Control with Random Communication Delays via a Discrete-Time Jump System Approach

Lin Xiao [1] Arash Hassibi Jonathan P. How

Information Systems Laboratory
Stanford University
Stanford, CA 94305, USA

Abstract—Digital control systems with random but bounded delays in the feedback loop can be modeled as finite-dimensional, discrete-time jump linear systems, with the transition jumps being modeled as finite-state Markov chains. This type of system can be called a "stochastic hybrid system". Due to the structure of the augmented state-space model, control of such a system is an output feedback problem, even if a state feedback law is intended for the original system. We propose a V-K iteration algorithm to design switching and non-switching controllers for such systems. This algorithm involves an iterative process that requires the solution of a convex optimization problem constrained by linear matrix inequalities at each step.

Keywords: V-K iteration, jump linear system, random delays, stochastic stability, linear matrix inequality (LMI), bilinear matrix inequality (BMI).

1 Introduction

In many complex systems, particularly those with remote sensors, actuators and processors, a communication network can be used to gather sensor data and send control signals. However, the utilization of a multi-user network with random demands affecting the network traffic could result in random delays in the feedback loop, from the sensors to the processors, and/or from the processors to the actuators. These delays will deteriorate the system performance as well as stability.

This type of system can be modeled as finite-dimensional, discrete-time jump linear systems [10, 5], with the transition jumps being modeled as finite-state Markov processes. The stability results of Markovian jump linear systems are well estabilished [6, 9, 10], and state feedback problem for such systems can be formulated as a convex optimization over a set of LMI's [1, 2], thus can be very efficiently solved by interior-point methods [4, 12]. However, for system with random delays, the augmented discrete-time jump system has a special structure, which leads to an output feedback control problem, even if a state feedback law is intended for the original system. This change in the problem formulation greatly complicates the control design within the convex optimization framework. This paper proposes a V-K iteration algorithm to design switching and non-switching output feedback stabilizing controllers for such systems. This control synthesis algorithm involves an iterative process that requires the solution of a convex optimization problem constrained by LMI's at each step.

[1]Contact author. E-mail: `lxiao@stanford.edu`. Lin Xiao is supported by a Stanford Graduate Fellowship.

In section 2, the general system setting with random communication delays is described and modeled as a jump linear system. Stability results for discrete-time jump systems are reviewed in section 3. In section 4, we describe the V-K iteration algorithm used to design stabilizing controllers. In section 5, both switching and non-switching controllers are designed for a cart and inverted pendulum system, which is robust to random delays in the feedback loop.

2 System Modeling

2.1 Delayed state feedback model

First consider the simple system setup in Figure 1. The discrete-time linear time-invariant (LTI) plant model is

$$x(k+1) = Ax(k) + Bu(k) \tag{1}$$

where $x(k) \in \mathbf{R}^n$, $u(k) \in \mathbf{R}^m$. There are random but bounded delays from the sensor to the controller. The mode-dependent switching state feedback control law is

$$u(k) = K_{r_s(k)} x(k - r_s(k)) \tag{2}$$

where $\{r_s(k)\}$ is a bounded random integer sequence with $0 \le r_s(k) \le d_s < \infty$, and d_s is the finite delay bound. If we augment the state variable

$$\tilde{x}(k) = \begin{bmatrix} x(k)^T & x(k-1)^T & \dots & x(k-d_s)^T \end{bmatrix}^T$$

where $\tilde{x}(k) \in \mathbf{R}^{(d_s+1)n}$, then the closed-loop system is

$$\tilde{x}(k+1) = (\tilde{A} + \tilde{B} K_{r_s(k)} \tilde{C}_{r_s(k)}) \tilde{x}(k) \tag{3}$$

where

$$\tilde{A} = \begin{bmatrix} A & 0 & \dots & 0 & 0 \\ I & 0 & \dots & 0 & 0 \\ 0 & I & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & I & 0 \end{bmatrix} \qquad \tilde{B} = \begin{bmatrix} B \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\tilde{C}_{r_s(k)} = \begin{bmatrix} 0 & \dots & 0 & I & 0 & \dots & 0 \end{bmatrix}$$

and $\tilde{C}_{r_s(k)}$ has all elements being zero except for the $r_s(k)$th block being an identity matrix. Equation (3) corresponds to a discrete-time jump linear system. Notice that these equations are in the form of an output feedback control problem, even if a state feedback control law (2) was intended for the original system (1).

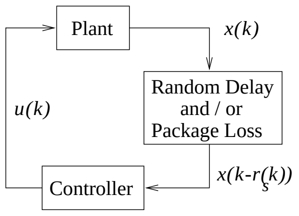


Figure 1: Control over networks

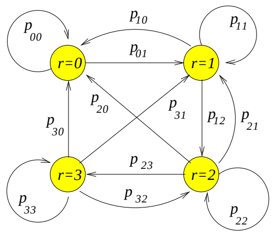


Figure 2: Markovian Jump States

One of the difficulties with this approach is how to model the $r_s(k)$ sequence. One way is to model the transitions of the random delays $r_s(k)$ as a finite state Markov process [11, 5]. In this case we have

$$\mathbf{Prob}\{r_s(k+1) = j \,|\, r_s(k) = i\} = p_{ij} \tag{4}$$

where $0 \le i, j \le d_s$. This model is quite general, communication package loss in the network can be included naturally as explained below. The assumption here is that the controller will always use the most recent data. Thus, if we have $x(k - r_s(k))$ at step k, but there is no new information coming at step $k+1$ (data could be lost or there is longer delay), then we at least have $x(k - r_s(k))$ available for feedback. So in our model of the system in Figure 1, the delay $r_s(k)$ can increase at most by 1 each step, and we constrain

$$\mathbf{Prob}\{r_s(k+1) > r_s(k) + 1\} = 0$$

However, the delay $r_s(k)$ can decrease as many steps as possible. Decrement of $r_s(k)$ models communication package loss in the network, or disregarding old data if we have newer data coming at the same time. Hence the structured transition probability matrix is

$$P_s = \begin{bmatrix} p_{00} & p_{01} & 0 & 0 & \dots & 0 \\ p_{10} & p_{11} & p_{12} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & 0 \\ & & & & & p_{d_s-1,d_s} \\ p_{d0} & p_{d1} & p_{d2} & p_{d3} & \dots & p_{d_s d_s} \end{bmatrix} \tag{5}$$

where

$$0 \le p_{ij} \le 1 \quad \text{and} \quad \sum_{j=0}^{d_s} p_{ij} = 1 \tag{6}$$

because each row represents the transition probabilities from a fixed state to all the states. The diagonal elements are the probabilities of data coming in sequence with equal delays. The elements above the diagonal are the probabilities of encountering longer delays, and the elements below the diagonal indicate package loss or disregarding old data. Figure 2 shows a four state transition diagram with such a structure, which clearly shows that we can jump from $r = 0$ to $r = 1$ and from all other r's to $r = 0$, but we cannot jump directly from $r = 0$ to $r = 2$ or $r = 3$. We can use the mode-dependent switching controller if we know the delay steps on-line, and this is the case if we use time-stamped data in the network communication. We will mention how to handle the situation where the transition probabilities are uncertain in next section.

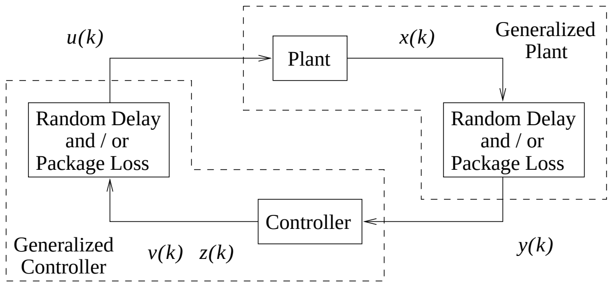


Figure 3: Control over networks: the general case

2.2 General model with dynamic output feedback

Next we consider a more general model with dynamic feedback controller (see Figure 3)

$$\begin{aligned} z(k+1) &= Fz(k) + Gy(k) \\ v(k) &= Hz(k) + Jy(k) \end{aligned} \tag{7}$$

In this case we use a mode-independent controller to simplify the notation. More importantly, because it is hard to predict the delays from the controller to the actuator at the time the control signal is calculated, the mode-independent controller is probably the most relevant for this application. To proceed, augment the controller state variable $\tilde{z}(k) = [\; z(k)^T \quad v(k)^T \quad \dots \quad v(k-d_a) \;]^T$, where d_a is the finite bound for the random delays $r_a(k)$ from the controller to the actuator. The generalized controller can be written as

$$\begin{aligned} \tilde{z}(k+1) &= \tilde{F}\tilde{z}(k) + \tilde{G}y(k) \\ u(k) &= \tilde{H}_{r_a(k)}\tilde{z}(k) + \tilde{K}_{r_a(k)}y(k) \end{aligned} \tag{8}$$

where

$$\tilde{F} = \begin{bmatrix} F & 0 & \dots & 0 & 0 \\ H & 0 & \dots & 0 & 0 \\ 0 & I & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & I & 0 \end{bmatrix} \qquad \tilde{G} = \begin{bmatrix} G \\ J \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\tilde{H}_{r_a(k)} = \begin{cases} \begin{bmatrix} H & 0 & \dots & 0 & 0 & \dots & 0 \end{bmatrix} & \text{if } r_a(k) = 0 \\ \begin{bmatrix} 0 & \dots & 0 & I & 0 & \dots & 0 \end{bmatrix} & \text{if } r_a(k) \neq 0 \end{cases}$$

$$\tilde{K}_{r_a(k)} = \begin{cases} J & \text{if } r_a(k) = 0 \\ 0 & \text{if } r_a(k) \neq 0 \end{cases}$$

Here we use the control signals $v(k), \dots, v(k-d_a)$ generated at different steps for analysis and design purpose. After the controller (7) is designed using the generalized model, we need not to store them in real-time control applications. The transition probability matrix P_a has the same structure as P_s in (5). Combined with the generalized plant model

$$\begin{aligned} \tilde{x}(k+1) &= \tilde{A}\tilde{x}(k) + \tilde{B}u(k) \\ y(k) &= C\tilde{C}_{r_s(k)}\tilde{x}(k) \end{aligned} \tag{9}$$

and using the state variable $\bar{x} = [\; \tilde{x}^T \quad \tilde{z}^T \;]^T$, we can write the generalized closed-loop system dynamics as

$$\bar{x} = (\bar{A} + \bar{B}\bar{K}_{r_a(k)}\bar{C}_{r_s(k)})\bar{x}(k) \tag{10}$$

where

$$\bar{A} = \begin{bmatrix} \tilde{A} & 0 \\ 0 & 0 \end{bmatrix} \qquad \bar{B} = \begin{bmatrix} 0 & \tilde{B} \\ I & 0 \end{bmatrix}$$

$$\bar{C}_{r_s(k)} = \begin{bmatrix} 0 & I \\ C\tilde{C}_{r_s(k)} & 0 \end{bmatrix} \quad \bar{K}_{r_a(k)} = \begin{bmatrix} \tilde{F} & \tilde{G} \\ \tilde{H}_{r_a(k)} & \tilde{K}_{r_a(k)} \end{bmatrix}$$

This general system is also a jump linear system. The Markovian jump parameter now becomes $r(k) = (r_s(k),\ r_a(k))$, and the transition probability matrix is $P = P_s \otimes P_a$, where $\otimes$ denotes the matrix Kronecker product [7]. So, both static and dynamic feedback with random delays can be formulated as discrete-time jump system control problems. Thus all stability results and design algorithms in following sections apply to both cases.

3 Jump System Control

In this section we consider the general (closed-loop) jump linear system

$$x(k+1) = A_{r(k)}x(k) \tag{11}$$

where $x(k) \in \mathbf{R}^n$, the Markovian integer jump parameter $r(k) \in \{0, \ldots, d\}$ is described by (4), and $A_{r(k)} \in \{A_0, A_1, \ldots, A_d\}$. Note this model can represent (3) or (10) in the previous section. Define the mode indicator function

$$I_i(k) = \begin{cases} I, & \text{if } r(k) = i \\ 0, & \text{if } r(k) \neq i \end{cases}$$

There are two ways ([6], [1]) to define the mode-dependent covariance matrices

$$\begin{array}{lrcl} \text{(a)} & M_i^a(k) & = & \mathbf{E}[x(k)x(k)^T I_i(k)] \qquad (12) \\ \text{(b)} & M_i^b(k) & = & \mathbf{E}[x(k)x(k)^T I_i(k-1)] \qquad (13) \end{array}$$

They satisfy the linear recursions

$$M_i^a(k+1) = \sum_{j=0}^{d} p_{ji} A_j M_j^a(k) A_j^T \tag{14}$$

$$M_i^b(k+1) = A_i \left(\sum_{j=0}^{d} p_{ji} M_j^b(k) \right) A_i^T \tag{15}$$

respectively. All these equations hold for $i = 0, 1, \ldots, d$. The mode-independent covariance matrix is

$$M(k) = \mathbf{E}[x(k)x(k)^T] = \sum_{i=0}^{d} M_i^a(k) = \sum_{i=0}^{d} M_i^b(k)$$

The system is mean square stable if $\lim_{k\to\infty} M(k) = 0$ regardless of $x(0)$. A necessary and sufficient condition is

$$\lim_{k\to\infty} M_i^a(k) = 0 \ \text{ or } \ \lim_{k\to\infty} M_i^b(k) = 0,\ i = 0, 1, \ldots, d$$

Theorem 3.1 *[6] The mean square stability of system (11) is equivalent to the existence of symmetric positive definite matrices $Q_0, Q_1, \ldots, Q_d$ satisfying any one of the following 4 conditions:*

1. $$A_i \left(\sum_{j=0}^{d} p_{ji} Q_j \right) A_i^T < Q_i, \quad i = 0, \ldots, d \tag{16}$$
2. $$A_j^T \left(\sum_{i=0}^{d} p_{ji} Q_i \right) A_j < Q_j, \quad j = 0, \ldots, d \tag{17}$$
3. $$\sum_{j=0}^{d} p_{ji} A_j Q_j A_j^T < Q_i, \quad i = 0, \ldots, d \tag{18}$$
4. $$\sum_{i=0}^{d} p_{ji} A_i^T Q_i A_i < Q_j, \quad j = 0, \ldots, d \tag{19}$$

This theorem can be proven using the same technique as in [4] page 136-137. One interesting property is that stability of every mode is neither sufficient nor necessary for mean square stability of the jump system [9].

If the transition probability matrix P is only known to belong to the polytope $\Pi = Co\{P_1, P_2, \ldots, P_t\}$, then a sufficient and necessary condition for the jump system to be mean square stable for every $P \in \Pi$ is that Theorem 3.1 holds for every vertex P_i of the polytope [1].

The decay rate is defined as the largest $\beta > 1$ such that $\lim_{k\to\infty} \beta^k M(k) = 0$. A lower bound of the decay rate $\beta = 1/\alpha$ must satisfy the inequalities (16–19) by replacing Q_i or Q_j on the right hand side by αQ_i or αQ_j.

4 $V - K$ Iteration

Now we consider the problem of using output feedback control to stabilize the jump system (11). The closed-loop system dynamics are

$$x(k+1) = \left(A_{r(k)} + B_{r(k)}K_{r(k)}C_{r(k)}\right) x(k) \tag{20}$$

which can represent the static feedback case (3) or the dynamic output feedback case (10), and is more general than these cases. Since the four conditions in Theorem 3.1 are equivalent, we can use any one of them to describe our algorithm. For example, condition (19) with a decay rate $\beta = 1/\alpha$ for the closed-loop system (20) becomes

$$\sum_{i=0}^{d} p_{ji}(A_i + B_iK_iC_i)^T Q_i (A_i + B_iK_iC_i) < \alpha Q_j$$

Using Schur complements, it can be written in an equivalent form

$$\begin{bmatrix} \alpha Q_j & (A_0+B_0K_0C_0)^T Q_0 & \cdots & (A_d+B_dK_dC_d)^T Q_d \\ Q_0(A_0+B_0K_0C_0) & p_{j0}^{-1} Q_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ Q_d(A_d+B_dK_dC_d) & 0 & \cdots & p_{jd}^{-1} Q_d \end{bmatrix} > 0 \tag{21}$$

which must be true for $j = 0, 1, \ldots, d$. Use $\alpha = 1$ for the stability problem and $\alpha = 1/\beta$ for decay rate β. These are coupled bilinear matrix inequalities (BMI) in the variables α, K_i and Q_i, $i = 0, 1, \ldots, d$, and every K_i and Q_i appear in all of the $d+1$ inequalities. If we fix α and the K_i's, then these equations are LMI's in the Q_i's, and vice versa.

A lower bound for the decay rate $\beta = 1/\alpha$ can be found by solving the following optimization problem

$$\begin{array}{ll} \text{minimize} & \alpha \\ \text{subject to} & (21) \text{ for } j = 0, 1, \ldots, d \\ & \text{and} \quad Q_0 > 0, \ldots, Q_d > 0 \end{array} \tag{22}$$

If we fix the K_i's, this corresponds to a generalized eigenvalue problem, which is quasi-convex in α and the Q_i's; if we fix the Q_i's, it is a eigenvalue problem, which is convex in α and the K_i's. Both of these problems can be solved very efficiently by convex optimization [4].

Note that we want to design a controller for a specified transition probability matrix P. However, it is often difficult to get an initial guess that works well for this P. So we start with a simple P_0 for which it is easy to design a stabilizing controller, and then change P in an outer-loop as we proceed through the design iteration. Now, using a similar idea as in [3], we give the V-K iteration algorithm used to design a stabilizing controller for the system model developed in Section 2.

1. Design a LQR (or LQG) for dynamic output feedback) controller K for the plant (1) without considering delays in the loop. Let $K_i = K$, for $i = 0, 1, \dots, d$, and initialize the transition probability matrix, let $P = P_0$.
2. Design Iterations
   Repeat{
   (a) V-step. Given the controllers $K_i,\ i = 0, \dots, d$, solve the LMI feasibility problem (21) for all $j = 0, \dots, d$ with $\alpha = 1$ to find $Q_i,\ i = 0, \dots, d$ to prove that the K_i's stabilize the jump system.
   (b) K-step. Given $Q_i,\ i = 0, \dots, d$ found in the V-step, solve the eigenvalue problem (22) to find the $K_i,\ i = 0, \dots, d$ which maximize the decay rate of the closed-loop jump system with respect to the Q_i's.
   (c) Δ-step. Perturb the transition probability matrix P by adding a small perturbation matrix Δ_{ij}: $P \Leftarrow P + \Delta_{ij}$.

   } Until the desired transition probability matrix P is reached or the V-step is not feasible.

Since the LQR (LQG) controller corresponds to the no delay case, a reasonable initial transition probability matrix is

$$P_0 = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & \dots & 0 \end{bmatrix} \approx \begin{bmatrix} 1-d\epsilon & \epsilon & \dots & \epsilon \\ 1-d\epsilon & \epsilon & \dots & \epsilon \\ \vdots & \vdots & \vdots & \vdots \\ 1-d\epsilon & \epsilon & \dots & \epsilon \end{bmatrix}$$

To solve it numerically, we replace p_{ji} by a very small positive number ϵ whenever it is zero in our structured transition probability matrix (5). At the same time, we must slightly change the matrix elements in the same row to keep the constraints (6) being satisfied. For ϵ small enough, The first V-step will always be feasible. For each succeeding iteration step we add a small perturbation matrix Δ_{ij} to the transition probability matrix, e.g.

$$\Delta_{01} = \begin{bmatrix} -s & s & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}, \ \Delta_{11} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ -s & s & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

$$\Delta_{12} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ -s & 0 & s & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}, \quad \dots$$

where s is a small number, say 0.01, for example. In the outer-loop of the design iterations, we will keep adding Δ_{ij} to P until p_{ij} is reached, and then start adding the next small perturbation matrix to P. The small perturbation sequence will not be unique, and we can add small perturbations to several matrix elements at the same time (see the example in Section 5). One general principle is to perturb the system from shorter delays to longer delays, ie, $\Delta_{01} \to \Delta_{11} \to \Delta_{12} \to \dots$, until the desired P is reached.

Remarks:

- A more aggressive initial transition probability matrix can be used in the design procedure, as long as the initial controller is a feasible solution of the V-step.
- In the V-step, given the controllers, instead of solving the LMI feasibility problem (21), we could solve the generalized eigenvalue problem (22).
- We can do more than one V-K iteration for every Δ, i.e., change step (a) and (b) into an inner V-K iteration loop as used in [3].
- Using different forms in Theorem 3.1 or their mixtures can lead to different designs.
- Notice that the output feedback problem (20) has a special structured $C_{r(k)}$ as in (3), which allow us to use LQR (or LQG) design to start the iteration. For more general output feedback control problem, finding a feasible initial design is very difficult.

If the Markov jump parameters $r(k)$ is not known on-line, a non-switching controller can be designed by adding the constraint $K_0 = K_1 = \dots = K_d$ to the K-step.

5 Examples

Consider the cart and inverted pendulum problem in Figure 4(a). This is a fourth order unstable system. The state variables are $[\ x \quad \dot{x} \quad \theta \quad \dot{\theta}\ ]^T$. The parameters used are: $m_1 = 1\ kg$, $m_2 = 0.5\ kg$, $L = 1\ meter$, and no friction surfaces. The sampling time is $T_s = 0.1$ second. Assume the case where random delays only exist from the sensor to the controller, and they are bounded by 2: $r(k) \in \{0, 1, 2\}$. The controllers are designed using the linearized model, but the computer simulation uses the nonlinear dynamics. The initial condition for simulation is $\theta(0) = 0.1\ rad$, and all other initial states are zero.

5.1 State feedback controllers

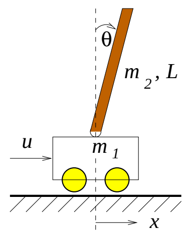
Given the "expected" transition probability matrix

$$P = \begin{bmatrix} 0.5 & 0.5 & 0 \\ 0.3 & 0.6 & 0.1 \\ 0.3 & 0.6 & 0.1 \end{bmatrix}$$

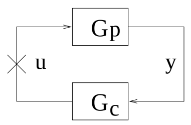
We want to design both switching and non-switching state feedback controllers for the jump system. First we design an LQR state feedback controller using weighting matrices $Q_x = I_4$ for the states and $R_u = 1$ for the control signal. We get

$$K = \begin{bmatrix} 0.5959 & 1.5087 & 30.4620 & 7.1349 \end{bmatrix}$$

Although this controller can stabilize the system when there is no delay in the feedback loop, it cannot stabilize the system in the mean square sense when random delays are



(a) Cart and Pendulum



(b) Break the loop

Figure 4: Cart and Inverted Pendulum Example

added in the loop. This can be verified by the fact that $K_0 = K_1 = K_2 = K$ is not a feasible solution to the LMI's (21). This fixed LQR controller also fails to stabilize the system in many of the simulation runs (using different random seeds), but not always.

Next we design a switching state feedback controller using the V-K iteration algorithm described in Section 4. The initial transition probability matrix and the small perturbation matrix used are

$$P_0 = \begin{bmatrix} 0.499 & 0.499 & 0.002 \\ 0.480 & 0.510 & 0.010 \\ 0.480 & 0.510 & 0.010 \end{bmatrix}, \quad \Delta = \begin{bmatrix} 0.00 & 0.00 & 0.00 \\ -0.02 & 0.01 & 0.01 \\ -0.02 & 0.01 & 0.01 \end{bmatrix}$$

The perturbation matrix was added in the design outer-loop until the ideal transition probability matrix P was reached. There were 10 design iterations and it took less than one hour running on a Sun SPARC work station. The mode-dependent switching controllers are

$$\begin{aligned} K_0 &= \begin{bmatrix} 0.6439 & 1.9292 & 30.1913 & 7.3256 \end{bmatrix} \\ K_1 &= \begin{bmatrix} 0.5010 & 1.7637 & 28.1023 & 7.7458 \end{bmatrix} \\ K_2 &= \begin{bmatrix} 1.8192 & 7.8869 & 22.4488 & 13.3279 \end{bmatrix} \end{aligned}$$

Adding the constraint $K_0 = K_1 = K_2$ to the K-step in the design iteration, we obtained the mode-independent non-switching controller

$$K_0 = K_1 = K_2 = \begin{bmatrix} 0.7382 & 2.0934 & 27.7030 & 8.1315 \end{bmatrix}$$

Notice that in our augmented state-space model (3), both $A+BK_1C_1$ and $A+BK_2C_2$ with the above two controllers have eigenvalues located outside the unit circle, but the jump system is mean square stable. Although the switching controller had a slightly better performance than the non-switching one in most simulations, these results show that the difference is small.

5.2 Dynamic output feedback controllers

For another transion probability matrix

$$P = \begin{bmatrix} 0.36 & 0.64 & 0 \\ 0.36 & 0.32 & 0.32 \\ 0.36 & 0.32 & 0.32 \end{bmatrix} \tag{23}$$

which corresponds to a longer average delay in stationary distribution, the above design iteration using state feedback

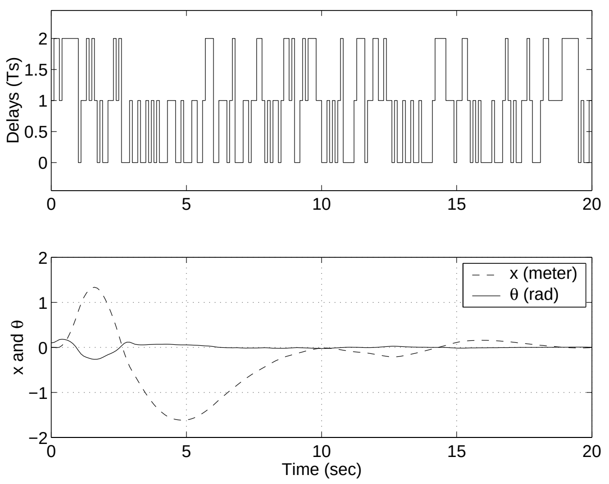


Figure 5: Random Delays and Initial Condition Response

couldn't reach the new desired transion probability matrix. But using the same design technique, we successfully designed a non-switching dynamic output feedback controller for the system. The output information used is $y = [x \ \ \theta]^T$. We started from an LQG controller

$$\begin{aligned} F &= \begin{bmatrix} 0.9191 & 0.1067 & 0.2524 & 0.0313 \\ -0.0585 & 1.1352 & 3.0493 & 0.6228 \\ 0.1207 & -0.0102 & -0.0030 & 0.0550 \\ 0.4705 & -0.2078 & -6.1384 & 0.1169 \end{bmatrix} \\ G &= \begin{bmatrix} 0.0835 & -0.1331 \\ 0.1119 & -0.6566 \\ -0.1248 & 0.8961 \\ -0.5527 & 3.9675 \end{bmatrix} \\ H &= \begin{bmatrix} 0.5969 & 1.5087 & 30.4620 & 7.1349 \end{bmatrix} \\ J &= \begin{bmatrix} 0 & 0 \end{bmatrix} \end{aligned}$$

The initial transition probability matrix and the small perturbation matrix used are

$$P_0 = \begin{bmatrix} 0.98 & 0.01 & 0.01 \\ 0.98 & 0.01 & 0.01 \\ 0.98 & 0.01 & 0.01 \end{bmatrix}, \quad \Delta = \begin{bmatrix} -0.02 & 0.02 & 0.00 \\ -0.02 & 0.01 & 0.01 \\ -0.02 & 0.01 & 0.01 \end{bmatrix}$$

The iterative design procedure provides a controller which makes the closed-loop system mean square stable

$$\begin{aligned} F &= \begin{bmatrix} 0.8662 & 0.0673 & 0.3343 & -0.1235 \\ -0.3148 & 1.1528 & -0.9257 & 0.5878 \\ -0.0461 & 0.0329 & 0.0870 & 0.1574 \\ 0.4985 & -0.2166 & -0.5426 & 0.2106 \end{bmatrix} \\ G &= \begin{bmatrix} 0.1273 & -0.6876 \\ 0.3330 & 3.2806 \\ 0.0482 & 1.2398 \\ -0.5265 & -1.0774 \end{bmatrix} \\ H &= \begin{bmatrix} -3.6345 & 2.0084 & -10.6230 & 7.7037 \end{bmatrix} \\ J &= \begin{bmatrix} 3.8954 & 41.6199 \end{bmatrix} \end{aligned} \tag{24}$$

In this case, there were 32 design iterations and it took more than one day running on the same computer.

Figure 5 shows one simulation run of the Markovian jump delays according to the given transition probability matrix,

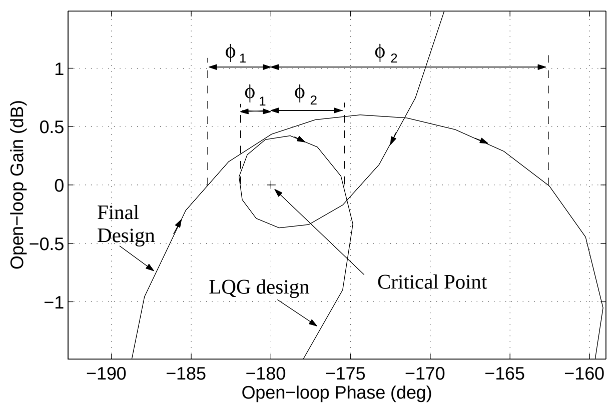


Figure 6: Phase Margins for Stability

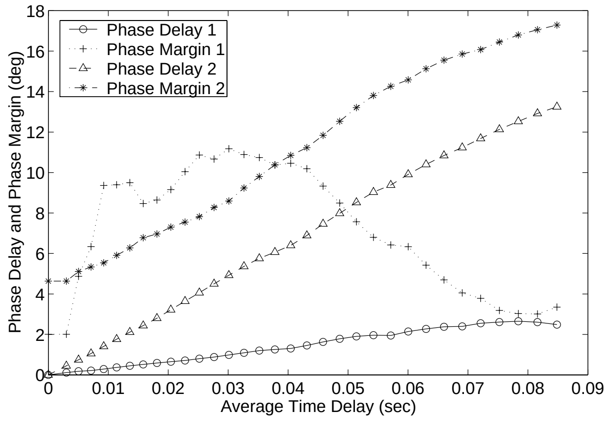


Figure 7: Phase Margin versus Average Delay

and the initial condition response of the closed-loop system using the controller (24). All other controllers designed before cannot stabilize the system.

To have a better understanding of the design algorithm, we can compare the average phase delay caused by the random time delays and the phase margin of the system. To simplify the analysis, we break the loop at the control signal (see Figure 4(b)) so that a single-input and single-output open-loop system is resulted and we can use intuitive classical control analysis tools. Figure 6 shows the log-magnitude versus linear-phase plot of the open loop system. The curve encircles the the critical point once (the plant has one unstable pole) and has two phase margins to be considered for stability. We can see that both phase margins ϕ_1 and ϕ_2 at the two cross over frequencies are increased through the V-K iteration. Figure 7 shows the phase margins and average phase delays versus the average time delays corresponding to the transition probability matrices used in each iteration step. To keep the closed-loop system stable, intuitively, the phase margin curves must stay above the corresponding (at the same crossover frequency) average phase delay curves, and this is precisely the case shown in Figure 6. An interesting phenomenon is that, to keep the system stable, the algorithm has to give up the redundant phase margins at the low crossover frequency to maintain necessary phase margins at the high crossover frequency as the average time delay increases.

6 Conclusions

This paper proposes a V-K iteration algorithm to design stabilizing controllers for special structured discrete-time jump linear systems, which are used to model control systems with random but bounded delays in the feedback loop. This algorithm involves solving a convex optimization problem constrained by LMI's at each step. The LMI's for each iteration are obtained by giving a small perturbation to the Markovian transition probability matrix. An example was included to demostrate the effectiveness of the approach. Current work is investigating how to include performance criteria such as H_2 and H_∞ norms into the V-K iteration design method.

References

[1] M. Ait-Rami and L. El Ghaoui: *Robust State-feedback Stabilization of Jump Linear Systems via LMIs*, Proceedings IFAC Symposium on Robust Control, September 1994.

[2] M. Ait-Rami and L. El Ghaoui: *H_∞ State-feedback Control of Jump Linear Systems*, Proceedings IEEE Conference on Decision and Control, December 1995.

[3] D. Banjerdpongchai: *Parametric Robust Controller Synthesis using Linear Matrix Inequalities.* PhD dissertation, Stanford University, 1997.

[4] S. Boyd, L. El Ghaoui, E. Feron, V. Balakrishnan: *Linear Matrix Inequalities in System and Control Theory.* volume 15 of *Studies in Applied Mathematics.* SIAM, Philadelphia, PA, June 1994.

[5] H. Chan and Ü. Özgüner: *Optimal Control of Systems over a Communication Network with Queues via a Jump System Approach.* In Proc. IEEE Conference on Control Applications, 1995.

[6] O. L. V. Costa and M. D. Fragoso: *Stability Results for Discrete-Time Linear Systems with Markovian Jumping Parameters.* Journal of Mathematical Analysis and Applications **179**, 154-178, 1993.

[7] A. Graham: *Kronecker Products and Matrix Calculus: with Applications.* Ellis Horwoods Ltd., England, 1981.

[8] A. Hassibi, S. Boyd and J. P. How: *Control of Asynchronous Dynamical Systems with Rate Constraints on Events.* In Proc. IEEE Conf. on Decision and Control, 1999.

[9] Y. Ji and H. J. Chizeck: *Jump Linear Quadratic Gaussian Control: Steady-State Solution and Testable Conditions.* Control Theory and Advanced Technology, Vol. **6**, No. **3**, 289-319, 1990.

[10] R. Krtolica, Ü. Özgüner, H. Chan, H.Göktas, J. Winkelman and M. Liubakka: *Stability of Linear Feedback Systems with Random Communication Delays.* Int. Jour. Control, Vol. **59**, No. 4, 925-953, 1994.

[11] Johan Nilson: *Real-Time Control Systems with Delays.* PhD dissertation, Department of Automatic Control, Lund Institute of Technology, 1998.

[12] Shao-Po Wu and S. Boyd: *SDPSOL: A Parser/Solver for Semidefinite Programming and Determinant Maximization Problems with Matrix Structure. User's Guide. Version Beta.* Stanford University, June 1996.