found in order to have a valid bound on the decrease of the Lyapunov function. In 30, the assumption is made that $\alpha_3 > 0 = \lambda_{\min}(Q)$, where $\lambda_{\min}(Q)$ the smallest eigenvalue of the positive definite matrix Q. This means that, if it's possible to find a solution for Q, there's an infinite range of matrices Q that also fulfil equation (30) (for $Q \in \mathbb{R}^{n\times n}$, with $n \geq 2$), since all other eigenvalues of Q can be freely assigned as long as they meet $\lambda_i(Q) \geq \lambda_{\min}(Q) > 0$. In particular, this means that if a positive definite matrix Q can be found, a valid structure for Q is $Q = \epsilon I$, where $\lambda_i(Q) = \lambda_{\min}(Q) = \epsilon > 0$. This reduces the complexity of the problem from having to find a full $n \times n$ matrix to having to find just a single parameter $\epsilon > 0$.

$$(F_e(h,\tau) - M_e(h,\tau)K)^\intercal P(F_e(h,\tau) - M_e(h,\tau)K) - P \preceq -\epsilon I \tag{33}$$

A further simplification can be made to this problem. It's easier to see that eq. (33) is linear in the parameter P, and no terms in the equation are independent of P. This means the problem is homogeneous in P. This is a useful property, since it means the solution to the problem will be insensitive to scalar transformations. Hence $F(P) \preceq -\epsilon I \Leftrightarrow F(P) \preceq -I$ and the final version of the problem to solve is:

$$(F_e(h,\tau) - M_e(h,\tau)K)^\intercal P(F_e(h,\tau) - M_e(h,\tau)K) - P \preceq -I \tag{34}$$

For a given K, and the matrices $F_e(h,\tau)$ and $M_e(h,\tau)$ varying within a finite set of matrices according to the various delays, stability is guaranteed if a matrix $P \succeq 0$ can be found such that $V(x_k) = x_k^\intercal P x_k$ is a common Lyapunov function to all combinations of $F_e(h,\tau)$ and $M_e(h,\tau)$. For the problem at hand, this means solving four LMIs for matrix P.

$$\begin{cases} (F_e(h,0.2h) - M_e(h,0.2h)K)^\intercal P(F_e(h,0.2h) - M_e(h,0.2h)K) - P \preceq -I \\ (F_e(h,0.5h) - M_e(h,0.5h)K)^\intercal P(F_e(h,0.5h) - M_e(h,0.5h)K) - P \preceq -I \\ (F_e(h,1.0h) - M_e(h,1.0h)K)^\intercal P(F_e(h,1.0h) - M_e(h,1.0h)K) - P \preceq -I \\ (F_e(h,1.5h) - M_e(h,1.5h)K)^\intercal P(F_e(h,1.5h) - M_e(h,1.5h)K) - P \preceq -I \end{cases} \tag{35}$$

5.1.1 Common Structure for Delays

Note that the set of LMIs above will look for a matrix $P \in \mathbb{R}^{n\times n}$, the same size as the matrix $F_e(h,\tau)$. This is a problem, since for delays $\tau \in [0,h)$ a matrix $F_e(h,\tau) \in \mathbb{R}^{3\times 3}$ was defined, and for delays $\tau \in [h,2h)$, the matrix needs to be $F_e(h,\tau) \in \mathbb{R}^{4\times 4}$. Solving this requires a common size set of matrices for both sets of delays. This is done by rewriting the expressions for $\tau \in [0,h)$ in the

framework used for $\tau \in [h, 2h)$. The resulting set of matrices is:

$$\begin{bmatrix} x_{k+1} \\ u_k \\ u_{k-1} \end{bmatrix} = \underbrace{\begin{bmatrix} F(h) & M_1(h,\tau') & 0 \\ 0 & 0 & 0 \\ 0 & I & 0 \end{bmatrix}}_{F_e(h,\tau')} \begin{bmatrix} x_k \\ u_{k-1} \\ u_{k-2} \end{bmatrix} + \underbrace{\begin{bmatrix} M_0(h,\tau') \\ I \\ 0 \end{bmatrix}}_{M_e(h,\tau')} u_k \tag{36}$$

For the standard controller, $K = [\bar{K} \;\; 0 \;\; 0]$, the corresponding closed loop then becomes:

$$x_e(k+1) = \big(F_e(h,\tau') - M_e(h,\tau')K\big)\, x_e(k) = \begin{bmatrix} F(h) - M_0(h,\tau')\bar{K} & M_1(h,\tau') & 0 \\ -\bar{K} & 0 & 0 \\ 0 & I & 0 \end{bmatrix} x_e(k) \tag{37}$$

With this structure, finding a common matrix P is now feasible.

5.2 Controller design for maximization of sampling intervals

The objective in this section is to design a controller K such that the sampling intervals h are maximized. Note that the change in sampling interval will also affect the delays the system is subject to, since the set of delays is $\mathcal{T} = \{0.2h, 0.5h, h, 1.5h\}$.

Further note that simply designing a controller to guarantee stability via eigenvalue analysis for the largest delay $\tau = 1.5h$ will not guarantee stability of the system, even if the resulting controller also guarantees that the eigenvalues of the closed loops with delays $\mathcal{T} = \{0.2h, 0.5h, h\}$ also meet $\|\lambda\| < 1$. As such, this section follows an LMI based approach to the problem.

Starting from equation (34), the approach will look to optimize both K and P in the same LMI. This is problematic since the equation isn't linear in these variables and therefore the problem is nonlinear and nonconvex. Getting around this issue involves the application of some techniques from the LMI lectures [3], further clarified in Theorem 4 of [4].

The first step is to lift the problem into a higher dimension via the Schur Complement. Recognising in the structure $A - BD^{-1}C$ in equation (34), the problem is re-written as:

$$\begin{bmatrix} -P & (F_e(h,\tau') - M_e(h,\tau')K)^\mathsf{T} \\ (F_e(h,\tau') - M_e(h,\tau')K) & -P^{-1} \end{bmatrix} \preceq -I \tag{38}$$

The dependence on both P and P^{-1} is still a problem. A congruent transformation can then be applied by pre- and post-multiplying the invertible symmetric matrix $T = \mathrm{diag}\{P^{-1},\ I\}$. Sylverster's

law of inertia states that if $P \succeq 0$ then $T^\intercal P T \succeq 0$, so the resulting solution will still be valid.

$$\begin{bmatrix} -P^{-1} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} -P & (F_e(h,\tau') - M_e(h,\tau')K)^\intercal \\ (F_e(h,\tau') - M_e(h,\tau')K) & -P^{-1} \end{bmatrix} \begin{bmatrix} -P^{-1} & 0 \\ 0 & I \end{bmatrix} = \\ \begin{bmatrix} -P^{-1} & P^{-1}\left(F_e(h,\tau') - M_e(h,\tau')K\right)^\intercal \\ \left(F_e(h,\tau') - M_e(h,\tau')K\right)P^{-1} & -P^{-1} \end{bmatrix} = \\ \begin{bmatrix} -P^{-1} & P^{-1}F_e(h,\tau')^\intercal - P^{-1}K^\intercal M_e(h,\tau')^\intercal \\ F_e(h,\tau')P^{-1} - M_e(h,\tau')KP^{-1} & -P^{-1} \end{bmatrix} \tag{39}$$

Denoting $M = P^{-1}$ and $Y = KP^{-1}$, a linear relation in M and Y is finally obtained:

$$\begin{bmatrix} -M & MF_e(h,\tau')^\intercal - Y^\intercal M_e(h,\tau')^\intercal \\ F_e(h,\tau')M - M_e(h,\tau')Y & -M \end{bmatrix} \preceq -I \tag{40}$$

Solving the problem for matrices M and Y, matrices P and K can be recovered:

$$P = M^{-1}, \qquad K = YP \tag{41}$$

The same procedure can be applied to the four LMIs in eq. (35). As such, a systematic way has been found to solve for both the Lyapunov function and the controller, given a sampling interval h.

The procedure now is quite simple, as one can simply keep increasing the sampling time h until a solution to the LMIs in eq. (35) can no longer be found. The final solution is the final sampling time h which admits a valid solution. The best controller, in turn, is the one found for the largest admissible value of $h = h_{\max}$. This controller is then checked a posteriori for stability in all other values of $h < h_{\max}$.

This is not the most correct way of going about this procedure. When trying to establish stability for an interval $h = (0, h_{\max}]$, the resulting set of LMIs would in essence, be infinite. As such, the most precise way of going about this would be to use a polytopic over-approximation of the system and prove stability for that finite set. This was not covered during the first lectures of the course, and therefore will not be implemented here.

The result from this procedure is a controller which is able to stabilize the system up to $h_{\max} = 0.132s$, $K = [27.15,\ \ 8.49,\ \ 0.49,\ \ 0.20]$.

As a benchmark, the original static controller $\bar{K}$ was able to maintain stability up to $h_{\max} = 0.077s$, and as such, the controller found represents an improvement of 71% over the initial controller.

5.3 Periodic Time Delays

The special case where the time delays are periodic can, at first, be handled in the exact same way as on did for the case of uncertain delays. In theory, however, this may not be the best way to go about it. Since the system behaves periodically, for a given controller, the behaviour will be as follows:

$$\begin{aligned} x_e(k+1) &= F_1^{cl} x_e(k) = \left(F_e(h, 0.2h) - M_e(h, 0.2h)K\right) x_e(k) \\ x_e(k+2) &= F_2^{cl} x_e(k+1) = \left(F_e(h, h) - M_e(h, h)K\right) x_e(k+1) \\ x_e(k+3) &= F_3^{cl} x_e(k+2) = \left(F_e(h, 0.5h) - M_e(h, 0.5h)K\right) x_e(k+2) \end{aligned} \tag{42}$$

The behaviour is cyclic form here on. As such, one can simply take a look at the behaviour of the system from $x_e(k)$ to $x_e(k+3)$. given controller, the behaviour will be as follows:

$$x_e(k+3) = F_3^{cl} F_2^{cl} F_1^{cl} x_e(k) \tag{43}$$

The resulting multiplication will be a single linear system whose stability can be analysed via either LMIs or, since it's a single system, via the more conventional eigenvalue based analysis. The simplified (single) set of LMIs will simply be:

$$\begin{bmatrix} -P^{-1} & P^{-1}\left(F_3^{cl} F_2^{cl} F_1^{cl}\right)^\intercal \\ \left(F_3^{cl} F_2^{cl} F_1^{cl}\right) P^{-1} & -P^{-1} \end{bmatrix} \preceq -I \tag{44}$$

or alternatively,

$$\max \operatorname{eig}(F_3^{cl} F_2^{cl} F_1^{cl}) < 1 \tag{45}$$

5.4 Controller design for periodic time delays

The approach taken here was the same as in Section 5.2. The now three LMIs were solved to find a valid matrix P for increasing values of h until some $h_{\max}$ for which the solver could no longer find a valid matrix P. The resulting controller was then used to check for stability in a finite range of points $h \in (0, h_{\max}]$ (which, again, is not the most accurate way of going about this process). The resulting controller, $K = [26.77 \quad 8.37 \quad 0.67 \quad 0.00]$, was able to guarantee stability up until $h_{\max} = 0.195s$.

Given the possibility of analysing this system via eigenvalue analysis, the controller found can be used to build the plot in Figure 11 Note that the eigenvalues stay really close to 1 for the majority of the range analysed, implying performance won't be the best throughout. Focusing only on stability, however, note that, despite the LMIs guaranteeing stability up to $h_{\max} = 0.195s$, eigenvalue analysis clearly remains stable up to $h_{\max} = 0.259s$, a lot more than initially considered.

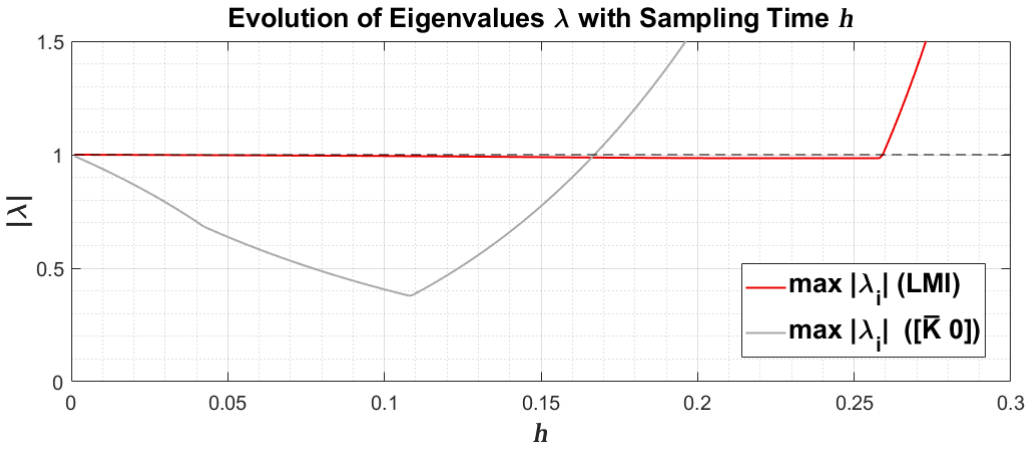

Figure 11: Evolution of closed loop eigenvalues with τ for the controller resulting from the LMI analysis.

The obvious reason for this is the fact that the method used does not take into account the periodicity of the system, but merely an uncertain delay at each time step, with no associated structure. Periodicity is included in the LMI analysis, but so are a whole range of other possibilities that could result in the system becoming unstable for smaller values of h than those achievable if only periodicity was taken into account.

As such, it's natural that the controller built for this unstructured, worst case scenario, performs better than expected in this structured case.

Further, though not taken into account here, there could be ways to design specifically for this periodic behaviour, which would allow for controller design specifically catered to this behaviour, and could result in even better performance than was reached here.

As a benchmark, the original static controller $\bar{K}$ was able to maintain stability up to $h_{\max} = 0.167s$, and as such, the controller found represents an improvement of 55% over the initial 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 [sc4100], 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.