A Controller Design for Networked Control Systems with Random Delays via the Jump Linear System Approach, which reduces the Effects of the Delay

Rainer Blind, Frank Allgöwer

***Abstract*— In this work, the controller design for a Networked Control System (NCS) with random time delays is considered. Therefore, the NCS is reformulated as Jump Linear System (JLS). Then a standard controller design method, which stabilizes the JLS in the second moment sense, is extended such that all sub-systems/sub-matrices of the JLS have some common eigenmodes. For the NCS with random time delays this translates to a reduction of the negative effects caused by the delay. We will also show, that a controller, which gives n common eigenmodes, where n is the dimension of the system, contains a predictor term.**

Moreover, we propose a new controller type which gives one common eigenmode and can be easily realized if there is delay between the controller and the actuator.

***Index Terms*— NCS, JLS, random delay, common eigenmode**

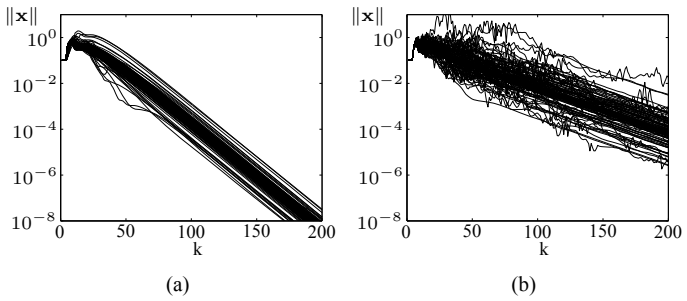


Fig. 1. Some simulations of a JLS.

I. Introduction and Motivation

Since the development of the first packet based networks, they continuously became cheaper, faster and more and more reliable. Hence, control engineers like to close their loops via such networks, which is then called a Networked Control System (NCS). Unfortunately, packet based networks have two weaknesses: Packet delay and loss. Thus, much research is necessary to overcome these weaknesses, see e.g. the special issues and sections on NCS in [1]–[3] and the references therein.

One approach to analyze and stabilize a NCS is to reformulate it as Jump Linear System (JLS), see [4]–[8]. This work also builds up on this idea. In contrast to previous works, we will not only stabilize the system but present an extension, which reduces the negative effects caused by a random delay.

In this work, a discrete-time system with a random delay $d \in \mathcal{D} := \{0, 1, \ldots, \bar{d}\}$ is considered:

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + \mathbf{b}u(k-d), \quad \mathbf{x} \in \mathbb{R}^n,\ u \in \mathbb{R}. \quad (1)$$

Using a delay dependent state feedback controller $u = \mathbf{k}_d^T\mathbf{x}$, $\mathbf{k}_d \in \mathbb{R}^n$, the closed loop system becomes:

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + \mathbf{b}\mathbf{k}_d^T\mathbf{x}(k-d). \quad (2)$$

If the delay d can be modeled by an independent and identically distributed (iid) random process or a Markov chain, then the closed loop system (2) can be analyzed by reformulating it as JLS. Therefore, the augmented state vector $\mathbf{z}(k) \in \mathbb{R}^{n\cdot(\bar{d}+1)}$ is defined as:

$$\mathbf{z}(k) := \begin{bmatrix}\mathbf{x}(k)^T & \mathbf{x}(k-1)^T & \cdots & \mathbf{x}(k-\bar{d})^T\end{bmatrix}^T. \quad (3)$$

Instead of (2) we now write:

$$\mathbf{z}(k+1) = \mathcal{A}_d\mathbf{z}(k), \quad (4)$$

where $\mathcal{A}_d$ is defined as follows:

$$\mathcal{A}_d = \begin{bmatrix} A & \leftarrow \mathbf{b}\mathbf{k}_d^T \rightarrow \\ I & 0 \ \ldots \ 0 & 0 \\ 0 & I \ \ldots \ 0 & 0 \\ \vdots & \vdots \ \ddots \ \vdots & \vdots \\ 0 & 0 \ \ldots \ I & 0 \end{bmatrix}. \quad (5)$$

Note, that the position of the matrix $\mathbf{b}\mathbf{k}_d^T$ is determined by the delay d.

Often, the controller $\mathbf{k}_d^T$ is designed such that the second moment decay rate of the JLS (4) is minimized, see [4]–[8]. In this work, we are more interested in reducing the negative effects caused by the delay than minimizing the second moment decay rate. This interest is motivated by an observation during our previous work.

In [7] and [8], we tested different algorithms, which minimize the second moment decay rate. In order to get a realistic picture of the resulting closed loop system, we simulated each system 100 times and plotted $\|\mathbf{x}\|$ on a logarithmic scale. Figure 1(a), which is taken from [8], shows such simulations. Note, that these simulations look more like the simulations of an ordinary linear system and not like the ones we expected from a jump linear system.

We discovered a similar effect for the mode-independent non-switching controller in the example of Section 5.1 of [5], after simulating the closed loop system 100 times and plot-

This work was supported by the Deutsche Forschungsgemeinschaft Priority Program 1305: Control Theory of Digitally Networked Dynamical Systems

R. Blind and F. Allgöwer are with the Institute for Systems Theory and Automatic Control, University of Stuttgart, Stuttgart, Germany. {blind, allgower}@ist.uni-stuttgart.de.

ISBN 978-963-311-369-1

TABLE I
THE EIGENMODE, WHICH IS ALMOST IDENTICAL FOR THE THREE MATRICES.

	eigenvalue	eigenvector
$\mathcal{A}_0$	0.9559	[0.5023, -0.2265, 0.0106, -0.0048, 0.5255, -0.2369, 0.0111, -0.0050, 0.5497, -0.2479, 0.0116, -0.0053]
$\mathcal{A}_1$	0.9549	[0.4997, -0.2307, 0.0110, -0.0051, 0.5233, -0.2416, 0.0115, -0.0054, 0.5480, -0.2530, 0.0121, -0.0056]
$\mathcal{A}_2$	0.9537	[0.4967, -0.2353, 0.0116, -0.0055, 0.5208, -0.2468, 0.0121, -0.0058, 0.5461, -0.2587, 0.0127, -0.0061]

ting $\|\mathbf{x}\|$ on a logarithmic scale. These simulations are shown in Figure 1(b). Although, there is some chaotic behavior, there are also some straight lines, which would correspond to a negative exponential decay, without any jumping.

First, we thought this is due to a bug in our simulations, or some strange numerical behavior. Fortunately this is not the case. We realized that all the closed loop matrices $\mathcal{A}_d$ have one eigenvalue, which is almost identical. Moreover, the corresponding eigenvectors are also almost identical. Table I shows the numerical values of this common eigenmode for the cited example in [5] and we see that they are amazingly similar.

Now, the straight lines can be explained as follows: First, suppose that there is one common eigenvalue λ with corresponding eigenvector $\mathbf{r}$, i.e $\mathcal{A}_i\mathbf{r} = \lambda\mathbf{r}, \forall i \in \mathcal{D}$. If the system is started on this eigenvector, i.e. $\mathbf{z}(k) = \mathbf{r}$ then it stays there because $\mathbf{z}(k+1) = \mathcal{A}_d\mathbf{z}(k) = \mathcal{A}_d\mathbf{r} = \lambda\mathbf{r} = \lambda\mathbf{z}(k),\ \forall d \in \mathcal{D}$. Note, that we have $\mathbf{z}(k+1) = \lambda\mathbf{z}(k)$ here, which corresponds to an ordinary linear system. For our NCS (1) this translates to a non delayed system.

Similar thoughts hold if there is more than one common eigenmode. In this case, the system can be divided in two parts: one part, which will be called the common subspace $\mathbb{A} \subset \mathbb{R}^\nu$ with the common eigenmodes and the residual part, which will be called $\mathbb{B} := \mathbb{R}^\nu \setminus \mathbb{A}$. Again, if the system is started in the common subspace, i.e. $\mathbf{z}(k) \in \mathbb{A}$, then it stays there, i.e. $\mathbf{z}(k+1) \in \mathbb{A}$. Moreover, the system behaves like an ordinary linear system. Finally, suppose the system is started somewhere else. Then there will be some jumping due to the sub-dynamics in $\mathbb{B}$. However, if the dynamics in $\mathbb{B}$ are faster than the ones in $\mathbb{A}$, then the dynamics in the common subspace $\mathbb{A}$ will dominate the entire dynamics. The jumping disappears after some time and the system behaves like an ordinary linear system.

Motivated by this observation, we will develop a design procedure, where this is an explicit design goal. We show, how to design a delay dependent controller $\mathbf{k}_d^T$ such that all $\mathcal{A}_i$ have one common eigenmode. Moreover, we will see that the proposed method is not restricted to one common eigenmode but is also able to design a controller $\mathbf{k}_d^T$ such that there are up to n common eigenmodes. Interestingly, we will see that this controller predicts future states.

The remainder of this work is organized as follows. In Sec. II, we give the main theorems to design a controller which gives common eigenmodes. Then, in Sec. III we propose a new controller type, which also gives one common eigenmode and is furthermore easy to realize if there is delay between the controller and the actuator. In Sec. IV, we show that the proposed method is correlated to the usage of a predictor. Finally, we visualize the benefits of the proposed method by an example in Sec. V and conclude the work in Sec. VI.

II. MAIN PART

In order to design a controller which gives common eigenmodes, we first give two lemmas, which simplify the eigenvalue equation of (5). Then, we derive Theorem 1 which gives a constraint for the controller $\mathbf{k}_d^T$ guaranteeing common eigenvalues. Finally, Theorem 2 shows how to get common eigenvectors. We will give the proofs in the appendix to improve readability.

The following lemma shows that the dimension of the eigenvalue equation of (5) can be reduced:

Lemma 1: The eigenvalue equation of (5) *can be written as:*

$$\begin{aligned}&\det(\lambda I_{n\cdot(\bar{d}+1)} - \mathcal{A}_i)\\ &\quad= \det(\lambda^{\bar{d}+1} I_n - \lambda^{\bar{d}} A - \lambda^{\bar{d}-i}\mathbf{b}\mathbf{k}_i^T), \quad i \in \mathcal{D}, \quad (6)\end{aligned}$$

where $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix.

Before we give the next lemma, we first have to clarify the notation, which is necessary to access the individual elements of the different controllers $\mathbf{k}_i^T$. We write $k_{i,j}$ for the j-th element of the i-th controller. Moreover, to improve readability and simplify the notation, we start to count these elements from 0.

The following lemma further simplifies the eigenvalue equation of (5):

Lemma 2: Suppose the system is given in controllable normal form. Then, we can write the right hand side of Eq. (6) *as:*

$$\begin{aligned}&\det(\lambda^{\bar{d}+1} I_n - \lambda^{\bar{d}} A - \lambda^{\bar{d}-i}\mathbf{b}\mathbf{k}_i^T)\\ &\quad= \lambda^{(n-1)\bar{d}}\Big(\lambda^{n+i} + \sum_{j=0}^{n-1}\lambda^{j+i}a_j - \sum_{j=0}^{n-1}\lambda^j k_{i,j}\Big), i \in \mathcal{D}.\end{aligned}$$

To guarantee that all $\mathcal{A}_i$ have one common eigenvalue λ, we require $\det(\lambda I - \mathcal{A}_i) = 0,\ \forall i \in \mathcal{D}$. Using Lemma 2, we get the following Theorem:

Theorem 1: Suppose the system is given in controllable normal form. Then all $\mathcal{A}_i$ have one common eigenvalue λ if the following equation holds for all $i \in \mathcal{D}$:

$$\lambda^{n+i} + \sum_{j=0}^{n-1}\lambda^{j+i}a_j - \sum_{j=0}^{n-1}\lambda^j k_{i,j} = 0. \quad (7)$$

In order to use this equality constraint within an LMI solver, we write Eq. (7) in the following vector notation:

$$\underbrace{\begin{bmatrix} k_{i,0} & \dots & k_{i,n-1} \end{bmatrix}}_{\mathbf{k}_i^T} \underbrace{\begin{bmatrix} 1 \\ \lambda \\ \vdots \\ \lambda^{n-1} \end{bmatrix}}_{\tilde{\lambda}} = \underbrace{\lambda^{n+i} + \sum_{j=0}^{n-1} \lambda^{j+i} a_j}_{c_i}, \ \forall i \in \mathcal{D}.$$

Note that some LMI solver can not handle equality constraints directly. In this case, it is possible to use $\mathbf{k}_i^T \tilde{\lambda} + \epsilon > c_i > \mathbf{k}_i^T \tilde{\lambda} - \epsilon$ as constraints. In doing so, the resulting eigenvalues will not be exactly identical but by choosing a small ϵ, they will be very close.

Note, that Eq. (7) is not restricted to one common eigenvalue. If we want to have m common eigenvalues $\lambda_1, \dots, \lambda_m$, then we can write:

$$\underbrace{\begin{bmatrix} k_{i,0} & \dots & k_{i,n-1} \end{bmatrix}}_{\mathbf{k}_i^T} \underbrace{\begin{bmatrix} 1 & \dots & 1 \\ \lambda_1 & \dots & \lambda_m \\ \vdots & \dots & \vdots \\ \lambda_1^{n-1} & \dots & \lambda_m^{n-1} \end{bmatrix}}_{\tilde{\Lambda}} = \mathbf{c}_i^T, \quad \forall i \in \mathcal{D},$$

where the elements of $\mathbf{c}_i^T \in \mathbb{R}^m$ are:
$c_{i,l} := \lambda_l^{n+i} + \sum_{j=0}^{n-1} \lambda_l^{j+i} a_j$.

Corollary 1: Suppose, we want n common eigenvalues, i.e. $m = n$, which are chosen such that $\tilde{\Lambda}$ has full rank. Then $\mathbf{k}_i^T$ can be calculate via: $\mathbf{k}_i^T = \mathbf{c}_i^T \tilde{\Lambda}^{-1}$. In doing so, all the coefficients of $\mathbf{k}_d^T$ are determined and hence also the second moment decay rate.

Remark 1: Obviously, we have to choose $\lambda_i < 1$, $i = 1, \dots, n$ to get a stabilizing controller. Unfortunately, we nevertheless have no clue about the resulting second moment decay rate in advance.

Now, we know how to design a controller such that there is one or even more common eigenvalue. At first glance, asking for common eigenvectors might be a very difficult or even impossible task. Fortunately the following theorem states that this is for free:

Theorem 2: Suppose, there is a controller such that Eq. (7) *of Theorem 1 holds. Then the corresponding eigenvectors are also identical.*

III. A New Controller Type

In general, there are two controller types: The *delay-dependent* controller $\mathbf{k}_d^T$, where there is no correlation between the different control vectors $\mathbf{k}_i^T$, $i \in \mathcal{D}$ and the *delay-independent* controller $\mathbf{k}^T$, where the control vector does not depend on the current delay.

Obviously, the delay-dependent controller is more complex but will often give a better performance. Unfortunately, there is an implementation problem if there is delay between the controller and the actuator. The controller can not know the delay d which will occur while sending the packet to the actuator. Hence the controller can not decide which control vector $\mathbf{k}_i^T$ to choose.

In order to combine the pros of both types, we propose a third kind of controller: $\mathbf{k}_d^T = \lambda^d \mathbf{k}^T$. The calculation of this control law can be shared between the controller and the actuator. The controller calculates $u = \mathbf{k}^T \mathbf{x}$, which is a delay-independent control law and sends it to the actuator. The actuator, which is aware of the occurred delay d, calculates the λ^d part.

This controller type not only simplifies the implementation problem but also gives a common eigenmode:

Theorem 3: Suppose, $\mathbf{k}_d^T = \lambda^d \mathbf{k}^T$ is used as control law, where λ is an eigenvalue of the non delayed closed loop. Then λ is also an eigenvalue of all $\mathcal{A}_i$, i.e. $\det(\lambda I - \mathcal{A}_i) = 0$, $\forall i \in \mathcal{D}$.

Proof: This can be best seen by using $\mathbf{k}_i^T = \lambda^i \mathbf{k}^T$ in Eq. (6). ∎

Note, that this approach can also be used to treat the loss of packets. In this case, the packet loss is modeled as delay: If a packet is lost, then the previous one is still there and is obviously delayed. Thus, the actuator can reuse (and modify) the previous packet as follows: $u(k) = \lambda u(k-1)$.

IV. Interpretation

It is easy to see, that the λ^d term of the new controller type can be interpreted as a predictor. We can also interpret the case of n common eigenmodes by a prediction of the closed loop system:

For $d = 0$, i.e. the non delayed case, we use $u(k) = \mathbf{b}\mathbf{k}_0^T \mathbf{x}(k)$, where $\mathbf{k}_0^T$ is found by any standard method, e.g. pole placement or LQR. For an arbitrary delay $d \geq 1$, we can predict the current state $\mathbf{x}(k)$ of the closed loop from the delayed state $\mathbf{x}(k-d)$ as follows: $\mathbf{x}(k) = (A + \mathbf{b}\mathbf{k}_0^T)^d \mathbf{x}(k-d)$ and thus get:

$$u(k) = \mathbf{b} \underbrace{\mathbf{k}_0^T (A + \mathbf{b}\mathbf{k}_0^T)^d}_{=:\mathbf{k}_d^T} \mathbf{x}(k-d).$$

Hence, we have

$$\mathbf{k}_d^T := \mathbf{k}_0^T (A + \mathbf{b}\mathbf{k}_0^T)^d. \tag{8}$$

Theorem 4: The delay dependent controller $\mathbf{k}_d^T$ given in Corollary 1 and the one given by Eq. (8) *are identical.*

Proof: First, note that $\mathbf{c}_i^T = \mathbf{c}_0^T \mathrm{diag}(\lambda_1, \dots, \lambda_n)^i, i = 1, \dots, \bar{d}$. Thus, we can write $\mathbf{k}_i^T \tilde{\Lambda} = \mathbf{c}_0^T \mathrm{diag}(\lambda_1, \dots, \lambda_n)^i$. For $i = 0$ this is $\mathbf{k}_0^T \tilde{\Lambda} = \mathbf{c}_0^T$ and it follows that $\mathbf{k}_i^T = \mathbf{k}_0^T \tilde{\Lambda} \mathrm{diag}(\lambda_1, \dots, \lambda_n)^i \tilde{\Lambda}^{-1}$. In Corollary 1, we assumed that the system is given in controllable normal form. Note that $\tilde{\Lambda}$ is the matrix of the corresponding right eigenvectors of the closed loop and thus we have:
$\mathbf{k}_i^T = \mathbf{k}_0^T \tilde{\Lambda} \mathrm{diag}(\lambda_1, \dots, \lambda_n)^i \tilde{\Lambda}^{-1} = \mathbf{k}_0^T (A + \mathbf{b}\mathbf{k}_0^T)^i$ if the system is given in controllable normal form. ∎

Note, that this work was motivated by the observation, that minimizing the second moment decay rate often results in common eigenmodes. We have shown that getting n common eigenmodes is identical to the usage of a predictor. Interestingly, this nicely fits to [9], where the authors state that every stabilizing dead-time controller has an observer-predictor-based structure.

It is well known, that a predictor is very sensitive to model uncertainties. Thus, designing the controller such that there

are n common eigenmodes might not be the best choice. It might be better to get only one or two common eigenmodes to compensate the negative effects of the delay, and use the residual degree of freedom for robustness issues.

Note, that the common eigenmodes are independent of the delay probabilities. Hence, uncertainties in the delay process do not effect the proposed method.

V. Example

In this section, we use an example to visualize the benefits of the proposed method. Therefore, we design three controllers with common eigenmodes and compare them to another controller which just stabilizes the system in the second moment sense. We use the inverted pendulum on a cart example given in [5] and [6]: $\mathbf{x}(k+1) = A\mathbf{x}(k) + \mathbf{b}u(k-d)$, where the values of A and $\mathbf{b}$ are:

$$A = \begin{bmatrix} 1 & 0.1 & -0.0166 & -0.0005 \\ 0 & 1 & -0.3374 & -0.0166 \\ 0 & 0 & 1.0996 & 0.1033 \\ 0 & 0 & 2.0247 & 1.0996 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 0.0045 \\ 0.0896 \\ -0.0068 \\ -0.1377 \end{bmatrix}.$$

For simplicity, we model the random delay by an iid random process with the probability vector $\mathbf{p}^T = \begin{bmatrix} 0.3 & 0.6 & 0.1 \end{bmatrix}$.

Our controller design algorithm is based on the Cone Complementary Linearization (CCL) [10], as suggested in [6], which we already used in [8]. In order to get common eigenvalues, we added the constraints discussed in Sec. II. For all examples, we set $\epsilon = 10^{-9}$.

After the controller design, we simulated each closed loop system with an initial deviation of the angle, i.e. $\mathbf{x}(0) = \begin{bmatrix} 0 & 0.1 & 0 & 0 \end{bmatrix}^T$ and plotted $\|\mathbf{x}\|$ on a logarithmic scale. Due to the randomness of the time delay, one simulation run is definitively not enough to get a realistic picture of its behavior. Hence, we simulated each closed loop system 100 times.

The following controller is used for comparison:

$$\begin{aligned} \mathbf{k}_0^T &= \begin{bmatrix} 0.134686 & 1.30997 & 21.4185 & 5.79433 \end{bmatrix}, \\ \mathbf{k}_1^T &= \begin{bmatrix} 0.64929 & 0.947411 & 23.3122 & 6.70352 \end{bmatrix}, \\ \mathbf{k}_2^T &= \begin{bmatrix} 0.105784 & -0.335455 & 11.0665 & 4.57635 \end{bmatrix}. \end{aligned}$$

This controller gives a second moment decay rate of 0.9038 but no common eigenmodes. Figure 2(a) shows the 100 simulation runs of the closed loop with this controller. Although the system decays to the origin, there is some chaos and inconsistency. Hence, we are not very satisfied with this result.

Now, we present some controllers with the feature presented in this work. We start with a controller, which gives two common eigenmodes, with eigenvalues at 0.9 and 0.89:

$$\begin{aligned} \mathbf{k}_0^T &= \begin{bmatrix} 1.40881 & 2.82344 & 32.6137 & 6.27701 \end{bmatrix}, \\ \mathbf{k}_1^T &= \begin{bmatrix} 0.284209 & 0.716302 & 21.2214 & 5.19977 \end{bmatrix}, \\ \mathbf{k}_2^T &= \begin{bmatrix} -0.881911 & -1.59426 & 7.09703 & 2.75821 \end{bmatrix}. \end{aligned}$$

The following table lists the four most important eigenvalues of the resulting closed loop matrices:

$\mathcal{A}_0$	0.890001	0.899998	0.791201±0.355636j
$\mathcal{A}_1$	0.890054	0.899961	0.79658 ±0.380964j
$\mathcal{A}_2$	0.890002	0.899998	0.789565±0.361021j

In addition to the desired eigenvalues at 0.9 and 0.89, there is a pair of conjugate complex eigenvalues ($\approx 0.79 \pm 0.36j$), which are almost identical. As already noted in the introduction this pair of complex poles appeared without our intention just by minimizing the second moment decay rate.

Furthermore, $\mathcal{A}_1$ has an additional eigenvalue at 0.816025 and $\mathcal{A}_2$ has two additional eigenvalues at -0.560614 and 1.39068. The second moment decay rate is 0.81 and hence the entire system is stable.

Figure 2(b) shows the 100 simulation runs of the corresponding closed loop system. Compared with the previous figure, we see a much smoother and consistent decay. The simulations look almost like the simulations of an usual linear system. Nevertheless, at the very beginning, there is some jumping which is also the reason, why there is not only one line. Note, that the oscillation at the beginning is due to the almost identical conjugate complex eigenvalues.

As highlighted in Corollary 1, it is also possible to design a controller $\mathbf{k}_d^T$ such that there are n common eigenmodes. For the next controller, we set them to 0.9, 0.89, 0.88 and 0.87:

$$\begin{aligned} \mathbf{k}_0^T &= \begin{bmatrix} 0.129222 & 0.434337 & 20.182703 & 4.077384 \end{bmatrix}, \\ \mathbf{k}_1^T &= \begin{bmatrix} 0.044039 & 0.160944 & 16.995251 & 3.873278 \end{bmatrix}, \\ \mathbf{k}_2^T &= \begin{bmatrix} -0.037927 & -0.110152 & 13.673295 & 3.425691 \end{bmatrix}. \end{aligned}$$

In addition to the desired eigenvalues, $\mathcal{A}_1$ has one more eigenvalue at 0.659199 and $\mathcal{A}_2$ has two more eigenvalues at -0.532337 and 1.19154. The second moment decay rate is 0.8098 and hence the entire system is stable.

Figure 2(c) shows the 100 simulation runs of the corresponding closed loop system. We also simulated the system without any delay, using the controller $\mathbf{k}_0^T$, which corresponds to ordinary pole placement. This simulation is shown as a bold red line in Figure 2(c) and we see that the effect of the random delay sequence is almost eliminated here.

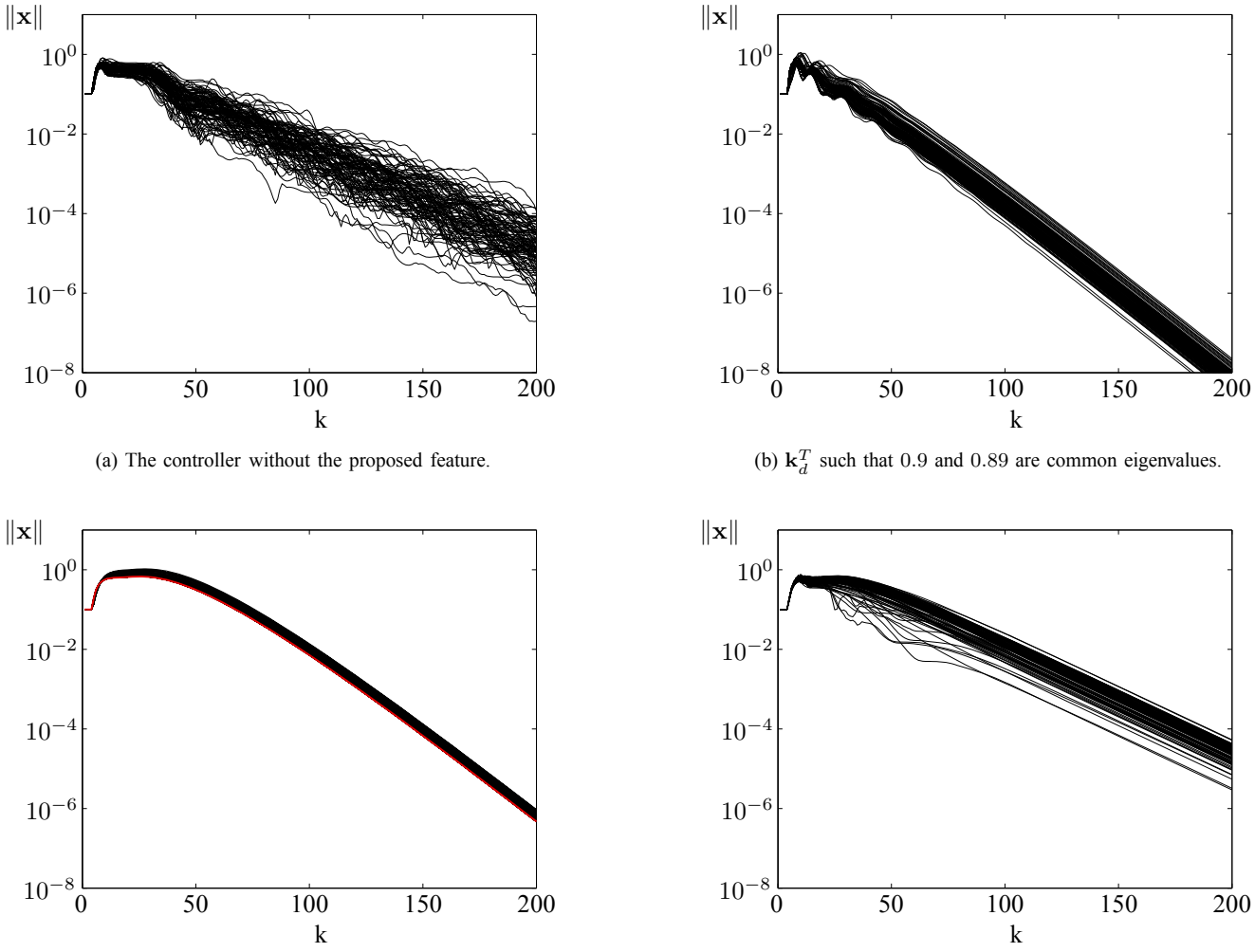
Finally, we give an example for the new controller type $\mathbf{k}_d^T = \lambda^d \mathbf{k}^T$, introduced in Sec. III. Therefore, we set the desired eigenvalue λ to 0.94 and got the following controller:

$$\mathbf{k}^T = \begin{bmatrix} 0.127998 & 0.443548 & 20.7205 & 4.9462 \end{bmatrix}.$$

Using this controller and checking the eigenvalues, we see that $\mathcal{A}_0$ has one at 0.940001, $\mathcal{A}_1$ one at 0.940004 and $\mathcal{A}_2$ has one at 0.939992. Moreover, the second moment decay rate is 0.8836. Figure 2(d) shows the 100 simulation runs with this controller. These simulations do not look as good as the two previous ones. However, keeping the reduced complexity in mind and compared with Figure 2(a) they look very good.

VI. Conclusion

In this work, we proposed a method to design a controller for a NCS such that the closed loop is not only second moment stable, but also has one (or even more) common eigenmode. We showed the benefits of this approach by an



(a) The controller without the proposed feature.

(b) $\mathbf{k}_d^T$ such that 0.9 and 0.89 are common eigenvalues.

(c) The controller which gives four common eigenmodes.

(d) $\mathbf{k}_d^T = \lambda^d \mathbf{k}^T$ such that 0.94 is a common eigenvalue.

Fig. 2. The simulation runs of the closed loop system with the different controllers.

example. In order to solve the controller design problem, we had to add equality constraints to the CCL algorithm, which we formerly used to minimize the second moment decay rate. Other algorithms, which synthesize controllers for a NCS with delay might also be modified such that there are common eigenmodes.

As noted in the introduction, we observed this effect while minimizing the second moment decay rate of JLS (4). Hence, the interesting question is whether or not common eigenmodes are necessary and/or sufficient for a minimal second moment decay rate.

We have seen, that getting n common eigenmodes is identical to the usage of a predictor. Finding a trade-off between the sensitivity to model uncertainties, an acceptable second moment decay rate and the common eigenmodes feature will be part of our ongoing work.

VII. Acknowledgement

We would like to thank Ulrich Münz for some very helpful comments.

Appendix I
Proof of Lemma 1

In order to simplify $\det(\lambda I_{n\cdot(\bar{d}+1)} - \mathcal{A}_i)$, we first write $(\lambda I_{n\cdot(\bar{d}+1)} - \mathcal{A}_i)$ in detail. It is:

$$\begin{bmatrix} \lambda I - A & \leftarrow -\mathbf{b}\mathbf{k}_i^T \rightarrow & & & \\ -I & \lambda I & \dots & 0 & 0 \\ 0 & -I & \ddots & \vdots & 0 \\ \vdots & \vdots & \ddots & \lambda I & \vdots \\ 0 & 0 & \dots & -I & \lambda I \end{bmatrix}.$$

Counting from 0, we multiply the l-th block column with $\lambda^{\bar{d}-l}$, to get:

$$\begin{bmatrix} \lambda^{\bar{d}+1} I - \lambda^{\bar{d}} A & \leftarrow -\lambda^{\bar{d}-i}\mathbf{b}\mathbf{k}_i^T \rightarrow & & & \\ -\lambda^{\bar{d}} I & \lambda^{\bar{d}} I & \dots & 0 & 0 \\ 0 & -\lambda^{\bar{d}-1} I & \ddots & \vdots & 0 \\ \vdots & \vdots & \ddots & \lambda^2 I & \vdots \\ 0 & 0 & \dots & -\lambda I & \lambda I \end{bmatrix}.$$

$$\tilde{A}_n = \begin{bmatrix} \lambda^{\bar{d}+1} & -\lambda^{\bar{d}} & 0 & \dots & 0 \\ 0 & \lambda^{\bar{d}+1} & -\lambda^{\bar{d}} & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda^{\bar{d}+1} & -\lambda^{\bar{d}} \\ a_0\lambda^{\bar{d}} - k_{i0}\lambda^{\bar{d}-i} & a_1\lambda^{\bar{d}} - k_{i1}\lambda^{\bar{d}-i} & \dots & \dots & a_{n-1}\lambda^{\bar{d}} - k_{in-1}\lambda^{\bar{d}-i} + \lambda^{\bar{d}+1} \end{bmatrix} \tag{9}$$

Remember, that if a matrix B is derived from a matrix A by multiplying one column of A with $\alpha \in \mathbb{R}$, then $\det(B) = \alpha \det(A)$ (see e.g. [11], Proposition 2.7.2). We will take care of this at the end of this proof.

Now, we want to eliminate the elements in the lower left part of this matrix. Therefore, we first add the last block column to the last but one, then the last but one to the last but second and so on. Remember, that adding one column to another does not change the determinant of the matrix. Finally, we add the second block column to the first one and end up with:

$$\begin{bmatrix} \Xi & -\lambda^{\bar{d}-i}\mathbf{b}\mathbf{k}_i^T \rightarrow & \dots & \\ 0 & \lambda^{\bar{d}} I & \dots & 0 & 0 \\ 0 & 0 & \ddots & \vdots & 0 \\ \vdots & \vdots & \ddots & \lambda^2 I & \vdots \\ 0 & 0 & \dots & 0 & \lambda I \end{bmatrix},$$

where $\Xi = \lambda^{\bar{d}+1} I - \lambda^{\bar{d}} A - \lambda^{\bar{d}-i}\mathbf{b}\mathbf{k}_i^T$. Note, that the $-\lambda^{\bar{d}-i}\mathbf{b}\mathbf{k}_i^T$ entry is in the first i columns due to this summations.

Calculating the determinant of the previous matrix and keeping the multiplications of the l-th block column with $\lambda^{\bar{d}-l}$ in mind, we obtain Lemma 1.

Appendix II
Proof of Lemma 2

Here, we further simplify $\det(\lambda^{\bar{d}+1} I_n - \lambda^{\bar{d}} A - \mathbf{b}\mathbf{k}_i^T \lambda^{\bar{d}-i})$, assuming that the system is given in controllable normal form. Therefore, we first introduce the shortcut: $\tilde{A}_n := (\lambda^{\bar{d}+1} I_n - \lambda^{\bar{d}} A - \mathbf{b}\mathbf{k}_i^T \lambda^{\bar{d}-i})$, where the index n denotes that $A \in \mathbb{R}^{n \times n}$. Eq. (9) at the top of this page, shows $\tilde{A}_n$ in detail.

In order to calculate $\det(\tilde{A}_n)$, we expand the determinant along the first column to get:

$$\lambda^{\bar{d}+1} \det \tilde{A}_{(n-1)} + \lambda^{(n-1)\bar{d}}(a_0\lambda^{\bar{d}} - k_{i0}\lambda^{\bar{d}-i}).$$

Following this recursion, we obtain Lemma 2.

Appendix III
Proof of Theorem 2

Here, we have to find a vector $\begin{bmatrix} \mathbf{r}_0^T & \mathbf{r}_1^T & \dots & \mathbf{r}_{\bar{d}}^T \end{bmatrix}^T$ such that the following equation holds for all $i \in \mathcal{D}$.

$$\begin{bmatrix} A & \leftarrow \mathbf{b}\mathbf{k}_i^T \rightarrow & & & \\ I & 0 & \dots & 0 & 0 \\ 0 & I & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & I & 0 \end{bmatrix} \begin{bmatrix} \mathbf{r}_0 \\ \mathbf{r}_1 \\ \mathbf{r}_2 \\ \vdots \\ \mathbf{r}_{\bar{d}} \end{bmatrix} = \lambda \begin{bmatrix} \mathbf{r}_0 \\ \mathbf{r}_1 \\ \mathbf{r}_2 \\ \vdots \\ \mathbf{r}_{\bar{d}} \end{bmatrix}.$$

From the first block row, we obtain:

$$A\mathbf{r}_0 + \mathbf{b}\mathbf{k}_i^T \mathbf{r}_i = \lambda \mathbf{r}_0. \tag{10}$$

The residual rows are:

$$\mathbf{r}_j = \lambda \mathbf{r}_{j+1}, \quad j = 0, \dots, \bar{d} - 1. \tag{11}$$

$$\Rightarrow \mathbf{r}_0 = \lambda^i \mathbf{r}_i. \tag{12}$$

Using Eq. (12) in Eq. (10) gives us:

$$A\lambda^i \mathbf{r}_i + \mathbf{b}\mathbf{k}_i^T \mathbf{r}_i = \lambda^{i+1} \mathbf{r}_i, \tag{13}$$

which holds due to the assumption and Lemma 1.

Remember the definition of the augmented state vector $\mathbf{z}(k) := \begin{bmatrix} \mathbf{x}(k)^T & \mathbf{x}(k-1)^T & \cdots & \mathbf{x}(k-\bar{d})^T \end{bmatrix}^T$. If $\mathbf{z}(k)$ is a common eigenvector of the matrices $\mathcal{A}_i$, then Eq. (11) can be interpreted as $\mathbf{x}(k) = \lambda \mathbf{x}(k-1)$, independent of the occurred delay. This is exactly what we wanted.

Note that similar thoughts can be used to show that the left eigenvectors are also identical.

References

[1] Special Section on Networks and Control, *IEEE Control Systems Magazine*, vol. 21, no. 1, 2001.

[2] Special Issue on Networked Control Systems, *IEEE Transactions on Automatic Control*, vol. 49, no. 9, 2004.

[3] Special Issue on Technology of Networked Control Systems, *Proceedings of the IEEE*, vol. 95, no. 1, 2007.

[4] R. Krtolica, Ü. Özgüner, H. Chan, H. Göktas, J. Winkelman, and M. Liubakka, "Stability of linear feedback systems with random communication delays," *International Journal of Control*, vol. 59, no. 4, pp. 925–953, 1994.

[5] L. Xiao, A. Hassibi, and J. P. How, "Control with random communication delays via a discrete-time jump system approach," in *Proceedings of the American Control Conference*, vol. 3, 2000, pp. 2199–2204.

[6] L. Zhang, Y. Shi, T. Chen, and B. Huang, "A new method for stabilization of networked control systems with random delays," *IEEE Transactions on Automatic Control*, vol. 50, no. 8, pp. 1177–1181, 2005.

[7] R. Blind, "Stabilization and transient behavior of networked control systems with stochastic delay and loss of packets," Master's thesis, Institute for Systems Theory and Automatic Control, University of Stuttgart, Germany, 2007.

[8] R. Blind, U. Münz, and F. Allgöwer, "Modeling, analysis, and design of networked control systems using jump linear systems," *Automatisierungstechnik*, vol. 56, no. 1, pp. 20–28, January 2008.

[9] L. Mirkin and N. Raskin, "Every stabilizing dead-time controller has an observer-predictor-based structure," *automatica*, vol. 39, no. 10, pp. 1747–1754, 2003.

[10] L. E. Ghaoui, F. Oustry, and M. AitRami, "A cone complementary linearization algorithm for static output-feedback and related problems," *IEEE Transactions on Automatic Control*, vol. 42, no. 8, pp. 1171–1176, 1997.

[11] D. S. Bernstein, *Matrix Mathematics: Theory, Facts, and Formulas with Application to Linear Systems Theory*. Princeton University Press, 2005.