STABILITY AND PERFORMANCE OF CONTROL SYSTEMS WITH LIMITED FEEDBACK INFORMATION

A Dissertation

Submitted to the Graduate School
of the University of Notre Dame
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

by

Qiang Ling, B.S., M.S.

Michael D. Lemmon, Director

Graduate Program in Electrical Engineering
Notre Dame, Indiana
May 2005

STABILITY AND PERFORMANCE OF CONTROL SYSTEMS WITH LIMITED FEEDBACK INFORMATION

Abstract

by

Qiang Ling

This thesis studies linear control systems with limited feedback information. The focus is on two types of limitations on the feedback information, dropout and quantization. By dropout, we mean that the desired feedback measurement is missed. By quantization, we mean the feedback measurement is described by a finite number of bits, which introduces "measurement error". This thesis analyzes the effect of dropout and quantization on stability and performance of control systems and develops synthesis methods that improve system performance in the face of such limited information.

We consider two dropout models, independent and identically distributed (i.i.d.) processes and Markov chains. For a control system with i.i.d. dropouts, we provide a necessary and sufficient stability condition and a closed-form expression for the output's power spectral density (PSD). Based on this PSD result, we identify an equivalent linear time-invariant (LTI) system which can be used to do synthesis. As an example, we design the optimal dropout compensator. For a control system with dropouts governed by a Markov chain, we provide a necessary and sufficient stability condition and a method to compute performance measured by the output's power. Based on the performance result, we propose a method

to design the optimal dropout policy which minimizes the degradation of system performance under the given dropout rate constraint. We extend the performance results on Markovian dropouts to distributed control systems.

For a quantized control system, we derive the minimum constant bit rate to guarantee stability. A dynamic bit assignment policy (DBAP) is proposed to achieve such minimum bit rate. We also study the performance of quantized systems. For a noise-free quantized system, we prove that DBAP is the optimal quantization policy with performance measured by the L-2 norm of the quantization error. For a quantized system with bounded noise, performance is measured by the ultimate upper bound of quantization error. We present both a lower bound and an upper bound on the optimal performance for quantized systems with bounded noise. The upper bound can always be achieved by the proposed DBAP. So DBAP is at least a sub-optimal quantization policy with the known performance gap.

CONTENTS

FIGURES

TABLES

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Professor Michael Lemmon, for his advisement, his encouragement and his support during my Ph.D. research. He has continuously provided me with enthusiasm, vision and wisdom. He is an outstanding researcher to have as a role model for a Ph.D. student.

I would like to thank my thesis committee members, Professor Panos Antsaklis, Professor Peter Bauer and Professor Sharon Hu, for their helpful comments.

I thank all my friends at Notre Dame, who helped make my time here enjoyable and memorable.

Finally I am indebted to my parents and family for always supporting and encouraging me.

This research was partially supported by the Army Research Office under Grant DAAG19-01-0743 and by the National Science Foundation under Grants NSF-CCR02-8537 and NSF-ECS02-25265. They are gratefully acknowledged.

CHAPTER 1

INTRODUCTION

A conventional sampled-date control system is shown in figure 1.1. That system has two feedback paths, the one from the sensor to the controller and the one from the controller to the sensor as shown in figure 1.1. The two feedback

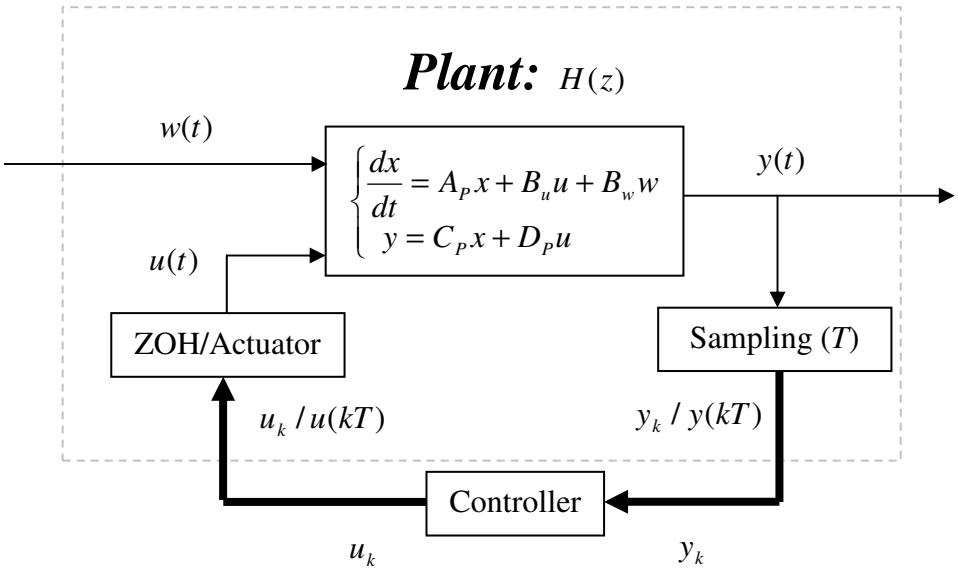


Figure 1.1. A sampled-data system

paths are assumed to be perfect in the sense that the controller knows the **exact** value of $\{y(kT)\}$ at sample instants $\{0, T, 2T, \cdots\}$ and the actuator also receives the control variable $\{u[kT]\}$ **without error** at the sample instants. *If the above feedback information is limited, i.e. imperfect, what will happen?*

This thesis considers two types of limitations on the feedback information; dropout and quantization. By dropout, we mean that the desired feedback measurement is not received by the actuator. For control systems, measurements that are delivered with long delays are usually useless for control purposes and are often dropped. By quantization, we mean that the feedback measurement is described by a finite number of bits. There is, of course, a *"measurement error"* due to the quantization of the feedback signal. The main goal of this thesis is to analyze the effect of dropouts and quantization on the stability and performance of control systems.

This introduction is divided into 3 parts. Section 1.1 discusses the motivation for studying control systems with limited feedback information. Section 1.2 reviews the related literature. Section 1.3 gives a summary of the chapters that follow.

1.1 Motivation

This section considers two classes of control systems with limited feedback information, computer-controlled systems (CCS) and networked control systems (NCS). They are presented below.

1.1.1 Computer-controlled systems

Before computer-controlled systems emerged in the 1950s, the controller in figure 1.1 was usually realized by hardware, e.g. an electrical network of resistors and capacitors. Such hardware controllers are difficult to design. It is also difficult to maintain these controllers. For example, if one wants to change a single parameter of the controller, the whole electrical network may have to be rebuilt because the resistors and capacitors are coupled together. When computers were introduced into control systems to act as controllers, we obtained a computer-controlled system (CCS) [3]. In a CCS, controllers are realized by software and their design and maintenance are greatly simplified, e.g. the control parameter change can be easily realized through setting a constant in a program. This benefit, however, is achieved at the cost of additional ADC (analog to digital conversion) and DAC (digital to analog conversion) and possible failure in accomplishing the control task.

Because the variables of the plant, $\{y(kT)\}$ and $\{u(kT)\}$, are continuous-valued and the variables of the computer are discrete-valued, we require an interface converting the continuous signal into the discrete signal (ADC) and vice versa (DAC). ADC is, in fact, quantization. When the precision of ADC is high, the quantization error may be neglected or treated as uniformly distributed noise [27]. But high precision ADC's (analog to digital converters) are usually expensive. One may, therefore, adopt a less expensive ADC with fewer quantization levels. The coarse quantization available on such an ADC can greatly degrade overal system performance. So it is important to quantify the performance degradation under a given quantization sensitivity.

In a CCS, the controller and the data transmission (from the sensor to the

controller and from the controller to the actuator) are realized through processes executed by a central processing unit (CPU). If the CPU is too busy, some processes are not executed before their deadlines so that some expected variables are not available, i.e. dropped. One may use a computer dedicated to a CCS to avoid such process failures. But such dedicated hardware may not be an acceptable design due to economic or power considerations. Another promising choice is to compute the performance degradation due to process failure. If the degradation is acceptable, no extra computation (CPU) cost will be needed.

From the above arguments, we see that a CCS is a system whose performance is limited by dropped and quantized data in the feedback path. By analyzing the effect of dropout and quantization on system performance, we may put the *appropriate* requirements on ADC converter and CPU that result in cost-effective design meeting customer expectations.

1.1.2 Networked control systems

In recent years there has been considerable interest in networked control systems (NCS) [7] [65] [70]. *What is an NCS?* Briefly speaking an NCS is a sampled-data system whose feedback loops are closed over non-deterministic digital communication networks. NCS has many advantages, such as efficient resource sharing, easy maintenance, high system reliability and lower price [7].

In order to give a clear picture of an NCS, we compare the NCS in figure 1.2 to the conventional sampled-data system in figure 1.1. The main difference between the NCS and the conventional sampled-data system is the introduction of the communication part, which includes the network and the related converters. In the following, we briefly describe the mechanism by which an NCS works.

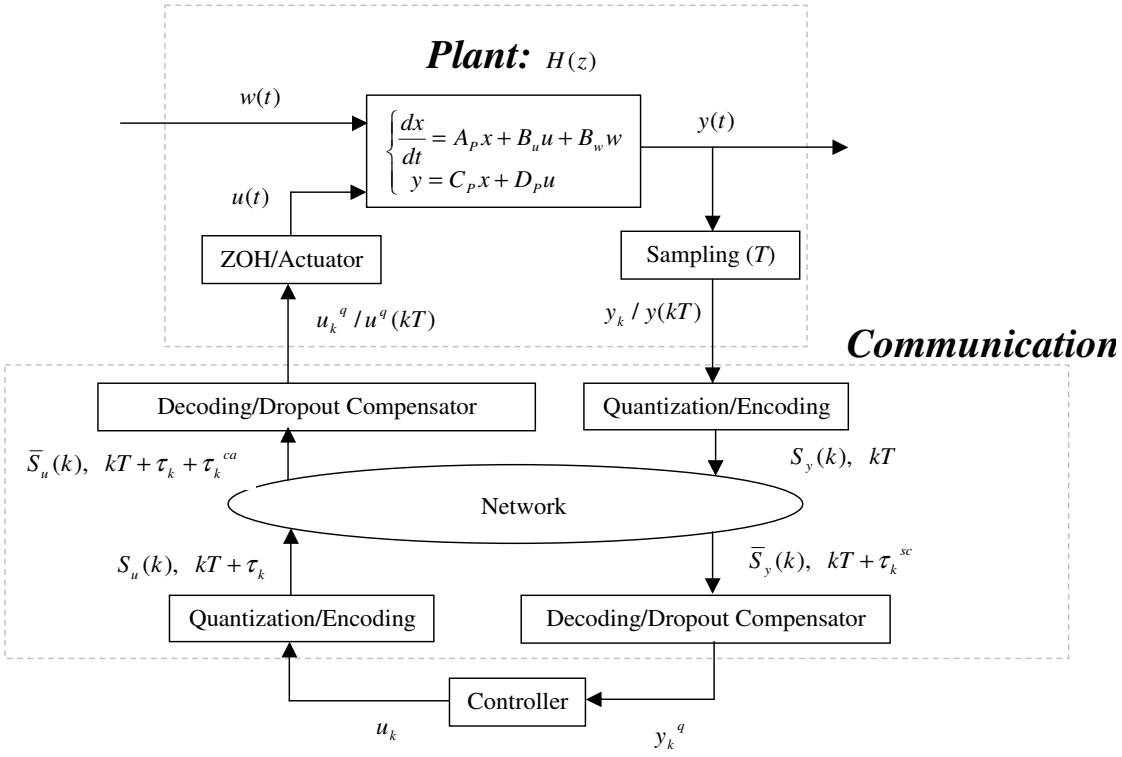


Figure 1.2. A Networked Control System

Because the network is digital with finite bandwidth and the "currency" of the network is the *bit*, any data must be converted into bits before transmission. y_k, as a continuous-valued signal, must be quantized into a discrete-valued signal. There are a finite number of such discrete values. The encoder maps every discrete value to a unique bit sequence of finite length. Such a bit sequence is often called a symbol. $S_y(k)$ is the symbol produced at time k for the value y_k. Quantization and encoding are usually jointly carried out, so we view them as a whole.

$S_y(k)$ is sent to the controller through a network at time kT. Because the network is non-deterministic (unreliable), the received symbol $\overline{S}_y(k)$ may be different from the original symbol $S_y(k)$ and there may be a delay of τ_k^{sc} in the symbol's

transmission. By the decoding policy, $\overline{S}_y(k)$ is converted into y_k^q, which is an estimate of y_k with a delay of τ_k^{sc}. Besides delay, dropout is also an important property of the NCS. Some symbols may be dropped by the the network due to transmission error. Some greatly delayed symbols are also treated as dropouts because in control systems, data whose delivery is greatly delayed may be more harmful to system stability than receiving no data at all [55]. In the event of dropouts, no symbol is received by the decoder at the controller side, but the controller still needs an input. A dropout compensator may therefore be used to generate an estimate $y^q(k)$ based on the previously received data. We, therefore, combine decoding and dropout compensation together to generate $y^q(k)$.

With the estimate $y^q(k)$, the controller produces a control u_k. u_k is usually a continuous-valued signal. In order to reach the ZOH (zero-order hold)/actuator through the network, u_k has to follow the same coding/decoding procedure as y_k, endures quantization error, transmission delay τ_k^{ca} and possible dropouts. So what the ZOH/actuator receives is only u_k^q, an estimate of u_k with the delay of τ_k^{ca}. u_k^q is a discrete-time signal. In order to control a continuous-time plant, u_k^q is usually converted by a ZOH. An actuator may be required for power amplification. So we view the ZOH and the actuator as a whole.

From the above description of the NCS, we see that the communication link introduces errors between the measured and true feedback signal. These errors are due to

1. **Dropout**.

2. **Delay:** $\tau_k^{sc} > 0$ and $\tau_k^{ca} > 0$.

3. **Quantization error:** Due to the finite network bandwidth.

Among the non-ideal feedback limitations listed above, we focus on dropout and quantization. Because of dropout and quantization, the system suffers from estimation error, i.e. $y_k \neq y_k^q$ and/or $u_k \neq u_k^q$ which may degrade system performance. It is critical to quantitatively analyze the performance degradation due to dropout and quantization. With the achieved analysis results, we may reduce the system resources, such as network bandwidth and the network's quality of service required to achieved a desired performance level.

1.2 Literature review

As mentioned before, this thesis focuses on dropout and quantization in control systems with limited feedback information. For a control system with feedback dropouts, if a measurement packet is dropped, then the system works in the open loop; otherwise, the feedback loop is closed. The system therefore jumps between two configurations: open and closed loops. We can, therefore, find a tight relationship between such systems and *jump linear systems*, which caught much interest in the early 1990s. In this section, we first review the literature on jump linear systems. Second we review the recent progress on network control systems, a class of control systems with limited feedback information. At the end, we give a brief review on quantized control systems.

1.2.1 Related work on Jump Linear Systems

A jump linear system can be modeled as follows [36]:

$$\begin{cases} x_{k+1} &= A(r_k)x_k + B(r_k)u_k \\ P(r_{k+1} = j | r_k = i) &= p_{ij} \end{cases} \tag{1.2.1}$$

where $x_k \in R^n$ is the state, $u_k \in R^m$ is the input, and $\{r_k\}$ is a Markov chain taking values in a finite set $S = \{1, 2, \cdots, s\}$ with the transition probabilities defined in eq. 1.2.1 where $p_{i,j}$ stands for the probability of a transition from state i to state j. $A(r_k)$ and $B(r_k)$ are functions of r_k. When $r_k = r_i$ $(i \in S)$, $A(r_k) = A_i$ and $B(r_k) = B_i$ where A_i and B_i are specified real-valued matrices with appropriate dimensions.

In [36] three types of second moment stability are defined for the jump linear system in eq. 1.2.1.

Definition 1.2.1 *For the jump linear system in eq. 1.2.1 with $u_k = 0$, the equilibrium point 0 is* [1]

1. *stochastically stable, if for every initial state (x_0, r_0)*

$$\mathbf{E}\left[\sum_{k=0}^{\infty} \|x_k(x_0, r_0)\|^2 \quad | \quad x_0, r_0\right] < \infty, \tag{1.2.2}$$

   where $\|\cdot\|$ denotes Euclidean norm of a vector.

2. *mean square stable, if for every initial state (x_0, r_0)*

$$\lim_{k \longrightarrow \infty} \mathbf{E}\left[\|x_k(x_0, r_0)\|^2 \quad | \quad x_0, r_0\right] = 0, \tag{1.2.3}$$

3. *exponentially mean square stable, if for every initial state (x_0, r_0), there exist constants $0 < \alpha < 1$ and $\beta > 0$ such that for all $k \geq 0$*

$$\mathbf{E}\left[\|x_k(x_0, r_0)\|^2 \quad | \quad x_0, r_0\right] < \beta \alpha^k \|x_0\|^2, \tag{1.2.4}$$

   where α and β are independent of x_0 and r_0.

[1] $x_k(x_0, r_0)$ denotes the state at time k for initial conditions x_0 and r_0.

In [36], the above 3 types of stability are proven to be equivalent. So we can focus our study on mean square stability without loss of generality. In [36], a necessary and sufficient condition for mean square stability is provided.

Theorem 1.2.1 *([36]) The jump linear system in eq. 1.2.1 with $u_k = 0$ is mean square stable, if and only if, for any given set of symmetric matrices $\{W_i > 0 : i \in S\}$, the following coupled matrix equations have positive definite solutions $\{M_i\}$.*

$$\sum_{j=1}^{s} p_{ij} A_i^T M_j A_i - M_i = -W_i \tag{1.2.5}$$

Remark: Although the above condition is necessary and sufficient, equation 1.2.5 must be verified having a solution $\{M_i\}_{i=1,\cdots,s}$ for arbitrary positive matrices $\{W_i\}_{i=1,\cdots,s}$. Fortunately the above stability condition can be reformulated into the stability of a compound matrix, which is much easier to test.

Theorem 1.2.2 *The jump linear system in eq. 1.2.1 with $u_k = 0$ is mean square stable, if and only if the following matrix $A_{[2]}$ has all eigenvalues within the unit circle.*

$$A_{[2]} = diag\left(A_i \otimes A_i\right)_{i=1,2,\cdots,s} \left(Q^T \otimes I_{n^2}\right) \tag{1.2.6}$$

where $Q = (p_{ij})_{s\times s}$ is the probability transition matrix of the Markov chain $\{r_k\}$.

In [51], the above stability condition was extended to continuous-time jump linear systems. Although the condition was claimed to be necessary and sufficient in [51], only sufficiency has actually been proven. It is [17] that rigorously proves the necessity for the first time [23]. We came up with a parallel proof of necessity. Compared to the one in [17], our proof is more direct [15]. We present our proof in chapter 3.

The previous literature [34] has studied controllability and stabilizability of jump linear systems as well as stability. Performance problem, particularly the linear quadratic Gaussian (LQG) problem, was also studied for jump linear systems [35] [51].

Theorem 1.2.3 *([35], [51]) The LQG controller for continuous-time jump linear systems is the combination of a linear quadratic regulator (LQR) and a Kalman state filter, i.e. the separation principle is satisfied.*

The above theorem can be extended to discrete-time cases. In [55], the above optimal control law is verified for a special class of discrete jump linear systems.

1.2.2 Related work on the limited feedback information of networked control systems

There are four major types of non-ideal properties of feedback information in an NCS, including sampling, delay, dropout and quantization. The discussion on quantization is moved into subsection 1.2.3. Among the first three types of non-ideality, dropout is our main focus. We, however, still review the results on sampling and delay so that a whole picture on an NCS can be presented.

1.2.2.1 Sampling

Because of sampling, the feedback information is available *only* at sampling instants. Compared to the continuous-time feedback systems where the measurement is always available, sampling may be treated as a kind of temporal limitation of feedback information. Although there are some non-periodic sampling methods [4], the main sampling methods usually assume a periodic form. Sampling period T is a kind of measure of the attention a system catches [13], or how limited

the feedback imformation is. Larger T may mean less resource occupation and is more desirable. Much effort was made to increase T on the condition that the performance degradation is acceptable.

In [66], a continuous-time plant is controlled over a network feedback loop. At time instants t_i, the measurement $y(t_i)$ is sent so that $\hat{y}(t_i) = y(t_i)$, where $\hat{y}(t)$ is the input to the controller, i.e. the controller computes the control variable from $\hat{y}(t)$. during the transmission intervals, zero order hold is used, i.e. $\hat{y}(t) = \hat{y}(t_i)$ for $t \in [t_i, t_{i+1})$. The transmission intervals are bounded from above, i.e. $t_{i+1} - t_i \leq \tau$. The NCS is viewed as an approximation of the directly linked continuous-time system, whose approximation error is $\hat{y}(t) - y(t) \neq 0$. It was proven that when τ is small enough, the stability of the NCS can be guaranteed [66]. The maximum allowable transfer interval (MATI), i.e. the upper bound of τ, was derived. This MATI is a function of the parameters of the NCS [66]. It deserves special attention that the transmission instants are not required to be equal-distance in [66].

In [52], a special sample-hold function is used for the system shown in figure 1.3. $\hat{x}(kh) = x(kh)$ at time instants kh $(k = 0, 1, 2, \cdots)$. During the update intervals $(t \in (kh, (k+1)h))$, $\hat{x}(t)$ is governed by the state estimator dynamics.

$$\begin{cases} \dot{\hat{x}} &= \hat{A}\hat{x} + \hat{B}u \\ u(t) &= K\hat{x}(t) \\ \hat{x}(kh) &= x(kh) \end{cases} \tag{1.2.7}$$

where K is the state feedback gain. Because $\hat{A}$ and $\hat{B}$ may not perfectly match A and B, the sampling period h cannot be infinite. The maximum allowable sampling periond, h is derived [52]. The work was extended to output feedback NCS by using a state estimator [52]. It was also applied to the NCS with delay

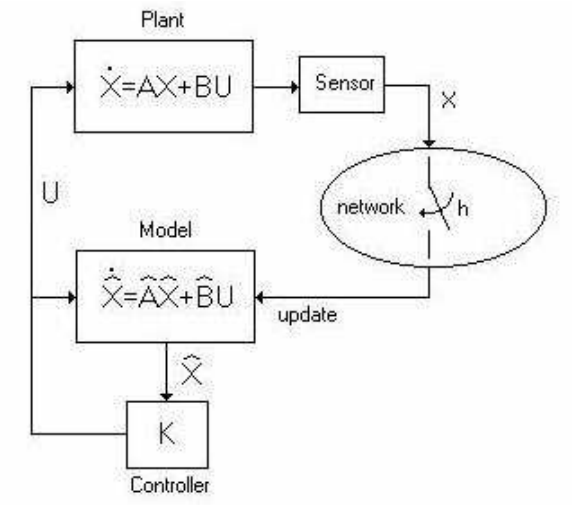


Figure 1.3. A model-based Networked Control System [52]

by constructing state predictor based on the known delay [52].

1.2.2.2 Dropout and delay

Dropout means there is no expected measurement at some sampling instants, i.e. the feedback information is lost. So dropout is a kind of temporal limitation of feedback information. Delay means that the feedback information is received a little later than the expected instants. But there is no information loss in the delayed measurement. So it may not be appropriate to treat delay as a limitation of feedback information and this thesis doesn't consider delay under the main goal of *"limited feedback information"*. The reason to mention delay here is that the literature on dropout usually also studies delay and the methods to treat dropout usually follow the same routines for delay. For completeness, we review dropout and delay as a whole.

The network delay and/or network dropout can be governed by a stochastic model, which includes Markov chains and i.i.d. (independent identically dis-

tributed) processes.

In [55], two models of the time-varying network delay are studied: Markov chain and i.i.d. process. The NCS with network delay takes the following form.

$$\begin{cases} x_{k+1} &= Ax_k + \Gamma_0(\tau_k)u_k + \Gamma_1(\tau_k)u_{k-1} + v_k \\ y_k &= Cx_k + w_k \end{cases} \tag{1.2.8}$$

where $x_k \in R^n$, $u_k \in R^m$, $v_k \in R^n$, $y_k \in R^l$ and $w_k \in R^l$. τ_k is the network delay, modeled by a Markov chain or an i.i.d. process. u_k is the control to be designed. v_k and w_k are uncorrelated Gaussian white noise with zero mean. $\Gamma_0(\tau_k)$ and $\Gamma_1(\tau_k)$ are the input matrices determined by the delay τ_k. Under the specified stochastic delay model, the performance defined in a quadratic form, $\mathbf{E}\left[x_k^T S_1 1 x_k + u_k^T S_2 2 u_k\right]$, was computed. Furthermore, an optimal LQG controller was designed, which is a combination of the Kalman filter and the optimal state feedback controller.

In [55], dropout is specially considered, where dropout is called *vacant sampling*. Dropout is modeled as a Markov chain with two states, *"dropout"* and *"no dropout"*. The dynamics equation with dropout is similar to eq. 1.2.8. Two heuristic compensation strategies in event of dropouts are proposed: reusing latest control, i.e. $u_k = u_{k-1}$, and constructing control from the estimate of the dropped measurement. Examples were used to compare the two strategies.

In [28], a first order NCS with both input and output noise is studied. In the NCS, the feedback measurement is dropped in an i.i.d. fashion. In the event of dropouts, the control will be set to 0. The work presented a necessary and sufficient condition for mean square stability and found the optimal feedback gain to minimize the output power. In [59], a necessary and sufficient stability condi-

tion of an NCS with i.i.d. dropouts takes an LMI form and stabilizability with constant state feedback gain was discussed. A Matlab toolbox: Jitterbug [42] was developed, which can numerically compute both the output power and the output's power spectral density under the given network delay and/or dropout modeled as a Markov chain.

By assuming stochatic models on the NCS, aggressive results can be obtained. For example, necessary and sufficient stability conditions can be established. It is, however, always disputable whether the stochastic models are practical. So some researchers choose weaker a weaker class of models which bound the dropout rate and the network delay from above.

In [70], the NCS with dropout is studied. It replaces the true switched system with an "averaged system" and then provides some sufficient stability conditions on the system. In the stability condition, only average dropout rate is used and there is no assumption on the dropout pattern. So the results are quite general, though, at the cost of more conservatism. [70] also gave a good start on network scheduling of the NCSs. A network is shared among a group of NCSs. It was shown that the sampling periods of the NCSs can be chosen in some optimal way to obtain best performance and satisfy the network sharing constraint (all packets can be successfully transmitted). When the network is overloaded, i.e. the sharing constraint cannot be satisfied, the stability of all the NCSs may still be guaranteed by dropping some packets. The dropout policy design was demonstrated through examples

In [6], discrete network delay is considered, which is really dropout. In the event of dropouts, the latest control is reused. In that work, the number of consecutive dropouts at time k, d_k, is bounded, i.e. $d_k \leq M$. Then the control

takes the following form.

$$
\begin{aligned}
u_k &= c_0(k)\hat{u}_k + c_1(k)\hat{u}_{k-1} + \cdots + c_M\hat{u}_{k-M} \\
c_i(k) &= \begin{cases} 1, & d_k = i \\ 0, & d_k \neq i \end{cases}
\end{aligned}
$$

where u_{k-l} is the control variable designed for $d_k = l$. If one substitutes the above control law into the system dynamics equation, e.g. an ARMA (autoregressive moving average) model, we obtain a dynamic system with time-varying parameters $c_i(n)$. So the system with uncertain dropouts is converted into a system with uncertain parameters. Based on results from robust control theory, a sufficient condition for the asymptotic stability of the NCS with discrete network delay (dropout) can be established [6].

1.2.3 Related work on quantization

Because of quantization, the received feedback measurement is different from the one originally sent. Quantization can be treated as a kind of spatial limitation of feedback information. The literature review here briefly goes through the history of quantization. A detailed (and technical) review will be found in later chapters.

Quantization is an old topic. At the beginning of computer-controlled systems, quantization is treated as ADC (analog-digital) conversion. The quantization (ADC) policies at that time are usually static, i.e. the quantization result depends only on the current measurement input. The major advantage of static quantization policies is the simplicity of their coding/decoding schemes. They may be realized through hardware, e.g. an ADC converter. Such quantization policies

usually choose uniform sensitivity. Their weakness lies in possible overflow and underflow, i.e. the distinguishable measurement magnitude is bounded both from below and from above due to the finiteness of the number of quantization levels. It was proven that static policies with a finite number of quantization levels cannot achieve asymptotic stability [19]. The best thing to expect with a finite number of quantization levels is locally practical stability (i.e. states converge into a bounded set under "good" initial condition) [69] [5] [21]. The aforementioned static uniform quantization policy processes all possible measurement with the same sensitivity. This may result in significant relative quantization error for measurements with small magnitude and may therefore degrade the system performance. This effect can be magnified by an example with an infinite number of quantization levels, where constant uniform sensitivity is enforced everywhere, particularly around the origin and asymptotic stability cannot be obtained [19].

After digital communication networks are introduced into control systems (networked control systems), quantization, named as network quantization here, caught more and more attention because all variables sent over such networks have to be quantized before transmission. Network quantization may be realized through software with a powerful processor and the quantization complexity is no longer a bottleneck. One may therefore pursue more complex quantization policies for better performance. One choice is static logarithmic quantization policies, where the quantization levels are marked in a logarithmic scale. Such logarithmic policies can guarantee almost constant relative quantization error. When an infinite number of quantization levels are available, asymptotic stability under static logarithmic quantization can be obtained [25] [33]. It was shown in [20] that the logarithmic quantizer is the least dense among all static quantizers to guarantee

asymptotic stability. When there is only a finite number of bits, it is still *impossible* to achieve asymptotic stability with the static logarithmic quantization policies.

Based on the above arguments, one may ask *whether asymptotic stability can be obtained with a finite number of bits?* An **affirmative** answer is given in [14]. The creativity in [14] is to choose a dynamic quantization range. In order to illustrate the dynamic procedure in [14], we compare it against a static quantization policy with static quantization range P_0 [2]. If the state is known to lie in a set $P[k]$ at time k, then the system dynamics and the control law are used to predict the state range $P[k+1]$ at time $k+1$. $P[k+1]$ must be a subset of P_0; otherwise an overflow happens. Under the static policy, P_0 is partitioned with the given number of bits. Under the dynamic policy in [14], $P[k+1]$ is partitioned with the same number of bits. It is usually the case that $P[k+1]$ is much smaller than P_0. So smaller quantization errors can be expected under the dynamic policy. It is further proven that asymptotic stability can be achieved if the number of quantization levels is above a threshold, i.e. a sufficient stability bound on the bit rate is achieved.

A sufficient stability condition is established in [14]. In order to save bandwidth, one pursues the minimum bit rate for stability, i.e. a necessary and sufficient stability condition. Following the dynamic quantization range idea in [14], Tatikonda obtained the desired minimum bit rate for asymptotic stability [61] [62]. In [53], stability is interpreted in the stochastic sense, i.e. a certain moment of state converges to 0. The same minimum bit rate as the one in [62] is proven to guarantee stochastic stability. The minimum bit rate in [62] [53] is achieved through a time-varying bit-rate configuration. Real networks, however, prefer to constant bit rates under bandwidth and power constraints [29]. In [48], the con-

[2]The quantization range of a static policy is a constant set, denoted as P_0.

stant bit rate constraint is considered and the minimum bit rate for asymptotic stability is derived.

In the above paragraphs, noise-free quantized systems are assumed and stability is characterized as asymptotic stability. If the quantized systems are perturbed by bounded noise, stability should be interpreted as BIBO (bounded-input-bounded-output) stability. The minimum bit rate for quantized systems with bounded noise has been derived for both time-varying bit rate configuration [62] and constant bit rate configuration [45].

Based on the previous minimum bit rate results, we know that the quantized control system is stable (either asymptotically stable or BIBO stable) if an enough number of bits are available. For a stable quantized system, we may further require a minimum level of performance [3]. More specifically we pursue the quantization policy which yields the best performance. In [63], the perturbation noise is assumed to be Gaussian and the performance is defined as a linear quadratic form of the system state. It is shown that the optimal performance decomposes into two terms. One is determined by the perfect (without quantization) state feedback. The other comes from the quantization error. The optimal quantization policy is the one which minimizes the quantization term in the optimal performance equation. Although the results in [63] are intuitively pleasing, they are less realistic because of their estimation policies. In [63], the state is estimated as the conditional mean based on the received bit squence. It is not trivial to compute such conditional mean because the quantization operation is non-linear. It may therefore difficult to find a recursive way of computing the mean. The only possible way is to compute it with the probability density function (pdf) of the state. So the state's pdf has to be updated at every step which may be unrealistically

[3]There are various performance indexes which will be specified late.

complicated. In [40], a scalar quantized system is studied, the input noise is assumed to be bounded and performance is measured by the eventual upper bound of quantization error. The relationship between the number of quantization levels and the packet dropout pattern is established. Note that a simple estimation policy is chosen in [40] and the proposed quantization policy is realistic. The results in [40] are extended to 2-dimensional systems in [46] and [47]. In [22], a noise-free quantized system is considered and the convergence rate of the quantization error is used to measure system performance. It proves that the quantization policy proposed in [48] achieves the best (fastest) convergence rate.

Up to now, we have reviewed the literature on quantized control systems. In fact, quantization is also a hot topic in the information theory community [27]. The idea of quantization in that community is to transmit a random process through a channel with finite bandwidth. The configuration is similar to ours. We, however, cannot apply those results directly to our quantized control systems. The major reason is **"feedback"**. In the information theory community, the source of quantization may be a Guassian process, which is independent of the quantization error. The quantization error is used only for performance evaluation. In the quantized control community, the source of quantization is usually the system's state (or output). The quantized state is fed back for control purpose. Specifically we consider the following quantized control system.

$$\begin{cases} x[k+1] &= Ax[k] + Bu[k] \\ u[k] &= Fx^q[k] \end{cases} \tag{1.2.9}$$

where $x[k] \in R^n$ is the state, $x^q[k]$ is the quantized version of $x[k]$. If we treat the quantized state as the summation of the true state and the quantization error,

we can say that the quantization error is fed back. If the quantization error is large, the control will be less desirable and the range of state will be enlarged so that the quantization error will be increased. Note that we ususally have a fixed number of quantization bits. If the quantization range is large, the quantization error is also large; vice versa. If the number of available bits is too small, the above "positive" feedback, i.e. large quantization error $\longrightarrow$ large state range $\longrightarrow$ larger quantization error $\longrightarrow$ ..., may destabilize the whole system.

1.3 Summary of thesis

This thesis focuses on two types of limitations on feedback information, dropout and quantization. It is organized in the following way.

- In chapter 2, we consider a linear control system with i.i.d. (independent identically distributed) dropouts. We study the stability condition, derive the closed-form expression for the output's power spectral density (PSD), propose an equivalent linear time-invariant (LTI) system that has the same stability condition and the same output PSD as the original time-varying system with dropouts. Therefore the synthesis of the original system can be performed for the equivalent LTI system. As an example, we show a dropout compensator design, which is formulated into an optimization problem among a group of LQG controllers.

- In chapter 3, we study a linear control system with dropouts governed by a Markov chain. We provide a necessary and sufficient stability condition and compute its performance characterized by the noise attenuation property. Based on the achieved performance results, we design the optimal dropout policy under an average dropout rate constraint. The obtained op-

timal dropout policy is enforced on a real-time control system as a novel QoS (quality of service) constraint. Simulations show that the scheduling algorithms considering the optimal dropout policies well outperform the others. The performance computation and optimal dropout policy results are also extended to distributed control systems.

- In chapter 4, we study the stability of a quantized control system under the constant bit rate constraint. Minimum bit rate is obtained for both noise-free quantized systems (for asymptotic stability) and quantized systems with bounded noise (for BIBO stability). The minimum bit rate results based on state feedback are extended to the output feedback cases.

- In chapter 5, we study the performance of a 2-dimensional quantized control system under the constant bit rate constraint. The performance is measured by the quantization error. The main goal is to find the quantization policy that optimizes such performance. For noise-free quantized systems, we obtain the optimal quantization policy, which is a modified version of the policy in [48], named DBAP (dynamic bit assignment policy). For quantized systems with bounded noise, we derive both upper and lower bounds on the optimal performance. The upper bound can be achieved with DBAP. The lower bound provides a threshold for achievable performance under a given number of bits. If the performance is required to be better than some level, we can predict the required number of bits based on the upper bound. All performance results may be extended to multiple-dimensional diagonalizable quantized systems.

- In chapter 6, we summarize the achieved results.

- In appendix A, we present mathematical preliminaries on stochastic processes and Kronecker product, which may be used by Chapters 2 and 3.

CHAPTER 2

STABILITY AND PERFORMANCE OF CONTROL SYSTEMS WITH I.I.D. FEEDBACK DROPOUTS

This chapter studies linear control systems with the feedback information dropped in an i.i.d. (independent identically distributed) fashion. It is organized as follows. Section 2.1 presents a mathematical model for such systems. Section 2.2 provides a necessary and sufficient condition for stability and wide sense stationarity of such systems. The main result, a closed-form expression of the output's power spectral density (PSD), is presented in section 2.3. Based on the PSD results, a linear time-invariant (LTI) system is obtained in section 2.4. That LTI system is equivalent to the original time-varying system in the sense of the same stability condition and the same output's PSD. This equivalent system is easy to deal with and can be used to do synthesis for the original system. As an example, section 2.5 designs the optimal dropout compensator for the equivalent system. Section 2.6 places some concluding remarks. To improve readability, all proofs are moved to the appendix, section 2.7.

2.1 System model

We consider a single-input-single-output (SISO) control system with i.i.d. dropouts in figure 2.1. The system has two inputs, w and d. w is white noise

with zero mean and unit variance. d is an i.i.d. binary process with the distribution of

$$P(d[k]=1)=\varepsilon, P(d[k]=0)=1-\varepsilon \tag{2.1.1}$$

where ε is the dropout rate. The loop function $L(z)$ is strictly proper. The output signal y drives a data dropout model:

$$\overline{y}[k]=\begin{cases} y[k], & d[k]=0 \\ \hat{y}[k], & d[k]=1 \end{cases} \tag{2.1.2}$$

where $\{\hat{y}[k]\}$ is the output of the dropout compensator $F(z)$. $F(z)$ is assumed to be strictly proper. The control signal of $L(z)$ is assumed to be $u[k]=w[k]+\overline{y}[k]$. The output power, $\mathbf{E}\left[y^2\right]$ [1] , is taken as the control performance. Such performance characterizes the noise attenuation of y with respect to w and d.

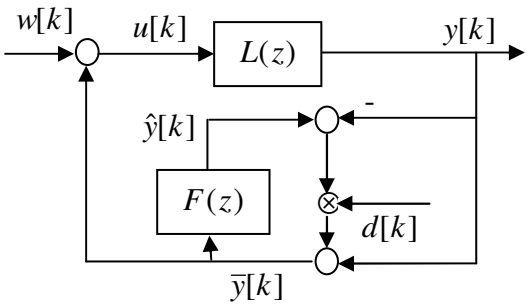


Figure 2.1. A control system with i.i.d. dropouts and dropout compensation

[1]It will be proven in theorem 2.2.2 that the process $\{y[k]\}$ is wide sense stationary. Based on the definition of wide sense stationarity in appendix A, we know $\mathbf{E}\left[y^2[k]\right]$ is constant with respect to k. So the time index k is dropped in the output power.

Based on the above model, we know the system will jump in an i.i.d. fashion between two configurations: open loop and closed loop. So the state space equation of the system takes the following form:

$$\begin{cases} x[k+1] &= A[k]x[k] + Bw[k] \\ y[k] &= Cx[k] \end{cases} \tag{2.1.3}$$

where $P(A[k] = A_{close}) = 1 - \varepsilon$, $P(A[k] = A_{open}) = \varepsilon$. $A_{close}, A_{open}, B, C$ are system parameters, which come from $L(z)$ and $F(z)$. Assume that $L(z)$ and $F(z)$ have state space realizations, $L(z) \overset{s}{=} \left[\begin{array}{c|c} A_h & B_h \\ \hline C_h & 0 \end{array}\right]$ and $F(z) \overset{s}{=} \left[\begin{array}{c|c} A_f & B_f \\ \hline C_f & 0 \end{array}\right]$, then $A_{close} = \begin{bmatrix} A_h + B_h C_h & 0 \\ B_f C_h & A_f \end{bmatrix}, A_{open} = \begin{bmatrix} A_h & B_h C_f \\ 0 & A_f + B_f C_f \end{bmatrix}, B = \begin{bmatrix} B_h \\ 0 \end{bmatrix}, C = \begin{bmatrix} C_h & 0 \end{bmatrix}$.

Throughout this chapter, the following assumptions are taken:

1. w is a white input noise with zero mean and unit variance.
2. The disturbance process $\{w[k]\}$ is independent from the dropout process $\{d[k]\}$.
3. The initial time of the system is $-\infty$.

2.2 Stability and wide sense stationarity

For a control system, stability is the most important property. We establish the following stability condition for the system in eq. 2.1.3.

Theorem 2.2.1 *The system in equation 2.1.3 is mean square stable if and only*

if $A_{[2]} = (1-\varepsilon)A_{close} \otimes A_{close} + \varepsilon A_{open} \otimes A_{open}$ *has all eigenvalues within the unit circle, where* $\otimes$ *is Kronecker product defined in Appendix A.*

Theorem 2.2.1 is a special case of Theorem 3.2.1 where dropouts are governed by a Markov chain. So the proof of Theorem 2.2.1 can be obtained directly from Theorem 3.2.1 which is presented in Chapter 3.

If a system is mean square stable, we know the initial condition will be eventually forgotten. By assumption 3, the initial time is $-\infty$. Therefore we can assume a zero initial state without loss of generality. Thus the output $y[k]$ will be uniquely determined by the input noise sequence $\{w[k]\}$. Because of the stationarity of $\{w[k]\}$ ($\{w[k]\}$ is white), $\{y[k]\}$ is expected to be wide sense stationary too (the definition of wide sense stationarity is presented in Appendix A).

Theorem 2.2.2 *All linear outputs of the system in equation 2.1.3, i.e. the output with the form of* $z[k] = Ex[k] + Fw[k]$*, are wide sense stationary if* $A_{[2]} = (1-\varepsilon)A_{close} \otimes A_{close} + \varepsilon A_{open} \otimes A_{open}$ *has all eigenvalues within the unit circle.*

Again the stationarity in theorem 2.2.2 is a special case of the stationarity in theorem 3.2.2 (presented in Chapter 3) where Markovian dropouts are considered. So we omit the proof of theorem 2.2.2.

It can be seen that the system is wide sense stationary if it is mean square stable by theorems 2.2.1 and 2.2.2. Throughout this chapter, we will assume the system is mean square stable. So the output $\{y[k]\}$ is wide sense stationary and we can compute the power spectral densities and cross spectral densities of the system with i.i.d. dropouts.

2.3 Power spectral density

This section states a closed-form expression for the output's power spectral density (PSD). The system's output power is then computed from the PSD. Simulation results are used to verify the correctness of our predicted system's output power. The main result is stated below in theorem 2.3.1. The theorem is proven in the appendix (section 2.7).

Theorem 2.3.1 *Consider the system in equation 2.1.3. Let* $\tilde{y}[k] = y[k] - \overline{y}[k]$. *If the system is mean square stable, the power spectral densities and the ouput's power can be computed as*

$$\begin{cases} S_{yy}(z) &= \left|\frac{1}{1-D(z)L(z)}\right|^2 |L(z)|^2 S_{ww}(z) + \left|\frac{D(z)L(z)}{1-D(z)L(z)}\right|^2 \frac{\Delta}{1-\varepsilon} \\ S_{\tilde{y}\tilde{y}}(z) &= \left|\frac{L(z)(D(z)-1)}{1-D(z)L(z)}\right|^2 S_{ww}(z) + \left|\frac{D(z)(1-L(z))}{1-D(z)L(z)}\right|^2 \frac{\Delta}{1-\varepsilon}, \\ \mathbf{E}\left[y^2\right] &= \frac{1}{2\pi}\int_{-\pi}^{\pi} S_{yy}\left(e^{j\omega}\right) d\omega \end{cases} \tag{2.3.4}$$

where $|\cdot|$ *means magnitude,* $D(z) = \frac{1-\varepsilon}{1-\varepsilon F(z)}$. *When* $\varepsilon = 0$, $\Delta = 0$; *when* $\varepsilon > 0$, Δ *is the unique positive solution of the following equation*

$$\begin{aligned} \Delta = \quad & \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|\frac{L(e^{j\omega})(D(e^{j\omega})-1)}{1-D(e^{j\omega})L(e^{j\omega})}\right|^2 S_{ww}(e^{j\omega}) d\omega \\ & + \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|\frac{D(e^{j\omega})(1-L(e^{j\omega}))}{1-D(e^{j\omega})L(e^{j\omega})}\right|^2 d\omega \frac{1}{1-\varepsilon}\Delta \end{aligned} \tag{2.3.5}$$

Remark: Δ is an important parameter in the the output's PSD expression in eq. 2.3.4. The proof of theorem 2.3.1 shows that Δ corresponds to the reconstruction error $\tilde{y}[k] = y[k] - \overline{y}[k]$.

$$\begin{aligned} \Delta &= \mathbf{E}\left[\left(y[k] - \overline{y}[k]\right)^2\right] \\ &= \varepsilon\left[\left(y[k] - \hat{y}[k]\right)^2\right] \end{aligned} \tag{2.3.6}$$

where the second equality comes from the fact that $\overline{y}[k] = y[k]$ for $d[k] = 0$ and $\overline{y}[k] = \hat{y}[k]$ for $d[k] = 1$ and $P(d[k] = 1) = \varepsilon$. By eq. 2.3.6, Δ is determined by the compensation error $y[k] - \hat{y}[k]$.

Remark: The power spectral density $S_{yy}(z)$ in equation 2.3.4 consists of two terms. The first term is the usual term we would expect to see if a wide sense stationary process w were driving a linear time-invariant (LTI) system with a feedback gain $D(z)$. The second term in equation 2.3.4 models the explicit effect the dropout process d has on the system's output.

Theorem 2.3.1 can be used to make quantitative predictions about the system performance as a function of the dropout rate ε. The following example illustrates such a prediction. Consider a feedback control system with an unstable loop function $L(z) = \frac{z+0.8}{z^2+z+1.7}$ and a dropout compensator $F(z) = \frac{1}{z}$. This dropout compensator simply reuses the latest feedback measurement when a dropout occurs. The input noise, w, is white Gaussian with zero mean and unit variance. By theorem 2.2.1, the system is mean square stable for dropout rates below 7.0%. A `Matlab simulink` model was written to simulate this system. The simulations experimentally estimated the output power using the time average

$$\hat{\mathbf{E}}_L\left[y^2\right] = \frac{1}{L}\left(y^2[L+1] + y^2[L+2] + \cdots + y^2[L+L]\right), \tag{2.3.7}$$

where L is half of the length of the simulation run. By the ergodicity theorem (theorem 3.3.2) in chapter 3, we know that when $\varepsilon < 3.9\%$, the system's ergodicity is guaranteed, i.e. $\lim_{L\longrightarrow\infty} \hat{\mathbf{E}}_L\left[y^2\right] = \mathbf{E}\left[y^2\right]$.

The control system was simulated with various dropout rates between 0 and 6.4%. For each value of ε, we ran 5 different simulations for $200,000$ time steps and estimated the output power by equation 2.3.7. The simulation results are

shown in figure 2.2. When $\varepsilon < 3.9\%$, the figure shows close agreement between the experimentally estimated and theoretically predicted output power. The disagreement between the predicted and experimentally estimated output power is significant if $\varepsilon > 4.4\%$. We believe this is because the system is no longer ergodic under the large dropout rates. Remember that we are using time-averages to estimate the output power. We can only expect such averages to be a good estimate of the power when the system is ergodic.

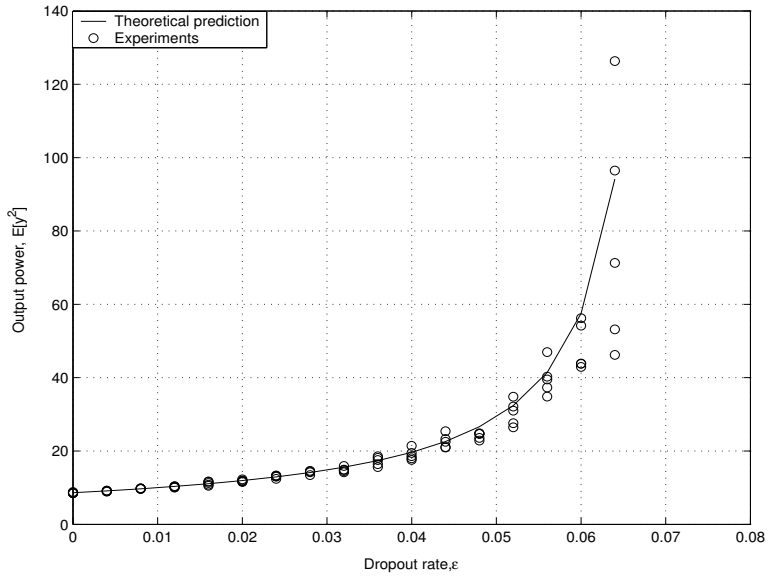


Figure 2.2. output power under different dropout rates

2.4 Equivalent linear time-invariant system

By theorem 2.3.1, we obtained a closed-form expression for the ouput's PSD, $S_{yy}(z)$. Such a PSD can also be generated by a constrained LTI system. This LTI system is shown in figure 2.3. It is equivalent to the original system in the sense that both system generate the same output's PSD.

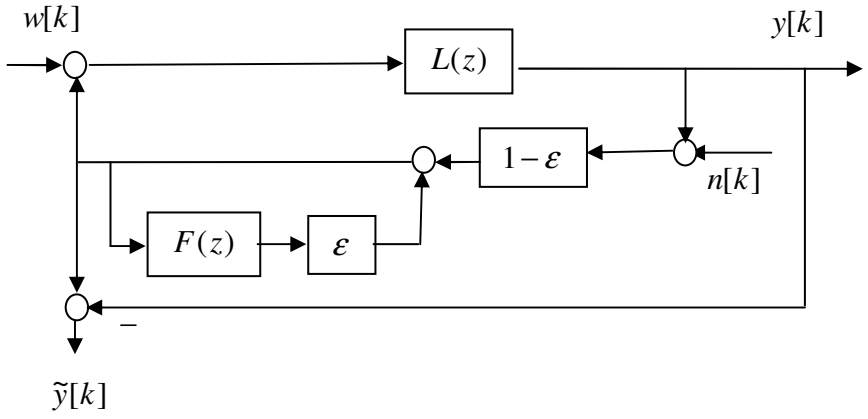


Figure 2.3. Equivalent system

The equivalent LTI system is driven by two zero-mean white Gaussian noise processes, w and n, whose variances are 1 and $\frac{\mathbf{E}[\tilde{y}^2]}{1-\varepsilon}$ respectively. Note that the variance of the noise process n is dependent on the variance of one output $\tilde{y}$. Obviously, the first question we must answer is whether or not this particular LTI system is well-posed. In other words, does there exist an input noise n such that

$$\mathbf{E}\left[n^2\right] = \frac{\mathbf{E}\left[\tilde{y}^2\right]}{1-\varepsilon} \tag{2.4.8}$$

That question is answered in the affirmative by the following theorem. The theorem is proven in the appendix.

Theorem 2.4.1 *Consider the LTI system shown in figure 2.3 with exogenous white zero-mean Gaussian inputs w and n where $\mathbf{E}[w^2] = 1$. If the closed-loop system is internally stable and the transfer function $G_{\tilde{y}n}(z)$ satisfies the inequality,*

$$\|G_{\tilde{y}n}\|_2^2 < 1 - \varepsilon, \tag{2.4.9}$$

then there exists a unique noise signal n (by unique, we mean the variance of n is unique) such that

$$\mathbf{E}\left[\tilde{y}^2\right] = (1-\varepsilon)\mathbf{E}\left[n^2\right] \tag{2.4.10}$$

The LTI system whose existence was established in theorem 2.4.1 is equivalent to the original time-varying system in figure 2.1 in the sense that both systems have the same stability conditions and generate the same power spectral densities. This assertion is stated in the following theorem whose proof will be found in the appendix (section 2.7).

Theorem 2.4.2 *When $\varepsilon > 0$, the LTI system in figure 2.3 is asymptotically stable and its transfer function $G_{\tilde{y}n}(z)$ satisfies*

$$\|G_{\tilde{y}n}\|_2^2 < 1 - \varepsilon \tag{2.4.11}$$

if and only if the system in eq. 2.1.3 is mean square stable. Furthermore, when the two systems are stable, they will generate the same power spectral densities, $S_{yy}(z)$ and $S_{\tilde{y}\tilde{y}}(z)$.

Remark: The power spectral density is a complete description of a process' second-order moments. By theorem 2.4.2, we know that the two systems, the equivalent LTI system and the original time-varying system, generate the same PSDs, $S_{yy}(z)$ and $S_{\tilde{y}\tilde{y}}(z)$. So they will also generate the same second-order moments, $\{R_{yy}[m]\}$ and $\{R_{\tilde{y}\tilde{y}}[m]\}$. In this chapter, we only care about second-order moments, such as $R_{yy}[0](= \mathbf{E}[y^2])$ and $R_{\tilde{y}\tilde{y}}[0](= \mathbf{E}[\tilde{y}^2])$. We can, therefore, use the equivalent LTI system to study the performance of the original time-varying system.

2.5 Optimal linear dropout compensation policy

In a system with dropouts, the dropout compensator is an essential part. When a packet is dropped, the dropout compensator will decide what to do in the absence of data from a pragmatic standpoint. It is important to design the optimal compensator. By optimal, we mean that the compensator achieves better performance than any other compensator.

There is little work on optimal dropout compensation. There are several heuristic compensation strategies, such as holding the latest received data [55], using constant data [28], and estimating the lost data based on the system dynamics [55]. Among these dropout compensators, which one is the best? Examples have shown that none of the aforementioned strategies is always the best [55] [43].

For mathematical tractability, we confine our attention to the optimal "linear" compensator. The system in figure 2.1 includes a linear time-invariant compensator $F(z)$. So we need to find the $F(z)$ that optimizes the performance, $\mathbf{E}[y^2]$. In [43], we solve this problem by the brute force methods. We first assume that

$F(z)$ has the following structure

$$F(z) = \frac{\sum_{j=1}^{N} b_j z^{-j}}{1 + \sum_{i=1}^{N} a_i z^{-i}} \tag{2.5.12}$$

where N is prespecified, and b_j and a_i are compensator parameters. Theorem 2.3.1 may be used to compute the performance, $\mathbf{E}[y^2]$, which is a function of b_j and a_i. We therefore optimize the function with respect to b_j and a_i. The problem inherent in this method is that we don't know how large N should be. This section uses the equivalent LTI system to propose a dropout compensator design method, which is not only more efficient than the one in [43], but also guarantees to be optimal over the class of linear time-invariant compensators.

Theorem 2.4.2 may be used to design $F(z)$ in the equivalent system of figure 2.3. In order to clarify the situation, let's redraw figure 2.3 as shown in figure 2.4.

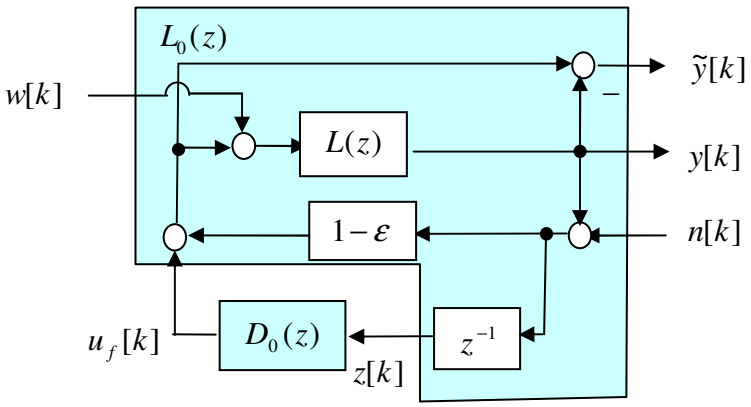


Figure 2.4. Redrawn equivalent system

In figure 2.4, $D_0(z)$ is a proper transfer function, which is related to $F(z)$ by [2]

$$D_0(z) = \frac{\varepsilon(1-\varepsilon)zF(z)}{1-\varepsilon F(z)} \tag{2.5.13}$$

Equation 2.5.13 presents an one-to-one map between $D_0(z)$ and $F(z)$. We can therefore design $F(z)$ through $D_0(z)$. As shown in figure 2.4, $D_0(z)$ constructs the control $\{u_f[k]\}$ from $\{z[k]\}$, the output of the generalized plant $L_0(z)$. We can therefore formulate the synthesis of $D_0(z)$ into the following optimization problem.

Optimization 2.5.1 $\min_{D_0(z)} \mathbf{E}\left[y^2\right]$ *subject to* [3]

$$\begin{aligned} \mathbf{E}\left[\tilde{y}^2\right] &= (1-\varepsilon)\mathbf{E}\left[n^2\right] \\ \mathbf{E}\left[w^2\right] &= 1 \end{aligned} \tag{2.5.14}$$

This particular optimization problem is awkward to solve directly because the dropout noise n has a variance that's proportional to the reconstruction error's variance $\mathbf{E}\left[\tilde{y}^2\right]$. The size of the reconstruction error's variance, of course, is dependent on our choice of $D_0(z)$. This means that both sides of the equality constraint are dependent on our choice of $D_0(z)$, thereby leading to a problem whose form is inconsistent with many optimization software packages.

In order to solve our synthesis problem, we need to recast the optimization problem in equation 2.5.1 into a more standard form. Without loss of generality, we take Δ as an additional design parameter that satisfies $\mathbf{E}\left[n^2\right] = \frac{1}{1-\varepsilon}\Delta$. We also note that the error signal $\tilde{y}$ can be rewritten as $\tilde{y}[k] = \varepsilon y[k] - u_f[k] - (1-\varepsilon)n[k]$.

[2] By eq. 2.5.13, we know that $F(z)$ is strictly proper is equivalent to that $D_0(z)$ is proper.

[3] Eq. 2.5.14 implies $\|G_{\tilde{y}n}\|_2^2 < 1-\varepsilon$ for $\varepsilon > 0$, which can be briefly proven as follows. By $\mathbf{E}\left[\tilde{y}^2\right] = \|G_{\tilde{y}n}\|_2^2\mathbf{E}\left[n^2\right] + \|G_{\tilde{y}w}\|_2^2\mathbf{E}\left[w^2\right]$, $\mathbf{E}\left[w^2\right] = 1 > 0$ and $\|G_{\tilde{y}w}\|_2^2 > 0$ for $\varepsilon > 0$, we get $\mathbf{E}[\tilde{y}^2] > \|G_{\tilde{y}n}\|_2^2\mathbf{E}[n^2]$. Combining the above inequality and eq. 2.5.14 yields $\|G_{\tilde{y}n}\|_2^2 < 1-\varepsilon$.

The reconstruction error's variance therefore can be written as

$$\mathbf{E}\left[\tilde{y}^2\right] = \mathbf{E}\left[\left(\varepsilon y - u_f\right)^2\right] + \left(1-\varepsilon\right)^2 \mathbf{E}\left[n^2\right] \tag{2.5.15}$$

Without loss of optimality, We rewrite the equality constraint in eq. 2.5.14 as an inequality constraint [4]

$$\begin{aligned} \mathbf{E}\left[\tilde{y}^2\right] &\leq (1-\varepsilon)\mathbf{E}\left[n^2\right] \\ &= \Delta \end{aligned}$$

The above inequality can be intuitively interpreted as a constraint on the reconstruction error. When the equality holds, the tightest constraint is obtained. So we expect the optimal configuration occurs when the equality holds, i.e. on the border of the feasible region. In the above inequality, Δ acts as an upper bound of the the reconstruction error's variance, $\mathbf{E}\left[\tilde{y}^2\right]$. If one substitutes eq. 2.5.15 into the above inequality, we obtain

$$\mathbf{E}\left[\left(\varepsilon y - u_f\right)^2\right] \leq \varepsilon\Delta \tag{2.5.16}$$

By multiplying both w and n by the same gain $\frac{1}{\sqrt{\Delta}}$, optimization problem 2.5.1 is transformed into the following problem.

Optimization 2.5.2 $\min_{\Delta} \min_{D_0(z)} \Delta \cdot \mathbf{E}\left[y^2\right]$ *subject to*

$$\begin{aligned} \mathbf{E}\left[\left(\varepsilon y - u_f\right)^2\right] &\leq \varepsilon \\ \mathbf{E}\left[n^2[k]\right] &= \frac{1}{1-\varepsilon}, \mathbf{E}\left[w^2[k]\right] = \frac{1}{\Delta} \end{aligned} \tag{2.5.17}$$

[4] In fact, Δ is the upper bound of the acceptable reconstruction error's variance.

This particular characterization of the synthesis problem is now in a more "standard" form that can be tackled by existing optimization software.

We solved optimization problem 2.5.2 in two steps. We first note that the inner optimization problem takes the form of a standard linear-quadratic Gaussian (LQG) synthesis [2]. We incorporated the constraint into the performance index as a penalty and solved the unconstrained optimization problem for the augmented performance index $\mathbf{E}\left[y^2 + \lambda\left(\varepsilon y - u_f\right)^2\right]$ where λ is a specified positive number. The solution of this optimization problem is a standard LQG controller, denoted as $D_{\Delta,\lambda}(z)$. It can be shown that smaller λ will lead to smaller $\mathbf{E}[y^2]$. This relationship between λ and $\mathbf{E}[y^2]$ stems from the fact that λ plays the role of a weighting function in the LQG performance objective. A small λ, therefore, corresponds to a larger penalty being assigned to $\mathbf{E}[y^2]$. The idea, therefore, is to search for the smallest λ whose corresponding controller $D_{\Delta,\lambda}(z)$ satisfies the constraint in eq. 2.5.17. We denote $\Delta\mathbf{E}[y^2]$, under the smallest λ, as $p(\Delta)$. This is exactly the optimal value for the inner part of optimization 2.5.2. This inner optimization problem was solved for a range of fixed Δ, so that $p(\Delta)$ now becomes a univariate function showing how the optimum performance $\mathbf{E}[y^2]$ varies as a function of the upper bound of the reconstruction error's variance Δ.

We used the above approach to design an optimal dropout compensator for the plant in section 2.3. We refer to this as the LQG dropout compensator. We compared the LQG compensator's performance against 3 popular heuristics. The first heuristic sets $F(z) = 0$ and corresponds to zeroing the control signal when a dropout occurs. The second heuristic is $F(z) = z^{-1}$ which is equivalent to reusing the last feedback measurement when a dropout occurs. The third heuristic uses an $F(z)$ that minimizes the reconstruction error Δ. We refer to this as the recon-

struction compensator. The output power achieved by all dropout compensators is plotted as a function of the average dropout rate ε in figure 2.5. The figure shows that the reconstruction estimator and LQG compensation schemes clearly outperform the other two heuristics. The LQG compensator actually does a little better than the reconstruction compensator and surprisingly it's minimum value does not occur for $\varepsilon = 0$. This is because the LQG compensator is a better regulator than the default unity output feedback.

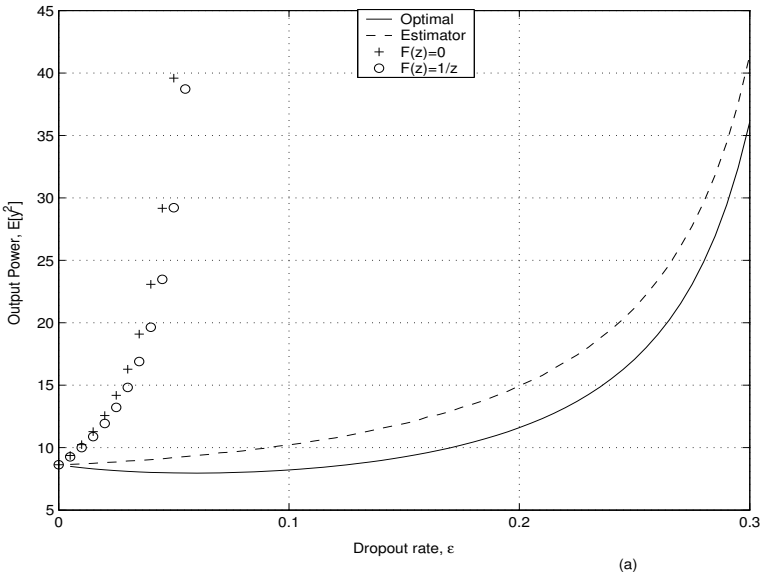


Figure 2.5. Performance comparisons under different dropout compensators

As mentioned in the introduction of this section, the two terms in equation 2.3.4 suggest that the optimal dropout compensator does not always attempt to minimize the reconstruction error. This fact is illustrated in figure 2.6. This figure

plots the optimum performance level, $p(\Delta)$, achieved for reconstruction errors in the range $0 < \Delta < 0.8$ assuming $\varepsilon = 0.1$. Note that this function is not a monotonically increasing function of Δ. It has a definite global minimum that appears to occur for a reconstruction error variance, Δ, of about 0.38. This result confirms our earlier suspicion that the optimal dropout compensation should not always try to minimize the reconstruction error.

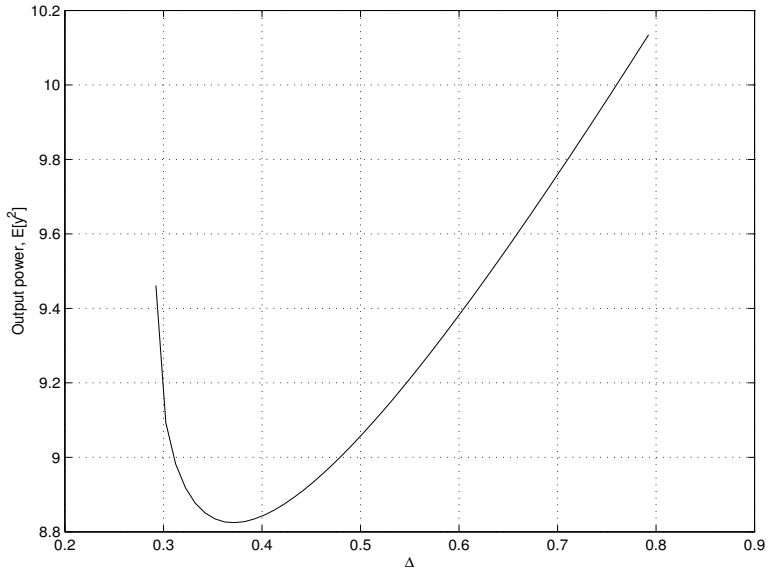


Figure 2.6. Value function $p(\Delta)$ for $\varepsilon = 0.1$

2.6 Conclusions

This chapter studied a control system with i.i.d. dropouts. We derived a necessary and sufficient stability condition and the performance results (a closed-

form expression of the output's power spectral density). The PSD results were used to obtain a linear time-invariant system that is equivalent to the original time-varying system with dropouts in the sense of the same stability condition and the same output's PSD. This equivalent system was used for synthesis. As a synthesis example, we designed an optimal dropout compensator for the system with i.i.d. dropouts.

Because the power spectral density is a full description of the process' second-order moments, we know that the "equivalent" system is equivalent to the original system with respect to second-order moments. Besides the aforementioned dropout compensator, we can follow the same procedure to design an optimal controller (with performance measured by the output process' second-order moment).

This chapter considered a SISO system. Future research should try to extend these methods to MIMO (multiple-input-multiple-output) systems.

2.7 Appendix: proofs

2.7.1 Proof of Theorem 2.3.1

Let h and f denote the impulse response functions for $L(z)$ and $F(z)$, respectively. Let w denote the exogenous disturbance signal. The signals, y, $\bar{y}$, and $\hat{y}$ represent the loop function's output signal, the control signal re-injected into the plant, and the dropout compensator's output signal, respectively. These three signals are related through the convolution equations,

$$\begin{cases} y &= h * (\overline{y} + w) \\ \hat{y} &= f * \overline{y} \end{cases} \tag{2.7.18}$$

We first compute the power cross-spectral densities $S_{\overline{y}w}(z)$, $S_{yw}(z)$ and $S_{\hat{y}w}(z)$ relating these output signals to the input w. From equation 2.7.18, we get

$$\begin{cases} S_{yw}(z) &= L(z)\left(S_{\overline{y}w}(z) + S_{ww}(z)\right) \\ S_{\hat{y}w}(z) &= F(z)S_{\overline{y}w}(z) \end{cases} \tag{2.7.19}$$

For any m, the correlation $R_{\overline{y}w}$ may be written as

$$\begin{aligned} R_{\overline{y}w}[m] &= \mathbf{E}\left[\overline{y}[k+m]w[k]\right] \\ &= \mathbf{E}\left[\overline{y}[k+m]w[k]|d[k+m]=0\right]P(d[k+m]=0) \\ &\quad +\mathbf{E}\left[\overline{y}[k+m]w[k]|d[k+m]=1\right]P(d[k+m]=1) \end{aligned}$$

Because $L(z)$ and $F(z)$ are strictly proper, we know that $y[k]$ and $\hat{y}[k]$ are independent of current and future dropouts, so that the last equation can be rewritten as

$$\begin{aligned} R_{\overline{y}w}[m] &= \mathbf{E}\left[y[k+m]w[k]\right]P(d[k+m]=0) \\ &\quad +\mathbf{E}\left[\hat{y}[k+m]w[k]\right]P(d[k+m]=1) \\ &= (1-\varepsilon)R_{yw}[m] + \varepsilon R_{\hat{y}w}[m] \end{aligned}$$

We then take the double-sided z-transform of the above equation to obtain

$$S_{\overline{y}w}(z) = (1-\varepsilon)S_{yy}(z) + \varepsilon S_{\hat{y}y}(z) \tag{2.7.20}$$

Combining equations 2.7.19 and 2.7.20 generates the following expressions for the cross-spectral densities,

$$\begin{cases} S_{\overline{y}w}(z) &= \frac{D(z)L(z)}{1-D(z)L(z)} S_{ww}(z) \\ S_{yw}(z) &= \frac{L(z)}{1-D(z)L(z)} S_{ww}(z) \\ S_{\hat{y}w}(z) &= \frac{D(z)L(z)F(z)}{1-D(z)L(z)} S_{ww}(z) \end{cases} \tag{2.7.21}$$

where $D(z) = \frac{1-\varepsilon}{1-\varepsilon F(z)}$.

The convolutions in equation 2.7.18 also generate the following equations

$$S_{\overline{y}y}(z) = L(z^{-1}) \left(S_{\overline{yy}}(z) + S_{\overline{y}w}(z)\right) \tag{2.7.22}$$

$$S_{yy}(z) = L(z)L(z^{-1}) \left(S_{\overline{yy}}(z) + S_{\overline{y}w}(z) + S_{\overline{y}w}(z^{-1}) + S_{ww}(z)\right) \tag{2.7.23}$$

$$S_{\hat{y}\hat{y}}(z) = F(z)F(z^{-1})S_{\overline{yy}}(z) \tag{2.7.24}$$

$$S_{\hat{y}y}(z) = F(z)S_{\overline{y}y}(z) \tag{2.7.25}$$

$$S_{\overline{y}\hat{y}}(z) = F(z^{-1})S_{\overline{yy}}(z) \tag{2.7.26}$$

There are six unknown spectral densities in these equations, including three PSD's S_{yy},$S_{\hat{y}\hat{y}}$, and $S_{\overline{yy}}$, and three cross-spectral densities $S_{\overline{y}y}$, $S_{\overline{y}\hat{y}}$, and $S_{\hat{y}y}$. There are, however, only 5 equations given above. Since there are six unknowns and only five equations, we must find another independent equation. The signal $\overline{y}$ is not related to y and $\hat{y}$ through a simple convolution because $\overline{y}$ switches between these two signals.

In order to properly model the correlation of such switching signals, it is convenient to define single-sided power spectral densities.

$$S^{+}_{xy}(z) = \sum_{m=1}^{\infty} R_{xy}[m]z^{-m}, \; S^{-}_{xy}(z) = \sum_{m=-\infty}^{-1} R_{xy}[m]z^{-m}$$

The above definitions imply

$$\begin{cases} S_{xy}(z) &= S^+_{xy}(z) + S^-_{xy}(z) + R_{xy}[0] \\ S^-_{xy}(z) &= S^+_{yx}(z^{-1}) \\ S_{yy}(z) &= S^+_{yy}(z) + S^+_{yy}(z^{-1}) + R_{yy}[0] \end{cases} \tag{2.7.27}$$

The sixth equation will be obtained by deriving an expression for $S_{\overline{yy}}(z)$. We first note that for $m > 0$,

$$\begin{aligned} R_{\overline{yy}}[m] &= \mathbf{E}\left[\overline{y}[k+m]\overline{y}[k]\right] \\ &= \mathbf{E}\left[\overline{y}[k+m]\overline{y}[k]|d[k+m]=0\right] P(d[k+m]=0) \\ &\quad +\mathbf{E}\left[\overline{y}[k+m]\overline{y}[k]|d[k+m]=1\right] P(d[k+m]=1) \end{aligned}$$

Because $L(z)$ and $F(z)$ are strictly proper, we know that

$$\begin{aligned} R_{\overline{yy}} &= \mathbf{E}\left[y[k+m]\overline{y}[k]\right] P(d[k+m]=0) \\ &\quad +\mathbf{E}\left[\hat{y}[k+m]\overline{y}[k]\right] P(d[k+m]=1) \\ &= (1-\varepsilon)R_{y\overline{y}}[m] + \varepsilon R_{\hat{y}\overline{y}}[m] \end{aligned}$$

which immediately implies that

$$S^+_{\overline{yy}}(z) = (1-\varepsilon)S^+_{y\overline{y}}(z) + \varepsilon S^+_{\hat{y}\overline{y}}(z) \tag{2.7.28}$$

From the PSD identities in equation 2.7.27, we know that

$$S^+_{y\overline{y}}(z) = S_{y\overline{y}}(z) - S^-_{y\overline{y}}(z) - S_{y\overline{y}}[0] \tag{2.7.29}$$

Following a similar derivation to that used in equation 2.7.28 we obtain, for $m < 0$,

$$R_{y\overline{y}}[m] = (1-\varepsilon)R_{yy}[m] + \varepsilon R_{y\hat{y}}[m] \tag{2.7.30}$$

Taking the single sided z-transform of equation 2.7.30 yields

$$S^{-}_{y\overline{y}}(z) = (1-\varepsilon)S^{-}_{yy}(z) + \varepsilon S^{-}_{y\hat{y}}(z) \tag{2.7.31}$$

Substituting eq. 2.7.31 into eq. 2.7.29 yields

$$S^{+}_{y\overline{y}}(z) = S_{y\overline{y}}(z) - (1-\varepsilon)S^{-}_{yy}(z) - \varepsilon S^{-}_{y\hat{y}}(z) - S_{y\overline{y}}[0] \tag{2.7.32}$$

Similarly we obtain

$$S^{+}_{\hat{y}\overline{y}}(z) = S_{\hat{y}\overline{y}}(z) - (1-\varepsilon)S^{-}_{\hat{y}y}(z) - \varepsilon S^{-}_{\hat{y}\hat{y}}(z) - S_{\hat{y}\overline{y}}[0] \tag{2.7.33}$$

We now substitute equations 2.7.32 and 2.7.33 into equation 2.7.28 to obtain

$$\begin{aligned} S^{+}_{\overline{y}\overline{y}}(z) &= (1-\varepsilon)S_{y\overline{y}}(z) + \varepsilon S_{\hat{y}\overline{y}}(z) - (1-\varepsilon)^2 S^{-}_{yy}(z) \\ &\quad -\varepsilon^2 S^{-}_{\hat{y}\hat{y}}(z) - \varepsilon(1-\varepsilon)S^{-}_{y\hat{y}}(z) - \varepsilon(1-\varepsilon)S^{-}_{\hat{y}y}(z) \\ &\quad -(1-\varepsilon)R_{y\overline{y}}[0] - \varepsilon R_{\hat{y}\overline{y}}[0] \end{aligned} \tag{2.7.34}$$

Substituting eq. 2.7.34 into the third identity in equation 2.7.27 yields

$$\begin{aligned} S_{\overline{yy}}(z) &= (1-\varepsilon)\left(S_{y\overline{y}}(z)+S_{y\overline{y}}(z^{-1})\right) \\ &\quad +\varepsilon\left(S_{\hat{y}\overline{y}}(z)+S_{\hat{y}\overline{y}}(z^{-1})\right) \\ &\quad -(1-\varepsilon)^2\left(S^{-}_{yy}(z)+S^{-}_{yy}(z^{-1})\right) \\ &\quad -\varepsilon^2\left(S^{-}_{\hat{y}\hat{y}}(z)+S^{-}_{\hat{y}\hat{y}}(z^{-1})\right) \\ &\quad -\varepsilon(1-\varepsilon)\left(S^{-}_{y\hat{y}}(z)+S^{-}_{\hat{y}y}(z^{-1})\right) \\ &\quad -\varepsilon(1-\varepsilon)\left(S^{-}_{y\hat{y}}(z^{-1})+S^{-}_{\hat{y}y}(z)\right) \\ &\quad -2(1-\varepsilon)R_{y\overline{y}}[0]-2\varepsilon R_{\hat{y}\overline{y}}[0]+R_{\overline{yy}}[0] \end{aligned}$$

We apply the properties of single-sided PSD's in eq. 2.7.27 to cancel the sum of single sided PSDs in the above equation to obtain our final expression

$$\begin{aligned} S_{\overline{yy}}(z) &= (1-\varepsilon)\left(S_{y\overline{y}}(z)+S_{y\overline{y}}(z^{-1})\right)+\varepsilon\left(S_{\hat{y}\overline{y}}(z)\right. \\ &\quad \left.+S_{\hat{y}\overline{y}}(z^{-1})\right)-(1-\varepsilon)^2S_{yy}(z)-\varepsilon^2S_{\hat{y}\hat{y}}(z) \\ &\quad -\varepsilon(1-\varepsilon)S_{y\hat{y}}(z)-\varepsilon(1-\varepsilon)S_{\hat{y}y}(z)+(1-\varepsilon)\Delta \end{aligned} \tag{2.7.35}$$

where

$$\begin{aligned} \Delta &= \left(-2\frac{\varepsilon}{1-\varepsilon}R_{\hat{y}\overline{y}}[0]+\frac{1}{1-\varepsilon}R_{\overline{yy}}[0]+\frac{\varepsilon^2}{1-\varepsilon}R_{\hat{y}\hat{y}}[0]\right. \\ &\quad \left.-2R_{y\overline{y}}[0]+(1-\varepsilon)R_{yy}[0]+\varepsilon R_{y\hat{y}}[0]+\varepsilon R_{\hat{y}y}[0]\right). \end{aligned} \tag{2.7.36}$$

Equations 2.7.22-2.7.26 and 2.7.35 represent 6 independent equations that we

can then solve for the 6 PSD's. In particular, solving for $S_{yy}(z)$ yields the following

$$S_{yy}(z) = \left|\frac{L(z)}{1-D(z)L(z)}\right|^2 S_{ww}(z) + \frac{1}{1-\varepsilon}\left|\frac{D(z)L(z)}{1-D(z)L(z)}\right|^2 \Delta \tag{2.7.37}$$

Because $\tilde{y}[k] = y[k] - \overline{y}[k]$, we know that

$$\begin{aligned} S_{\tilde{y}\tilde{y}}(z) &= S_{yy}(z) + S_{\overline{y}\overline{y}}(z) - S_{y\overline{y}}(z) - S_{\overline{y}y}(z) \\ &= \left|\frac{L(z)(D(z)-1)}{1-D(z)L(z)}\right|^2 S_{ww}(z) + \frac{1}{1-\varepsilon}\left|\frac{D(z)(1-L(z))}{1-D(z)L(z)}\right|^2 \Delta \end{aligned}$$

which matches the PSD's stated in the theorem.

A simpler, more meaningful, expression for Δ can be computed. With the switching property of $\overline{y}[k]$, we get the identities, such as $R_{\overline{y}y}[0] = \varepsilon R_{\hat{y}y}[0] + (1-\varepsilon)R_{yy}[0]$, and then use these identities to simplify the expression in eq. 2.7.36 to the form $\Delta = R_{\tilde{y}\tilde{y}}[0]$, where $\tilde{y} = y - \overline{y}$. When $\varepsilon = 0$, $\tilde{y} = 0$ and $\Delta = 0$.

Because the system in eq. 2.1.3 is stable in the mean square sense, $\tilde{y}$, as a linear output, has finite variance by theorem 2.2.2, i.e. $R_{\tilde{y}\tilde{y}}[0]$ is finite. When $\varepsilon > 0$, i.e. there is dropouts, we can show that $R_{\tilde{y}\tilde{y}}[0] > 0$.

Because $R_{\tilde{y}\tilde{y}}[0] = \frac{1}{2\pi}\int_{-\pi}^{\pi} S_{\tilde{y}\tilde{y}}(e^{j\omega})d\omega$, we can further reduce this expression to that stated in eq. 2.3.5. Therefore $\Delta = R_{\tilde{y}\tilde{y}}[0]$ is a positive solution to eq. 2.3.5 for $\varepsilon > 0$.

Take a close look at eq. 2.3.5. When $\varepsilon > 0$, the first term $\frac{1}{2\pi}\int_{-\pi}^{\pi}\left|\frac{L(e^{j\omega})(D(e^{j\omega})-1)}{1-D(e^{j\omega})L(e^{j\omega})}\right|^2 S_{ww}(e^{j\omega})d\omega > 0$. So eq. 2.3.5 has a positive solution implies the solution is unique. $\diamondsuit$

2.7.2 Proof of Theorem 2.4.1

Because the system is internally stable, a straightforward computation of $\mathbf{E}\left[\tilde{y}^2\right]$ for the LTI system shows that

$$\mathbf{E}\left[\tilde{y}^2\right] = \|G_{\tilde{y}w}\|_2^2 + \|G_{\tilde{y}n}\|_2^2 \, \mathbf{E}[k^2]$$

Let $\Delta = \mathbf{E}[k^2](1-\varepsilon) = \mathbf{E}[\tilde{y}^2]$, then the preceding equation takes the form,

$$\Delta = \|G_{\tilde{y}w}\|_2^2 + \|G_{\tilde{y}n}\|_2^2 \, \frac{\Delta}{1-\varepsilon} \tag{2.7.38}$$

Eq. 2.7.38 has a unique non-negative solution with respect to Δ if and only if $\frac{1}{1-\varepsilon}\|G_{\tilde{y}n}\|_2^2 < 1$. $\diamondsuit$

2.7.3 Proof of Theorem 2.4.2

The state space model of the equivalent system in figure 2.3 is

$$\left\{ \begin{array}{rcl} x_e[k+1] & = & A_e x_e[k] + B_w w[k] + B_n n[k] \\ y[k] & = & C_y x_e[k] \\ \tilde{y}[k] & = & C_{\tilde{y}} x_e[k] + (1-\varepsilon) n[k] \end{array} \right. \tag{2.7.39}$$

where

$$B_w = \left[\begin{array}{c} B_h \\ 0 \end{array} \right], \quad B_n = (1-\varepsilon) \left[\begin{array}{c} B_h \\ B_f \end{array} \right],$$
$$C_y = \left[\begin{array}{cc} C_h & 0 \end{array} \right], \quad C_{\tilde{y}} = \varepsilon \left[\begin{array}{cc} -C_h & C_f \end{array} \right]$$

and $A_e = (1-\varepsilon)A_0 + \varepsilon A_1$. The matrices A_0, A_1, B_h, B_f, C_h and C_f are defined in section 2.1.

2.7.3.1 Proof of stability equivalence

When the equivalent system is asymptotically stable and the constraint in eq. 2.4.11 is satisfied

The stability of the LTI system means that $\|G_{\tilde{y}n}\|_2^2 = C_{\tilde{y}} W_n C_{\tilde{y}}^2 + (1-\varepsilon)^2$ where W_n satisfies the Lyapunov equation

$$A_e W_n A_e^T + B_n B_n^T = W_n \tag{2.7.40}$$

Moreover, because all eigenvalues of A_e lie within the unit circle, we also know there exists a unique $P_0 > 0$ that satisfies the Lyapunov equation

$$A_e P_0 A_e^T + I = P_0 \tag{2.7.41}$$

Combining the assumption that $\|G_{\tilde{y}n}\|_2^2 < 1 - \varepsilon$ with the expression of $\|G_{\tilde{y}n}\|_2^2$ yields

$$C_{\tilde{y}} W_n C_{\tilde{y}}^T < \varepsilon(1-\varepsilon) \tag{2.7.42}$$

Because this is a strict inequality, we know there exists a small positive real number γ such that

$$C_{\tilde{y}} \left(W_n + \gamma P_0\right) C_{\tilde{y}}^T < \varepsilon(1-\varepsilon) \tag{2.7.43}$$

We now define a symmetric matrix $P = W_n + \gamma P_0$.

Based on the matrix definitions in eq. 2.7.39, we know that for any symmetric matrix $\overline{P}$

$$((1-\varepsilon)A_0\overline{P}A_0^T + \varepsilon A_1\overline{P}A_1^T) - A_e\overline{P}A_e^T = \frac{1}{\varepsilon(1-\varepsilon)}B_nC_{\tilde{y}}\overline{P}C_{\tilde{y}}^TB_n^T \qquad (2.7.44)$$

In particular, we set $\overline{P}$ equal to the matrix P defined in the preceding paragraph. For this particular P we know that $C_{\tilde{y}}PC_{\tilde{y}}^T < \varepsilon(1-\varepsilon)$, so that equation 2.7.44 becomes,

$$\begin{aligned}
(1-\varepsilon)A_0PA_0^T + \varepsilon A_1PA_1^T &\leq A_ePA_e^T + B_nB_n^T \\
&= (A_eW_nA_e^T + B_nB_n^T) + \gamma A_eP_0A_e^T \\
&= W_n + \gamma(P_0 - I) \\
&< W_n + \gamma P_0 \\
&= P
\end{aligned}$$

Therefore there exists a $P > 0$ such that

$$(1-\varepsilon)A_0PA_0^T + \varepsilon A_1PA_1^T < P. \qquad (2.7.45)$$

We now construct a free jump linear system with the system matrix, $A[k]$, of the original system in equation 2.1.3.

$$x[k+1] = A^T[k]x[k] \qquad (2.7.46)$$

We construct a candidate Lyapunov function $V[k] = x^T[k]Px[k]$. Because the

switching is i.i.d. in the jump linear system, we use equation 2.7.45 to show that

$$\begin{aligned}
& \mathbf{E}\left[V[k+1]\right] \\
= \; & \mathbf{E}\left[x^T[k+1]Px[k+1]\right] \\
= \; & \mathbf{E}\left[x^T[k]A[k]PA^T[k]x[k]\right] \\
= \; & \mathbf{E}\left[x^T[k]\mathbf{E}\left[A[k]PA^T[k]\right]x[k]\right] \\
= \; & \mathbf{E}\left[x^T[k]\left((1-\varepsilon)A_0PA_0^T+\varepsilon A_1PA_1^T\right)x[k]\right] \\
< \; & \mathbf{E}\left[x^T[k]Px[k]\right] \\
= \; & \mathbf{E}\left[V[k]\right]
\end{aligned}$$

So the system in eq. 2.7.46 is mean square stable. By theorem 2.2.1, we know $(1-\varepsilon)A_0^T \otimes A_0^T + \varepsilon A_1^T \otimes A_1^T = A_{[2]}^T$ has all eigenvalues within the unit circle. This implies that $A_{[2]}$ has all eigenvalues within the unit circle and we again use theorem 2.2.1 to infer the mean square stability of the original system.

When the original system is mean square stable We will first prove the equivalent system is asymptotically stable. We will prove there exist $P > 0$ such that

$$A_e P A_e^T < P \tag{2.7.47}$$

For any square matrix $Q > 0$, we construct the following matrix sequence

$$\begin{cases} P_{n+1} = (1-\varepsilon)A_0P_nA_0^T + \varepsilon A_1P_nA_1^T + Q \\ P_0 = Q \end{cases} \tag{2.7.48}$$

By the definition of $\{P_n\}$, we can prove

$$P_{n+1} \geq P_n \tag{2.7.49}$$

Use the operator $vec()$, we can transform eq. 2.7.48 into

$$vec(P_{n+1}) = A_{[2]} vec(P_n) + vec(Q) \tag{2.7.50}$$

Because $A_{[2]}$ is stable by theorem 2.2.1, we obtain the limit $\lim_{n \longrightarrow \infty} vec(P_n)$ exists and is finite. By the linearity of $vec()$, we know $\lim_{n \longrightarrow \infty} P_n$ exists and is finite and definite positive, denoted as P. By eq. 2.7.48, we get

$$P = (1-\varepsilon) A_0 P A_0^T + \varepsilon A_1 P A_1^T + Q \tag{2.7.51}$$

By the definition of A_e, we obtain

$$\begin{aligned} A_e P A_e^T &= \left((1-\varepsilon)A_0 + \varepsilon A_1\right) P \left((1-\varepsilon)A_0 + \varepsilon A_1\right)^T \\ &= (1-\varepsilon) A_0 P A_0^T + \varepsilon A_1 P A_1^T - \varepsilon(1-\varepsilon)\left(A_0 - A_1\right) P \left(A_0 - A_1\right)^T \\ &= P - Q - \varepsilon(1-\varepsilon)\left(A_0 - A_1\right) P \left(A_0 - A_1\right)^T \\ &< P \end{aligned}$$

The above third equality uses eq. 2.7.51. By $P > 0$ and $A_e P A_e^T < P$, we know A_e is stable, i.e. the equivalent system is asymptotically stable. In the following, we will prove the constraint in 2.4.11 is satisfied.

Because the original system is stable in the mean square sense, we know that by theorem 2.3.1, eq. 2.3.5 has a unique positive solution. We may use the

configuration of the equivalent system in figure 2.3 to rewrite eq. 2.3.5 as

$$\Delta = \|G_{\tilde{y}w}\|_2^2 + \|G_{\tilde{y}n}\|_2^2 \frac{1}{1-\varepsilon}\Delta \tag{2.7.52}$$

Because $\|G_{\tilde{y}w}\|_2^2 > 0$ for $\varepsilon > 0$, the uniqueness of the solution of the above equation yields

$$\|G_{\tilde{y}n}\|_2^2 < 1 - \varepsilon \tag{2.7.53}$$

2.7.3.2 PSD equivalence

We now show that both systems generate the same power spectral density. Since the equivalent system is stable, it will generate WSS signals y and $\tilde{y}$. The PSD's for these signals are readily computed as

$$\begin{aligned} S_{yy}(z) &= \left|\tfrac{L(z)}{1-D(z)L(z)}\right|^2 S_{ww}(z) + \left|\tfrac{D(z)L(z)}{1-D(z)L(z)}\right|^2 \mathbf{E}\left[n^2\right] \\ S_{\tilde{y}\tilde{y}}(z) &= \left|\tfrac{L(z)(D(z)-1)}{(1-D(z)L(z))}\right|^2 S_{ww}(z) + \left|\tfrac{D(z)(1-L(z))}{1-D(z)L(z)}\right|^2 \mathbf{E}\left[n^2\right] \end{aligned}$$

In the above expression, we have already considered the white nature of n, which yields $S_{nn}(z) = \mathbf{E}\left[n^2\right]$. From theorem 2.4.1, we know that $\mathbf{E}[n^2] = \mathbf{E}[\tilde{y}^2]/(1-\varepsilon)$. Define $\Delta_0 = \mathbf{E}\left[\tilde{y}^2\right]$. Then $\mathbf{E}\left[n^2\right] = \frac{\Delta_0}{1-\varepsilon}$ and Δ_0 satisfies eq. 2.4.11. By the uniqueness of the solution of eq. 2.4.11, we know $\Delta = \Delta_0$. Therefore $\mathbf{E}\left[n^2\right] = \frac{\Delta}{1-\varepsilon}$. Substituting this into the above equations for $S_{yy}(z)$ and $S_{\tilde{y}\tilde{y}}(z)$, yields the same PSD's found in theorem 2.3.1. $\diamondsuit$

CHAPTER 3

STABILITY AND PERFORMANCE OF CONTROL SYSTEMS WITH FEEDBACK DROPOUTS GOVERNED BY A MARKOV CHAIN

This chapter considers a control system with dropouts governed by a Markov chain. It is organized as follows. Section 3.1 presents a mathematical model for the concerned system. Section 3.2 provides a necessary and sufficient condition for stability and wide sense stationarity of such systems. Section 3.3 proposes a method to compute system performance measured by the output power. The computation may be cast as an LMI (linear matrix inequality). Section 3.4 utilizes the achieved performance result to design the optimal dropout policy under the overloading circumstances, for example the communication network is congested. The optimal dropout policy design is formulated as a BMI (bilinear matrix inequality) and the resulting optimization problem is solved with a branch-and-bound method. The optimal dropout policy we obtain works as a guideline for real-time engineers to design scheduling algorithms. We also did some experiments to verify the optimality of the proposed dropout policy. Section 3.5 extends the performance results of a system with feedback dropouts governed by a Markov chain, including the performance computation and the optimal dropout policy design, to distributed systems. Section 3.6 concludes this chapter with some final remarks. The appendix, section 3.7 includes all technical proofs.

3.1 System model

The block diagram of the concerned system is shown in figure 3.1. $H(z)$ is the loop function which is the combination of the plant and the controller. $H(z)$ generates the output signal y. The output signal y and the random dropout process d drive a model of the feedback channel which generates the feedback signal $\overline{y}$. The loop function, $H(z)$, is driven by the input signal $u = \overline{y} + w$ where w is an exogenous noise signal. The system performance is measured by the output power $\mathbf{E}[y^T y]$. [1] Such performance characterizes the noise attenuation of y with respect to w and d.

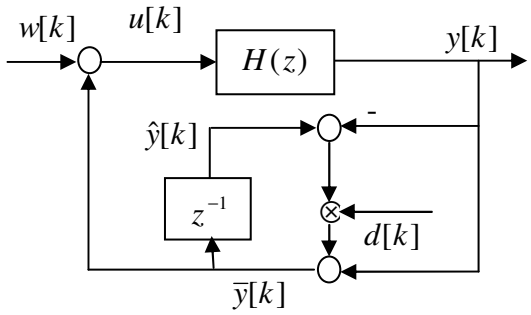


Figure 3.1. A control system with data dropouts

The system shown in figure 3.1 has two inputs. The exogenous input disturbance process w is assumed to be white with zero mean and unit variance. The other input is the dropout process $d = \{d[k]\}$. When $d[k] = 0$, the feedback

[1] It will be proven by Corollary 3.2.1 that $\mathbf{E}\left[y^T[k]y[k]\right]$ is constant with respect to n. Therefore the index n in $\mathbf{E}\left[y^T[k]y[k]\right]$ is dropped.

measurement is successfully transmitted over the channel and $\overline{y}[k] = y[k]$. When $d[k] = 1$, the measurement is dropped by the channel and the feedback signal simply reuses the last transmitted value, i.e. $\overline{y}[k] = \overline{y}[k-1]$. The random dropout process, d, has a distribution selected from an underlying Markov chain. The Markov chain is homogeneous, irreducible and aperiodic with N ($N < \infty$) states, $\{q_1, q_2, \cdots, q_N\}$, and the transition matrix $Q = (q_{ij})_{N\times N}$. At time step n, $d[k]$ is uniquely determined by the state of the Markov chain $q[k]$, i.e. there exists a function $d[k] = f(q[k])$.

We assume that H has a minimal state space realization, $H \stackrel{s}{=} \left[\begin{array}{c|c} A_P & B_P \\ \hline C_P & 0 \end{array}\right]$, where the column number of B_P equals to the row number of C_P. Thus a state space representation of the system in figure 3.1 is

$$\left\{\begin{array}{rcl} x[k+1] & = & A[q[k]]x[k] + Bw[k] \\ y[k] & = & Cx[k] \end{array}\right. , \tag{3.1.1}$$

where $x[k] \in \mathcal{R}^n$, $w[k] \in \mathcal{R}^p$, $y[k] \in \mathcal{R}^m (m = p)$. $\{A[q[k]]\}$ is a switching matrix-valued random process. When $d[k] = f(q[k]) = 0$, $A[k] = A_{close}$; when $d[k] = f(q[k]) = 1$, $A[q[k]] = A_{open}$. $A_{close} = \left[\begin{array}{cc} A_P + B_P C_P & 0 \\ C_P & 0 \end{array}\right]$, $A_{open} = \left[\begin{array}{cc} A_P & B_P \\ 0 & I_m \end{array}\right]$, $B = \left[\begin{array}{c} B_P \\ 0 \end{array}\right]$,$C = \left[\begin{array}{cc} C_P & 0 \end{array}\right]$, I_m denotes an identity matrix with the dimension of m. The 0 matrices have the appropriate dimensions. For notational convenience, $A[q[k]]$ is denoted as A_i when $q[k] = q_i$. Obviously $A_i = A_{close}$ when $d[k] = f(q_i) = 0$; $A_i = A_{open}$ when $d[k] = f(q_i) = 1$. With the above notational conventions, we

define the matrix

$$A_{[2]} = diag\left(A_i \otimes A_i\right)_{i=1,\cdots,N}\left(Q^T \otimes I_{n^2}\right) \tag{3.1.2}$$

Throughout this chapter, the following five assumptions are taken:

1. w is a zero-mean white noise process with unit variance.

2. The dropout process is a Markov chain which is finite, time-homogeneous, irreducible and aperiodic with transition matrix of $Q = (q_{ij})_{N\times N}$. The steady state of the Markov chain is denoted as $\pi = \begin{bmatrix} \pi_1 & \pi_2 & \cdots & \pi_N \end{bmatrix}$.

3. The disturbance process, w, is independent from the dropout process .

4. The initial time of the system is $-\infty$.

5. The matrix $A_{[2]}$, defined in equation 3.1.2, is stable, i.e. all eigenvalues of $A_{[2]}$ lie within the unit circle.

3.2 Stability and wide sense stationarity of control systems with feedback dropouts governed by a Markov chain

As mentioned in chapter 1, the system with feedback dropouts can be modeled as a jump linear system, so we adopt the results in [16], [51], [24] and [11] to get the following stability condition.

Theorem 3.2.1 *([16], [51], [24], [11]) Consider the free control system ($w = 0$) given in equation 3.1.1 under assumption 2. It is mean square stable if and only if the matrix defined in eq. 3.1.2, $A_{[2]}$, is stable, i.e. all eigenvalues of $A_{[2]}$ lie within the unit circle.*

In the previous literature, only a sufficiency proof was provided. It is in [17] that the necessity was rigorously proven for the first time [23]. We came up with an alternative proof for necessity that is more direct than the proof in [17] ([15]). We present our proof in the appendix, section 3.7.

If the system is mean square stable, its initial condition will be eventually forgotten. By assumption 4, the initial time of the system is set to $-\infty$, so the state $\{x[k]\}$ will be uniquely determined by the input noise sequence $\{w[k]\}$. The stationarity of $\{w[k]\}$ ($\{w[k]\}$ is white) guarantees that $\{x[k]\}$ is wide sense stationary, which is presented in the following theorem 3.2.2. The proof of theorem 3.2.2 will be found in the appendix.

Theorem 3.2.2 *Consider the system in equation 3.1.1. Under assumptions 1—5, the state process $x = \{x[k]\}$ is wide sense stationary.*

Remark: Comparing the conditions in theorems 3.2.1 and 3.2.2, we find that the system is WSS if it is mean square stable. Only when a system is stable does it make sense to study its wide sense stationarity. So the conditions in theorem 3.2.2 is not only sufficient but also necessary.

The following corollary extends theorem 3.2.2 to show that all linear outputs of the system are WSS. The proof of corollary 3.2.1 is similar to the one of theorem 3.2.2. So it is omitted.

Corollary 3.2.1 *Consider the system in equation 3.1.1. Under assumptions 1—5, any linear outputs are WSS, where a linear output stands for a signal of the form $z[k] = E[q[k]]x[k]+F[q[k]]w[k]$ and $E[q[k]]$ and $F[q[k]]$ are matrices switching between the closed-loop and open-loop configurations.*

Remark: Corollary 3.2.1 guarantees the wide sense stationarity for general linear outputs. It particularly holds for $y[k] = Cx[k]$. Therefore $\mathbf{E}\left[y[k]y^T[k]\right]$ is a

constant matrix with respect to n. Because $\mathbf{E}\left[y^T[k]y[k]\right] = Trace\left(\mathbf{E}\left[y[k]y^T[k]\right]\right)$, $\mathbf{E}\left[y^T[k]y[k]\right]$ is constant with respect to n and the index n in the output power can be dropped. We measure the system performance by $\mathbf{E}\left[y^T y\right]$.

3.3 Performance of control systems with feedback dropouts governed by a Markov chain

As mentioned in the beginning of this chapter, we measure the system performance by the output power. This section computes that power through a group of Lyapunov equations in theorem 3.3.1. The computed power is a function of the dropout Markov chain's transition matrix. Simulation results are presented to support the correctness of the result. In theorem 3.3.1, the power is measured by the ensemble average of the output power. In simulations, the power was measured by the the time average of the output power. Only when the system is ergodic are the two kinds of averages equal to each other. Theorem 3.3.2 provides sufficient conditions for the system to be ergodic.

Theorem 3.3.1 *Consider the system in equation 3.1.1 under assumptions 1—5. Denote the conditional correlations as* $P_i = \pi_i \mathbf{E}\left[x[k]x^T[k] \mid q[k-1] = q_i\right]$ *for* $i = 1, 2, \cdots, N$ [2] *. Then the power of* y *can be computed as*

$$\mathbf{E}\left[y^T y\right] = \text{Trace}\left(C \sum_{i=1}^{N} P_i C^T\right). \tag{3.3.3}$$

where P_i *satisfies the equation*

$$P_i = A_i \sum_{k=1}^{N} q_{ki} P_k A_i^T + \pi_i B B^T, \tag{3.3.4}$$

[2]It can be shown that $\mathbf{E}\left[x[k]x^T[k] \mid q[k-1] = q_i\right]$ is constant with respect to n due to the wide sense stationarity in Theorem 3.2.2.

$A_i = A_{close}$ *when* $f(q_i) = 0$, $A_i = A_{open}$ *when* $f(q_i) = 1$.

In [16], there is a formal computation of output power for a jump linear system. Theorem 3.3.1 can be viewed as an application of the results in [16]. We provide a proof of theorem 3.3.1 in the appendix for completeness.

Remark: Note that the existence of $P_i \geq 0$ in eq. 3.3.4 is equivalent to the mean square stability of the system [16]. From eq. 3.3.4, we see that $\mathbf{E}\left[y^T y\right]$ is a function with respect to $Q = (q_{ji})$, denoted as $f(Q)$. Let $f(Q) = +\infty$ if eq. 3.3.4 does not have non-negative definite solution, i.e. the system is not mean square stable.

Theorem 3.3.1 provides a method for predicting the output signal's power. We did simulations to verify the correctness of the prediction. We created a `Matlab simulink` model to generate the system's output signal under specified dropout conditions. We used these output traces to compute a time average of the output signal's power. This time average was taken as an estimate of the system's true expected output power $\mathbf{E}\left[y^T y\right] = Trace\left(devec\left(\mathbf{E}\left[y^{[2]}\right]\right)\right)$, where the operators $devec(\cdot)$ and $^{[2]}$ are defined in Appendix A. For a sample path of length $2L$ time steps, the time average can be computed as

$$\hat{\mathbf{E}}_L\left[y^T y\right] = Trace\left(devec\left(\hat{\mathbf{E}}_L\left[y^{[2]}\right]\right)\right), \tag{3.3.5}$$

where

$$\hat{\mathbf{E}}_L\left[y^{[2]}\right] = \frac{1}{L}\left(y^{[2]}[L+1] + y^{[2]}[L+2] + \cdots + y^{[2]}[L+L]\right). \tag{3.3.6}$$

The following theorem provides the conditions for $\hat{\mathbf{E}}_L\left[y^{[2]}\right]$ to approach $\mathbf{E}\left[y^{[2]}\right]$. The proof of theorem 3.3.2 is moved to the appendix for readability.

Theorem 3.3.2 *Under the assumptions of theorem 3.3.1, if the following two additional assumptions are satisfied,*

1. *w is i.i.d., whose fourth order moments exist, i.e.* $\mathbf{E}\left[\left(w^T[k]w[k]\right)^2\right] < \infty$.
2. $A_{[4]} = diag(A_i^{[4]})_{i=1,\cdots,N}(Q^T \otimes I_{n^4})$ *is stable, where n is the dimension of the system in 3.1.1.*

Then

$$\lim_{L\longrightarrow\infty} \hat{\mathbf{E}}_L\left[y^{[2]}\right] = \mathbf{E}\left[y^{[2]}\right] \text{ in mean square sense,} \tag{3.3.7}$$

where $\hat{\mathbf{E}}_L\left[y^{[2]}\right]$, the time average of the process $\{y^{[2]}[k]\}$, is computed through equation 3.3.6.

Remark: Theorem 3.3.2 may be used to determine the conditions for the time-average power $\hat{\mathbf{E}}_L\left[y^T y\right]$ (equation 3.3.5) to converge to the true ensemble average $\mathbf{E}\left[y^T y\right]$ predicted in theorem 3.3.1. When the conditions are satisfied, we can verify theorem 3.3.1 by simply comparing $\hat{\mathbf{E}}_L\left[y^T y\right]$ and $\mathbf{E}\left[y^T y\right]$. Otherwise, no conclusions can be made.

We applied theorems 3.3.1 and 3.3.2 on a simple example. The assumed plant was unstable with transfer function of $H(z) = \frac{z+2}{z^2+z+2}$. Assuming positive unity output feedback, the closed loop transfer function becomes $z^{-1} + 2z^{-2}$, which is stable. The dropout process therefore switches our system between a stable configuration and an unstable configuration. The input noise, w, is white zero-mean Gaussian with unit variance. The dropout Markov chain has $N = 3$ states. Its transition matrix is $Q = \begin{bmatrix} 1-\varepsilon & \varepsilon & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$, where ε is a parameter between 0

and 1. This particular Markov chain specifies a soft (m, k) scheduling model with $m = 2$, $k = 3$ [56].

We used theorems 3.2.1 and 3.2.2 to determine that the system is mean square stable and wide sense stationary when $\varepsilon \leq 24.9\%$. By theorem 3.3.2, we know when $\varepsilon < 6.1\%$, the convergence of the estimate in equation 3.3.7 can be guaranteed. This upper bound 6.1% is smaller than the upper stability bound 24.9% predicted by theorem 3.3.1. The system with i.i.d. dropouts was simulated with various dropout rates between 0 and 6%. For each value of ε, we ran 5 different simulations for $200,000$ time steps and then estimated the output power by the time average in equation 3.3.6. The simulation results are shown in figure 3.2. The figure shows close agreement between the predicted and the experimentally estimated power of the output. So we have high confidence in the correctness of the results stated in theorems 3.3.1 and 3.3.2.

For completeness, we also present results for the system with dropout rates between 7% and 24%. Figure 3.3 shows that the predicted and the experimentally estimated power of the outputs disagree under large dropout rates. As discussed earlier, the disagreement comes from the violation of the ergodicity conditions in theorem 3.3.2. Figure 3.3 still shows close agreement for dropout rates less than 10%. This thereby shows that theorem 3.3.2 is somewhat conservative in its estimate of the ergodic interval with respect to ε.

3.4 Optimal dropout policy

Theorem 3.3.1 establishes the relationship between control performance and the transition matrix of the dropout Markov chain, Q. In reality, the channel's quality of service (QoS), e.g. the network's QoS, is usually measured by

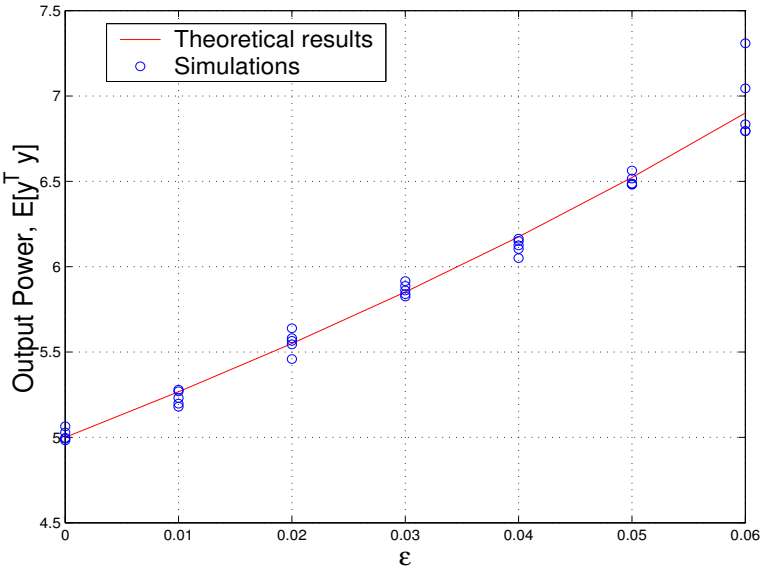


Figure 3.2. Output power($0 \leq \varepsilon \leq 6\%$)

its throughput rate (the number of packets that are successfully delivered per unit time) rather than Q. Since our feedback control systems strive to transmit a feedback measurement every sampling period, the average rate at which such feedback measurements are dropped actually measures the network's throughput.

Under some conditions, the dropout rate of a control system has to be above a certain level, i.e. some feedback packets of the system have to be dropped. The reason to drop packets lies in the resource sharing. We illustrate this motivation through a networked control system. In networked control systems, the feedback path is implemented over a network [7]. Such a network may be shared among control systems whose performance is measured by the output power and non-control systems whose performance is measured by the dropout rate. When the network is congested, every (control or non-control) system may have to drop some packets to reduce network congestion, which is exactly presented as the preceding constraint that the dropout rate is above a given threshold. There may exist

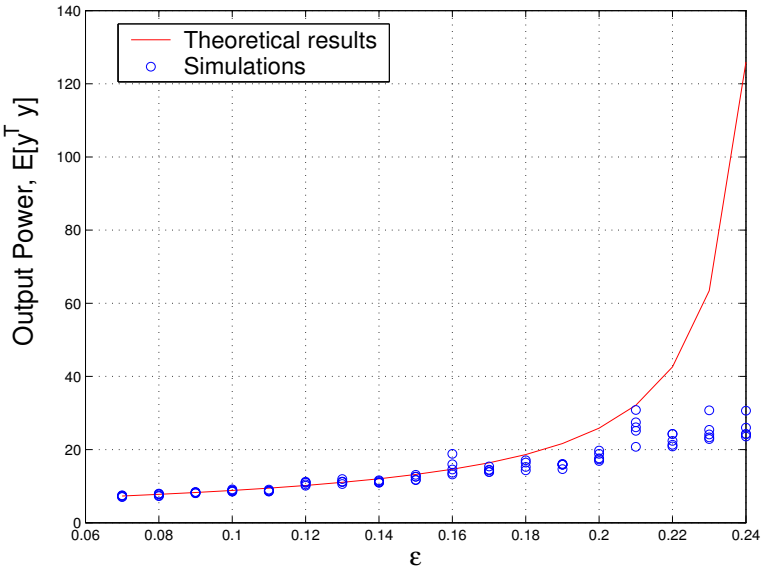


Figure 3.3. Output power($7\% \leq \varepsilon \leq 24\%$)

many dropout policies to realize the average dropout rate requirement. Our aim is to find the *optimal* dropout policy among all feasible ones. We choose the same performance measure as section 3.3, the output power. In a word, the objective of optimal dropout policy design is to minimize the output power with respect to all possible dropout policies under the average dropout rate constraint.

This section is organized as follows. In subsection 3.4.1, we review the literature on dropout policies. In subsection 3.4.2, we compare the performance of a control system under different dropout policies, where the implemented dropout policies have the same average dropout rate. Based on the comparisons, the optimal dropout policy design is formulated into an optimization problem. In subsection 3.4.3, several methods are proposed for solving the dropout policy optimization problem, including a gradient descent method and a branch-and-bound method. In subsection 3.4.4, the achieved optimal dropout policy acts as a guideline for real-time engineers to design scheduling algorithms. In subsection 3.4.5, a

helicopter simulator is used to verify the achieved optimal dropout policy.

3.4.1 Literature review on dropout policy

The impact that dropout policies have on system stability has been studied in [32] and [12]. Both of these papers consider a set of decoupled control systems that share a single communication network as the feedback paths. In [32], an admission policy is studied in which the control system with the largest error is given network access. This means that the other control systems must drop their measurements, i.e. working in the open loop. The paper then provides a sufficient condition on the control system's eigenvalues that ensures stability under the proposed admission (dropout) policy. In [12], examples are used to demonstrate that the occasional dropping of control tasks enhances overall system stability, i.e. there exists a dropout policy to both satisfy the resource sharing (dropout) requirement and guarantee the overall stability. Both of these paper only analyze the impact specific dropping protocols have on stability. They provide no guidelines for the selection of "good" dropout policy for performance.

The idea of using dropout policies to characterize network QoS has been recently employed to design real-time schedulers that are more responsive to the needs of control systems. Prior work in the real-time systems community has used ad hoc strategies such as "skip-over" [38] and the (m, k)-firm guarantee rule [57] [9] to control the impact dropouts have on control system performance. The skip-over constraint [38] can drop a packet every s consecutive transmission attempts. Systems satisfying the (m, k)-firm guarantee rule [57] guarantee that at least m out of k consecutive transmission attempts are successful. An example of a prioritized real-time system satisfying (m, k) constraints will be found in [9].

The skip-over and (m, k) constraints attempt to control the number of consecutive dropouts. The intuition behind this approach is that controlling the number of consecutive dropouts will minimize the impact such dropouts have on the application's performance. This heuristic reasoning has only been validated through simulations [57] on specific examples. This work, however, provides little in the way of concrete analysis suggesting that these particular dropout schemes always result in improved control system stability and performance.

More recently a novel Markov-chain (MC) constraint on the dropout policy was proposed [49] [50]. The MC-constraint is based on the optimal dropout policy results in [44] and further developed below. The value of the MC-constraint is that its enforcement ensures a specified level of application performance. The results in [49] [50] thereby provide a concrete example where control theoretic approaches guide the design of real-time schedulers. Subsection 3.4.4 presents such results in detail.

3.4.2 Performance comparison and problem formulation

Let's consider a system whose plant is unstable with transfer function $L(z) = \frac{z+2}{z^2+z+2}$. Assuming positive unity feedback, the closed loop transfer function becomes $z^{-1} + 2z^{-2}$, which is stable. The dropout process therefore switches our system between a stable configuration and an unstable configuration. The input disturbance, $w[k]$, is white noise with zero mean and unit variance. First we consider a special dropout process $\{d[k]\}$, which is generated by a Markov chain

$\{q[k]\}$ that has four states

$$q[k] = \begin{cases} q_1, & \text{if } d[k-1] = 0 \text{ and } d[k] = 0 \\ q_2, & \text{if } d[k-1] = 0 \text{ and } d[k] = 1 \\ q_3, & \text{if } d[k-1] = 1 \text{ and } d[k] = 0 \\ q_4, & \text{if } d[k-1] = 1 \text{ and } d[k] = 1 \end{cases}$$

In other words, the dropout decision, $d[k+1]$, is made based on the last two steps of the history, $d[k]$ and $d[k-1]$. When $q[k] = q_i$ $(i = 1, 2, 3, 4)$ the next measurement is dropped with probability ε_i. Denote the dropout rate vector as $\varepsilon = [\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4]^T$. With these notational conventions, the transition matrix from the dropout process' Markov chain is

$$Q = \begin{bmatrix} 1-\varepsilon_1 & \varepsilon_1 & 0 & 0 \\ 0 & 0 & 1-\varepsilon_2 & \varepsilon_2 \\ 1-\varepsilon_3 & \varepsilon_3 & 0 & 0 \\ 0 & 0 & 1-\varepsilon_4 & \varepsilon_4 \end{bmatrix}$$

Let $\pi = \begin{bmatrix} \pi_1 & \pi_2 & \pi_3 & \pi_4 \end{bmatrix}$ denote the steady state distribution for this Markov chain. π_i $(i = 1, \cdots, 4)$ are computed from the following equation.

$$\begin{cases} \sum_{j=1}^{4} q_{ji}\pi_j &= \pi_i, i = 1, \cdots, j \\ \sum_{i=1}^{4} \pi_i &= 1 \end{cases} \tag{3.4.8}$$

When the Markov chain characterized by Q is ergodic, equation 3.4.8 has a unique solution. Some remarks on relaxing this assumption will be placed in section 3.4.3.1.

The *average dropout rate* for this process can therefore be computed as

$$\overline{\varepsilon} = \sum_{i=1}^{4} \pi_i \varepsilon_i$$

From the structure of Q, we obtain

$$\overline{\varepsilon} = \pi_2 + \pi_4 \tag{3.4.9}$$

To emphasize the dependence of $\overline{\varepsilon}$ upon ε, we denote $\overline{\varepsilon}$ as $\overline{\varepsilon}(\varepsilon)$. In section 3.3, we denote $\mathbf{E}\left[y^T y\right]$ as $f(Q)$. Here Q is determined by ε. So we may replace $f(Q)$ with $f(\varepsilon)$.

A directed graph representing this 4-state Markov chain is shown in figure 3.4. We can change the actual dropout policy by changing the probabilities, ε_i. The table in figure 3.4 defines five different dropout policies that we denote as $\mathcal{P}_i$ for $i = 0$ to 4. The table's columns show the dropout probabilities ε_i for each state q_i as a function of the average dropout rate $\overline{\varepsilon}$. These dropout policies are described below:

- Process $\mathcal{P}_0$ assumes that at any state q_i, the next measurement can be dropped with probability $\overline{\varepsilon}$. $\mathcal{P}_0$, therefore, generates dropouts in an independent and identically distributed manner and so we refer to $\mathcal{P}_0$ as the i.i.d. dropout process

- Processes $\mathcal{P}_1$ and $\mathcal{P}_2$ are special cases of a dropout process that adheres to the so-called (m, k)-firm guarantee rule [57]. The (m, k)-firm guarantee rule is a heuristic constraint used by real-time system engineers. Dropout processes satisfying the (m, k)-firm guarantee rule require that there are least m successful transmissions in k consecutive attempts. A quick inspection

shows that process $\mathcal{P}_1$ satisfies a $(1,3)$-firm guarantee constraint and that process $\mathcal{P}_2$ satisfies a $(2,3)$-firm guarantee constraint.

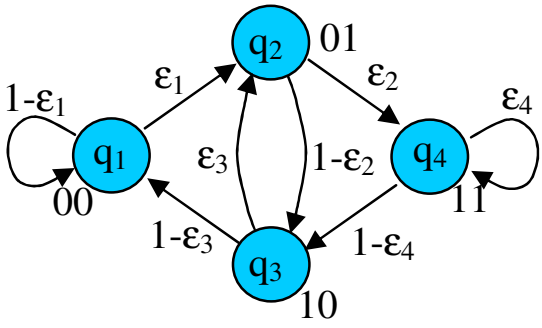
- Process $\mathcal{P}_3$ is a dropout process that satisfies the constraint that two consecutive dropouts occur with probability ε_1. So we refer to this as the "drop-2" process.

- Process $\mathcal{P}_4$ is a variant of process $\mathcal{P}_3$ that allows extra dropouts with a probability of ε'. Note that as ε' goes to zero, process $\mathcal{P}_4$ approaches the drop-2 process, $\mathcal{P}_3$.

When the dropout process adheres to a Markov chain, then there may be a maximum achievable average dropout rate, $\overline{\varepsilon}_{\max}$. For instance, since process $\mathcal{P}_2$ never drops more than 1 packet in 3 consecutive tries, its maximum average dropout rate will be one third. The maximum achievable dropout rates for each dropout process are tabulated in table 3.4.

Given a dropout process, we may use theorem 3.2.1 to determine the largest average dropout rate for which the control system is mean square stable. Let $\overline{\varepsilon}_s$ denote this largest stable dropout rate. A direct application of theorem 3.2.1, for example, shows that our control system is mean square stable under dropout process, $\mathcal{P}_2$, if and only if $\overline{\varepsilon} < 0.1662$. For this dropout process, therefore, $\overline{\varepsilon}_s = 0.1662$. The values for $\overline{\varepsilon}_s$ are tabulated in table 3.4. From this table it is apparent that the system's stability is highly dependent on the random process driving the data dropouts.

Given a dropout process, we may use theorem 3.3.1 to determine the output signal power of the example's control system. The results from this computation are shown in figure 3.5. In this figure, we see that the output signal power is a monotone increasing function of the average dropout rate, $\overline{\varepsilon}$. How quickly



Dropout Process	Dropout rate for state q_i (i=1-4) ε_1	ε_2	ε_3	ε_4	Maximum dropout rate $\overline{\varepsilon}_{max}$	Largest stable dropout rate $\overline{\varepsilon}_s$
$\mathcal{P}_0$ (i.i.d.)	$\overline{\varepsilon}$	$\overline{\varepsilon}$	$\overline{\varepsilon}$	$\overline{\varepsilon}$	$\overline{\varepsilon}$	0.189
$\mathcal{P}_1$ (1,3)	$-\frac{1}{2}+\frac{\sqrt{1+3\overline{\varepsilon}}}{2-2\overline{\varepsilon}}$	$-\frac{1}{2}+\frac{\sqrt{1+3\overline{\varepsilon}}}{2-2\overline{\varepsilon}}$	$-\frac{1}{2}+\frac{\sqrt{1+3\overline{\varepsilon}}}{2-2\overline{\varepsilon}}$	0	.667	0.189
$\mathcal{P}_2$ (2,3)	$\frac{\overline{\varepsilon}}{1-2\overline{\varepsilon}}$	0	0	0	0.333	0.166
$\mathcal{P}_3$ (drop 2)	$\frac{\overline{\varepsilon}}{2-3\overline{\varepsilon}}$	1	0	0	0.500	0.500
$\mathcal{P}_4$ (drop 2) ($\varepsilon'=0.3$)	$\frac{\overline{\varepsilon}(1-\varepsilon')}{2-\varepsilon'-(3-2\varepsilon')\overline{\varepsilon}}$	1	0	ε'	0.556	0.406

Figure 3.4. (top) Directed graph of 4-state Markov chain; (bottom) Table characterizing different dropout policies $\mathcal{P}_i$ $(i = 0 - 4)$ as a function of the average dropout rate $\overline{\varepsilon}$.

the output signal power increases, however, is dependent on the type of dropout process driving the system. In particular, for a given average dropout rate, the figure shows large difference between the performance levels achieved by different dropout policies. These results, therefore, suggest to us that the average dropout rate $\overline{\varepsilon}$ is a poor predictor of application (i.e. control system) performance. If we are to constrain network dropouts in a manner that provides guarantees on application performance, the results in figure 3.5 suggest we should attempt to

force the dropout process to adhere to an underlying Markov process.

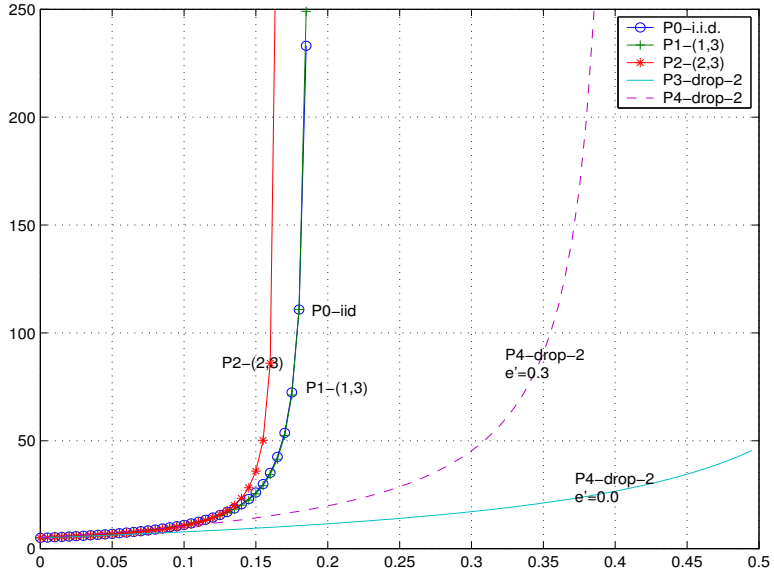


Figure 3.5. Performance of various dropout processes

It is interesting to observe that the drop-2 process, $\mathcal{P}_3$, significantly outperforms the dropout process $\mathcal{P}_2$ satisfying the $(2,3)$-firm guarantee rule. Not only does the drop-2 policy have lower output power, but its maximum stable dropout rate $\overline{\varepsilon}_s$ is much larger than the rate for dropout process $\mathcal{P}_2$. This is somewhat surprising, as the (m,k)-firm guarantee rule was suggested as a means of limiting the effect dropped data have on control system performance [57]. From a heuristic standpoint, the reasoning behind the (m,k) rule seems very plausible. By limiting consecutive dropouts, we may limit the effect of dropouts on system behavior. The results in figure 3.5, however, demonstrate that this reasoning need not always

hold true. For our particular control system, we get better performance by forcing the real-time system to always drop two consecutive measurements; a policy that runs directly counter to the intuition behind the (m, k)-firm guarantee rule. The optimality of the drop-2 rule will certainly not be valid for all control systems, but it is true in this particular case. It suggests, to our mind, that control theory may provide a useful tool by which engineers can guarantee the performance of real-time control systems in the presence of dropped data.

The fact that the drop-2 process outperforms all of the other dropout processes is no accident. In reviewing the statement of theorem 3.3.1, it is apparent that we can formulate an optimization problem of the following form

$$\begin{aligned} &\min_{\varepsilon_1,\cdots,\varepsilon_4} \quad f(\varepsilon), \\ &\text{subject to:} \quad \overline{\varepsilon}(\varepsilon) \geq \varepsilon_0 \\ &\qquad\qquad\quad 0 \leq \varepsilon_i \leq 1, i = 1, \cdots, 4 \end{aligned} \tag{3.4.10}$$

This optimization problem seeks to minimize the output signal power, with respect to the individual dropout probabilities ε_i in the Markov chain's transition matrix. A solution to this optimization problem, if it exists should be the "optimal" dropout process generated by a 4-state Markov chain. The constraints imposed on this optimization problem arise from three sources. The first is the structure constraints imposed by eq. 3.3.4 and 3.4.8. The second is the implicit stability constraint. It is imposed by setting $f(\varepsilon_i) = +\infty$ when the system is not mean square stable. The last one that $\overline{\varepsilon}(\varepsilon) \geq \varepsilon_0$ simply requires that the average dropout rate be greater than ε_0. The specified drop rate ε_0 has the physical interpretation of the resource occupation that has been *allocated* to the control system, thereby leaving sufficient space for other users. The regularizing constant

ε_0 is chosen by the resource manager and the resulting solution (if it exists) will always be a Markov chain whose average dropout rate equals ε_0.

We now consider the optimization problem 3.4.10. One may arise the following questions, "Does the problem have globally optimal solution", "how can we find the globally optimal solution if it exists?", etc. Because optimization problem 3.4.10 takes a nonlinear form, we don't have trivial answers to the previous questions. Subsection 3.4.3 will answer the questions concerned with optimization problem 3.4.10.

Based on the methods in subsection 3.4.3, we can solve optimization problem 3.4.10. Figure 3.6 plots the output signal power for this particular control system as a function of the average dropout rate for the "optimal dropout process". Solving optimization problem 3.4.10 for $\varepsilon_0 < 0.5$ produces the dropout process $\mathcal{P}_3$, *"drop-2"* described by the table in figure 3.4. Note, however, that this Markov chain is unable to support dropout rates greater than 0.5. It is still possible, however, to solve optimization problem 3.4.10 for average dropout rates above 0.5. In this case, we obtain a different optimal Markov chain whose structure is shown in figure 3.6. In other words, the optimal dropout process changes its structures in a discontinuous fashion as we increase the average dropout rate. In this particular example it was impossible to find a 4-state Markov chain dropout process that was able to support average dropout rates in excess of 0.533.

3.4.3 Solving optimization problem 3.4.10

3.4.3.1 Computations on $f(\varepsilon)$ and $\overline{\varepsilon}(\varepsilon)$

$f(\varepsilon)$ and $\overline{\varepsilon}(\varepsilon)$ are governed by coupled Lyapunov equations in 3.3.4 and 3.4.8. We assume that the markov chain characterized by Q is ergodic. Under that

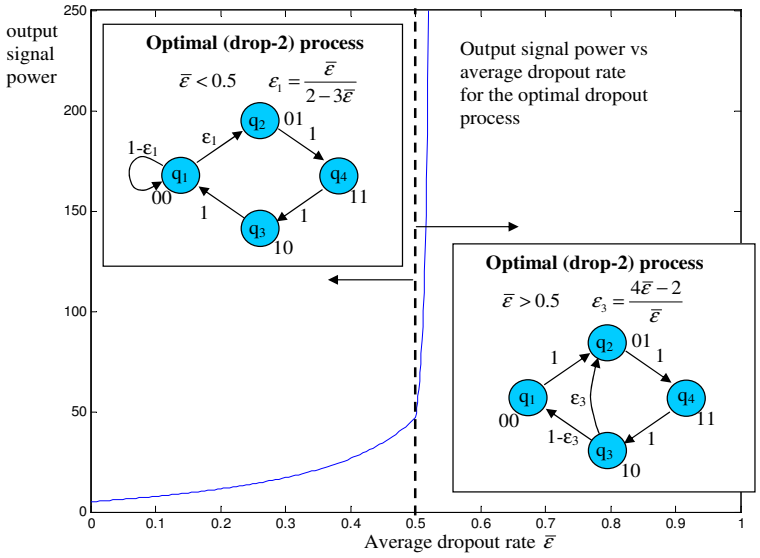


Figure 3.6. Performance vs. average dropout rate under the optimal dropout process

condition, we know the solution to 3.4.8 ($\{\pi_i, i = 1, \cdots, N\}$) exists and is unique. If the system is also mean square stable, then the solution to eq. 3.3.4 ($\{P_i, i = 1, \cdots, N\}$) also exists and is unique. Eq. 3.3.4 can be solved through the following LMI (linear matrix inequality) method.

$$\begin{cases} & f(\varepsilon_i) = \min_{P_i} Trace\left(C \sum_{i=1}^{N} P_i C^T\right) \\ s.t. & P_i - \alpha I < \sum_{j=1}^{N} q_{ji} A_j P_j A_j^T + \pi_i B B^T < P_i + \alpha I \\ & P_i \geq 0, i = 1, \cdots, N \end{cases} \tag{3.4.11}$$

where α is a small positive number to transform equality constraints to their strict inequality versions. It is interesting to note that the LMI problem in eq. 3.4.11 has feasible solutions if and only if the system is mean square stable [16]. Therefore we assign $f(\varepsilon) = +\infty$ if the LMI problem has no solution, which is consistent with our previous setting when the system is not mean square stable.

From eq. 3.3.4 and 3.4.8, we can get not only $f(\varepsilon)$ and $\overline{\varepsilon}(\varepsilon)$ but also their derivatives. Suppose the solution to eq. 3.4.8 is $\{\pi_i\}$. Because $\overline{\varepsilon}(\varepsilon) = \pi_2 + \pi_4$, we know

$$\frac{\partial \overline{\varepsilon}}{\partial \varepsilon_l} = \frac{\partial \varepsilon_2}{\partial \varepsilon_l} + \frac{\partial \varepsilon_4}{\partial \varepsilon_l} \tag{3.4.12}$$

where $l = 1, \cdots, N$ (N=4). We compute $\frac{\partial \pi_i}{\partial \varepsilon_l}$ by taking the derivative of eq. 3.4.8 with respect to ε_l

$$\begin{cases} \sum_{j=1}^{N} \frac{\partial q_{ji}}{\partial \varepsilon_l} \pi_j + \sum_{j=1}^{N} q_{ji} \frac{\partial \pi_j}{\partial \varepsilon_l} &= \frac{\partial \pi_i}{\partial \varepsilon_l}, i = 1, \cdots, N \\ \sum_{i=1}^{N} \frac{\partial \pi_i}{\partial \varepsilon_l} &= 1 \end{cases} \tag{3.4.13}$$

In the above equations, $\frac{\partial q_{ji}}{\partial \varepsilon_l}$ and π_j are known. Because of the ergodicity of the Markov chain, eq. 3.4.13 has a unique solution with respect to $\{\frac{\partial \pi_i}{\partial \varepsilon_l}\}_{i=1}^{N}$ for any $l = 1, \cdots, N$.

The above method can also be applied to compute $\frac{\partial f(\varepsilon)}{\partial \varepsilon_l}$.

$$\frac{\partial f(\varepsilon)}{\partial \varepsilon_l} = Trace\left(C \sum_{i=1}^{N} \frac{\partial P_i}{\partial \varepsilon_l} C^T \right) \tag{3.4.14}$$

$$s.t. \quad \sum_{j=1}^{N} q_{ji} A_j \frac{\partial P_j}{\partial \varepsilon_l} A_j^T + \sum_{j=1}^{N} \frac{\partial q_{ji}}{\partial \varepsilon_l} A_j P_j A_j^T + \frac{\partial \pi_i}{\partial \varepsilon_l} B B^T = \frac{\partial P_i}{\partial \varepsilon_l} \tag{3.4.15}$$

Again we know eq. 3.4.15 has a unique solution with respect to $\{\frac{\partial P_i}{\partial \varepsilon_l}\}_{i=1}^{N}$ if the system is mean square stable. Therefore we obtain the derivatives of $f(\varepsilon)$ in the mean square stable systems.

Remark: In the above discussions, we assume the Markov chain is ergodic. When $0 < \varepsilon_i < 1$ $(i = 1, \cdots, N)$, the Markov chain is ergodic. If that ergodicity assumption is violated, what will happen? Under that situation, the Markov chain

has multiple stationary distributions. For every stationary distribution, we can evaluate its average dropout rate and output power (performance). Among the stationary distributions satisfying the average dropout rate constraint, we will choose the one with the best performance (the minimum output power).

3.4.3.2 The existence of global optimal solution to optimization problem 3.4.10

The feasible set of optimization problem 3.4.10, denoted as S_f, is the cross product of the set $\{\varepsilon|0 \leq \varepsilon_i \leq 1, i = 1, \cdots, N\}$, the set $\{\varepsilon|\overline{\varepsilon}(\varepsilon) \geq \varepsilon_0\}$ and the set $\{\varepsilon|f(\varepsilon) < \infty\}$. The last set is really the one where $A_{[2]} = \left(Q^T \otimes I\right) diag(A_i \otimes A_i)_{i=1}^N$ is Schur stable, i.e. $\lambda_{max}(A_{[2]}) < 1$ ($\lambda_{max}(\cdot)$ denotes the maximum magnitude of eigenvalues of a matrix). Because $\overline{\varepsilon}(\varepsilon)$ is a continuous function with respect to ε, the set $\overline{\varepsilon}(\varepsilon_i) \geq \varepsilon_0$ is a closed set. Because $\lambda_{max}(A_{[2]})$ continuously depends on ε, $\lambda_{max}(A_{[2]}) \leq 1$ specifies a closed set. Therefore the set $S_c = \{\varepsilon|0 \leq \varepsilon_i \leq 1\} \cap \{\varepsilon|\overline{\varepsilon}(\varepsilon) \geq \varepsilon_0\} \cap \{\varepsilon|\lambda_{max}(A_{[2]}) \leq 1\}$ is a closed set. The difference between S_f and S_c is only the boundary, $\{\varepsilon_i|\lambda_{max}(A_{[2]}) = 1)\}$. From eq. 3.3.4, we know $f(\varepsilon)$ is $+\infty$ on that boundary. Therefore we know

$$\min_{\varepsilon \in S_f} f(\varepsilon) = \min_{\varepsilon \in S_c} f(\varepsilon) \tag{3.4.16}$$

Because of the continuity of $f(\varepsilon)$ for $\varepsilon \in S_c$ and S_c is a closed set, we know $f(\varepsilon)$ has a globally optimal (minimum) solution over S_c, which is denoted as $f(\varepsilon^*)$. Again from the continuity of $f(\varepsilon)$, we know for any $\delta > 0$, there exists θ such that

$$f(\varepsilon) - f(\varepsilon^*) < \delta f(\varepsilon^*), \forall \|\varepsilon - \varepsilon^*\|_2 < \theta \tag{3.4.17}$$

This property will be used to guarantee that the global optimal solution can be determined after a finite number of steps.

3.4.3.3 Gradient method to solve optimization problem 3.4.10

As we show in section 3.4.3.1, $f(\varepsilon)$, $\overline{\varepsilon}(\varepsilon)$ and their derivatives all can be computed. So we can use the gradient descent method to solve optimization problem 3.4.10. In section 3.4, we used the gradient descent method of MATLAB function $fmincon()$ to solve optimization 3.4.10. That method needs only the value functions of $f(\varepsilon)$ and $\overline{\varepsilon}(\varepsilon)$ and estimates the derivatives from the value functions.

The gradient descent method converges very fast. That method, however, may be stuck at local optima. Only when the performance index and the feasible set are all convex with respect to its arguments is the gradient descent method guaranteed to converge to the global optimal point. Unfortunately we don't know whether $f(\varepsilon)$ and $\overline{\varepsilon}(\varepsilon)$ are convex with respect to ε. So it is quite possible that the gradient descent method converges to a local optimal solution. We will search the globally optimal solution using a branch-and-bound method in the following subsection.

3.4.3.4 Branch-and-bound method to solve optimization problem 3.4.10

Optimization problem 3.4.10 is nonlinear. The branch-and-bound method will relax that problem into a linear problem, which will provide lower and upper bounds on the globally optimal solution. These lower and upper bounds will be loose when the feasible set is "large". By branching operations, the method partitions the feasible set into smaller subsets, over which the bounds will be tighter. After enough branching operations, the lower bound will approach the

upper bound. We can use the gap between the two bounds to approximate the error in our approximately optimal solution. During the branching operations, many subsets will be produced. In order to improve computational efficiency, we have to prune the generated subsets. The above bounding, branching and pruming ideas come from [64]. Their applications to our example will be explained as follows.

3.4.3.4.1 **Bounds on** $f(\varepsilon^*)$ Suppose ε lies in a box $B = [\underline{\varepsilon}, \overline{\varepsilon}]$ [3]. Following the arguments in section 3.4.3.2, we know that there exists an ε in $B \cap S_f$ that minimizes $f(\varepsilon)$. Let ε_B^* denote this optimal ε and let $f(\varepsilon_B^*)$ denote the performance at ε_B^*. $f(\varepsilon_B^*)$ is an upper bound on $f(\varepsilon^*)$.

$$f(\varepsilon^*) \leq f(\varepsilon_B^*) \tag{3.4.18}$$

We state the following upper and lower bounds on $f(\varepsilon_B^*)$.

Upper bound on $f(\varepsilon_B^*)$**:** For any $\varepsilon' \in B$ and $\overline{\varepsilon}(\varepsilon') \geq \varepsilon_0$, it is true that

$$f(\varepsilon') \geq f(\varepsilon_B^*) \tag{3.4.19}$$

It particularly holds for $\overline{\varepsilon}$, one vertex of B, i.e.

$$f(\overline{\varepsilon}) \geq f(\varepsilon_B^*), \text{if } \overline{\varepsilon}(\overline{\varepsilon}) \geq \varepsilon_0 \tag{3.4.20}$$

Lower bound on $f(\varepsilon_i^*)$**:** In optimization problem 3.4.10, ε_i has both a lower bound $\underline{\varepsilon}_i$ $(= 0)$ and an upper bound $\overline{\varepsilon}_i (= 1)$. These bounds are known. Based on the expression for Q, we obtain lower and upper bounds of q_{ji}, denoted as $\underline{q}_{ji}$

[3] $[\underline{\varepsilon}, \overline{\varepsilon}]$ stands for the set $\{\varepsilon = [\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4]^T | \underline{\varepsilon}_i \leq \varepsilon_i \leq \overline{\varepsilon}_i, i = 1, \cdots, 4\}$.

and $\overline{q}_{ji}$. As argued before, $f(\varepsilon)$ and $\overline{\varepsilon}(\varepsilon)$ are computed by solving eq. 3.3.4 and 3.4.8. These equations are non-linear (bi-linear to be more precise) with respect to variables ε_i, P_i and π_i. It is difficult to solve such non-linear equations. If we substitute the upper or lower bounds $\overline{q}_{ji}$ and $\underline{q}_{ji}$ for q_{ji} for these equations, we will get linear inequalities with respect to P_i and π_i. Eq. 3.3.4, for example, is relaxed into

$$\sum_{j=1}^{N} \underline{q}_{ji} A_j P_j A_j^T + \pi_i B B^T \leq P_i \tag{3.4.21}$$

$$\sum_{j=1}^{N} \overline{q}_{ji} A_j P_j A_j^T + \pi_i B B^T \geq P_i \tag{3.4.22}$$

We therefore formulate an optimization problem over these relaxed linear inequality constraints.

$$\begin{array}{ll} \min_{P_i,\pi_i} & Trace\left\{C \sum_{i=1}^{N} P_i C^T\right\} \\ \text{subject to} & \sum_{j=1}^{N} \underline{q}_{ji} A_j P_j A_j^T + \pi_i B B^T \leq P_i \\ & \sum_{j=1}^{N} \overline{q}_{ji} A_j P_j A_j^T + \pi_i B B^T \geq P_i \\ & \pi_2 + \pi_4 \geq \varepsilon_0 \\ & \sum_{j=1}^{N} \underline{q}_{ji} \pi_j \leq \pi_i \\ & \sum_{j=1}^{N} \overline{q}_{ji} \pi_j \geq \pi_i \\ & \sum_{i=1}^{N} \pi_i = 1 \end{array} \tag{3.4.23}$$

Denote the optimal solution to the above problem as $\underline{f}(\underline{\varepsilon}, \overline{\varepsilon})$. It is obvious that the feasible set for optimization problem 3.4.10 is a subset of that for optimization problem 3.4.23. The two optimization problems have the same performance

measurement, so

$$\underline{f}(\underline{\varepsilon}, \overline{\varepsilon}) \leq f(\varepsilon_B^*). \tag{3.4.24}$$

We now introduce a small positive number to convert the non-strict inequalities in optimization problem 3.4.23 into strict inequalities and then solve the resulting problem using the MATLAB LMI toolbox.

When $\overline{\varepsilon}_i - \underline{\varepsilon}_i$ is large, the above two bounds, especially the lower bound $\underline{f}(\underline{\varepsilon}, \overline{\varepsilon})$, can be very loose. But when $\overline{\varepsilon}_i - \underline{\varepsilon}_i$ is small, the two bounds will be quite tight. So an intuitive idea is to decrease $\overline{\varepsilon}_i - \underline{\varepsilon}_i$ in a systematic way. When this difference between $f(\overline{\varepsilon})$ and $\underline{f}(\underline{\varepsilon}, \overline{\varepsilon})$ is smaller than a specified tolerance, we can say that we have bounded the global optimal solution to the given tolerance level. The above arguments are realized by the branching procedure in the following subsection.

3.4.3.4.2 **Branching operation** Suppose we start from a box $B^{(0)} = [\underline{\varepsilon}^{(0)}, \overline{\varepsilon}^{(0)}]$. We equally partition every side of this box into two parts and get 2^N sub-boxes $B^{(1,j)} = [\underline{\varepsilon}^{(1,j)}, \overline{\varepsilon}^{(1,j)}]$ $(j = 1, \cdots, 2^N)$. It is interesting to note that

$$\overline{\varepsilon}_i^{(1,j)} - \underline{\varepsilon}_i^{(1,j)} = \frac{\overline{\varepsilon}_i^{(0)} - \underline{\varepsilon}_i^{(0)}}{2} \tag{3.4.25}$$

We therefore expect to get tighter lower and upper bounds on for all sub-boxes than the ones generated from the original box, i.e.

$$f(\overline{\varepsilon}^{(1,j)}) - \underline{f}(\underline{\varepsilon}^{(1,j)}, \overline{\varepsilon}^{(1,j)}) \leq f(\overline{\varepsilon}^{(0)}) - \underline{f}(\underline{\varepsilon}^{(0)}, \overline{\varepsilon}^{(0)}), j = 1, \cdots, 2^N. \tag{3.4.26}$$

The cost for this improvement is that we have to solve 2^N instances of optimization problem 3.4.23.

If we branch all available boxes at every step, then all side lengths will shrink to 2^{-L} of their original lengths after L steps of branching. The global optimal solution is inside one of these subsets. If L is large enough, we know the distance between the optimal point and any other point in the subset, $\frac{1}{2^L}\sqrt{N}$, is less than θ so that the performance difference is less than the specified level δ by eq. 3.4.17 (if $\|\varepsilon - \varepsilon^*\|_2 < \theta$, then $f(\varepsilon) - f(\varepsilon^*) < \delta f(\varepsilon^*)$). Under that situation, we may say that the global optimal solution has been achieved with a tolerance of δ. Of course, the smaller the δ, the more steps (the larger L) are needed. We now can give an upper bound on L to achieve the distance θ.

$$L_{max} = log_2 \left(\frac{\sqrt{N}}{\theta} \right) \tag{3.4.27}$$

3.4.3.4.3 **Pruning operation** Although we may achieve the global optimal solution within a specified tolerance by the above branching operations, there are a huge number of subsets generated by the branching operations. If we partition all subsets at every step, we will have 2^{LN} subsets after L-step branching, which may be beyond the available computing capability. So we must therefore prune the achieved subsets. The pruning criterion is based on two types of constraints.

Average dropout rate constraint If the maximum dropout rate of a subset is below the required level ε_0, then that subset should be eliminated. The set we consider is a box $[\underline{\varepsilon}, \overline{\varepsilon}]$. It can be proven that the maximum dropout rate is achieved by the upper vertex $\overline{\varepsilon}$ [4]. Correspondingly, the minimum dropout rate is achieved by the lower vertex $\underline{\varepsilon}$. If $\overline{\varepsilon}(\overline{\varepsilon}) < \varepsilon_0$, the subset $[\underline{\varepsilon}, \overline{\varepsilon}]$ will be eliminated.

[4] This statement is based on the proposition:

Proposition 3.4.1 *If $\varepsilon \geq \varepsilon'$ (in the sense that $\varepsilon_i \geq \varepsilon'_i$ $(i = 1, \cdots, N)$), then $\overline{\varepsilon}(\varepsilon) \geq \overline{\varepsilon}(\varepsilon')$.*

Upper bound of the global optimal solution The performance of any feasible point is an upper bound of the global optimal solution. If we know a feasible point ε' with performance $f(\varepsilon')$ and the solution to the relaxed LMI problem for a subset is above $f(\varepsilon')$, we will eliminate that subset.

Which feasible points should we use to estimate the upper bound on the globally optimal solution? We can choose the vertices of the subset. If a vertex satisfies the average dropout rate constraint, we will compute its performance. We store the best (minimum) performance of all available points in a variable $min\_perform$. Consider a subset $B' = [\underline{\varepsilon}', \overline{\varepsilon}']$. If

$$\underline{f}(\underline{\varepsilon}', \overline{\varepsilon}') \geq min\_perform, \tag{3.4.28}$$

then we discard B' because the performance will not be improved by searching the points of B'.

Besides the above pruning methods, we usually list all available subsets in an increasing order by $\underline{f}(\underline{\varepsilon}, \overline{\varepsilon})$. At each step we will branch only the first set in that list. The resulting 2^N subsets are then inserted back into the list based on their values of $\underline{f}(\underline{\varepsilon}, \overline{\varepsilon})$. The variable $min\_perform$ is updated by checking the vertices of the 2^N subsets. If $min\_perform$ is decreased, then the pruning operations are performed for the subsets in the list. This pruming mehtod can usually improve the computational efficiency by discarding the "useless" subsets.

3.4.3.5 Combination of gradient descent method and branch-and-bound method

Although the branch-and-bound method can identify the globally optimal solution (within a given tolerance), that method is computationally intensive. The gradient descent method usually converges quickly but to a locally optimal solu-

tion. We hope that by combining both methods, we can improve the computational efficiency of the branch-and-bound method.

The gradient descent method will always stop at a *feasible* point if a feasible initial condition is given. That feasible point can be used to update $min\_perform$, i.e. the best performance of available feasible points. The execution of the gradient descent method may help greatly in pruning the unnecessary subsets.

3.4.3.6 Algorithms for searching optimal dropout policy

Here we formally write down the ideas in subsection 3.4.3.5 in an algorithmic form. There are two searching strategies, the breadth-first (BF) and depth-first (DF) strategies. We describe the BF strategy in detail and briefly mention the DF strategy. At the end of this subsection, we will analyze the complexity of the algorithms, especially the DF strategy.

Breadth-first algorithm

1. Execute the gradient descent method. $min\_perform$ is set equal to the performance value attained by the "'optimal"' (probably locally optimal) solution.

2. Initialize a list of subsets $\mathcal{L} = \{B^{(0)}\}$, where $B^{(0)} = [\underline{\varepsilon}, \overline{\varepsilon}]$ is specified by the bounds $\underline{\varepsilon}_i = 0$, $\overline{\varepsilon}_i = 1$ $(i = 1, \cdots, N)$. Initialize the step index $k = 0$.

3. Choose the first subset in $\mathcal{L}$, $B^{(k)}$, and delete $B^{(k)}$ from $\mathcal{L}$, i.e. $L = L/B^{(k)}$. Check whether $\underline{f}(\underline{\varepsilon}, \overline{\varepsilon})$ is large enough over $B^{(k)}$, i.e.

$$\underline{f}(\underline{\varepsilon}, \overline{\varepsilon}) \geq \frac{1}{1+\delta} min\_perform, \tag{3.4.29}$$

   If eq. 3.4.29 is satisfied, then we have achieved globally optimal solution

within a tolerance of δ and we stop this algorithm. Otherwise, go to step 4.

4. Partition every side of $B^{(k)}$ into two equal parts. We get 2^N sub-boxes, listed as $B^{(k,j)}$ $(j = 1, \cdots, 2^N)$. For every sub-box, we do the following operations.

   - Check whether the average dropout rate constraint can be satisfied by the upper vertex of $B^{(k,j)}$. If not, we know the average dropout rate constraint can't be satisfied in $B^{(k,j)}$ and discard it. Otherwise, go forward to the next operations.
   - Compute the performance over the upper vertex of $B^{(k,j)}$, $\overline{\varepsilon}$. Update $min\_perform$ as

$$min\_perform = min(min\_perform,\ f(\overline{\varepsilon})) \tag{3.4.30}$$

   - Solve the optimization problem 3.4.23 over $B^{(k,j)}$. Check whether eq. 3.4.29 holds. If it is true, discard $B^{(k,j)}$; otherwise, insert $B^{(k,j)}$ into $\mathcal{L}$ in an increasing order of $\underline{f}(\underline{\varepsilon}, \overline{\varepsilon})$.

5. If $min\_perform$ was changed by eq. 3.4.30, check eq. 3.4.29 over all subsets of $\mathcal{L}$. If one subset satisfies eq. 3.4.29, it will be eliminated from $\mathcal{L}$.

6. If $\mathcal{L}$ is empty, the global optimal solution with tolerance of δ has been achieved by $min\_perform$; otherwise, $k = k + 1$ and go to step 3.

Depth-first (DF) algorithm The depth-first (DF) algorithm organizes all nodes (subsets) as the hierachical directed graph shown in figure 3.7.

In figure 3.7, $B^{(k)}$ is a node at the depth of k. After partitioning $B^{(k)}$, we get m $(m \leq 2^N)$ subnodes $B^{(k+1,j)}$ $(j = 1, \cdots, m)$. $B^{(k)}$ is the parents of $B^{(k+1,j)}$ as the solid arrow shows. $B^{(k+1,j)}$ $(j = 1, \cdots, m)$ are sibling to each

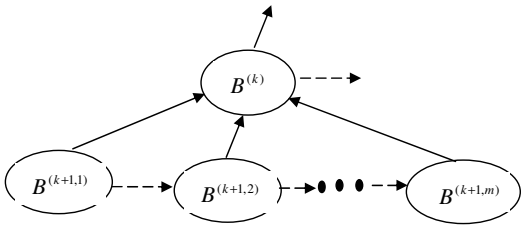


Figure 3.7. The ordering structure of all subsets in the depth-first algorithm

other. The nodes at the same depth (siblings) are ordered in an increasing way, i.e. $\underline{f}(\underline{\varepsilon}^{k+1,1}, \overline{\varepsilon}^{k+1,1}) \leq \underline{f}(\underline{\varepsilon}^{k+1,2}, \overline{\varepsilon}^{k+1,2}) \leq \cdots \leq \underline{f}(\underline{\varepsilon}^{k+1,m}, \overline{\varepsilon}^{k+1,m})$ ($B^{(k+1,j)} = [\underline{\varepsilon}^{k+1,j}, \overline{\varepsilon}^{k+1,j}]$ for $j = 1, \cdots, m$) as the dashed arrow shows. At the next step, $B^{(k+1,1)}$ will be processed.

Sometimes $m = 0$, i.e. no valid children of $B^{(k)}$ has been generated. It means that $B^{(k)}$ is useless for improving performance. We will discard it and process the next sibling of $B^{(k)}$. If $B^{(k)}$ has no sibling, we discard the parent of $B^{(k)}$ and process the sibling of that parent.

The above partitioning and retrospecting procedure will be continued until there are no nodes left in $\mathcal{L}$, which means that we have achieved the globally optimal solution within the specified tolerance.

Comparing the BF and DF algorithms, we see that the DF one digs a "hole" deep enough at one place before trying other places; the BF one digs holes everywhere. It is hard to say which one is better. For our specific problem, the two algorithms have almost the same time complexity [5] ; the DF one occupies

[5]It was shown that the globally optimal solution of that example is achieved by the gradient descent method. The aim of the branch-and-bound method is only to verify the global optimality of that solution. The BF and DF will expand the same number of nodes. So they have the same time complexity.

less memory. In the rest of the this section, we will express that time complexity scaled as a function of the order of the plant and the order of the Markov chain.

3.4.3.7 Complexity of the DF algorithm

The complexity of the algorithm depends on not only the complexity of a single step but also the number of steps.

3.4.3.7.1 **Complexity of a single step** $B^{(k)}$ is partitioned into m subnodes $B^{(k+1,j)}$ $(j = 1, \cdots, m)$. For every subnode $B^{(k+1,j)}$, we solve the LMI in equation 3.4.23. What is the complexity of that LMI? Suppose the order of the plant is M and the Markov chain has N states. There are N matrix decision variables $(P_i, i = 1, \cdots, N)$ with the order of M and N scalar decision variables $(\pi_i, i = 1, \cdots, N)$, $2N$ matrix constraints with the order of M, $2N + 2$ scalar constraints. The complexity is $O(N^3M^6)$ [26].

We evaluate one vertex of every subnode $B^{(k+1,j)}$ by solving the LMI in eq. 3.3.4. The time complexity of eq. 3.3.4 is also $O(N^3M^6)$.

Summing the above two types of complexity together, we get the complexity for a single step

$$O\left(m\left(N^3M^6\right)\right) \tag{3.4.31}$$

where $m \in [0, 2^N]$. The exact value of m depends on the pruning operations.

3.4.3.7.2 **The number of required steps** The number of required steps depends on not only the depth of the current node but also on the number of expanded subnodes at each level. Suppose the depth is d and there are m expanded

subnodes at every level. The total number of required steps is m^{d+1}. We now discuss the factors that effect m and d.

Comments on d As argued before, the lower bound $\underline{f}(\underline{\varepsilon}, \overline{\varepsilon})$ over a box B is tight when the size of B is small. At the depth of d, the length of every side of B is 2^{-d}. So the tightness of $\underline{f}(\underline{\varepsilon}, \overline{\varepsilon})$ will require large enough d.

Consider the physical meaning of ε_i $(i = 1, \cdots, N)$. They are dropout rates. For these dropout rates, the difference of 0.01 may be significant. In order to achieve the precision of 0.01 (i.e. the maximum side length is less than 0.01), d has to be larger than 7. So it is reasonable that $d > 7$.

If $min\_perform$ is updated based only on the evaluation of the vertices of boxes, i.e. no local optimization is utilized, the required d will be close to L_{max}. If we do local optimization over every box, the optimal solution (perhaps locally optimal solution) may perform better than the vertices of the box. Therefore by updating $min\_perform$ based on local optimization, we may expect d to be smaller than L_{max}.

Comments on m If there is no pruning operation, $m = 2^N$ and the number of required steps is 2^{Nd} at the depth of d. Suppose $N = 4$ and $d = 7$. That number is over 256 million. It may be impossible to deal with so many subsets. Fortunately pruning operations significantly reduce the number. For small d, the lower bound $\underline{f}(\underline{\varepsilon}, \overline{\varepsilon})$ is less tight, so m may be large. When d increases, that lower bound is tighter and more subsets are pruned so that m is reduced.

The above comments on d and m are only from a qualitative viewpoint. The exact values d and m are problem-dependent.

3.4.3.8 An example

We apply the combined method to our previous example $H(z) = \frac{z+2}{z^2+z+2}$. To our surprise, the results from the gradient descent method are globally optimal, which is guaranteed by the branch-and-bound method. Table 3.1 shows the cost to solve the optimization problem under different dropout rates and different tolerance δ. The CPU time comes from a Pentium-M 1.4GHz processor.

TABLE 3.1

THE COMPUTATION COST OF THE BRANCH-AND-BOUND METHOD ($N = 4$, $M = 3$)

ε_0 (Dropout Rate)	δ (tolerance)	Number of steps	Maximum depth	CPU times (seconds)
0.5	0.01	84	9	204.3
0.4	0.01	61	9	194.7
0.3	0.01	83	9	247.5
0.2	0.01	186	9	568.2
0.1	0.01	913	10	7041.2
0.5	0.001	102	12	245.5
0.4	0.001	89	12	305.3
0.3	0.001	131	12	422.8
0.2	0.001	324	13	2177.8

Table 3.1 consists of 5 columns, including ε_0 (the given dropout rate level), δ (the specified tolerance), number of steps, maximum depth (d), and the consumed CPU time (with the unit of second). We briefly go through table 3.1 as follows.

1. The number of computation steps is almostly proportional to the consumed CPU time because the time complexity of the DF algorithm is almost constant at every step (there is an exception for dropout rate of 0.4).

2. Under the given tolerance δ, the number of computation steps decreases as the dropout rate increases. One reason is that more subsets may be discarded due to violating the average dropout rate constraint under the higher dropout rate. Another reason is that the lower bound $\underline{f}(\underline{\varepsilon}, \overline{\varepsilon})$ is tighter under higher dropout rates than the one under lower dropout rates.

3. Under the same dropout rate, the number of computation steps and the maximum depth are both increased when δ is decreased. Smaller δ means more precise results are desired. So the size of stopping boxes will be smaller and the depth d will be increased.

Remarks: In this example, $N = 4$ and $M = 3$. By the previous analysis, we know the time complexity for a single step is almostly proportional to N^3M^6. So when N and M increases, the increasing of that complexity will be significant, although not exponentially. The branch-and-bound may work only for problems with small size, i.e. smaller N and M.

3.4.4 Guide the real-time scheduling with the achieved optimal dropout policy

In the early part of this section, we derived the optimal dropout policy, which yields the *"best"* performance under the given dropout rate constraint. One may

ask a question *whether this optimal dropout policy is useful in practice?* Affirmative answers have been obtained through our cooperation with real-time engineers.

In [49] and [50], a control system shares a single resource (a network or a CPU) with other control systems (tasks) or non-control systems [6] . An algorithm is expected to schedule the single resource among all control systems and non-control systems. It is shown that both dropout rate and window based QoS (Quality of Service) constraints, such as the *(m,k)*-firm constraint and the *skip-over* constraint, are not appropriate for control systems because of the lack of a control system performance guarantee. It is pointed out in [49] [50] that the achieved optimal dropout policy (a Markov chain) form optimization problem 3.4.10 can act as a novel QoS constraint on the control system, i.e. the scheduler tries to enforce the optimal Markov chain over the dropout decisions of that control system. In other words, our optimal policy works as a guideline to real-time scheduling algorithm design. Many simulations were done, which show that the scheduling algorithms based on the optimal Markov chain outperform previous scheduling algorithms, such as the EDF (Earliest Deadline First) algorithm. We may therefore conclude that our proposed optimal policy does benefit the real-time scheduling community. In the following, we briefly review some of simulation results in [49] [50].

3.4.4.1 Simulation results from [49]

In [49], a control system is assumed to share a single CPU resource with 4 non-control systems. It is assumed that the overall system is overloaded, i.e. certain percentage of dropouts is required. There are 3 priority groups in [49], *Must Finish* (MF), *Better Finish* (BF) and *Optional Finish* (OF) which are ordered in

[6] In the real-time scheduling community, a system is usually called a task, for example, a control task. This section interchangeably uses the two terms *system* and *task*.

the decreasing priority order. Non-control systems are assigned to either the MF group or the BF group. Control systems are assigned to either the MF group or the OF group. The tasks (systems) in the MF group and the BF group are executed by using the EDF (earliest deadline first) algorithm. The tasks (systems) in the OF group are executed by using a randomized priority assignment similar to [37].

The control system here is exactly the one in figure 3.1 with the loop function $H(z) = \frac{z+2}{z^2+z+2}$. For such control system, the optimal dropout policy is the *drop-2* policy proposed in subsection 3.4.2. This policy is shown in figure 3.8.

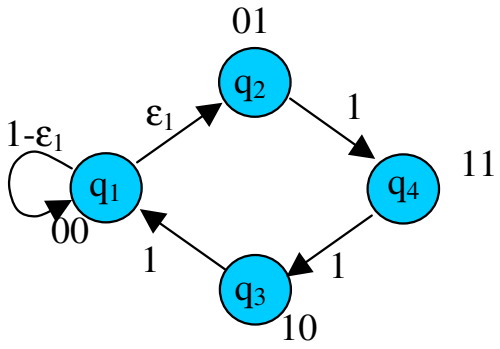


Figure 3.8. Drop-2 policy

In figure 3.8, when the state of the control system $q[n]$ stays in either q_3 or q_4, the *drop-2* policy requires to execute the control task successfully ("0" denots success and "1" denots failure), i.e. the dropout rate equals 0. So the control system is put in the MF group when $q[n] = q_3$ or $q[n] = q_4$. When $q[n] = q_2$, the

dropout rate is 1, i.e. the *drop-2* policy suggests to drop the control task to release the CPU resource. So the control system is put in the OF group when $q[n] = q_2$. When $q[n] = q_1$, the dropout rate ε_1 is between 0 and 1. *Which group should the control system be put in under the condition of* $q[n] = q_1$*?* Three different heuristic rules are proposed to answer this question.

1. **MC Driven Algorithm (MDA):** MDA first estimates the average dropout rate $\overline{\varepsilon}$ of the control sytem by some offline algorithms. Then the individual dropout rate ε_1 is got from the relationship between $\overline{\varepsilon}$ and ε_1 in figure 3.8. When $q[n] = q_1$, a uniformly distributed random number $r \in [0, 1)$ is generated. If $r < \varepsilon_1$, the control system is put in the OF group; otherwise, it is put in the MF group.

2. **Dropout-rate Driven Algorithm (DDA):** DDA maintains a long dropout history window of the control system. From that history window, the average dropout rate of the control system is estimated, which is denoted as $\tilde{\varepsilon}$. If $\tilde{\varepsilon}$ is above a given upper bound ε^u, i.e. the current average dropout rate is too high, the control system is put in the MF group. If $\tilde{\varepsilon}$ is below a given lower bound ε^l, the control system is put in the OF group. If $\varepsilon^l \leq \tilde{\varepsilon} \leq \varepsilon^u$, the algorithm finds the individual dropout rate ε_1 corresponding to the average dropout rate $\tilde{\varepsilon}$ as MDA does. The control system is put in the OF group with probability $\varepsilon_1(\tilde{\varepsilon})$ (In order to emphasize the dependence of ε_1 on $\tilde{\varepsilon}$, we add the arguement $\tilde{\varepsilon}$).

3. **Feedback Driven Algorithm (FDA):** FDA also maintains a long dropout history window of the control system. Besides the average dropout rate $\tilde{\varepsilon}$, FDA estimates the individual dropout rates, for example, estimates ε_1 with $\tilde{\varepsilon}_1$ based on the history window. As DDA does, FDA puts the control

system in the MF group when $\tilde{\varepsilon} > \varepsilon^u$ or in the OF group when $\tilde{\varepsilon} < \varepsilon^l$. On the condition of $\varepsilon^l \leq \tilde{\varepsilon} \leq \varepsilon^u$, if $\tilde{\varepsilon}_1 > \varepsilon_1(\tilde{\varepsilon})$, the control system is put in the MF group; otherwise in the OF group.

In the simulations, 246 task sets were generated. Every task set consists of a control system and 4 non-control systems. For every task set, the EDF algorithm and 3 algorithms based on the optimal dropout policy (MDA, DDA, FDA) are implemented. EDF is compared against MDA, DDA and FDA. The performance of the non-control systems is measured by their average dropout rates. The performance of the control system is measured by the output power $\mathbf{E}[y^T y]$. In the comparisons, policy A is stated to beat policy B if the output power of the control system under policy A is less than the one under policy B **and** the average dropout rate of every non-control system under policy A is less than the one under policy B. If the differece between two dropout rates is less than a specified *dropout rate accuracy*, the two rates are said to be the same.

A comparison of these results is shown in the following table.

In table 3.2, "EDF vs MDA" means EDF beats MDA based on the above comparision law. The entry for "EDF vs MDA" under the accuracy 0.001 is 0, which means there is 0 task sets (out of 246 ones) where EDF outperforms MDA. The entry for "MDA vs EDF" under the accuracy 0.001 are 49, which means there is 49 task sets (out of 246 ones) where MDA outperforms EDF. Because $49 > 0$, we know MDA outperform EDF. Similarly we can know that both DDA and FDA outperform EDF. So table 3.2 confirms the efficiency of the enforced optimal dropout policy (a Markov chain).

Among the 5 systems in every task set, we have a special interest in the control system. In figure 3.9, each point in the plot depict the output power

TABLE 3.2

SCHEDULER SCORES FOR 246 TASK SETS. EACH TASK SET CONTAINS ONE CONTROL SYSTEM WITH THE OPTIMAL DROPOUT POLICY ENFORCED [49]

Dropout Rate Accuracy	EDF vs MDA	EDF vs DDA	EDF vs FDA	MDA vs EDF	DDA vs EDF	FDA vs EDF
0.001	0	0	0	49	43	58
0.01	0	0	0	170	166	88

obtained by applying one scheduling algorithm (dropout policy) to one task set. The performance under the optimal dropout policy corresponds to the curve below all the points in each plot. From the plots, one can readily see that the data corresponding to MDA, DDA and FDA are close to the optimal dropout curve while EDF in general is not. This is especially true when the dropout rate is high (say greater than 18%). A horizontal long-dashed line indicating power of 100 is also shown in the plot for reference. It is clear that EDF resulted in much more points above this line than the other algorithms. Since this specific control system is considered unstable when the power value is larger than 100, MDA, DDA and FDA again outperform EDF greatly in this regard.

3.4.4.2 Simulation results from [50]

Simulations in [49] assumed the configuration with a single control system (task) and several non-control systems (tasks). The optimal dropout policy (an MC constraint) is enforced on the control system as a QoS specification. The

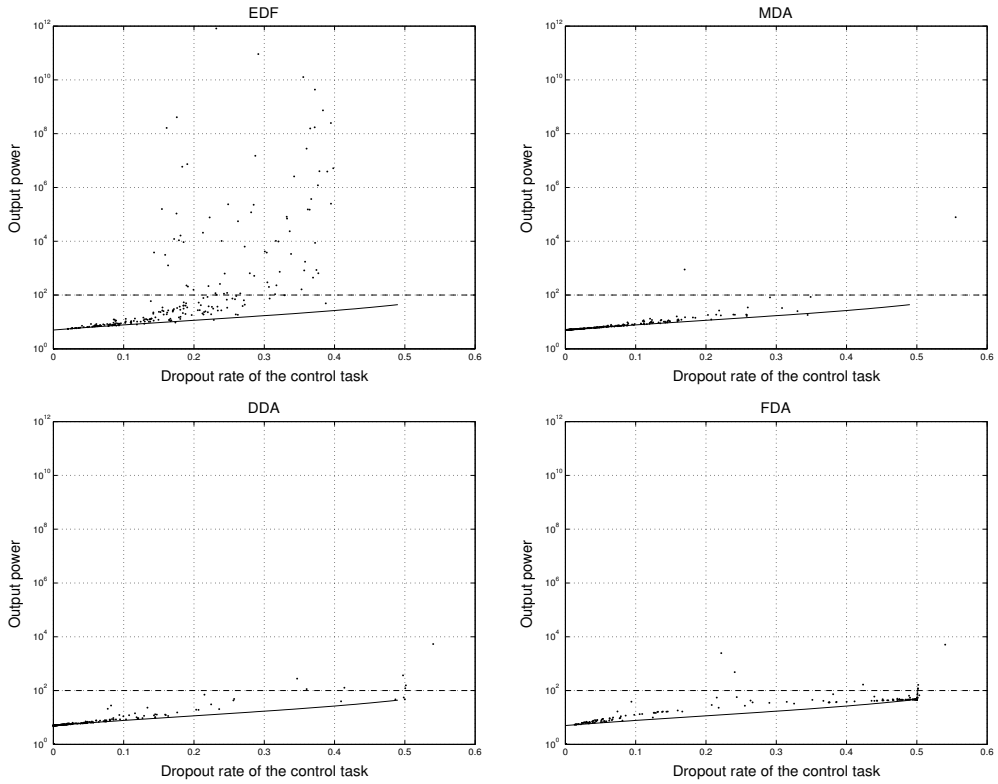


Figure 3.9. Output power vs. the dropout rate under four different schedulers (dropout policies) with 246 task sets [49].

non-control systems are scheduled based on the average dropout rate, which is more flexible than the MC contraint and may leave enough room for the control system to satisfy the MC constraint.

Now we extend the above configuration to more general cases where more than one, to say two, control systems have to be scheduled. In order to guarantee performance, every control system tries to follow their own MC constraints (i.e. the optimal dropout policies). It is quite possible that two control systems require access to the single resource at the same time. It is true that one control system will not get the expected resource so that it cannot follow its optimal dropout

policy. *Which system should get the resource?* In preliminary studies, EDF was implemented to resolve this access collision from control systems. It was shown by simulations that EDF does not work well. Therefore more deliberate methods are needed to resolve these access collisions inside every group.

In [50], there are also 3 priority groups in [49], *Must Finish* (MF), *Better Finish* (BF) and *Better Drop* (BD) which are ordered in the decreasing priority order. We may understand the BD group in [50] as the OF group in [49].

Simulations in [50] consider a task set with 5 control systems, among which 2 systems have the loop function of $H(z) = \frac{z+2}{z^2+z+2}$, named as MC1 and the others are an inverted pendulum [1], named as MC2. The optimal dropout policy of MC1 is the *drop-2* policy in subsection 3.4.2. The optimal dropout policy of MC2 is *(2,3)*-firm guarantee policy. For a control system, suppose its current state $q[n] = q_i$ and the individual dropout rate is ε_i as shown in figure 3.4 (a). If $\varepsilon_i = 0$, the control system is put in the MF group; if $\varepsilon_i = 1$, it is put in the BD group; otherwise, it is put in the BF group. Two strategies are proposed to resolve the access collision of control systems **inside** every group.

1. **State-Sensitivity based Algorithm (SSA):** SSA is based on the penalty due to loss of resource access. If a control system does not get access to the resource, its performance (as measured by the output power) will be degraded. The resource is assigned to the control system with the largest performance degradation. The performance degradation is computed from Theorem 3.3.1.

2. **Grouped Fair Dropout Rate algorithm (GFDR):** GFDR is based on the average dropout rate. A long dropout history window is maintained for every control system and the average dropout rate is computed based on

the history window. Inside every priority group, for example the MF group, the resource is assigned to the control system with the largest dropout rate. Sometime "weighted" dropout rates are used for decision. The insight of GFDR is to guarantee fairness. Note that such fairness is achieved under the dropout pattern specified by the MC constraints (the optimal dropout policies).

In simulations, we randomly generated 250 task sets, each of which contains 2 MC1 systems and 3 MC2 systems. Performance is measured by the normalized output power, which is computed as follows.

$$P = \min\left\{\frac{P_{raw} - P_0}{P_{unstable} - P_0}, 1\right\} \tag{3.4.32}$$

where P is the normalized performance, P_{raw} is the raw output power estimated from the simulations, P_0 is the output power *without* dropouts and $P_{unstable}$ is the upper bound on acceptable performance. If $P_{raw} > P_{unstable}$, we say the system is *"unstable"*. It can be seen the normalized performance P ranges from 0 to 1. The overall performance of 5 control systems is the average of the individual normalized performance of 5 systems. In the following figure, the histograms of output power under different scheduling policies are plotted. MDA, DDA and FDA are the scheduling approaches derived in [49]. WHSA1 and WHSA2 comes from the literature on window-based constraint [10]. WHSA1 is the weakly hard system scheduling algorithm puting the constraint of completing 2 *consecutive* jobs in any 4-job window on MC1 and the constraint of completing 2 *consecutive* jobs in any 3-job window on MC2. WHSA2 is the weakly hard system scheduling algorithm puting the constraint of completing *any* 2 jobs in any 4-job window on MC1 and the constraint of completing *any* 2 jobs in any 3-job window on MC2.

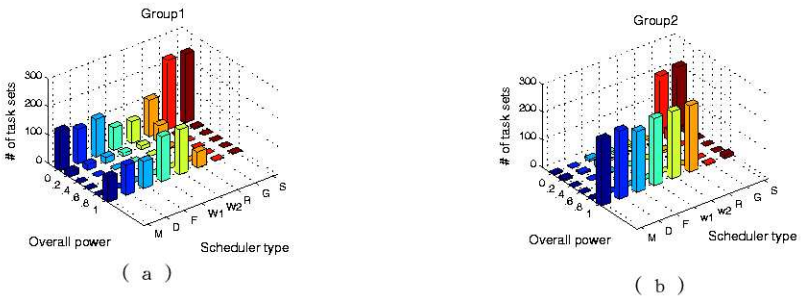


Figure 3.10. Histograms illustrating overall output signal power for task sets with utilization between 1.0-1.2 (group 1) and 1.2-1.4 (group 2) [50]

In figures 3.10 (a) and (b), "M", "D", "F", "w1", "w2", "R", "G", "S" represent MDA, DDA, FDA, WHSA1, WHSA2, FDR, GFDR and SSA respectively, and each "slice" of the "scheduler type" axis is a histogram of task sets' overall output signal powers resulted from using that type of scheduler. Figures 3.10(a) and (b) plot the distribution of the overall performance of two groups of task sets for each scheduling algorithm. Each group contains 250 task sets, and each task set consists of 5 tasks (2 MC1 systems and 3 MC2 systems). In group 1, each task set's utilization (or the load it presents to the system) is between 1.0 and 1.2 [7]. And in group 2 it is between 1.2 and 1.4. From figure 3.10, one can observe

[7]The task set's utilization is equivalent to dropout rate level. For example, if such utilization equals 1.2, the system is overloaded by 20%, i.e. the dropout rate has to be above 20%.

that the overall output signal power values of task sets scheduled by GFDR and SSA tend to be lower, while those of other algorithms tend to be larger. (Recall that for the control tasks in the experiments, lower output signal power indicates better performance.) If the overall performance of a task set equals 1, it indicates that the output signal power of a system is beyond the upper bound and the system is considered to be unstable (unacceptable). In both groups of task sets, GFDR and SSA, unlike other schedulers, rarely result in instability. The better performance of GFDR and SSA can be attributed to the fact that both of them enforce the MC constraints to resolve the resource collision from control systems. Figure 3.10(a) and (b) also show that as the task set's utilization (system load) increases, GFDR and SSA still manage to achieve good performance while the performance of other schedulers degrades much faster. In fact, when the system load is larger than 1.2, we observed that the system performances due to other schedulers are rarely acceptable (less than 1% of total number of task sets).

To examine the performance difference between to GFDR and SSA, we plot the stable (i.e. $P < 1$) task set percentage and the average overall output power at heavier system loads (task set's utilization is larger than 1.4) in figure 3.11(a) and (b). (The performance of SSA and GFDR under lighter system loads is similar, as shown in figure 3.10) For each utilization range, we generated a group of 250 task sets. GFDR and SSA are used to schedule task sets in each groups. From figure 3.11, one can see that SSA is indeed better than GFDR. Figure 3.11(a) shows that as the system load increases, the stable task set percentage of GFDR decreases much faster than that of SSA. For example, when the utilization is between 1.6 and 1.7, SSA doubles the stable task set percentage achieved by GFDR. In figure 3.11(b), one can see that the average overall output signal power due to SSA is also

better than that of GFDR. For the group of task sets whose utilization is between 1.6 and 1.7, the improvement is up to 33%. The better performance of SSA can be attributed to the careful analysis of the output power based on theorem 3.3.1 which leads to a more effective scheduling decision.

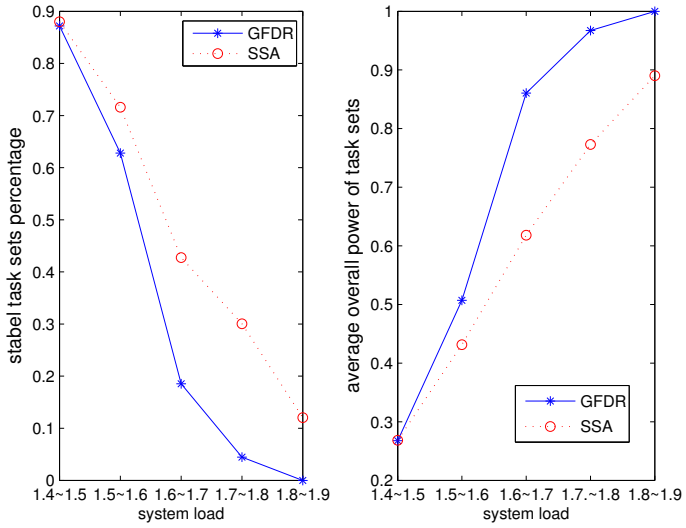


Figure 3.11. GFDR vs. SSA [50]

3.4.5 Verify the optimal dropout policy through a helicopter simulator

There is a hardware helicopter simulator in our lab. We tried to use it to verify the utility of the optimal dropout policy. The conclusion drawn from these experiments were inconclusive because the noise model for theoretical predictions (a white noise process) mismatches the noise model for the experiments (a bounded

noise process). The theoretical model in eq. 3.1.1 was, unfortunately, less realistic than expected. Now we start to talk about our experiments.

3.4.5.1 Control plant: a Helicopter Simulator

Our plant is a hardware helicopter simulator shown in figure 3.12. There

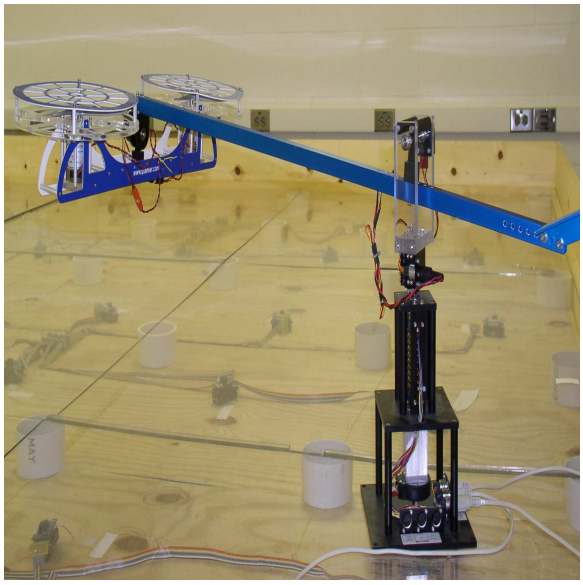


Figure 3.12. The helicopter simulator

are 3 DOF (degrees of freedom) in the helicopter simulator, including elevation dynamics, pitch dynamics and travelling dynamics. Denote the elevation angle as

ε, pitch angle as p and travelling rate as r. Their dynamics can be expressed as

$$\begin{cases} Elevation: \begin{cases} J_e \frac{d^2\varepsilon}{dt^2} = K_f l_a (V_f + V_b) = K_f l_a V_s \\ V_s = K_{ep}(\varepsilon - \varepsilon_c) + K_{ed}\frac{d\varepsilon}{dt} \end{cases} \\ Pitch: \begin{cases} J_p \frac{d^2 p}{dt^2} = K_f l_h (V_f - V_b) = K_f l_h V_d \\ V_d = K_{pp}(p - p_c) + K_{pd}\frac{dp}{dt} \end{cases} \\ Travelling: \begin{cases} J_t \frac{dr}{dt} = K_p l_a p \\ p_c = K_{rp}(r - r_c) + K_{ri}\int (r - r_c) \end{cases} \end{cases} \tag{3.4.33}$$

where ε_c, p_c and r_c are the desired setpoints, V_f and V_b are the input voltages on the front and back motors respectively, and the rest of the symbols are control parameters such as K_{ep}, K_{ed} or system parameters, such as J_e, J_p.

As eq. 3.4.33 shows, the dynamics of this three dimensonal system are all second order. This low order nature simplifies the performance analysis. But eq. 3.4.33 omits all nonlinear factors, such as friction and blackslash, and all noises.

3.4.5.2 Control configuration

In section 3.4.2, the dropout policy designed for a linear system is studied. We try to verify the theoretical optimal dropout policy results on the travelling dynamics. Our set point is $r_0 = 20 deg/sec$, i.e. the system dynamics are linearized at the point $r = r_0$. $\Delta r = r - r_0$ is governed by the third (travelling) sub-equation in eq. 3.4.33. Δr is also chosen as the output. In order to simplify notation, we represent Δr as r. Note $r = 0$ means the traveling rate is equal to r_0 for the rest of this subsection.

We discretize the traveling dynamics to get

$$\begin{cases} x_{k+1} &= Ax_k + B_u u_k + B_w w_k \\ y_k &= Cx_k \\ u_k &= f(y_k, y_{k-1}, \cdots) \end{cases} \tag{3.4.34}$$

where A, B_u and C are system parameters provided by the manufacturer. B_w stands for the input channel of the noise. We don't know the value of B_w. $\{u_k\}$ is the control variable, which is really the pitch angle p. $f(\cdot)$ is a PI law (proportional and integral law). w_k is noise, which may come from an exogenous disturbance, linearization error or other uncertainty. Note that u_k and w_k may have different input channels. So it is possible that $B_u \neq B_w$.

As mentioned above, we use a PI control law, which can be expressed as

$$u_k = f(y_k, y_{k-1}, \cdots) = K_{rp} y_k + K_{ri} T \left(y_{k-1} + y_{k-2} + \cdots\right) \tag{3.4.35}$$

In the event of dropouts, i.e. the measurement is not available, the lost measurement is replaced by 0. Why do we choose this dropout compensation policy? The aim of our control is to keep steady state, i.e. $y_k = 0$. So we expect y_k is not far from the the objective 0.

In section 3.4, we assume $\{w_k\}$ is white. Is that assumption valid? We did experiments to verify it. In the experiments, we chose $T = 20msec$. There is no dropout in the experiments. Thus the whole system can be described by the control diagram in figure 3.13. As figure 3.14 shows, the system exhibits a strong periodic oscillation. Its period T_{os} is exactly determined by the travelling rate, $T_{os} = 360/r_0$. One reasonable explanation to the above result is the periodicity of $\{w_k\}$. Why is $\{w_k\}$ periodic and $T_{os} = 360/r_0$? To our best knowledge, it

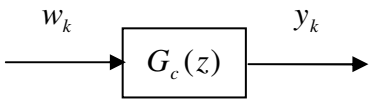


Figure 3.13. Control diagram

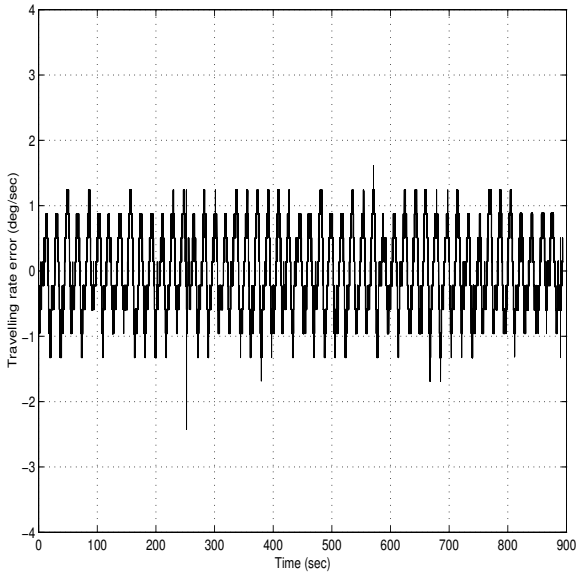


Figure 3.14. Response of travelling rate error

comes from the friction. When the helicopter simulator flies, its axis will rotate at the period of T_{os}. There are many (about 40) sliding contacts on the base of the simulator. This base may not be perfectly symmetric. The contacting friction may be stronger at some positions and weaker at the others. So the friction varies periodically at the period of T_{os}. In figure 3.13, feedback is used, which is expected to counteract the oscillation from w_k. So the original magnitude of w_k may be larger than the one in figure 3.14.

From the above analysis, we see that $\{w_k\}$ is almost deterministic and periodic.

One may wonder whether we can totally cancel it. It is hard to say *yes* or *no*. If $B_u = B_w$, i.e. the controller and the noise have the same input channel, we can cancel $\{w_k\}$ by setting $u_k = -w_k + u'_k$. If $B_u \neq B_w$, we have no way to cancel w_k. We, unfortunately, don't know whether $B_u = B_w$.

3.4.5.3 Performance analysis of the travelling dynamics under dropouts

Our main concern is to verify the results on the dropout policy in section 3.4. The system equation is

$$\begin{cases} x_{k+1} &= \begin{cases} A_{close}x_k + B_w w_k, & d_k = 0 \\ A_{open}x_k + B_w w_k, & d_k = 1 \end{cases} \\ y_k &= Cx_k \end{cases} \tag{3.4.36}$$

By estimating B_w with B_u and assuming $\{w_k\}$ is white with unit variance, we apply Theorem 3.3.1 to predict the performance $p = \mathbf{E}[y_k^2]$. We compare several policies shown in figure 3.15 To our disappointment, the difference between dropout policies is not significant. For example, when the average dropout rate is 25%, the difference among the four policies is less than 10%. By utilizing the design methods in subsection 3.4.2, we may find a better policy and amplify the difference. We now take a look at the maximum eigenvalue of system matries.

$$\lambda_{max}(A_{close}) = 0.9991, \lambda_{max}(A_{open}) = 1.0004 \tag{3.4.37}$$

where $\lambda_{max}(\cdot)$ stands for the maximum magnitude of the eigenvalues. It can be seen that $\lambda_{max}(A_{close})$ is quite close to $\lambda_{max}(A_{open})$, i.e. the difference between the closed-loop and open loop configurations is not significant. So it is not surprising

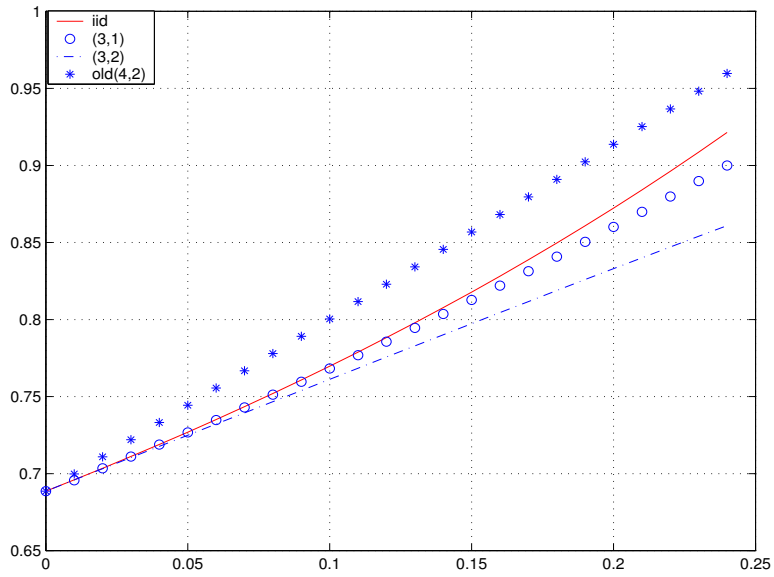


Figure 3.15. Theoretical predictions of different dropout policies

that there is little different between the various policies in figure 3.15.

Although the improvement due to the dropout policy is not very exciting, there is still improvement. The next question is whether such improvement (about 10%) can be demonstrated through experiments. We enforced i.i.d. and $(2, 3)$ policies in the experiments. The results are shown in figure 3.16. Comparing figure 3.15 and 3.16, we can list the following disagreement between empirical results and theoretical prediction.

- By theoretical prediction in figure 3.15, the performance is worse when the average dropout rate is higher. The experimental results, however, demonstrate little difference among different dropout rates.

- At the same dropout rate, the difference among different trials in figure 3.16 seems unacceptable, especially for the i.i.d. dropout policy. For example, the performance variation at the rate of 10% is about 100%. $(2, 3)$ policy

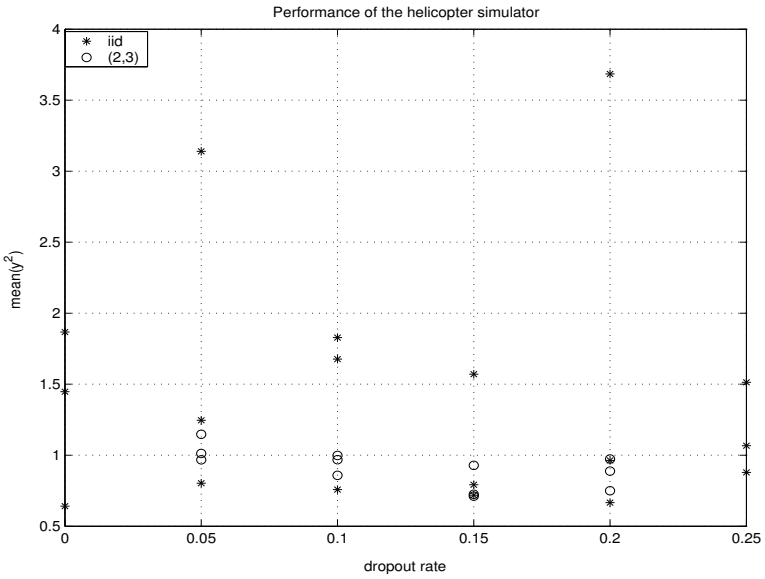


Figure 3.16. Performance of the plant under i.i.d. and $(2, 3)$ dropout policies

seems better with smaller performance variation.

- We expected the $(2, 3)$ policy to be better than the i.i.d. policy on the basis of our theoretic prediction. Due to the large performance variations, however, it is hard to see whether such performance predictions can be verified from the experiments.

Based on the above conclusions, we may wonder whether the theoretical prediction or the experiments are wrong. After carefully thinking, we find out **nothing is wrong** and the disagreement comes from that fact that the noise $\{w_k\}$ doesn't satisfy the white noise assumption for theoretical prediction. The following discussion attempts to explain the reasons for this the disagreement.

3.4.5.4 Why don't experimental results agree with the theoretical prediction?

The travelling dynamics are governed by eq. 3.4.34. For convenience, we rewrite that equation down.

$$\begin{cases} x_{k+1} &= Ax_k + B_u u_k + B_w w_k \\ y_k &= Cx_k \\ u_k &= f(y_k, y_{k-1}, \cdots) \end{cases} \tag{3.4.38}$$

The control law $f(\cdots)$ is a PI one, whose proportional gain is much larger than its integral gain. So u_k is approximately proportional to y_k when y_k is large enough. In event of dropouts, the measurement is replaced with 0. If the dropouts occur when y_k is large, u_k will deviate much from its desired value, which leads to inefficiently counteracting w_k. We can imagine at the next step, i.e. time $n+1$, y_{k+1} will be larger than the expected one. It can be seen that the larger the lost measurement y_k, the larger performance degradation at time $n+1$. So the wise dropout decision is to drop small y_k. But what did the experimental policies do?

I.i.d. policy drops data in an i.i.d. fashion. If we are lucky, the dropped measurments are all small and better performance, i.e. smaller output power, can be achieved. Otherwise, we drop some large measurements and the performance is made worse. This may be the reason that under the same dropout policy, different trials yield quite different performance.

The performance variation of different trials under *(2,3)* policy is smaller than the variation under i.i.d. policy. *Why?* It can be explained by the effect of consecutive dropouts. By the discussions in the last paragraph, if we drop y_k, y_{k+1} will be larger (perhaps much larger) than the expectation. The larger y_k, the larger degradation of y_{k+1}. If consecutive dropouts happen, i.e. y_{k+1} is also

dropped, we can imagine the degradation of y_{k+2} will be very significant. Therefore consecutive dropouts are even more harmful on performance than dropout rate. $(2,3)$ policy tries to separate dropouts apart (in any 3 consecutive packets, there is at most 1 dropout). So it is not surprising to see smaller performance deviation under the $(2,3)$ policy.

Now we are clear that it is the noise w_k that yields the disagreement between theoretical prediction and experimental results.

3.5 Spatially distributed control systems

This section extends the main result in theorem 3.3.1 to a spatially distributed control system that implements the control over a sensor-actuator network. The system under study is shown in figure 3.17. This system [18] is a linear array of masses that are linked together by springs. Each mass (node) has an embedded processor attached to it that can measure the node's local state (position and velocity) and then transmit that information to its neighboring nodes. This transmitted information is then used by the embedded processor to compute a control input.

We first assume that the continuous-time dynamics of the system have been discretized in time. So we let $x[n_1, n_2]$ denote the state of the n_2th node at time instant n_1. The state x is a 2-dimensional vector characterizing the position and velocity of the node with respect to its equilibrium position. The state satisfies the following recursive equations,

$$\begin{cases} x[n_1+1, n_2] &= Ax[n_1, n_2] + B(x[n_1, n_2-1] + x[n_1, n_2+1]) \\ & \quad +F(u[n_1, n_2] + w[n_1, n_2]) \\ z[n_1, n_2] &= Cx[n_1, n_2] \end{cases} \tag{3.5.39}$$

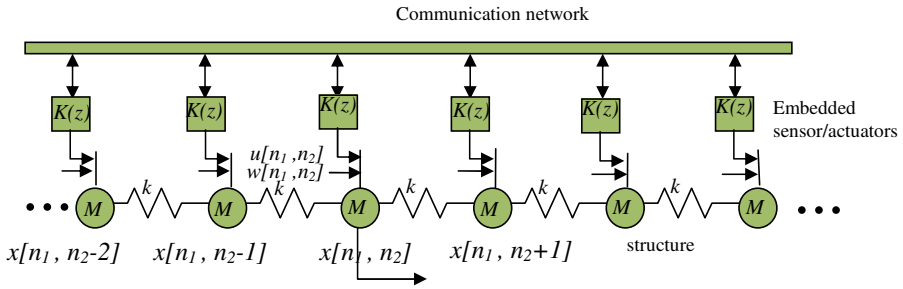


Figure 3.17. Distributed control system and Sensor-Actuator network

for $n_1 \geq 0$ and any n_2. $z[n_1, n_2]$ is an output signal that is used to characterize overall system performance. A, B, C and F are appropriately dimensioned real-valued matrices. There are two inputs to this equation; the disturbance $w[n_1, n_2]$ and the control $u[n_1, n_2]$. The disturbance w is a zero-mean white noise process in both time and space. The control input is computed by the embedded processor.

Each node has a processor attached to it. The processor measures the node's local state $x[n_1, n_2]$ and it transmits this information to its neighbors upon request. We assume that the nodes are synchronized in time and that in each sampling interval the node decides whether or not to access its neighbor's state. This means that a node first "requests" that its neighbors send data to it and then the processor computes its control input $u[n_1, n_2]$ upon receiving this data. If neighboring state information has been received, then the control input is computed according to the following equation,

$$u[n_1, n_2] = K_0 x[n_1, n_2] + K_1(x[n_1, n_2 - 1] + x[n_1, n_2 + 1]) \qquad (3.5.40)$$

where K_0 and K_1 represent control gain matrices that have been chosen by the control engineer. Since our network may occasionally drop measurements, the

processor needs to use a different control signal if the neighboring state data is not received. In this case, the processor simply sets $u[n_1, n_2] = 0$.

Data may be dropped by the network. These dropouts occur for two reasons. The first reason is that the medium is unreliable. A transmitted measurement has a finite probability f of being lost due to link failure. This probability is assumed to be statistically independent of the state of the measurement's source. Dropouts will also occur because a node explicitly decides NOT to request neighboring state measurements. This occurs because an *overload management policy* requires nodes to drop a certain percentage of measurements when the network is congested. In particular, the network assigns each processor a maximum allowable transmission rate which is represented as a lower bound, ε_0 on the node's actual dropout rate. The size of ε_0 depends on the amount of network congestion.

Because dropouts cause us to switch between two different control laws, the system's state space model takes the form of a *jump linear system* [51]. In particular, let's define a *dropout process* that is denoted as $d[n_1, n_2]$. It is a binary random process in which $d[n_1, n_2] = 1$ if a dropout occurs and is zero otherwise. Under the dropout process, our system equations take the form,

$$\begin{aligned} x[n_1+1, n_2] &= A[n_1, n_2]x[n_1, n_2] + B[n_1, n_2](x[n_1, n_2-1] + x[n_1, n_2+1]) \\ &\quad + Fw[n_1, n_2] \\ z[n_1, n_2] &= Cx[n_1, n_2] \end{aligned}$$

where $A[n_1, n_2]$ and $B[n_1, n_2]$ are matrix valued random processes such that

$$A[n_1, n_2] = \begin{cases} A_0 = A + FK_0 & \text{if no dropouts occur (i.e., } d[n_1, n_2] = 0) \\ A_1 = A & \text{if a dropout occurs (i.e., } d[n_1, n_2] = 1) \end{cases}$$

$$B[n_1, n_2] = \begin{cases} B_0 = B + FK_1 & \text{if no dropouts occur (i.e., } d[n_1, n_2] = 0) \\ B_1 = B & \text{if a dropout occurs (i.e., } d[n_1, n_2] = 1) \end{cases}$$

Application performance will be measured by the average power in the control system's output signal. This measures how well the system suppresses the effect that the disturbance $w[n_1, n_2]$ has on the system's shape. In particular we assume that w is a white noise process whose covariance matrix is $R = \mathbf{E}\left[w[n_1, n_2]w^T[n_1, n_2]\right]$. The control objective is to minimize the noise power in the node's state. So a natural measure of application performance is the average power,

$$\|z\|_{\mathcal{P}}^2 = \text{Trace}\left(\mathbf{E}\left[z[n_1, n_2]z^T[n_1, n_2]\right]\right) = \text{Trace}\left(C\overline{P}_0C^T\right)$$

where $\overline{P}_0$ is the variance matrix

$$\overline{P}_0 = \mathbf{E}\left[x[n_1, n_2]x^T[n_1, n_2]\right] \tag{3.5.41}$$

Note that this example presumes that all nodes are "identical" (spatially invariant) and temporally stationary, so that the above covariance matrix is independent of n_1 and n_2.

The following theorem extends theorem 3.3.1 to this spatially distributed system. This theorem provides an infinite set of equations that can be solved for the covariance matrix $\overline{P}_0$. This theorem's proof will be found in the appendix.

Theorem 3.5.1 *Let $w[n_1, n_2]$ be a zero-mean white noise process with variance R. Let $x[n_1, n_2]$ satisfy a jump linear system equation that is driven by a Markov chain with transition matrix $Q = [q_{ij}]_{N\times N}$ with stationary distribution $\pi = [\pi_1, \pi_2, \cdots, \pi_N]$.*

If $\mathbf{E}\left[x^T[n_1, n_2]x[n_1, n_2]\right] < \infty$ (i.e. mean square stable) , then

$$\overline{P}_0 = \mathbf{E}\left[x[n_1, n_2]x^T[n_1, n_2]\right] = \sum_{i=1}^{N} \overline{P}_0^{i,i} \tag{3.5.42}$$

where $\overline{P}_0^{i,i}$ satisfy the following infinite set of equations.

$$\begin{aligned}
\overline{P}_0^{i,i} &= \sum_{l=1}^{N}\left[q_{li}A_i\overline{P}_0^{l,l}A_i^T + 2\pi_i B_i\overline{P}_0^{l,l}B_i^T\right] \\
&\quad + \sum_{l,m=1}^{N}\left[\pi_i B_i\left(\overline{P}_2^{l,m} + \left(\overline{P}_2^{l,m}\right)^T\right)B_i^T\right] \\
&\quad + \sum_{l,m=1}^{N}\left[q_{li}A_i\left(\overline{P}_1^{l,m} + \left(\overline{P}_1^{l,m}\right)^T\right)B_i^T + q_{mi}B_i\left(\overline{P}_1^{l,m} + \left(\overline{P}_1^{l,m}\right)^T\right)A_i^T\right] + \pi_i R \\
\overline{P}_1^{i,j} &= B_i\left(\overline{P}_1^{i,j}\right)^T B_j^T + \sum_{l=1}^{N}\left[\pi_j q_{li}A_i\overline{P}_0^{l,l}B_j^T + \pi_i q_{lj}B_i\overline{P}_0^{l,l}A_j^T\right] \\
&\quad + \sum_{l,m=1}^{N}\left[q_{li}q_{mj}A_i\overline{P}_1^{l,m}A_j^T + \pi_i q_{lj}B_i\overline{P}_1^{l,m}B_j^T + \pi_j q_{mi}B_i\overline{P}_1^{l,m}B_j^T\right] \\
&\quad + \sum_{l,m=1}^{N}\left[\pi_j q_{li}A_i\overline{P}_2^{l,m}B_j^T + \pi_i q_{mj}B_i\overline{P}_2^{l,m}A_j^T + \pi_i\pi_j B_i\overline{P}_3^{l,m}B_j^T\right] \\
\overline{P}_k^{i,j} &= \sum_{l,m=1}^{N}\left[q_{li}q_{mj}A_i\overline{P}_k^{l,m}A_j^T + 2\pi_i\pi_j B_i\overline{P}_k^{l,m}B_j^T\right] \\
&\quad + \sum_{l,m=1}^{N}\left[\pi_j q_{li}A_i\left(\overline{P}_{k+1}^{l,m} + \overline{P}_{k-1}^{l,m}\right)B_j^T + \pi_i q_{mj}B_i\left(\overline{P}_{k+1}^{l,m} + \overline{P}_{k-1}^{l,m}\right)A_j^T\right] \\
&\quad + \sum_{l,m=1}^{N}\left[\pi_i\pi_j B_i\left(\overline{P}_{k-2}^{l,m} + \overline{P}_{k+2}^{l,m}\right)B_j^T\right]
\end{aligned} \tag{3.5.43}$$

for $k \geq 2$ and $i, j = 1, \ldots, N$.

Remark: Note that theorem 3.5.1 presumes the system is already mean square stable. This condition may be difficult, in general, to verify and we provide no characterization of the system's stability.

Remark: Equations 3.5.43 is an infinite set of linear equations that we solve for the matrices $\overline{P}_k^{i,j}$. In particular, these matrices are the following conditional expectations.

$$\begin{aligned}
\overline{P}_k^{i,j} &= \pi_i \pi_j \mathbf{E}\left[x[n_1,n_2]x^T[n_1,n_2+k] \quad | \quad q[n_1-1,n_2]=q_i,\right. \\
&\qquad \left. q[n_1-1,n_2+k]=q_j\right], k \neq 0 \\
\overline{P}_0^{i,i} &= \pi_i \mathbf{E}\left[x[n_1,n_2]x^T[n_1,n_2] \quad | \quad q[n_1-1,n_2]=q_i\right] \\
\overline{P}_0^{i,j} &= 0, i \neq j
\end{aligned}$$

We can solve these equations numerically in a recursive manner. In particular, we generate a sequence $\{\overline{P}_k^{i,j}[L]\}_{L=0}^{\infty}$ of matrices that converges to the true $\overline{P}_k^{i,j}$ as L goes to infinity. In particular, we insert the matrices $\overline{P}_k^{i,j}[L]$ into the righthand side of equation 3.5.43 and take the matrices on the lefthand side of the equation as $\overline{P}_k^{i,j}[L+1]$. By eq. 3.5.42, we estimate $\overline{P}_0$ by

$$\overline{P}_0[L] = \sum_{i=1}^{N} \overline{P}_0^{i,i}[L] \tag{3.5.44}$$

We want to compute $\overline{P}_0$. So if $\|\overline{P}_0[L+1] - \overline{P}_0[L]\|_\infty < \delta$ (δ is tolerance), we will stop the recursive computation.

There is a trick to pick the initial condition for $\{\overline{P}_k^{i,j}[0]\}$. We let

$$\begin{aligned}
\overline{P}_0^{i,i}[0] &= \pi_i R \\
\overline{P}_0^{i,j}[0] &= 0, \text{when } i \neq j \\
\overline{P}_k^{i,j}[0] &= 0, \text{when } k \neq 0
\end{aligned}$$

Based on the above initial conditions and the update rule of $\overline{P}_k^{i,j}[L]$, we know, at the recursion step L,

$$\overline{P}_k^{i,j}[L] = 0, \text{when } k \geq 2L+1 \tag{3.5.45}$$

So we only need to consider the recursion of $\overline{P}_k^{i,j}[L+1]$ $(k = 0, 1, 2, \cdots, 2L+2)$ [8], i.e. only $2L+3$ equations need to be computed. We can, therefore, obtain $\overline{P}_0$ under a specified dropout policy.

As we did in section 3.4, we can use theorem 3.5.1 to pose an optimization problem whose solution is the "optimal" dropout policy for this distributed system. To state this optimization problem, we again consider the 4-state Markov chain that was considered in section 3.4. Recall that ε_i is the probability of not transmitting a message when the Markov chain is in state q_i. We also assume that there is a finite probability f, that a transmitted message will not reach its

[8] Of course we may still do recursive computations on $\overline{P}_k^{i,j}[L+1]$ $(k \geq 2L+3)$, but we are sure they are all 0

destination. With these notational conventions we obtain the transition matrix,

$$Q = \begin{bmatrix} (1-\varepsilon_1)(1-f) & \varepsilon_1 + f(1-\varepsilon_1) & 0 & 0 \\ 0 & 0 & (1-\varepsilon_2)(1-f) & \varepsilon_2 + f(1-\varepsilon_2) \\ (1-\varepsilon_3)(1-f) & \varepsilon_3 + f(1-\varepsilon_3) & 0 & 0 \\ 0 & 0 & (1-\varepsilon_4)(1-f) & \varepsilon_4 + f(1-\varepsilon_4) \end{bmatrix}$$

We solved the optimization problem (eq. 3.4.10) for this specific system. The system matrices (A_i, B_i) when the Markov chain is in state q_i are

$$\begin{aligned} A_1 &= A_3 = A + BK \\ B_1 &= B_3 = 0 \\ A_2 &= A_4 = A \\ B_2 &= B_4 = B \end{aligned}$$

where $A = \begin{bmatrix} 0.9990 & 0.0100 \\ -0.1999 & 0.9990 \end{bmatrix}$, $B = \begin{bmatrix} 0.0005 & 0 \\ 0.1000 & 0 \end{bmatrix}$, $C = \begin{bmatrix} 1.0 & 0 \\ 0 & 0.1 \end{bmatrix}$, $F = B$, $K = \begin{bmatrix} -93.2580 & -10.4700 \\ 0 & 0 \end{bmatrix}$. From these equations we see that the closed-loop distributed system is really a group of decoupled subsystems. When a dropout occurs there is no control (i.e. $u[n_1, n_2] = 0$) and the physical coupling between subsystems reasserts itself.

For our particular problem, `Matlab`'s optimization toolbox was used to numerically solve the preceding optimization problem by the gradient descent method. These optimizations were done assuming a link failure rate, f, of 0.3 and an average dropout rate ε_0 between 0.05 and 0.85. The solutions are the transition

probabilities ε_i. We found that these probabilities took the following form,

$$[\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4] = \begin{cases} [x_1, 0, 0, 0], & \varepsilon_0 < 0.2237 \\ [1, 0, x_3, 0], & 0.2237 < \varepsilon_0 < 0.4117 \\ [1, x_2, 1, 0], & 0.4117 < \varepsilon_0 < 0.5833 \\ [1, 1, 1, x_4], & 0.5833 < \varepsilon_0 \end{cases}$$

where x_1, x_2, x_3, x_4 are determined by the average dropout rate condition. In reviewing the transition probabilities given above, it is apparent that the optimal policy is a soft $(2, 3)$-policy for $\varepsilon_0 < 0.2237$. For dropout rates above this level, however, the structure of the optimal policy changes to allow higher dropout rates.

The plot in figure 3.18 compares the performance under the "optimal" dropout policy (solid line) and an i.i.d. dropout policy (dashed line). This is the performance computed using our theoretical results. We also simulated the distributed system using a `MatLab SimuLink` model. All of the simulations assumed 27 nodes with the endpoints having free boundary conditions. The results for these simulations are plotted as $*$ and o. We simulated 3 different runs at 6 different dropout rates for 100,000 iterations. In the simulations, we used time averages to estimate the signal power. The theoretical predictions show that the optimal policy is indeed better than the i.i.d. policy. The simulation results show close agreement with the theoretical predictions for a wide range of dropout rates.

The Markov chains derived in this section form the basis of an overload management policy that is easily implemented on an embedded processor. In particular, a number of these "optimal" Markov chains would be determined for a range of overload conditions (ε_0) and a range of link failure rates (f). We store these transition probabilities in a table that is indexed with respect to ε_0 and f. The

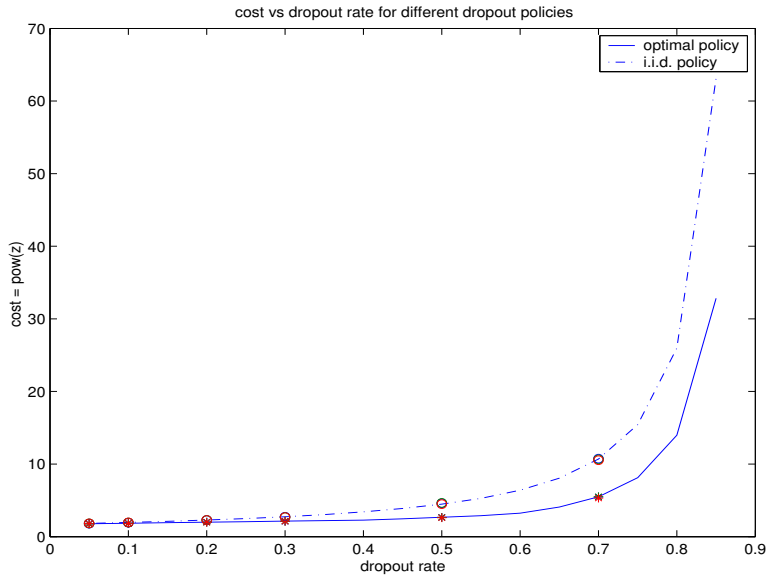


Figure 3.18. Simulation vs. theoretical results

overload management policy used by each node is a concrete instantiation of the optimal Markov chain whose transition probabilities are loaded from this table based on 1) the transmission rate $(1 - \varepsilon_0)$ that was allocated to the node and 2) based on the link failure rate (f) that was estimated by the node. What should be apparent is that the resulting policy is adaptive with respect to link failure rate and the allocated transmission rate . Moreover, since these chains are solutions to the optimization problem, we know that this policy degrades application performance as little as possible. In other words, we have provable guarantees that this approach makes optimum use of the allocated transmission rate to the local controller. The simulation and theoretical results shown in figure 3.18 suggest that hardware implementations of such policies should also perform well.

3.6 Conclusions

This chapter studies the control system with dropouts governed by a Markov chain. A necessary and sufficient condition is established for stability and wide sense stationary for such systems. The system performance, measured by the output power, is computed. Based on the achieved performance results, the optimal dropout policy is designed. The performance computation and optimal dropout policy results are extended to distributed systems.

This chapter obtains the optimal dropout policy by solving an optimization problem through gradient descent method or branch-and-bound method. The gradient descent method is only guaranteed to find a locally optimal solution. Although the branch-and-bound method is guaranteed to converge to a neighbor of the globally optimal solution, its computational complexity is high, especially when the dimension of the system or the order of the dropout Markov chain is large. So it is desirable to develop more efficient method to solve the optimization problem.

The optimal dropout policy in section 3.4 makes the decision when a packet can be dropped. It can also be viewed as a sampling method. Although the system is sampled periodically, some samples are dropped on purpose. Suppose the sampling period to be T_0 and the dropout rate to be ε_0. Then the average sampling period is

$$T = \frac{T_0}{1 - \varepsilon_0}. \tag{3.6.46}$$

T is a measurement of network utilization. The smaller T, the more network utilization. For specified T, we have infinite choices (T_0, ε_0) satisfying the constraint

in eq. 3.6.46. Of course we will prefer the choice which achieves the optimal (minimum) control performance. Is the periodic sampling ($T_0 = T, \varepsilon_0 = 0$) necessarily optimal? There is no answer yet.

3.7 Appendix: proofs

Throughout this section, the following notational conventions will be followed. The system transition matrix in equation 3.1.1 is

$$\Phi(k;m) = \begin{cases} \prod_{l=m}^{k-1} A[q[l]], & \text{if } m < k \\ I_n, & \text{if } m \geq k \end{cases},$$

where I_n is an identity matrix with the dimension of n. With this matrix, the system's state at time instant n can be expressed as $x[k] = \Phi(k;0)x[0]$ when the input disturbance $w = 0$. The following $n^2 \times (kn^2)$ matrix C_I is frequently used

$$C_I = \begin{bmatrix} I_{n^2} & I_{n^2} & \cdots & I_{n^2} \end{bmatrix}.$$

3.7.1 Preliminary lemmas

Denote the initial condition of the jump linear system in 3.1.1 as $x[0] = x_0$, $q[0] = q_0$ and the distribution of q_0 as $p = \begin{bmatrix} p_1 & p_2 & \cdots & p_N \end{bmatrix}$ ($P(q[0] = q_i \,|q_0) = p_i$).

A conditional expectation is defined as

$$\Phi_i[k] = P(q[k] = q_i \mid q_0)\mathbf{E}\left[(\Phi(k;0))^{[2]} \mid q[k] = q_i, q_0\right], \; i = 1, 2, \cdots, N. \quad (3.7.47)$$

Specifically $\Phi_i[0] = p_i I_{n^2}$ ($i = 1, \cdots, N$). Based on the definition of $\Phi_i[k]$, we

obtain

$$\mathbf{E}\left[(\Phi(k;0))^{[2]} \mid q_0\right] = \sum_{i=1}^{N} \Phi_i[k]. \tag{3.7.48}$$

By combining all $\Phi_i[k](i = 1, 2, \cdots, N)$ into a bigger matrix, we define

$$V_\Phi[k] = \begin{bmatrix} \Phi_1^T[k] & \Phi_2^T[k] & \cdots & \Phi_N^T[k] \end{bmatrix}^T. \tag{3.7.49}$$

Thus $V_\Phi[0] = p^T \otimes I_{n^2}$.

The necessity proof of theorem 3.2.1 needs the following 3 preliminary lemmas.

Lemma 3.7.1 *If the jump linear system in eq. 3.1.1 is mean square stable, then*

$$\lim_{k\to\infty} \mathbf{E}\left[(\Phi[k;0])^{[2]} | q_0\right] = 0, \ \forall q_0 \tag{3.7.50}$$

proof of Lemma 3.7.1: Because the system is mean square stable, we get

$$\lim_{k\longrightarrow\infty} \mathbf{E}\left[x^{[2]}[k] | x_0, q_0\right] = 0, \ \forall x_0, q_0 \tag{3.7.51}$$

The expression of $x[k] = \Phi(k;0)x_0$ yields

$$\lim_{k\longrightarrow\infty} \mathbf{E}\left[(\Phi(k;0)x_0)^{[2]} | x_0, q_0\right] = 0, \tag{3.7.52}$$

$\Phi(k;0)$ is an $n \times n$ matrix. So we denote it as $\Phi(k;0) = [a_1(k), a_2(k), \cdots, a_n(k)]$, where $a_i(k)$ is a column vector. By choosing $x_0 = e_i$ (e_i is an $R^{n\times 1}$ vector with the i^{th} element as 1 and the others as 0), eq. 3.7.52 yields

$$\lim_{k\longrightarrow\infty} \mathbf{E}\left[a_i^{[2]}[k] | q_0\right] = 0, i = 1, 2, \cdots, n \tag{3.7.53}$$

For any two random vectors y and z, we know

$$\left|\mathbf{E}[y^T z]\right| < \sqrt{\mathbf{E}[y^T y]\mathbf{E}[z^T z]} \tag{3.7.54}$$

By eq. 3.7.53, 3.7.54 and the definition of Kronecker product, we obtain

$$\lim_{k\longrightarrow\infty} \mathbf{E}\left[a_i[k] \otimes a_j[k] | q_0\right] = 0, i \neq j \tag{3.7.55}$$

By the definition of the Kronecker product, we know

$$\begin{aligned} & (\Phi(k;0))^{[2]} \\ = \; & [a_1[k] \otimes a_1[k], \cdots, a_1[k] \otimes a_n[k], \cdots, a_n[k] \otimes a_1[k], \cdots, a_n[k] \otimes a_n[k]] \end{aligned}$$

So eq. 3.7.53 and 3.7.55 yield

$$\lim_{k\to\infty} \mathbf{E}\left[(\Phi[k;0])^{[2]} | q_0\right] = 0, \ \forall q_0 \diamondsuit \tag{3.7.56}$$

Lemma 3.7.2 *If the jump linear system in eq. 3.1.1 is mean square stable, then*

$$\lim_{k\to\infty} \Phi_i[k] = 0, \ i = 1, \cdots, N; \ \forall q_0. \tag{3.7.57}$$

proof of Lemma 3.7.2: Choose any $z_0, w_0 \in R^n$. Lemma 3.7.1 guarantees

$$\lim_{k\longrightarrow\infty} \mathbf{E}\left[(z_0^{[2]})^T (\Phi(k;0))^{[2]} w_0^{[2]} \mid q_0\right] = 0. \tag{3.7.58}$$

By the definition of the Kronecker product, we know

$$\mathbf{E}\left[(z_0^{[2]})^T (\Phi(k;0))^{[2]} w_0^{[2]} \mid q_0\right] = \mathbf{E}\left[\left(z_0^T \Phi(k;0) w_0\right)^2 \mid q_0\right]. \tag{3.7.59}$$

By eq. 3.7.47, 3.7.48 and 3.7.59, we get

$$\begin{aligned} & \mathbf{E}\left[\left(z_0^T\Phi[k;0]w_0\right)^2 \mid q_0\right] \\ = & \sum_{i=1}^{N} P(q[k]=q_i \mid q_0)\mathbf{E}\left[\left(z_0^T\Phi[k,0]w_0\right)^2 \mid q[k]=q_i,q_0\right]. \end{aligned} \tag{3.7.60}$$

Because $P(q[k]=q_i \mid q_0)\mathbf{E}\left[\left(z_0^T\Phi[k;0]w_0\right)^2 \mid q[k]=q_i,q_0\right] \geq 0$, the combination of eq. 3.7.58 and 3.7.60 yields

$$\lim_{k\to\infty} P(q[k]=q_i \mid q_0)\mathbf{E}\left[\left(z_0^T\Phi(k;0)w_0\right)^2 \mid q[k]=q_i,q_0\right]=0, \tag{3.7.61}$$

$\Phi(k;0)$ is an $n\times n$ matrix, so it can be denoted as $\Phi(k;0)=(a_{mj}(k))_{m=1,\cdots,n;j=1,\cdots,n}$. In eq. 3.7.61, we choose $z_0=e_m$ and $w_0=e_j$ and get

$$\lim_{k\to\infty} P(q[k]=q_i \mid q_0)\mathbf{E}\left[\left(a_{mj}(k)\right)^2 \mid q[k]=q_i,q_0\right]=0, \tag{3.7.62}$$

where $i=1,2,\cdots,N$, $m=1,\cdots,n$ and $j=1,\cdots,n$. By the definition of $\Phi_i[k]$, we know the elements of $\Phi_i[k]$ take the form of

$$P(q[k]=q_i \mid q_0)\mathbf{E}\left[a_{m_1j_1}(k)a_{m_2j_2}(k) \mid q[k]=q_i,q_0\right]$$

where $m_1,m_2,j_1,j_2=1,\cdots,n$. So eq. 3.7.62 guarantees

$$\lim_{k\to\infty}\Phi_i[k]=0,\ \forall q_0.\diamondsuit \tag{3.7.63}$$

Lemma 3.7.3 *$V_\Phi[k]$ is governed by the following dynamic equation*

$$V_\Phi[k]=A_{[2]}\ V_\Phi[k-1]. \tag{3.7.64}$$

with $V_\Phi[0] = p^T \otimes I_{n^2}$.

proof of Lemma 3.7.3: By the definition in eq. 3.7.47, we recursively compute $\Phi_i[k]$ as follows.

$$\begin{aligned}
\Phi_i[k] &= P(q[k] = q_i \mid q_0)\mathbf{E}\left[(A[q[k-1]]\Phi(k-1;0))^{[2]} \mid q[k] = q_i, q_0\right] \\
&= P(q[k] = q_i \mid q_0)\mathbf{E}\left[(A[q[k-1]])^{[2]}(\Phi(k-1;0))^{[2]} \mid q[k] = q_i, q_0\right] \\
&= P(q[k] = q_i \mid q_0)\sum_{j=1}^{N} P(q[k-1] = q_j \mid q[k] = q_i, q_0) \\
&\quad \mathbf{E}\left[(A[q[k-1]])^{[2]}(\Phi(k-1;0))^{[2]} \mid q[k] = q_i, q[k-1] = q_j, q_0\right] \\
&= \sum_{j=1}^{N} A_j^{[2]} \left(P(q[k] = q_i \mid q_0)P(q[k-1] = q_j \mid q[k] = q_i, q_0)\right) \\
&\quad \mathbf{E}\left[(\Phi(k-1;0))^{[2]} \mid q[k] = q_i, q[k-1] = q_j, q_0\right]
\end{aligned}$$

Because $\Phi(k-1;0)$ depends on only $\{q[k-2], q[k-1], \cdots, q[0]\}$ and the jump sequence $\{q[k]\}$ is Markovian, we know

$$\begin{aligned}
&\mathbf{E}\left[(\Phi(k-1;0))^{[2]} \mid q[k] = q_i, q[k-1] = q_j, q_0\right] \\
=\ &\mathbf{E}\left[(\Phi(k-1;0))^{[2]} \mid q[k-1] = q_j, q_0\right] \qquad (3.7.65)
\end{aligned}$$

The probability $P(q[k] = q_i \mid q_0)P(q[k-1] = q_j \mid q[k] = q_i, q_0)$ can be computed

as

$$\begin{aligned} & P(q[k]=q_i \mid q_0)P(q[k-1]=q_j \mid q[k]=q_i,q_0) \\ = \;& P(q[k-1]=q_j, q[k]=q_i \mid q_0) \\ = \;& P(q[k]=q_i \mid q[k-1]=q_j,q_0)P(q[k-1]=q_j \mid q_0) \\ = \;& P(q[k]=q_i \mid q[k-1]=q_j)P(q[k-1]=q_j \mid q_0) \\ = \;& q_{ji}P(q[k-1]=q_j \mid q_0) \end{aligned} \tag{3.7.66}$$

Substitute eq. 3.7.65 and 3.7.66 into the expression of $\Phi_i[k]$, we get

$$\Phi_i[k] = \sum_{j=1}^{N} q_{ji} A_j^{[2]} \Phi_j[k-1], \tag{3.7.67}$$

After combining $\Phi_i[k](i = 1, 2, \cdots, N)$ into $V_\Phi[k]$ as eq. 3.7.49, we get

$$V_\Phi[k] = A_{[2]} V_\Phi[k-1] \tag{3.7.68}$$

We trivially get $V_\Phi[0]$ from $\Phi_i[0]$ by eq. 3.7.49. $\diamondsuit$

For any initial distribution p, we can put a bound on $\mathbf{E}\left[\Phi(k;0)\Phi^T(k;0)\right]$ in the following lemma.

Lemma 3.7.4 *Consider the system in equation 3.1.1 under assumptions 1—5. There exists a positive semi-definite matrix Φ_0 such that*

$$\mathbf{E}\left[\Phi(k;0)\Phi^T(k;0)\right] \leq \sigma_0^k \Phi_0$$

where $\sigma_0 = \lambda_{max}\left(A_{[2]}\right)$.

Proof: Considering the initial distribution of q_0, $\mathbf{E}\left[\Phi(k;0)\Phi^T(k;0)\right]$ can be com-

puted as

$$\mathbf{E}\left[\Phi(k;0)\Phi^T(k;0)\right] = \sum_{i=1}^{N} P(q_0 = q_i)\mathbf{E}\left[\Phi(k;0)\Phi^T(k;0)|q_0 = q_i\right] \quad (3.7.69)$$

By the definition of $V_\Phi[k]$, eq. 3.7.69 can be rewritten as

$$\mathbf{E}\left[\Phi(k;0)\Phi^T(k;0)\right] = \sum_{i=1}^{N} P(q_0 = q_i) devec\left(C_I V_{\Phi,q_i}[k]\right) \quad (3.7.70)$$

where we add a new subscript $_{q_i}$ in V_Φ to emphasize the initial condition $q_0 = q_i$. By Lemma 3.7.3, we know $V_{\Phi,q_i}[k]$ is expressed as

$$V_{\Phi,q_i}[k] = A_{[2]}^k V_{\Phi,q_i}[0], \quad (3.7.71)$$

where $V_{\Phi,q_i}[0]$ is determined by the initial condition. Because $\lambda_{max}(A_{[2]}) = \sigma_0$, we put the following bound,

$$devec\left(C_I V_{\Phi,q_i}[k]\right) \leq \sigma_0^k \Phi_{q_i} \quad (3.7.72)$$

where Φ_{q_i} is a constant semi-definite matrix which depends on q_i. Define $\Phi_0 = \sum_{i=1}^{N} \Phi_{q_i}$. Relaxing the bound in eq.3.7.72, we get

$$devec\left(C_I V_{\Phi,q_i}[k]\right) \leq \sigma_0^k \Phi_0 \quad (3.7.73)$$

Substituting eq. 3.7.73 for all i into eq. 3.7.70 yields

$$\mathbf{E}\left[\Phi(k;0)\Phi^T(k;0)\right] = \sigma_0^k \Phi_0 \quad \diamondsuit \quad (3.7.74)$$

The shift-invariance property of the dropout Markov chain yields that $\mathbf{E}\left[\Phi(k;l)\Phi^T(k;l)\right] = \mathbf{E}\left[\Phi(k-l;0)\Phi^T(k-l;0)\right]$. So the following upper bound holds

$$\mathbf{E}\left[\Phi(k;l)\Phi^T(k;l)\right] \leq \sigma_0^{(k-l)}\Phi_0. \tag{3.7.75}$$

When the system in equation 3.1.1 is asymptotically stable in the mean square sense, we can ignore the initial state by taking the initial time at $-\infty$ and express the state $x[k]$ as the following lemma.

Lemma 3.7.5 *Consider the system in equation 3.1.1 with initial time of $-\infty$. Under assumptions 1—5, the state $x[k]$ can be expressed with the following infinite series.*

$$x[k] = \sum_{l=0}^{\infty} \Phi(k;k-l)Bw[k-l-1]. \tag{3.7.76}$$

Furthermore,

$$\mathbf{E}\left[x^T[k]x[k]\right] < \infty.$$

Proof: If the infinite series in equation 3.7.76 makes sense, $x[k]$ can obviously be computed as the equation. So we just need to prove that the infinite series in equation 3.7.76 is convergent in the mean square sense.

Denote the partial summation in equation 3.7.76 as

$$S(p,q) = \sum_{l=p}^{p+q} \Phi(k;k-l)Bw[k-l-1], \tag{3.7.77}$$

where $p \geq 0$, $q \geq 1$. It can be shown that

$$\begin{aligned} & \sqrt{\mathbf{E}\left[S^T(p,q)S(p,q)\right]} \qquad (3.7.78) \\ \leq & \sum_{l=p}^{p+q} \sqrt{\mathbf{E}\left[\left(\Phi(k;k-l)Bw[k-l-1]\right)^T\left(\Phi(k;k-l)Bw[k-l-1]\right)\right]} \end{aligned}$$

Now consider a single term of the summation in equation 3.7.78.

$$\begin{aligned} & \mathbf{E}\left[\left(\Phi(k;k-l)Bw[k-l-1]\right)^T\left(\Phi(k;k-l)Bw[k-l-1]\right)\right] \\ & = \text{Trace}\left(\mathbf{E}\left[\Phi(k;k-l)Bw[k-l-1]w^T[k-l-1]B^T\Phi^T(k;k-l)\right]\right) \\ & = \text{Trace}\left(\mathbf{E}\left[\Phi(k;k-l)BR_{ww}[0]B^T\Phi^T(k;k-l)\right]\right). \end{aligned}$$

Let $\sigma_B = \lambda_{max}(BR_{ww}[0]B^T)$, then

$$\begin{aligned} & \mathbf{E}\left[\left(\Phi(k;k-l)Bw[k-l-1]\right)^T\left(\Phi(k;k-l)Bw[k-l-1]\right)\right] \\ & \leq \sigma_B \text{Trace}\left(\mathbf{E}\left[\Phi(k;k-l)\Phi^T(k;k-l)\right]\right). \end{aligned}$$

By equation 3.7.75, we put an upper bound:

$$\begin{aligned} & \mathbf{E}\left[\left(\Phi(k;k-l)Bw[k-l-1]\right)^T\left(\Phi(k;k-l)Bw[k-l-1]\right)\right] \qquad (3.7.79) \\ \leq & \sigma_B\sigma_0^l \text{Trace}(\Phi_0). \end{aligned}$$

With the preceding relation, we get

$$\begin{aligned} \sqrt{\mathbf{E}\left[S^T(p,q)S(p,q)\right]} & \leq \sqrt{\sigma_B \text{Trace}(\Phi_0)}\sum_{l=p}^{p+q}\sigma_0^{l/2} \\ & \leq M\sigma_0^{p/2}, \end{aligned}$$

where $M = \frac{\sqrt{\sigma_B \text{Trace}(\Phi_0)}}{1-\sqrt{\sigma_0}}$.

Because $\sigma_0 < 1$, $\lim_{p\to\infty} \sup_{q\geq 1} \mathbf{E}\left[S^T(p,q)S(p,q)\right] = 0$. So we know that the summation in equation 3.7.76 is convergent in the mean square sense.

If we set $p = 0$ and $q = \infty$, $S(p,q) = x[k]$. So

$$\begin{aligned} \mathbf{E}\left[x^T[k]x[k]\right] &\leq M \\ &< \infty. \qquad \diamondsuit \end{aligned}$$

3.7.2 proof of Theorem 3.2.1

proof of Necessity of Theorem 3.2.1: By Lemma 3.7.2, we get

$$\lim_{k\longrightarrow\infty} V_\Phi[k] = 0 \tag{3.7.80}$$

By Lemma 3.7.3, we get $V_\Phi[k] = A^k_{[2]} V_\Phi[0]$ and $V_\Phi[0] = p^T \otimes I_{n^2}$. Therefore eq. 3.7.80 yields

$$\lim_{k\longrightarrow\infty} A^k_{[2]}(p^T \otimes I_{n^2}) = 0 \tag{3.7.81}$$

for any p, the initial distribution of q_0.

$A^k_{[2]}$ is a $Nn^2 \times Nn^2$ matrix. We write $A^k_{[2]}$ as $A^k_{[2]} = [A^1(k), A^2(k), \cdots, A^N(k)]$ where $A^i(k)$ $(i = 1, \cdots, N)$ is a $Nn^2 \times n^2$ matrix. By taking $p_i = 1$ and $p_j = 0$ $(j = 1, \cdots, i-1, i+1, \cdots, N)$, eq. 3.7.81 yields

$$\lim_{k\longrightarrow\infty} A^i(k) = 0 \tag{3.7.82}$$

Thus we get

$$\lim_{k\longrightarrow\infty} A^k_{[2]} = 0. \tag{3.7.83}$$

So $A_{[2]}$ is Schur stable.

proof of Sufficiency of Theorem 3.2.1: Under the given initial condition $x[0] = x_0, q[0] = q_0$, the state $x[k]$ at time n can be expressed as

$$x[k] = \Phi(k;0)x_0 \tag{3.7.84}$$

The conditional variance of $x[k]$ can be computed as

$$\begin{aligned}
& \mathbf{E}\left[x^T[k]x[k]|x_0, q_0\right] \\
= \;& Trace\left(devec\left(\mathbf{E}\left[\Phi^{[2]}(k;0)|q_0\right]x_0^{[2]}\right)\right) \\
= \;& Trace\left(devec\left(\sum_{i=1}^{N} P(q_0 = q_i)\mathbf{E}\left[\Phi^{[2]}(k;0)|q_0 = q_i\right]x_0^{[2]}\right)\right) \\
= \;& Trace\left(devec\left(\sum_{i=1}^{N} P(q_0 = q_i)C_I V_{\Phi,q_i}[k]x_0^{[2]}\right)\right) \\
= \;& Trace\left(devec\left(\sum_{i=1}^{N} P(q_0 = q_i)C_I A_{[2]}^k V_{\Phi,q_i}[0]x_0^{[2]}\right)\right)
\end{aligned} \tag{3.7.85}$$

where the first equality comes from eq. 3.7.84 and the definitions of $devec(\cdot)$ and $^{[2]}$, the third equality comes from the definition of $V_\Phi[k]$ (kote that we add the subscript q_i to emphasize the dependence of $V_\Phi[k]$ on $q_0 = q_i$), and the fourth equality comes from Lemma 3.7.3. By assumption 5, we know $A_{[2]}$ is stable. So $\lim_{k\longrightarrow\infty} A_{[2]}^k = 0$. Considering the expression in eq. 3.7.85, we obtain

$$\lim_{k\longrightarrow\infty} \mathbf{E}\left[x^T[k]x[k]|x_0, q_0\right] = 0, \forall x_0, q_0.\diamondsuit \tag{3.7.86}$$

3.7.3 proof of Theorem 3.2.2

As shown by lemma 3.7.5, $x[k]$ exists in the mean square sense and has finite variance. So we just need to prove the mean of $x[k]$ is constant and the correlation $\mathbf{E}\left[x[k+m]x^T[k]\right]$ is shift-invariant with respect to n.

For reference convenience, we write down the expression of $x[k]$ from equation 3.7.76 again.

$$x[k] = \sum_{l=0}^{\infty} \Phi(k; k-l)Bw[k-l-1]. \tag{3.7.87}$$

So the mean of $x[k]$ can be computed as

$$\begin{aligned} \mathbf{E}[x[k]] &= \sum_{l=0}^{\infty} \mathbf{E}[\Phi(k; k-l)B]\mathbf{E}\left[w[k-l-1]\right] \\ &= 0, \end{aligned}$$

where the first equality follows from the independence between dropouts and w; the second equality follows from the fact w is zero-mean.

The correlation $\mathbf{E}\left[x[k+m]x^T[k]\right]$ can be expressed as

$$\begin{aligned} &\mathbf{E}\left[x[k+m]x^T[k]\right] \\ = &\sum_{k_1=0}^{\infty}\sum_{k_2=0}^{\infty} \mathbf{E}\left[\Phi(k+m; k+m-k_1)Bw[k+m-k_1-1]w^T[k-k_2-1]\right. \\ &\left. B^T\Phi^T(k; k-k_2)\right] \end{aligned}$$

By the white nature of w, we know

$$w[k+m-k_1-1]w^T[k-k_2-1] = \begin{cases} R_{ww}[0], & \text{When } m-k_1 = -k_2 \\ 0 & otherwise \end{cases}$$

Combine the above two equations, we obtain

$$\mathbf{E}\left[x[k+m]x^T[k]\right] = \sum_{l=0}^{\infty} \mathbf{E}\left[\Phi(k+m;k-l)BR_{ww}[0]B^T\Phi^T(k;k-l)\right].$$

Because the dropout Markov chain is time-homogeneous and the initial time is set to $-\infty$, the dropout Markov chain stays in the steady state.
Then $\mathbf{E}\left[\Phi(k+m;k-l)BR_{ww}[0]B^T\Phi^T(k;k-l)\right]$ is shift-invariant with respect to k. Therefore $\mathbf{E}\left[x[k+m]x^T[k]\right]$ is constant with respect to k and $\{x[k]\}$ is WSS.
$\diamondsuit$

3.7.4 Proof of Theorem 3.3.1

Corollary 3.2.1 guarantees that $\{y[k]\}$ is WSS. Then the power of y can be computed as

$$\begin{aligned}
\mathbf{E}\left[y^T[k]y[k]\right] &= \text{Trace}\left(\mathbf{E}\left[y[k]y^T[k]\right]\right) \\
&= \text{Trace}\left(\text{devec}\left(\mathbf{E}\left[y^{[2]}[k]\right]\right)\right) \\
&= \text{Trace}\left(\text{devec}\left(C^{[2]}\mathbf{E}\left[x^{[2]}[k]\right]\right)\right).
\end{aligned}$$

The above equation shows $\mathbf{E}\left[x^{[2]}[k]\right]$ has to be computed in order to get $\mathbf{E}\left[y^T[k]y[k]\right]$. Because the initial time is set to $-\infty$, the dropout Markov chain can be assumed to remain at steady state, i.e. $P(q[k]=q_i)=\pi_i$ $(i=1,2,\cdots,N)$. Let $P_i[k]=\pi_i\mathbf{E}\left[x^{[2]}[k] \mid q[k-1]=q_i\right]$. Then

$$\mathbf{E}\left[x^{[2]}[k]\right] = \sum_{i=1}^{N} P_i[k].$$

$P_i[k+1]$ can be recursively computed as follows.

$$\begin{aligned}
& P_i[k+1] \\
= \;& \pi_i \mathbf{E}\left[(A[k]x[k] + Bw[k])^{[2]} \mid q[k] = q_i\right] \\
= \;& \pi_i A_i^{[2]} \mathbf{E}\left[x^{[2]}[k] \mid q[k] = q_i\right] + \pi_i B^{[2]} \mu_{w_2} \\
= \;& \pi_i A_i^{[2]} \sum_{l=1}^{N} \mathbf{E}\left[x^{[2]}[k] \mid q[k] = q_i, q[k-1] = q_l\right] P(q[k-1] = q_l \mid q[k] = q_i) \\
& + \pi_i B^{[2]} \mu_{w_2} \\
= \;& \pi_i A_i^{[2]} \sum_{l=1}^{N} \mathbf{E}\left[x^{[2]}[k] \mid q[k-1] = q_l\right] P(q[k-1] = q_l \mid q[k] = q_i) + \pi_i B^{[2]} \mu_{w_2} \\
= \;& A_i^{[2]} \sum_{l=1}^{N} q_{li} P_l[k] + \pi_i B^{[2]} \mu_{w_2}, \qquad (3.7.88)
\end{aligned}$$

where $\mu_{w_2} = vec(R_{ww}[0])$ ($vec(\cdot)$ is defined in appendix A, the second equality follows from $\mathbf{E}\left[x[k] \otimes w[k]\right] = 0$ (because $x[k]$ depends linearly on only the past noise inputs $\{w[k-1], w[k-2], \cdots\}$ and w is white); the fourth equality follows from the Markov property of the dropouts.

Let $V_P[k] = \begin{bmatrix} P_1^T[k] & P_2^T[k] & \cdots & P_N^T[k] \end{bmatrix}^T$. Then the recursive computations on $P_i[k]$ yield

$$V_P[k+1] = A_{[2]} V_P[k] + \pi^T \otimes (B^{[2]} \mu_{w_2}). \qquad (3.7.89)$$

Because the initial time is set to $-\infty$, the solution of the above equation is

$$\begin{aligned}
V_P[k] &= \sum_{l=-\infty}^{k} A_{[2]}^{k-l} \left(\pi^T \otimes (B^{[2]} \mu_{w_2})\right) \\
&= \sum_{m=0}^{\infty} A_{[2]}^{m} \left(\pi^T \otimes (B^{[2]} \mu_{w_2})\right) \\
&= constant,
\end{aligned}$$

where the second equality comes from the substitution of the variable , $m = k - l$; the infinite series in the second equality exists and is bounded because $A_{[2]}$ is stable.

Because $V_P[k] = constant$, $P_i[k]$ is also constant with respect to k. Then $P_i[k]$ and $P_i[k+1]$ in equation 3.7.88 all can be simplified as P_i. Taking the operation $devec(\cdot)$ over eq. 3.7.88 will yield the final result. $\diamondsuit$

3.7.5 Proof of Theorem 3.3.2

In this proof, the dropout Markov chain is assumed to stay at the steady state $\pi = \begin{bmatrix} \pi_1 & \pi_2 & \cdots & \pi_N \end{bmatrix}$, i.e. $P(q[k] = q_i) = \pi_i$. The notations in the proof of theorem 3.3.1, P_i and $V_P[k]$, are reused. Because $V_P[k]$ is constant with respect to n (with the initial time $-\infty$), it is usually simply denoted as V_P.

At the begining, the wide sense stationarity of $\{y^{[2]}[k]\}$ is proved.

$$y^{[2]}[k] = C^{[2]} x^{[2]}[k]. \tag{3.7.90}$$

So $\{y^{[2]}[k]\}$ is WSS if $\{x^{[2]}[k]\}$ is WSS. From the proof of theorem 3.2.2, we know $\mathbf{E}\left[x^{[2]}[k]\right] = \mathbf{E}\left[x[k+0] \otimes x[k]\right]$ is constant with respect to k, which is denoted as μ_{x_2}. From the proof of theorem 3.3.1, we know

$$\begin{aligned} \mu_{x_2} &= \sum_{i=1}^{N} P_i \\ &= C_I \sum_{l=0}^{\infty} A_{[2]}^{l} \left(\pi^T \otimes \left(B^{[2]} \mu_{w_2}\right)\right). \end{aligned}$$

To prove the wide sense stationarity of $\{x^{[2]}[k]\}$, we only need to show that $\mathbf{E}\left[\left(x^{[2]}[k+m]\right)^T x^{[2]}[k]\right]$ exists and is shift-invariant with respect to k. Before

that, we establish the following two propositions

$$\begin{aligned} \lim_{k\longrightarrow\infty} x^{[2]}[k]\otimes x^{[2]}[k] &= 0, \text{when } w=0; && (3.7.91)\\ \mathbf{E}\left[\left(x^{[2]}[k]\right)^T x^{[2]}[k]\right] &< \infty, \text{when } w \text{ is an i.i.d. process.} && (3.7.92)\end{aligned}$$

Obviously $x^{[2]}[k]\otimes x^{[2]}[k] = x^{[4]}[k]$. Define

$$P_{4,i}[k] = \pi_i \mathbf{E}\left[x^{[4]}[k] \mid q[k-1]=q_i\right],$$

and

$$V_{P_4}[k] = \left[\begin{array}{cccc} P_{4,1}^T[k] & P_{4,2}^T[k] & \cdots & P_{4,N}^T[k] \end{array}\right]^T.$$

When $w = 0$, we get $V_{P_4}[k] = A_{[4]}V_{P_4}[k-1]$ with arguments similar to the ones used in the proof of theorem 3.2.1. Because of the stability of $A_{[4]}$, we know $\lim_{n\longrightarrow\infty} V_{P_4}[k] = 0$. Because $\mathbf{E}\left[x^{[2]}[k]\otimes x^{[2]}[k]\right] = (C_I\otimes I_{n^2})V_{P_4}[k]$, $\lim_{n\longrightarrow\infty} x^{[2]}[k]\otimes x^{[2]}[k] = 0$, i.e. the proposition in equation 3.7.91 holds. Then we know the effect of the initial conditions on the variance (correlation) of $x^{[2]}[k]$ can be eventually forgotten. So we can assume zero initial conditions without loss of generality.

Because the proposition in equation 3.7.91 holds, the expression of $x[k]$ in equation 3.7.76 still holds in the fourth moment sense. We write it down again.

$$x[k] = \sum_{l=0}^{\infty} \Phi(k;k-l)Bw[k-l-1].$$

Denote the partial summation in the above infinite series as

$$S(p,q) = \sum_{l=p}^{p+q} \Phi(k;k-l)Bw[k-l-1],$$

where $p \geq 0$, $q \geq 1$. It can be shown that [9]

$$\begin{aligned}
& \sqrt[4]{\mathbf{E}\left[\left(S^{[2]}(p,q)\right)^T S^{[2]}(p,q)\right]} \\
\leq & \sum_{l=p}^{p+q} \sqrt[4]{\mathbf{E}\left[\mathrm{Trace}\left(\left(\Phi(k;k-l)Bw[k-l-1]\right)\left(\Phi(k;k-l)Bw[k-l-1]\right)^T\right)\right]^2} \\
\leq & \sum_{l=p}^{p+q} \sqrt[4]{n\mathrm{Trace}\left(\mathbf{E}\left[\left(\Phi(k;k-l)Bw[k-l-1]\right)\left(\Phi(k;k-l)Bw[k-l-1]\right)^T\right]^2\right)}
\end{aligned} \tag{3.7.94}$$

Because $\lambda_{max}(A_{[4]}) = \sigma_4 < 1$, we can similarly put the following upper bound.

$$\mathbf{E}\left[\Phi(k;k-l)\left(\Phi(k;k-l)\right)^T\right]^2 \leq \Phi_4\sigma_4^l$$

where Φ_4 is a constant positive definite matrix.

Thus we get an upper bound on the single term in eq. 3.7.94.

$$\begin{aligned}
& n\mathrm{Trace}\left(\mathbf{E}\left[\left(\Phi(k;k-l)Bw[k-l-1]\right)\left(\Phi(k;k-l)Bw[k-l-1]\right)^T\right]^2\right) \\
< & \ \sigma_4^k M_4
\end{aligned}$$

[9] Eq. 3.7.94 comes from the fact that

$$\mathbf{E}\left[\left(a_1+a_2+\cdots+a_N\right)^2\right] \leq N\left(\mathbf{E}[a_1^2]+\mathbf{E}[a_2^2]+\cdots+\mathbf{E}[a_N^2]\right) \tag{3.7.93}$$

where a_i is random variable.

where M_4 is a positive constant. Then we get the upper bound on $S(p,q)$

$$\sqrt[4]{\mathbf{E}\left[\left(S^{[2]}(p,q)\right)^T S^{[2]}(p,q)\right]} < \frac{\sqrt[4]{M_4}}{1-\sqrt[4]{\sigma_4}}\sigma_4^{p/4}$$

If we set $p=0$ and $q=\infty$, $S(p,q)=x[k]$. So

$$\begin{aligned}\mathbf{E}\left[\left(x^{[2]}[k]\right)^T x^{[2]}[k]\right] &\leq \frac{\sqrt[4]{M_4}}{1-\sqrt[4]{\sigma_4}} \\ &< \infty\end{aligned}$$

i.e. the proposition in equation 3.7.92 holds. Then we know that
$\mathbf{E}\left[\left(x^{[2]}[k+m]\right)^T x^{[2]}[k]\right]$ exists. Following the proof of theorem 3.2.2, we know $\mathbf{E}\left[\left(x^{[2]}[k+m]\right)^T x^{[2]}[k]\right]$ is shift-invariant with respect to k. So $\{x^{[2]}[k]\}$ is WSS.

In order to guarantee the ergodicity of $\{x^{[2]}[k]\}$, we have to show that

$$\lim_{m\longrightarrow\infty}\mathbf{E}\left[x^{[2]}[k+m]\otimes x^{[2]}[k]\right]-\mu_{x_2}^{[2]}=0 \tag{3.7.95}$$

Define $F_i[m]=\pi_i\mathbf{E}\left[x^{[2]}[k+m]\otimes x^{[2]}[k]\mid q[k+m-1]=q_i\right]$ (Because of the shift-invariance of the dropouts and the input noise,
$\mathbf{E}\left[x^{[2]}[k+m]\otimes x^{[2]}[k]\mid q[k+m-1]=q_i\right]$ is constant with respect to k), $V_F[m]=\begin{bmatrix} F_1^T[m] & F_2^T[m] & \cdots & F_N^T[m]\end{bmatrix}^T$. Then

$$\mathbf{E}\left[x^{[2]}[k+m]\otimes x^{[2]}[k]\right]=(C_I\otimes I_{n^2})V_F[m].$$

When $m \geq 1$, $F_i[m]$ can be recursively computed as

$$\begin{aligned} F_i[m] &= \pi_i \mathbf{E}\left[(A[k+m-1]x[k+m-1] \right. \\ &\qquad \left. +Bw[k+m-1])^{[2]} \otimes x_n^{[2]} \mid q[k+m-1] = q_i\right] \\ &= \pi_i (A_i^{[2]} \otimes I_{n^2}) \mathbf{E}\left[x^{[2]}[k+m-1] \otimes x^{[2]}[k] \mid q[k+m-1] = q_i\right] \\ &\quad +(\pi_i B^{[2]} \mu_{w_2}) \otimes \mathbf{E}\left[x^{[2]}[k] \mid q[k+m-1] = q_i\right] \\ &= (A_i^{[2]} \otimes I_{n^2}) \sum_{l=1}^{N} q_{li} F_l[m-1] + (B^{[2]} \mu_{w_2}) \otimes \sum_{l=1}^{N} q_{li}(m) P_l. \end{aligned}$$

where $\mu_{w_2} = vec(R_{ww}[0])$, $q_{li}(m) = P(q[k+m-1] = q_i \mid q[k-1] = q_l)$ and $q_{li} = P(q[k] = q_i \mid q[k-1] = q_l)$. We want to determine $F_i[m]$ when $m \longrightarrow \infty$.

We know $\lim_{m \longrightarrow \infty} q_{li}(m) = \pi_i$, so that

$$\lim_{m \longrightarrow \infty} \sum_{l=1}^{N} q_{li}(m) P_l = \pi_i \sum_{l=1}^{N} P_l = \pi_i \mu_{x_2}.$$

Then we can replace the update of $F_i[m]$ in the above equation with its limiting value as $m \to \infty$ to obtain

$$F_i[m] = (A_i^{[2]} \otimes I_{n^2}) \sum_{l=1}^{N} q_{li} F_l[m-1] + \pi_i (B^{[2]} \mu_{w_2}) \otimes \mu_{x_2} \tag{3.7.96}$$

So

$$V_F[m] = (A_{[2]} \otimes I_{n^2}) V_F[m-1] + (\pi \otimes (B^{[2]} \mu_{w_2})) \otimes \mu_{x_2} \tag{3.7.97}$$

Based on the above update, we get the following limit

$$\begin{aligned}\lim_{m\longrightarrow\infty} V_F[m] &= \sum_{l=0}^{\infty}(A_{[2]}\otimes I_{n^2})^l\left((\pi\otimes(B^{[2]}\mu_{w_2}))\otimes\mu_{x_2}\right)\\ &= \sum_{l=0}^{\infty}(A_{[2]}^l\otimes I_{n^2})\left((\pi\otimes(B^{[2]}\mu_{w_2}))\otimes\mu_{x_2}\right)\\ &= \left(\sum_{l=0}^{\infty}A_{[2]}^l(\pi\otimes(B^{[2]}\mu_{w_2}))\right)\otimes\mu_{x_2}\end{aligned}$$

Recall that

$$\mu_{x_2} = C_I\sum_{l=0}^{\infty}A_{[2]}^l(\pi\otimes(B^{[2]}\mu_{w_2}))$$

and

$$\mathbf{E}\left[x^{[2]}[k+m]\otimes x^{[2]}[k]\right] = (C_I\otimes I_{n^2})V_F[m].$$

So we get

$$\begin{aligned}&\lim_{m\longrightarrow\infty}\mathbf{E}\left[x^{[2]}[k+m]\otimes x^{[2]}[k]\right]-\mu_{x_2}\otimes\mu_{x_2}\\ &= (C_I\otimes I_{n^2})\left(\textstyle\sum_{l=0}^{\infty}A^l(\pi\otimes(B^{[2]}\mu_{w_2}))\right)\otimes\mu_{x_2}-\mu_{x_2}\otimes\mu_{x_2}\\ &= \left(C_I\textstyle\sum_{l=0}^{\infty}A^l(\pi\otimes(B^{[2]}\mu_{w_2}))\right)\otimes(I_{n^2}\mu_{x_2})-\mu_{x_2}\otimes\mu_{x_2}\\ &= \mu_{x_2}\otimes\mu_{x_2}-\mu_{x_2}\otimes\mu_{x_2}\\ &= 0.\end{aligned}$$

This means that $\{x^{[2]}[k]\}$ is ergodic and so $\{y^{[2]}[k]\}$ is also ergodic. $\diamondsuit$

3.7.6 Proof of Theorem 3.5.1:

Note that if $q[n_1, n_2] = q_i$ and $q[n_1, n_2 + k] = q_j$, then the states at time $n_1 + 1$ can be written as

$$\begin{aligned} x[n_1+1, n_2] &= A_i x[n_1, n_2] + B_i(x[n_1, n_2-1] + x[n_1, n_2+1]) + Fw[n_1, n_2] \\ x[n_1+1, n_2+k] &= A_j x[n_1, n_2+k] + B_j(x[n_1, n_2+k-1] + x[n_1, n_2+k+1]) \\ &\quad + Fw[n_1, n_2+k] \end{aligned}$$

Here, when $q[n_1, n_2] = q_i$, we denote $A(q[n_1, n_2])$ and $B(q[n_1, n_2])$ as A_i and B_i respectively.

So we use this to write out for $k \geq 2$,

$$\begin{aligned} & \overline{P}_k^{i,j} \\ = \;& \pi_i \pi_j \mathbf{E}\left[x[n_1+1, n_2] x^T[n_1+1, n_2+k] \;\middle|\; q[n_1, n_2] = q_i, q[n_1, n_2+k] = q_j\right] \\ = \;& \pi_i \pi_j A_i \mathbf{E}\left[x[n_1, n_2] x^T[n_1, n_2+k] \;\middle|\; q[n_1, n_2] = q_i, q[n_1, n_2+k] = q_j\right] A_j^T \\ + \;& \pi_i \pi_j A_i \mathbf{E}\left[x[n_1, n_2] x^T[n_1, n_2+k-1] \;\middle|\; q[n_1, n_2] = q_i, q[n_1, n_2+k] = q_j\right] B_j^T \\ + \;& \pi_i \pi_j A_i \mathbf{E}\left[x[n_1, n_2] x^T[n_1, n_2+k+1] \;\middle|\; q[n_1, n_2] = q_i, q[n_1, n_2+k] = q_j\right] B_j^T \\ + \;& \pi_i \pi_j B_i \mathbf{E}\left[x[n_1, n_2-1] x^T[n_1, n_2+k] \;\middle|\; q[n_1, n_2] = q_i, q[n_1, n_2+k] = q_j\right] A_j^T \\ + \;& \pi_i \pi_j B_i \mathbf{E}\left[x[n_1, n_2-1] x^T[n_1, n_2+k-1] \;\middle|\; q[n_1, n_2] = q_i, q[n_1, n_2+k] = q_j\right] B_j^T \\ + \;& \pi_i \pi_j B_i \mathbf{E}\left[x[n_1, n_2-1] x^T[n_1, n_2+k+1] \;\middle|\; q[n_1, n_2] = q_i, q[n_1, n_2+k] = q_j\right] B_j^T \\ + \;& \pi_i \pi_j B_i \mathbf{E}\left[x[n_1, n_2+1] x^T[n_1, n_2+k] \;\middle|\; q[n_1, n_2] = q_i, q[n_1, n_2+k] = q_j\right] A_j^T \\ + \;& \pi_i \pi_j B_i \mathbf{E}\left[x[n_1, n_2+1] x^T[n_1, n_2+k-1] \;\middle|\; q[n_1, n_2] = q_i, q[n_1, n_2+k] = q_j\right] B_j^T \\ + \;& \pi_i \pi_j B_i \mathbf{E}\left[x[n_1, n_2+1] x^T[n_1, n_2+k+1] \;\middle|\; q[n_1, n_2] = q_i, q[n_1, n_2+k] = q_j\right] B_j^T \end{aligned}$$

There are nine conditional expectations in the above equation. The first expectation can be simplified as follows,

$$
\begin{aligned}
\pi_i\pi_j \quad & \mathbf{E}\left[x[n_1,n_2]x^T[n_1,n_2+k]|q[n_1,n_2]=q_i,q[n_1,n_2+k]=q_j\right] \\
= \quad & \sum_{l,m=1}^{N}\pi_i\pi_j\mathbf{E}\left[x[n_1,n_2]x^T[n_1,n_2+k] \quad | \quad q[n_1,n_2]=q_i,q[n_1,n_2+k]=q_j,\right. \\
& \qquad\qquad\qquad\qquad \left. q[n_1-1,n_2]=q_l,q[n_1-1,n_2+k]=q_m\right] \\
& P(q[n_1-1,n_2]=q_l,q[n_1-1,n_2+k]=q_m|q[n_1,n_2]=q_i,q[n_1,n_2+k]=q_j) \\
= \quad & \sum_{l,m=1}^{N}\pi_i\pi_j\mathbf{E}\left[x[n_1,n_2]x^T[n_1,n_2+k]|q[n_1-1,n_2]=q_l,q[n_1-1,n_2+k]=q_m\right] \\
& P(q[n_1-1,n_2]=q_l|q[n_1,n_2]=q_i)P(q[n_1-1,n_2+k]=q_m|q[n_1,n_2+k]=q_j) \\
= \quad & \pi_i\pi_j\sum_{l=1}^{N}\sum_{m=1}^{N}\overline{P}_k^{l,m}\frac{q_{mj}q_{li}}{\pi_i\pi_j} \\
= \quad & \sum_{l=1}^{N}\sum_{m=1}^{N}q_{li}q_{mj}\overline{P}_{l,m}(L)
\end{aligned}
$$

The second expectation can be simplified as shown below. The third, fourth, and seventh expectations have similar derivations and aren't shown.

$$
\begin{aligned}
& \pi_i\pi_j\mathbf{E}\left[x[n_1,n_2]x^T[n_1,n_2+k-1] \quad | \quad q[n_1,n_2]=q_i,q[n_1,n_2+k]=q_j\right] \\
= \quad & \pi_i\pi_j\mathbf{E}\left[x[n_1,n_2]x^T[n_1,n_2+k-1] \quad | \quad q[n_1,n_2]=q_i\right] \\
= \quad & \pi_i\pi_j\sum_{l,m=1}^{N}\mathbf{E}\left[x[n_1,n_2]x^T[n_1,n_2+k-1]|q[n_1,n_2]=q_i,q[n_1-1,n_2]=q_l,\right. \\
& \left. q[n_1-1,n_2+k-1]=q_m\right]P(q[n_1-1,n_2]=q_l,q[n_1-1,n_2+k-1]=q_m|q[n_1,n_2]=q_i) \\
= \quad & \pi_i\pi_j\sum_{l,m=1}^{N}\mathbf{E}\left[x[n_1,n_2]x^T[n_1,n_2+k-1]|q[n_1-1,n_2]=q_l,q[n_1-1,n_2+k-1]=q_m\right] \\
& P(q[n_1-1,n_2]=q_l \quad | \quad q[n_1,n_2]=q_i)P(q[n_1-1,n_2+k-1]=q_m) \\
= \quad & \pi_j\sum_{l,m=1}^{N}q_{li}\overline{P}_{k+1}^{l,m}
\end{aligned}
$$

The fifth expectation can be simplified as shown below. The sixth, eight, and ninth expectations have similar deriviations and aren't shown,

$$\begin{aligned} & \pi_i\pi_j \mathbf{E}\left[x[n_1, n_2-1]x^T[n_1, n_2+k-1] | q[n_1, n_2] = q_i, q[n_1, n_2+k] = q_j\right] \\ = \; & \pi_i\pi_j \mathbf{E}\left[x[n_1, n_2-1]x^T[n_1, n_2+k-1]\right] \\ = \; & \pi_i\pi_j \sum_{l,m=1}^{N} \overline{P}_k^{l,m} \end{aligned}$$

With these simplifications inserted into the original system equaqtion, we obtain the third equation in theorem 3.5.1. A similar derivation can be used to obtain the first two equations. $\diamondsuit$

CHAPTER 4

STABILITY OF QUANTIZED CONTROL SYSTEMS

4.1 Introduction

In recent years there has been a considerable amount of work studying the stability [1] of quantized feedback control systems [19][69] [5] [21] [20] [33] [25] [14] [41] [62] [61] [53] [30]. These papers may be classified into two groups: static and dynamic quantization policies. Static policies [19] [69] [5] [21] [20] [33] [25] presume that data quantization at time k is only dependent on the data at time k. Such policies are sometimes said to be memoryless. In dynamic policies [14] [41] [62] [61] [53] [30] data quantization at time k depends on data at time instants less than or equal to k. The major advantage of static quantization policies is the simplicity of their coding/decoding schemes. In [19], however, it was proven that static policies with a finite number of quantization levels cannot achieve asymptotic stability. A finite number of quantization levels can only achieve practical stability (i.e. states converge into a bounded set) [69] [5] [21]. When an infinite number of quantization levels are available, sufficient bounds for asymptotic stability under static quantization were derived in [33] [25] using robust stability methods. It was shown in [20] that the least dense static quantizer with an infinite number of quantization levels is the logarithmic quantizer.

[1] For noise-free quantized systems, the concerned stability is asymptotic stability, i.e. the state converges to 0. For quantized systems with bounded noise, the concerned stability is BIBO (bounded-in-bounded-output) stability, i.e. the state will be eventually bounded.

Dynamic policies have been shown to achieve asymptotic stability with a finite number of quantization levels [14]. These policies presume that the state, $x[k] \in R^N$, at time instant k lies inside a set $P[k]$ called the *uncertainty set*. If $P[k]$ converges to 0, i.e. every point in $P[k]$ converges to 0, then the system is asymptotically stable. The basic approach was introduced in [14]. In that paper the uncertainty set, $P[k]$, is a rectangle centered at the origin. $P[k]$ is partitioned into M^N small rectangles. Denote the small rectangles as $P_i[k]$ $(i = 0, 1, \cdots, M^N - 1)$. If $x[k] \in P_j[k]$ then the index j is transmitted. Based on the received symbol, j, the encoder knows $x[k] \in P_j[k]$ and comes up with an estimate of $x[k]$ for control purpose. Note that the encoder may receive j with certain delay. At time $k+1$, the uncertainty set $P_j[k]$ is propagated to set $P[k+1]$ using what we know of the plant's dynamics. That paper provided sufficent conditions for the convergence of the sequence, $\{P[k]\}$, to zero. A later paper [41] extends [14] by removing the center assumption on $P[k]$ and providing a tighter sufficient stability condition.

A significant generalization of the approach in [14] was presented in [62] [61]. Suppose the eigenvalues of the quantized system are denoted as λ_i for $i = 1, \ldots, N$ and assume $P[k]$ is shaped like a rectangle. Let the i^{th} side of $P[k]$ be equally partitioned into $2^{R_i[k]}$ parts, i.e. $R_i[k]$ bits are assigned to the i^{th} dimension ($R_i[k]$ must be an integer). The total number of bits is $R[k] = \sum_{i=1}^{N} R_i[k]$, i.e. there are $Q[k] = 2^{R[k]}$ quantization levels at time k. The approach discussed in [62] [61] assumes a time-varying bit rate policy in which $R[k]$ varies over time, but has an average value $\overline{R} = \lim_{T\to\infty} \frac{1}{T} \sum_{k=0}^{T-1} R[k]$. In [62] [61], it is shown that the quantized system is asymptotically stabilizable if and only if

$$\overline{R} > \sum_{i=1}^{N} \max(0, \log_2 |\lambda_i|) \tag{4.1.1}$$

In [53], stability is interpreted in the stochastic sense. A necessary and sufficient condition similar to equation 4.1.1 is provided for the moment of $x[k]$ to converge to 0.

In real networks, constant bit rates may be more desirable than time-varying bit rates due to power and bandwidth efficiency [29]. This chapter therefore focuses on constant bit rate policies in which $R[k] = R$ for all k and R is a constant. In [62] a sufficient condition for asymptotic stability under constant bit rates is given as

$$R = \sum_{i=1}^{N} R_i > \sum_{i=1}^{N} \max\left(0, \lceil \log_2 |\lambda_i| \rceil\right). \tag{4.1.2}$$

where $\lceil \cdot \rceil$ means $\lceil x \rceil = \min\{n | n > x, n \in \mathcal{N}\}$. There can be a significant gap between the bounds in equations 4.1.1 and 4.1.2, so it is natural to ask whether there exists a tighter bound than the one in equation 4.1.2 for the constant bit rate case. That is precisely the question addressed in this chapter.

This chapter shows that a lower bound on the number of quantization levels required for asymptotic stability is given by the equation

$$Q = 2^R \geq \left\lceil \prod_{i=1}^{N} \max(1, |\lambda_i|) \right\rceil. \tag{4.1.3}$$

We then introduce a **dynamic bit assignment policy** (DBAP) that actually achieves this bound. This bit assignment is done as follows. Suppose $P[k]$ is a parallelogram, there are $Q = 2^R$ quantization levels and Q is an integer. At every step, only the "longest" side of $P[k]$ (in the sense of a weighted length) is equally partitioned into Q parts; the other sides aren't partitioned. Because no side is always the longest, the bit assignments are dynamic rather than static. We prove

that the lower bound in equation 4.1.3 is realized by this policy in section 4.4. In other words, we achieve the minimum bit rate for asymptotic stability under the constant bit rate constraint [48].

Up to now we have reviewed the literature on asymptotic stability, i.e. the concerned stability for noise free quantized systems. Besides noise free quantized system, there are some quantized systems perturbed by bounded noise. Significant attention has also been caught on stability of such bounded system, i.e. BIBO stability. In [30], sufficient BIBO stability conditions are provided for the quantized systems with diagonalizable system matrices. In [62], the minimum bit rate for BIBO stability is derived under the time-varying bit rate configuration. The achieved minimum (average) bit rate is exactly specified by eq. 4.1.1. When the constant bit rate constraint is considered, it is shown in [62] that BIBO stability is achievable if the bit rate R satisfies eq. 4.1.2. Again there may exist a gap between the bounds in eq. 4.1.1 and 4.1.2. In section 4.5, we derive the minimum bit rate for BIBO stability under the constant bit rate configuration, which is specified by eq. 4.1.3 and achieved by a modified version of DBAP.

Most of the aforementioned papers focus on state quantization. In reality, the system state is not necessarily available. Perhaps what we have is only output. For such systems, there are two types of quantization strategies, quantizing the output directly or first estimating the state and then quantizing the estimated state. In [61] [54], the second strategy is implemented and the minimum bit rate (for both asymptotic stability and BIBO stability) is achieved under the time-varying bit rate configuration. In [58], the first strategy, i.e. the direct output quantization, is implemented under the constant bit rate constraint and sufficient (BIBO) stability conditions are achieved based on l_1 control theories. The novelty of [58] lies in its

bit allocation, which allocates the available bits to both the current output and the previous outputs. In section 4.7, we comment on implementing the second strategy, i.e. quantizing the estimated state, on an output feedback system and obtaining the minimum bit rate (for both asymptotic stability and BIBO stability) under the constant bit rate constraint.

This chapter is organized as followes. Section 4.2 formally defines the quantized feedback control system. Section 4.3 reviews notational conventions and preliminary results. Section 4.4 derives the minimum constant bit rate for asymptotic stability of noise free quantized systems. Section 4.5 derives the minimum constant bit rate for BIBO stability of quantized systems with bounded noise. Section 4.6 comments on extending the results on the state feedback systems to the output feedback case and derives the minimum constant bit rate for asymptotic stability and BIBO stability. Section 4.7 summarizes the achieved results. The proofs to all theorems are found in the appendix, section 4.8.

4.2 Quantized feedback control systems

This paper studies a quantized feedback control system with dropouts, which is shown in figure 4.1. For the system in figure 4.1, let $x[k] \in R^N$ be the state. The state, $x[k]$, is quantized into one of Q symbols and sent over the communication channel. The transmitted symbol $s[k]$ is received by the decoder with one step delay $s'[k] = s[k-1]$ or is dropped $s'[k] = \phi$. A dropout is denoted as that the decoder receives the empty signal, ϕ. We assume that the average dropout rate is denoted as ε with a detailed dropout model given later. The decoder uses the received symbols $\{s'[k]\}$ to estimate the state. This estimate is denoted as $x^q[k]$, which can also be viewed as a quantized version of $x[k]$. The control input $u[k]$

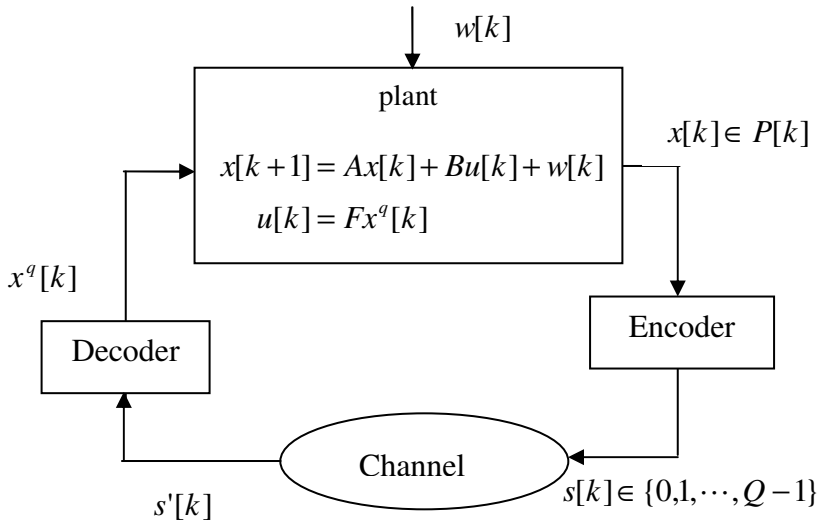


Figure 4.1. Quantized feedback control system

is then constructed from $x^q[k]$. In figure 4.1, the input signal, $w[k]$, represents an exogenous noise that satisfies

$$\|w\|_\infty \leq M \tag{4.2.4}$$

A noise free system is a special case of the systems in figure 4.1 by setting $M = 0$.

The plant in figure 4.1 is a discrete-time linear system whose state equations are

$$\begin{cases} x[k+1] &= Ax[k] + Bu[k] + w[k] \\ u[k] &= Kx^q[k] \end{cases} \tag{4.2.5}$$

We study stability under the following assumptions

1. (A, B) is controllable where A is in real Jordan canonical form, i.e. $A = diag(J_1, J_2, \cdots, J_p)$ [31]. J_i is an $n_i \times n_i$ real matrix with a single real

eigenvalue λ_i or a pair of conjugate eigenvalues λ_i and $\overline{\lambda}_i$. All eigenvalues λ_i are assumed to be unstable, i.e. $|\lambda_i| > 1$.

2. The initial condition $x[0]$ lies in a parallelogram $P[0]$.

3. Transmitted symbols, $s[k]$, are dropped at the rate of ε symbols per transmission. The precise definition of ε will be found in equation 4.2.7. We assume that the encoder and decoder both know whether a dropout has occurred.

4. Both the encoder and the decoder know the system matrices (A and B), the coding-decoding policy and the control law. They also agree upon the initial uncertainty set, i.e. $P[0]$.

Assumption 2 requires that the initial state is known to lie within a specified parallelogram $P[0]$. This set may be written as

$$P[0] = x^q[0] + U[0]$$

where $x^q[0]$ is the center of $P[0]$ and $U[0]$ is a parallelogram centered at the origin and defined in equations 4.3.10-4.3.11.

Assumption 3 comes from the non-determinism of the network. We introduce a dropout indicator $d[k]$,

$$d[k] = \begin{cases} 1, & \text{the symbol at time } k \text{ is dropped} \\ 0, & \text{otherwise} \end{cases} \tag{4.2.6}$$

We assume that the dropout model satisfies

$$\varepsilon = \lim_{L \longrightarrow \infty} \frac{1}{L} \sum_{i=1}^{L} d[i + k_0], \tag{4.2.7}$$

for all $k_0 \geq 0$ where ε is the "average" dropout rate and the convergence in equation 4.2.7 is uniform with respect to k_0.

Assumption 4 requires that the coder and the decoder deal with the same initial uncertainty, and share the same coding-decoding policy and control law so that the symbol produced by the encoder can be correctly interpreted by the decoder. This is a strong assumption for it requires that the encoder and decoder are "synchronized". Maintaining such synchronization in a fault-tolerant manner requires further study, but that study is not done in this thesis.

4.3 Preliminary results

This section introduces notational conventions and outlines a proof for the bound in equation 4.1.3. For the matrix A in assumption 1, let

$$\gamma(A) = \prod_{i=1}^{p} \left(\max(1, |\lambda_i|)\right)^{n_i} \tag{4.3.8}$$

We assume all eigenvalues of A are unstable. So $\gamma(A) = |\det(A)|$, where $\det(\cdot)$ is the determinant of a matrix.

The state $x[k]$ at time k is quantized with respect to a parallelogram representing the quantization "uncertainty". These uncertainty sets are represented as

$$P[k] = x^q[k] + U[k] \tag{4.3.9}$$

where $x^q[k] \in R^N$ is the center of $P[k]$ and $U[k]$ is a parallelogram with its center at the origin. The parallelogram $U[k]$ is formally represented by a set of vectors $\{v_{i,j}[k] \in R^{n_i}\}$ where $i = 1, \ldots, p$ and $j = 1, \ldots, n_i$. The "side" of the parallelogram associated with the ith Jordan block in A is denoted as the convex hull

$$S_i[k] = \text{Co}\left\{v \ : \ v = \sum_{j=1}^{n_i} (\pm\frac{1}{2}) v_{i,j}[k]\right\} \tag{4.3.10}$$

The entire parallelogram, $U[k]$, may therefore be expressed as the Cartesian product of the sides, $S_i[k]$. In other words

$$U[k] = \prod_{i=1}^{p} S_i[k] \tag{4.3.11}$$

We illustrate the definitions in eq. 4.3.10 and 4.3.11 through a 2-dimensional example. Suppose $A = \begin{bmatrix} \lambda & 1 \\ 0 & \lambda \end{bmatrix}$. $U[k]$ is specified by $\{v_{1,1}[k], v_{1,2}[k]\}$. $S_1[k]$ in eq. 4.3.10 is the convex hull of 4 vertices.

$$\begin{aligned} S_1[k] &= \text{Co}\left\{\frac{1}{2}v_{1,1}[k] + \frac{1}{2}v_{1,2}[k], \frac{1}{2}v_{1,1}[k] - \frac{1}{2}v_{1,2}[k]\right. \\ &\quad \left. -\frac{1}{2}v_{1,1}[k] + \frac{1}{2}v_{1,2}[k], -\frac{1}{2}v_{1,1}[k] - \frac{1}{2}v_{1,2}[k]\right\} \\ U[k] &= \prod_{i=1}^{1} S_i[k] = S_1[k] \end{aligned}$$

The volume of U is defined as $\text{vol}(U) = \int_{x \in U} 1 \cdot dx$. The "size" of $U[k]$ is measured by its diameter $d_{\max}(U[k])$. The diameter of U is defined as

$$d_{\max}(U) = \sup_{x,y \in U} \|x - y\|_2 \tag{4.3.12}$$

where $\|\cdot\|_2$ denotes Euclidean 2-norm of a vector. The quantization error is defined as $e[k] = x[k] - x^q[k]$. By equation 4.3.9, we know $e[k] \in U[k]$. When a quantization policy is used, we will generate a sequence of uncertainty sets, $\{U[k]\}$. The following lemma asserts that the convergence of the diameter of $U[k]$ is equivalent to asymptotic stability of the system.

Lemma 4.3.1 *The system in equation 4.2.5 is asymptotically stable (with $M = 0$) if and only if the sequence of uncertainty sets, $\{U[k]\}$, satisfies*

$$\lim_{k \longrightarrow \infty} d_{\max}(U[k]) = 0. \tag{4.3.13}$$

Lemma 4.3.1 can be proven in a manner analogous to that found in Lemma 3.5.1 of [61]. For a quantized system with bounded noise, we can get results similar to those in Lemma 4.3.1.

Lemma 4.3.2 *The system in equation 4.2.5 is BIBO stable (with $M > 0$) if and only if the sequence of uncertainty sets, $\{U[k]\}$, satisfies*

$$\limsup_{k \longrightarrow \infty} d_{\max}(U[k]) < \infty. \tag{4.3.14}$$

Remark: By Lemmas 4.3.1 and 4.3.2, we know the convergence and boundedness of the state $\{x[k]\}$ are equivalent to the convergence and boundedness of the quantization error $\{e[k]\}$ with $e[k] \in U[k]$. Therefore the ongoing discussions will focus on the evolution of $U[k]$.

A lower bound on the number of quantization levels required to stabilize the feedback control system is stated below in theorem 4.3.1. We only sketch the proof of this theorem as the proof's method directly follows that used in [30].

Theorem 4.3.1 *Under assumptions 1 - 4, if the quantized feedback system in equation 4.2.5 (with $M = 0$) can be asymptotically stabilized, then the number of quantization levels, Q, satisfies*

$$Q \geq \left\lceil \gamma(A)^{\frac{1}{1-\varepsilon}} \right\rceil \tag{4.3.15}$$

Sketch of Proof: The volume of $U[k]$ (in the worst case) is updated by

$$\text{vol}(U[k+1]) \begin{cases} \geq \frac{|\det(A)|}{Q}\text{vol}(U[k]), & d[k] = 0 \\ = |\det(A)|\text{vol}(U[k]), & d[k] = 1 \end{cases}$$

Because of asymptotic stability, lemma 4.3.1 implies $vol(U[k]) \to 0$ as $k \to \infty$. This volume limit, together with the dropout rate of ε, yields

$$\frac{|\det(A)|}{Q^{1-\varepsilon}} < 1 \tag{4.3.16}$$

Because $\gamma(A) = |\det(A)|$ and Q is an integer, we obtain the lower bound in equation 4.3.15. $\diamondsuit$

Similarly we can get a corollary corresponding to Theorem 4.3.1 for the quantized system with bounded noise. Its proof is omitted.

Corollary 4.3.1 *Under assumptions 1 - 4, if the quantized feedback system in equation 4.2.5 (with $M > 0$) can be stabilized in the BIBO sense, then the number of quantization levels, Q, satisfies*

$$Q \geq \left\lceil \gamma(A)^{\frac{1}{1-\varepsilon}} \right\rceil \tag{4.3.17}$$

4.4 Stability of noise free quantized systems

This section presents the *dynamic bit assignment policy* (algorithm 4.4.1) and states a theorem (theorem 4.4.1) asserting that the lower bound in Theorem 4.3.1 is achieved by this bit assignment policy. Therefore the minimum bit rate (for asymptotic stability) is achieved under the constant bit rate constraint.

The following algorithm dynamically quantizes the state $x[k]$ for the feedback system in equation 4.2.5 under assumptions 1- 4. The algorithm updates a parallelogram, $P[k]$ containing the state at time k. This parallelogram, $P[k]$, is characterized by, $x^q[k]$, the center of the parallelogram, and $U[k]$, the uncertainy set. The uncertainty set $U[k]$ is formed from a set of vectors $\{v_{i,j}[k] \in R^{n_i}\}$ ($i = 1, \ldots, p$ and $j = 1, \ldots, n_i$) according to equations 4.3.10-4.3.11. The uncertainty set $U^{(I,J)}[k]$ is a modification of $U[k]$ that is formed from the vectors $\left\{v'_{i,j}[k]\right\}$ where $v'_{i,j} = v_{i,j}$ if $(i,j) \neq (I,J)$ and $v'_{i,j} = v_{i,j}/Q$ if $(i,j) = (I,J)$. The basic variables updated by this algorithm are therefore the collection of vectors $\{v_{i,j}[k]\}$ and $x^q[k]$. The quantized signal that is sent between the encoder and decoder at time k is denoted as $s[k]$. This quantized signal is equal to one of Q discrete symbols. The following algorithm consists of two tasks that are executed concurrently, the *encoder* and *decoder* tasks. Each task's first step starts its execution at the same time instant.

Algorithm 4.4.1 Dynamic Bit Assignment:

Encoder/Decoder initialization:

Initialize $x^q[0]$ *and* $\{v_{i,j}[0]\}$ *so that* $x[0] \in x^q[0] + U[0]$ *and set* $k = 0$.

Encoder Task:

1. **Select** *the indices* (I, J) *by*

$$(I, J) = \arg\max_{i,j} \|J_i v_{i,j}[k]\|_2 .$$

2. **Quantize** *the state* $x[k]$ *by setting* $s[k] = s$ *if and only if*

$$x[k] \in x^q[k] + x_s^{(I,J)} + U^{(I,J)}[k]$$

where

$$x_s^{(I,J)} = \begin{bmatrix} 0 & \cdots & 0 & v^T & 0 & \cdots & 0 \end{bmatrix}^T \tag{4.4.18}$$

and $v = \frac{-Q+(2s-1)}{2Q} v_{I,J}[k]$ *for* $s = 1, \ldots, Q$.

3. **Transmit** *the quantized symbol* $s[k]$ *and wait for acknowledgement*

4. **Update** *the variables*

$$\begin{aligned} v_{i,j}[k+1] &= J_i v_{i,j}[k] \\ x^q[k+1] &= (A + BK)x^q[k] \end{aligned}$$

5. **If decoder ack received:**

$$\begin{aligned} v_{I,J}[k+1] &:= \frac{1}{Q} v_{I,J}[k+1] \\ x^q[k+1] &:= x^q[k+1] + A x_{s[k]}^{(I,J)} \end{aligned}$$

where $x_{s[k]}^{(I,J)}$ *is defined in equation 4.4.18.*

6. *Update time,* $k := k + 1$ *and return to step 1.*

Decoder Task:

1. **Update** *the variables*

$$\begin{aligned} v_{i,j}[k+1] &= J_i v_{i,j}[k] \\ x^q[k+1] &= (A+BK)x^q[k] \end{aligned}$$

2. **Wait** *for quantized data, $s[k]$, from encoder.*

3. **If data received:**

$$\begin{aligned} v_{I,J}[k+1] &:= \frac{1}{Q} v_{I,J}[k+1] \\ x^q[k+1] &:= x^q[k+1] + A x^{(I,J)}_{s[k]} \end{aligned}$$

where $x^{(I,J)}_{s[k]}$ is defined in equation 4.4.18. Then send ack back to the encoder.

4. Update time index, $k := k+1$, and return to step 1.

Remark: This algorithm assumes the variables $\{v_{i,j}[k]\}$ and $x^q[k]$ are "synchronized" at the beginning of the kth time interval. Furthermore, we assume the "ack" from decoder to the encoder is reliably transmitted.

Remark: The decision in step 1 of the encoder algorithm is made on the uncertainty set at time $k+1$, rather than k. This was motivated by preliminary studies which showed that using the kth uncertainty set may perform poorly when some of the λ_i are large.

Theorem 4.4.1 *Let $Q = \left\lceil \gamma(A)^{\frac{1}{1-\varepsilon}} \right\rceil$. The feedback system in equation 4.2.5 is asymptotically stable under the quantizer in algorithm 4.4.1. Furthermore for any*

$\Delta\eta > 0$, *there exists a finite* $\lambda_{\Delta\eta} > 0$ *such that*

$$d_{\max}(U[k]) \leq \lambda_{\Delta\eta} \left(\eta + \Delta\eta\right)^{k/N} \tag{4.4.19}$$

where $\eta = \frac{\gamma(A)}{Q^{1-\varepsilon}}$

In order to improve readability, we move the proof of theorem 4.4.1 to the appendix, section 4.8.

Remark: Theorem 4.4.1 characterizes the bit rate R with the number of quantization levels Q, i.e. $Q = 2^R$. In that theorem, we enforce the constant bit rate constraint only through the integer constraint on Q, which may not be enough. For example, $Q = 3$ may require $R = 1.59$bits which can not be realized by a constant bit rate constraint. In fact the constant bit rate constraint is equivalent to the integer constraint on R. With this constraint considered, we obtain the following minimum bit rate

Corollary 4.4.1 *The minimum constant bit rate to guarantee asymptotic stability of noise-free quantized systems is*

$$R = \left\lceil \frac{1}{1-\varepsilon} log_2 \left(\gamma(A)\right) \right\rceil \tag{4.4.20}$$

The proof to the above Corollary directly comes from Theorems 4.3.1 and 4.4.1 by considering $Q = 2^R$ and the integer constraint on R. So it is omitted here.

Remark: We now compare the two sufficient stability conditions in equations

4.1.2 and 4.1.3. For convenience, we rewrite these two conditions as

$$Q = 2^R \geq \prod_{i=1}^{N} 2^{\lceil \log_2(|\lambda_i|) \rceil} \tag{4.4.21}$$

$$Q \geq \lceil \prod_{i=1}^{N} |\lambda_i| \rceil \tag{4.4.22}$$

Considering the ceiling operations above, we know that the bound in equation 4.4.22 is equal to or smaller than that in equation 4.4.21. We use the following example to illustrate this difference more clearly. Let $A = \begin{bmatrix} 1.8 & 0 \\ 0 & 1.1 \end{bmatrix}$. The bound in equation 4.4.21 is $Q \geq 4$. The bound in equation 4.4.22 is $Q \geq 2$. So the latter bound is better (smaller).

We offer an intuitive explanation for this difference. The quantization policy in [61] deals seperately with the two subsystems

$$x_1[k+1] = 1.8x_1[k] + b_1 u[k] \tag{4.4.23}$$

$$x_2[k+1] = 1.1x_2[k] + b_2 u[k] \tag{4.4.24}$$

Every subsystem is unstable and therefore needs at least 2 quantization levels (1 bit). So by equation 4.1.2, we need at least $2 \times 2 = 4$ quantization levels. Although the two subsystems are unstable, however, it can be seen that they are not *too unstable*. If we assign 2 quantization levels to every subsystem, there exists excess stability margin because $\frac{1.8}{2} < 1$ and $\frac{1.1}{2} < 1$. This paper's dynamic bit assignment policy considers the two subsystems as a whole. It should be possible to combine the two stability margins together so that fewer quantization levels are required. This is precisely the case in this example. Figure 4.2 shows the reponse of the quantized system under our dynamic bit assignment policy in which *only 1 bit* is

used to quantize the feedback. The plot clearly shows that this system converges to zero. The "chattering" in this plot arises from the fact that the algorithm quantizes only one component of the state at a time.

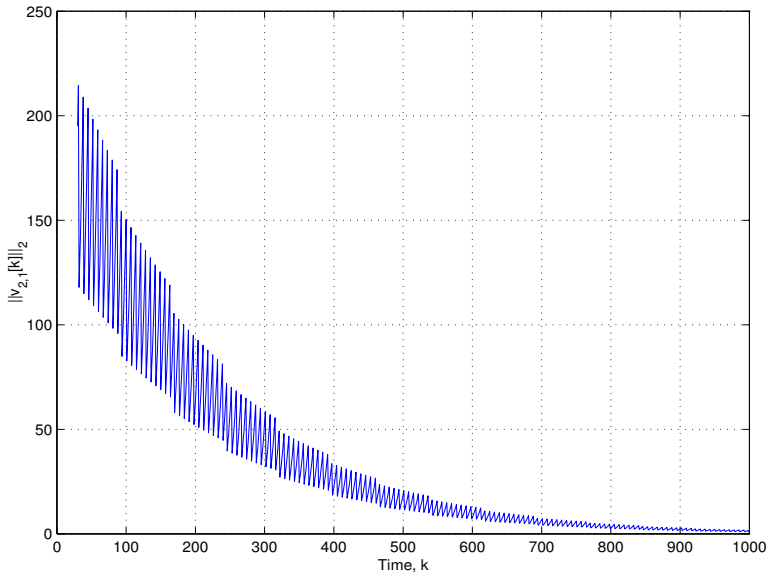


Figure 4.2. Response of quantized system

4.5 Stability of quantized systems with bounded noise

The plant of this section is a quantized system with bounded noise ($M > 0$). The main concern is BIBO stability, i.e. the eventual boundedness of the state. Section 4.4 considers a noise-free quantized system and studies asymptotic stability, i.e. the convergence of the state. The difference between the plants in two sections is the injected bounded noise. For linear systems, such noise

will not affect stability, i.e. the asymptotic stability of the noise-free linear system implies the BIBO stability of the corresponding linear system with bounded noise. The quantized systems are, however, nonlinear due to the nonlinear quantization operation. So it is not trivial to derive the minimum bit rate for BIBO stability, although we know the minimum bit rate for asymptotic stability. In this section, we consider the constant bit rate constraint as argued in section 4.1.

4.5.1 Mathematical preliminaries

In section 4.3, we assume the uncertainty set $U[k]$ in eq. 4.3.9 is a parallelogram. For the quantized system with bounded noise, we assume $U[k]$ to be a special parallelogram, a rectangle. More specifically, $U[k]$ is a rectangle with side lengths $\{L_{i,j}[k]\}_{i=1,\cdots,p;j=1,\cdots,n_i}$.

$$U[k] = \prod_{i=1}^{p} \prod_{j=1}^{n_i} [-L_{i,j}[k], L_{i,j}[k]] \tag{4.5.25}$$

We stack all side lengths into a vector

$$L[k] = \left[L_{1,1}[k], \cdots, L_{1,n_1}[k], \cdots, L_{p,1}[k], \cdots, L_{p,n_p}[k]\right]^T \tag{4.5.26}$$

For notation simplicity, we denote $U[k]$ as

$$U[k] = rect(L[k]) \tag{4.5.27}$$

Correpsonding to the block diagonal structure of $A = diag(J_i)_{i=1}^p$, we define two matrices $H = diag(H_i)_{i=1}^p$ and $K = diag(K_i)_{i=1}^p$ where H_i and K_i have the same dimension as J_i $(i = 1, \cdots, p)$. H_i and K_i are defined as follows.

When λ_i is real,

$$H_i = I,$$
$$K_i = \begin{bmatrix} |\lambda_i| & 1 & 0 & \cdots & 0 \\ 0 & |\lambda_i| & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & |\lambda_i| \end{bmatrix}_{n_i \times n_i}.$$

When $\lambda_i = |\lambda_i| e^{j\theta_i}$ is complex,

$$H_i = diag\left(r(\theta_i)^{-1}, \cdots, r(\theta_i)^{-1}\right),$$
$$K_i = \begin{bmatrix} |\lambda_i| I & E & 0 & \cdots & 0 \\ 0 & |\lambda_i| I & E & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & |\lambda_i| I \end{bmatrix}_{n_i \times n_i}$$

where $r(\theta_i) = \begin{bmatrix} cos(\theta_i) & sin(\theta_i) \\ -sin(\theta_i) & cos(\theta_i) \end{bmatrix}$, $E = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$.

Define the new state variable $z[k] = H^k x[k]$ [61]. The transformed system's dynamics satisfies the following state equations [61].

$$z[k+1] = HAz[k] + H^{k+1}Bu[k] + \overline{w}[k] \tag{4.5.28}$$

where $\overline{w}[k] = H^{k+1}w[k]$. By the boundedness of $w[k]$ and the structure of H, we know $\overline{w}[k]$ is still bounded,

$$\|\overline{w}[k]\|_\infty \leq \overline{M} \tag{4.5.29}$$

where $\overline{M} = 2M$.

The boundedness of $z[k]$ is equivalent to that of $x[k]$, so we will study $z[k]$ instead of $x[k]$. The state $z[k]$ lies in a rectangle $P_z[k]$.

$$z[k] \in P_z[k] = z^q[k] + U_z[k]. \tag{4.5.30}$$

where $z^q[k]$ is the center of $P_z[k]$ and $U_z[k] = rect(L_z[k])$ with $rect(\cdot)$ defined in eq. 4.5.27. The update rules of $z^q[k]$ and $L_z[k]$ will be presented later.

By Lemma 4.3.2 and the fact $\sup_{x[k]\in U[k]} \|x[k] - x^q[k]\|_2 \leq d_{max}(U[k]) \leq 2\sup_{x[k]\in U[k]} \|x[k]-x^q[k]\|_2$, we know that the eventual boundedness of $x[k]-x^q[k]$ is equivalent to that of $x[k]$. By the transformation $z[k] = H^k x[k]$ and $z^q[k] = H^k x^q[k]$ and the boundedness of all elements of H^k and $\left(H^k\right)^{-1}$, we know the boundedness of $x[k] - x^q[k]$ is equivalent to that of $z[k] - z^q[k]$. The bound of $z[k] - z^q[k]$ is measured by $L_z[k]$. Therefore the eventual boundedness of $x[k]$ can be equivalently studied through $L_z[k]$. For notation simplicity, we omit the subscript $_z$ in $P_z[k]$, $U_z[k]$ and $L_z[k]$ from now. Such omission should not result in any confusion since the rest of this section studies only $z[k]$ and $z^q[k]$.

4.5.2 Minimum bit rate for BIBO stability

By Corollary 4.3.1, we get a lower bound on Q for BIBO stability. This section introduces a modified version of the **dynamic bit assignment policy** (DBAP) in section 4.4 and proves it achieves that bound, thereby demonstrating the minimum bit rate for BIBO stability as specified by Corollary 4.3.1 is achievable by DBAP.

DBAP updates a rectangle, $P[k]$ containing the state $z[k]$ at time k. This rectangle, $P[k]$, is characterized by, $z^q[k]$, the center of the rectangle, and $U[k]$, the uncertainty set. The uncertainty set $U[k]$ is formed from its side lengths $L[k]$

according to equation 4.5.27. $L^{I_k,J_k}[k]$ is a modification of $L[k]$ through

$$L_{i,j}^{I_k,J_k}[k] = \begin{cases} L_{i,j}[k], & (i,j) \neq (I_k, J_k) \\ L_{i,j}[k]/Q, & (i,j) = (I_k, J_k) \end{cases} \tag{4.5.31}$$

The basic variables updated by this algorithm are therefore $L[k]$ and $z^q[k]$.

At every time step, the side of $U[k]$ with the longest weighted length $a_{i,j}^2 L_{i,j}[k]$ $(i = 1, \cdots, p; j = 1, \cdots, n_i)$ is partitioned into Q equal parts, where $a_{i,j}$ is defined in equation 4.5.32. Thus $U[k]$ is partitioned into Q smaller rectangles. The index of the smaller rectangle which $z[k]$ lies within is sent from the encoder to the decoder at time k and is denoted as $s[k]$. The following algorithm describes two tasks that are executed concurrently, the *encoder* and *decoder* algorithms. Each task's first step starts its execution at the same time instant. Before presenting the quantization algorithm, we define the weighting parameters $a_{i,j}$ $(i = 1, \cdots, P; j = 1, \cdots, n_i)$, together with 3 other parameters ρ_0, ε_1 and ε_0.

$$a_{i,j} = \begin{cases} \rho_0^{-n_i+j}, & \text{real } \lambda_i; \\ \rho_0^{-\frac{n_i}{2}+\frac{j}{2}}, & \text{complex } \lambda_i, \text{ even } j \\ \rho_0^{-\frac{n_i}{2}+\frac{j+1}{2}}, & \text{complex } \lambda_i, \text{ odd } j \end{cases} \tag{4.5.32}$$

$$\begin{cases} \rho_0 > \max_{i=1}^{p} \left(\frac{3Q}{\varepsilon_0}, (|\lambda_i| + \varepsilon_0)Q \right) \\ Q^{1-\varepsilon-\varepsilon_1} > (1 + Q\varepsilon_0)^N \gamma(A) \\ \varepsilon_1 > 0, \varepsilon_0 > 0 \end{cases} \tag{4.5.33}$$

Algorithm 4.5.1 Dynamic Bit Assignment:

Encoder/Decoder initialization:

Initialize $z^q[0]$ *and* $L[0]$ *so that* $z[0] \in z^q[0] + U[0]$ *and set* $k = 0$.

Encoder Algorithm:

1. **Select** *the indices* (I_k, J_k) *by*

$$(I_k, J_k) = \arg\max_{i,j} \left(a_{i,j}^2 L_{i,j}[k]\right) \tag{4.5.34}$$

2. **Quantize** *the state* $z[k]$ *by setting* $s[k] = s$ *if and only if*

$$z[k] \in z^q[k] + z_s^{(I_k,J_k)} + rect(L^{(I_k,J_k)}[k])$$

where

$$z_s^{(I_k,J_k)} = \begin{bmatrix} 0 & \cdots & 0 & l & 0 & \cdots & 0 \end{bmatrix}^T \tag{4.5.35}$$

and $l = \frac{-Q+(2s-1)}{2Q} L_{I_k,J_k}[k]$ *for* $s = 1, \ldots, Q$.

3. **Transmit** *the quantized symbol* $s[k]$ *and wait for acknowledgement*

4. **Update** *the variables*

$$\begin{aligned} L[k+1] &= KL[k] + [\overline{M}, \cdots, \overline{M}]^T \\ z^q[k+1] &= HAz^q[k] + H^{k+1}BGH^{-k}z^q[k] \end{aligned}$$

5. **If decoder ack received:**

$$\begin{aligned} L[k+1] &= KL^{I_k,J_k}[k] + [\overline{M}, \cdots, \overline{M}]^T \\ z^q[k+1] &= z^q[k+1] + HAz_{s[k]}^{(I_k,J_k)} \end{aligned}$$

where $z_{s[k]}^{(I_k,J_k)}$ *is defined in equation 4.5.35.*

6. *Update time,* $k = k + 1$ *and return to step 1.*

Decoder Algorithm:

1. **Update** *the variables*

$$\begin{aligned} L[k+1] &= KL[k] + [\overline{M}, \cdots, \overline{M}]^T \\ z^q[k+1] &= HAz^q[k] + H^{k+1}BGH^{-k}z^q[k] \end{aligned}$$

2. **Wait** *for quantized data,* $s[k]$*, from encoder.*

3. **If data** $s[k]$ **received:**

$$\begin{aligned} L[k+1] &= KL^{I_k,J_k}[k] + [\overline{M}, \cdots, \overline{M}]^T \\ z^q[k+1] &= z^q[k+1] + HAz_{s[k]}^{(I_k,J_k)} \end{aligned}$$

where (I_k, J_k) *is determined similarly to equation 4.5.34 and* $z_{s[k]}^{(I_k,J_k)}$ *is defined in equation 4.5.35. Then send ack back to the encoder.*

4. *Update time index,* $k = k+1$*, and return to step 1.*

Remark: In algorithm 4.5.1, the side is measured by the weighted length $a_{i,j}^2 L_{i,j}[k]$ rather than the direct length $L_{i,j}[k]$. We demonstrate its motivation through an example. Let $A = \begin{bmatrix} \lambda & 1 \\ 0 & \lambda \end{bmatrix}$, where $\lambda > 1$. Suppose we make the partition decision based on $L_{1,j}[k]$ $(j = 1, 2)$. When $L_{1,1}[k] > L_{1,2}[k]$, we can follow the procedure in algorithm 4.5.1 to obtain

$$\begin{cases} L_{1,1}[k+1] &= \frac{\lambda}{Q} L_{1,1}[k] + L_{1,2}[k] + M \\ L_{1,2}[k+1] &= \lambda L_{1,2}[k] + M \end{cases} \tag{4.5.36}$$

We will encounter a problem when $L_{1,1}[k]$ is close to $L_{1,2}[k]$, i.e. $L_{1,1}[k] \approx L_{1,2}[k]$. Under that condition, equation 4.5.36 can be written as

$$\begin{cases} L_{1,1}[k+1] & \approx & (1+\frac{\lambda}{Q})L_{1,1}[k] + M \\ L_{1,2}[k+1] & = & \lambda L_{1,2}[k] + M \end{cases} \tag{4.5.37}$$

It can be seen that both $L_{1,1}[k]$ and $L_{1,2}[k]$ have been increased. So the uncertainty of $U[k]$ has been increased, which is not what we expect. If we keep the above partition decision based on $L_{1,j}[k]$, then $L_{1,j}[k]$ $(j = 1, 2)$ will eventually diverge. The reason for such divergence is the coupling between $L_{1,1}[k]$ and $L_{1,2}[k]$. When $L_{1,1}[k]$ and $L_{1,2}[k]$ are comparable, partitioning $L_{1,1}[k]$ may decrease neigher $L_{1,1}[k]$ nor $L_{1,2}[k]$, as shown above. But when $L_{1,1}[k]$ is much larger than $L_{1,2}[k]$, partitioning $L_{1,1}[k]$ does decrease it. This fact inspired us to use the weighted lengths $a_{1,j}^2 L_{1,j}[k]$, where $a_{1,1} = \rho_0^{-1} << 1$ and $a_{1,2} = 1$. If $a_{1,1}^2 L_{1,1}[k] > a_{1,2}^2 L_{1,2}[k]$, it is guaranteed that

$$L_{1,1}[k] >> L_{1,2}[k] \tag{4.5.38}$$

So $L_{1,1}[k+1] < L_{1,1}[k]$ (when $L_{1,1}[k]$ is large compared to M), i.e. the uncertainty is decreased.

The above discussions are applicable for general J_i. Note that $a_{i,n_i} = 1$ $(i = 1, \cdots, P)$. If $n_i = 1$, the weighted length rule reduces into the direct length rule.

Remark: Now we compare the quantization policy in this section and the quantization policy in section 4.4. The obvious difference between them is that noise-free systems are considered in section 4.4 and systems with bounded noise are studied here. Because of the exogenous noise, the method in section 4.4 is not applicable here. In section 4.4, the uncertainty set $U[k]$ takes the form of a super-

parallelogram. It deserves special attention that $U[k]$ is the tightest uncertainty set in the sense that every point in $U[k]$ can be reached by some initial condition. In this section, the uncertainty set $U[k]$ is assumed to be rectangular. After one step, $U[k]$ evolves into a set $U_1[k+1]$. Due to the effect of noise, $U_1[k+1]$ is neither a rectangle nor a parallelogram for a non-diagonalizable system matrix. We therefore overbound $U_1[k+1]$ with a larger rectangle $U[k+1]$. So the uncertainty set in this paper is not as tight as that in section 4.4. One may suspect that the quantization policy in this paper is more conservative and would need more quantization levels, i.e. larger Q than that in section 4.4. But suprisingly, this does not appear to be the case. Theorem 4.5.1 states that the system in this paper can be stabilized with the same smallest Q as that in section 4.4.

Theorem 4.5.1 *Let* $Q = \left\lceil \gamma(A)^{\frac{1}{1-\varepsilon}} \right\rceil$. *The quantized linear system in equation 4.2.5 is BIBO stable under the dynamic quantizer in algorithm 4.5.1, i.e. the state* $x[k]$ *is eventually bounded for bounded noise* $w[k]$.

The proof is found in the appendix, section 4.8.

Theorem 4.5.1 characterizes the bit rate R with the number of quantization levels Q, i.e. $Q = 2^R$. Similarly to Corollary 4.4.1, we may consider the integer contraint on R and obtain following minimum bit rate result.

Corollary 4.5.1 *The minimum constant bit rate to guarantee BIBO stability for quantized systems with bounded noise is*

$$R = \left\lceil \frac{1}{1-\varepsilon} log_2 \left(\gamma(A)\right) \right\rceil \tag{4.5.39}$$

The proof to the above Corollary directly comes from the ones of Corollary 4.3.1 and Theorem 4.5.1 by considering $Q = 2^R$ and the integer constraint on R. So it

is omitted here.

4.6 Minimum constant bit rate for stability of quantized systems under output feedback

In sections 4.4 and 4.5, we consider the minimum *constant* bit rate for stability of quantized systems *under state feedback*. If what we have is only output rather than state, *what is the minimum bit rate?* In [61] and [54], the question is answered under the time-varying bit rate configuration. We give an answer to the question under the constant bit rate constraint. The concerned plant is

$$\begin{cases} x[k+1] &= Ax[k] + Bu[k] + w[k] \\ y[k] &= Cx[k] \end{cases} \tag{4.6.40}$$

where (A, B) is controllable, (C, A) is observable and $w[k]$ is bounded. The output $y[k]$ is quantized into R bits where R is an integer, i.e. the constant bit rate constraint is considered. How large should R be in order to guarantee BIBO stability? Because state is the full description of the system, the minimum bit rate under state feedback is a lower bound for output feedback, which is presented in the following corollary.

Corollary 4.6.1 *The minimum bit rate to guarantee BIBO stability under output feedback must satisfy*

$$R \geq \left\lceil \frac{1}{1-\varepsilon} log_2 \left(\gamma(A)\right) \right\rceil \tag{4.6.41}$$

A lower bound on stabilizable R is provided in eq. 4.6.41. We will construct a quantization policy which guarantees the equality in eq. 4.6.41 holds, i.e. the

minimum bit rate is achieved.

Because the system in eq. 4.6.40 is observable, we can construct a Luenberger observer.

$$\hat{x}[k+1] = A\hat{x}[k] + Bu[k] + L(C\hat{x}[k] - y[k]) \tag{4.6.42}$$

where L is chosen so that $A - LC$ is stable. Denote the observation error as $\hat{e}[k] = x[k] - \hat{x}[k]$. $\hat{e}[k]$ is governed by

$$\hat{e}[k+1] = (A - LC)\hat{e}[k] + w[k] \tag{4.6.43}$$

Because $(A - LC)$ is stable, $\{\hat{e}[k]\}$ is bounded

$$\|\hat{e}[k]\|_\infty \leq E < \infty, \forall k \geq 0 \tag{4.6.44}$$

Let $\overline{w}[k] = L\hat{e}[k]$. By the boundedness of $\{\hat{e}[k]\}$, we know $\{\overline{w}[k]\}$ is also bounded

$$\|\overline{w}[k]\|_\infty \leq \overline{M}, \forall k \geq 0 \tag{4.6.45}$$

The observer in eq. 4.6.42 is man-made and its state is available for the encoder. We may virtually consider eq. 4.6.42 as our quantized system, i.e.

$$\begin{cases} \hat{x}[k+1] &= A\hat{x}[k] + Bu[k] + \overline{w}[k] \\ u[k] &= F\hat{x}^q[k] \end{cases} \tag{4.6.46}$$

where $\hat{x}^q[k]$ is the quantized version of $\hat{x}[k]$ and F is a stabilizing state feedback gain, i.e. $A + BF$ is stable. Eq. 4.6.46 is exactly the model considered in section 4.5. By theorem 4.5.1 (Corollary 4.5.1), we know when $R = \left\lceil \frac{1}{1-\varepsilon} log_2 \left(\gamma(A)\right) \right\rceil$,

the eventual boundedness of $\hat{x}[k]$ can be guaranteed by algorithm 4.5.1. Because $x[k] = \hat{x}[k] + \hat{e}[k]$, we get

$$\|x[k]\|_\infty \leq \|\hat{x}[k]\|_\infty + \|\hat{e}[k]\|_\infty \tag{4.6.47}$$

Consider the bound on $\hat{e}[k]$ in eq. 4.6.44 and the eventual boundedness of $\hat{x}[k]$, we know $x[k]$ is eventually bounded. Therefore we get the following statement.

Proposition 4.6.1 *The minimum constant bit rate to guarantee BIBO stability of quantized systems under output feedback is*

$$R = \left\lceil \frac{1}{1-\varepsilon} log_2 \left(\gamma(A)\right) \right\rceil \tag{4.6.48}$$

Remark: For noise-free quantized system under output feedback, we may achieve the similar minimum constant bit rate as proposition 4.6.1.

4.7 Conclusions

This chapter considers the constant bit rate constraint in quantization and presents the minimum number of quantization levels for asymptotic stability (for a noise-free quantized system) and the minimum number of quantization levels for BIBO stability (for a quantized system with bounded noise) under such constraint. Its major contributions lie in the proposed dynamic quantization policy, DBAP which can achieved the minimum number of quantization levels. Such results are extended to achieve the minimum constant bit rate for stability with the integer constraint enforced on the bit rate. The minimum constant bit rate result is also extended to the quantized system under output feedback.

4.8 Appendix: proofs

4.8.1 Proof to Theorem 4.4.1

The following lemma follows from basic algebra, so its proof is omitted.

Lemma 4.8.1 *Let J_i be as defined in assumption 1. For any non-zero $v_i \in R^{n_i}$,*

$$\lim_{k \longrightarrow \infty} \frac{\|J_i^{k+1} v_i\|_2}{\|J_i^k v_i\|_2} \quad = \quad |\lambda_i|$$

By algorithm 4.4.1, we know $v_{i,j}[k]$ is a scaled version of $J_i^k v_{i,j}[0]$. Therefore lemma 4.8.1 guarantees that for any $\varepsilon_0 > 0$, there exists K_1 such that

$$(1 - \varepsilon_0)|\lambda_i| \leq \frac{\|J_i v_{i,j}[k]\|_2}{\|v_{i,j}[k]\|_2} \leq (1 + \varepsilon_0)|\lambda_i|, \tag{4.8.49}$$

for $k \geq K_1$ and any i and j.

Define the average dropout rate as

$$\overline{\varepsilon}_{l,k} = \frac{1}{l} \sum_{i=0}^{l-1} d[k+i] \tag{4.8.50}$$

Since $\overline{\varepsilon}_{l,k} \longrightarrow \varepsilon$ as $l \longrightarrow \infty$, we know that for any $\delta_0 > 0$, there exists $M > 0$ such that

$$\varepsilon - \delta_0 \leq \overline{\varepsilon}_{l,k} \leq \varepsilon + \delta_0, \tag{4.8.51}$$

for all $l \geq M$ and all k.

We prove that the uncertainty set $U[k]$ converges to zero by first showing that the "volume" of this set (as defined by the product of side lengths $p[k] = \prod_{i=1}^{p} \prod_{j=1}^{n_i} \|v_{i,j}[k]\|_2$) converges exponentially to zero.

Lemma 4.8.2 *Assume* $Q \geq \lceil \gamma(A)^{\frac{1}{1-\varepsilon}} \rceil$ *and let* $\eta = \frac{\gamma(A)}{Q^{1-\varepsilon}}$. *For any* $\Delta\eta > 0$, *there exist constants* $p_{\Delta\eta}$ *and* K_3 *such that for all* $k \geq K_3$

$$p[k] \leq p_{\Delta\eta}(\eta + \Delta\eta)^k \tag{4.8.52}$$

Proof: For any small numbers ε_0, δ_0, there exists K_3 and M such that equation 4.8.49 and 4.8.51 hold. So we limit our attention to $k \geq K_3$ and $l \geq M$. From time k to $k + l - 1$, there are $(1 - \overline{\varepsilon}_{l,k})l$ sucessfully quantized measurements. So

$$p[k+l] = \frac{1}{Q^{(1-\overline{\varepsilon}_{l,k})l}} \prod_{i=1}^{p} \prod_{j=1}^{n_i} \|J_i^l v_{i,j}[k]\|_2.$$

Equations 4.8.49 and 4.8.51 let us bound $p[k+l]$ as

$$p[k+l] \leq \left(\frac{\gamma(A)}{Q^{1-\varepsilon-\delta_0}} (1+\varepsilon_0)^N \right)^l p[k]$$

Note that if $Q \geq \lceil \gamma(A)^{1/(1-\varepsilon)} \rceil$, then $\eta < 1$. Choose K_3 and M large enough to make ε_0 and δ_0 arbitrarily small. We can couple this choice with the fact that $\eta < 1$ to infer that $\frac{\gamma(A)}{Q^{1-\varepsilon-\delta_0}}(1+\varepsilon_0)^N < \min(1, \eta + \Delta\eta)$. If we let

$$p_{\Delta\eta} = (\max(p[K_3], \cdots, p[K_3 + M - 1])) \left(\frac{\gamma(A)}{Q^{1-\varepsilon-\delta_0}} (1+\varepsilon_0)^N \right)^{-M-K_3},$$

then $p[k] \leq p_{\Delta\eta} (\eta + \Delta\eta)^k$ for $k \geq K_3$. $\diamondsuit$

For the preceding lemma to imply that $U[k]$ goes to zero, we must establish that each side of the parallelgram gets quantized an infinite number of times. In particular let $T_{i,j}$ denote the time instants when side $v_{i,j}$ was successfully quan-

tized. In other words,

$$T_{i,j} = \{k \,:\, I_k = i, J_k = j, d[k] = 0\}$$

Define $\mathcal{T}_\infty = \{(i,j) \,:\, \text{card}(T_{i,j}) = \infty\}$ where $\text{card}(I)$ is the cardinality of set I. The following lemma shows that $\text{card}(T_{i,j}) = \infty$,

Lemma 4.8.3 *If* $v_{i,j}[0] \neq 0$, *then* $\text{card}(T_{i,j}) = \infty$

Proof: This lemma is proven by contradiction. Suppose $v_{I,J}[0] \neq 0$ but $\text{card}(T_{I,J}) < \infty$, then there exists a large number K_u such that $K_u \geq K_1$ and such that the side $v_{I,J}$ is never quanitzed after time K_u. The update rule for $v_{I,J}$ in our algorithm requires $v_{I,J}[k+1] = J_I v_{I,J}[k]$ for all $k \geq K_u + 1$. Applying lemma 4.8.1 to this equation yields $\|J_I v_{I,J}[k]\|_2 \geq c_0((1-\epsilon_0)|\lambda_I|)^{k-K_u}$ for all $k \geq K_u$ where $c_0 = \|J_I v_{I,J}[K_u]\|_2$. By choosing ϵ_0 small enough, we can guarantee $(1-\epsilon_0)|\lambda_i| > 1$ for all $i = 1, \ldots, p$, which implies that $\|J_I v_{I,J}[k]\|$ is bounded below by a monotone increasing function of k.

Now consider any other side $v_{i,j}$ where $(i,j) \neq (I,J)$ and $\text{card}(T_{i,j}) = \infty$. Define $K_{i,j} = \min\{k \,|\, k \in T_{i,j}, k \geq K_u\}$. In other words, $K_{i,j}$ is the first time instant after K_u when side $v_{i,j}$ is quantized again. From our algorithm, we know that

$$\|v_{i,j}[K_{i,j}+1]\|_2 = \frac{1}{Q}\|J_i v_{i,j}[K_{i,j}]\|_2 \geq \frac{1}{Q}\|J_I v_{I,J}[K_u]\|_2 = c_Q \tag{4.8.53}$$

where $c_Q = c_0/Q \neq 0$. For $k \geq K_{i,j}+1$, if $v_{i,j}[k]$ is not successfully quantized, then

$$\|v_{i,j}[k+1]\|_2 = \|J_i v_{i,j}[k]\|_2 \geq (1-\epsilon_0)|\lambda_i|\|v_{i,j}[k]\|_2 \tag{4.8.54}$$

If $v_{i,j}[k]$ is successfully quantized then

$$\|v_{i,j}[k+1]\|_2 = \frac{1}{Q}\|J_i v_{i,j}[k]\|_2 \geq \frac{1}{Q}\|J_I v_{I,J}[K_u]\|_2 = c_Q \tag{4.8.55}$$

Combining equations 4.8.53, 4.8.54, and 4.8.55, in addition to $(1-\epsilon_0)|\lambda_i| > 1$, guarantees $\|v_{i,j}[k]\|_2 \geq c_Q$ for all $k \geq K_{i,j}+1$. Now define the product of part of the side lengths as $p'[k] = \prod_{(i,j)\in\mathcal{T}_\infty} \|v_{i,j}[k]\|_2$ and let $\overline{K} = \max_{(i,j)\in\mathcal{T}_\infty} K_{i,j}+1$ By equation 4.8.55 we know that for $k \geq \overline{K}$

$$p'[k] \geq c_Q^{N'} \tag{4.8.56}$$

where $N' = \text{card}(\mathcal{T}_\infty)$. Equation 4.8.56 is an eventual lower bound on $p'[k]$.

We may repeat the procedure used in lemma 4.8.2 to obtain an upper bound on $p'[k]$ of the form

$$p'[k] \leq p'_{\delta\eta'}(\eta' + \Delta\eta')^k \tag{4.8.57}$$

where $\Delta\eta' > 0$ is any chosen tolerance, $p'_{\Delta\eta'}$ is a constant, and $\eta' = \frac{1}{Q}\prod_{(i,j)\in\mathcal{T}_\infty}|\lambda_i| < \frac{\gamma(A)}{Q^{1-\epsilon}} < 1$. We choose $\Delta\eta'$ small enough so that $\eta' + \Delta\eta' < 1$. Thus $\lim_{k\to\infty} p'[k] = 0$, which contradicts the eventual lower bound in equation 4.8.56. $\diamondsuit$

This note assumes that $v_{i,j}[0] \neq 0$ for all i,j. So lemma 4.8.3 guarantees $\text{card}(T_{i,j}) = \infty$ for all i,j. Thus there must exist $K_2 > K_1$ such that

$$\text{card}\left(T_{i,j} \cap [K_1, K_2]\right) \geq M \tag{4.8.58}$$

for all i,j, where $[K_1,K_2]$ is the set of integers from K_1 to K_2. In the following discussion, we assume $k \geq K_2$ and we let $(i_0,j_0) = \arg\min_{i,j} \|J_i v_{i,j}[k]\|_2$. We

define

$$\overline{l}(i_0, j_0, M, k) = \min \{m : \text{card}\,([k-m, k-1] \cap T_{i_0,j_0}) = M\} \tag{4.8.59}$$

where $\overline{l}(i_0, j_0, M, k)$ is the shortest length of time prior to time instant k in which the side v_{i_0,j_0} was quantized exactly M times.

The following lemma establishes the "fairness" of the algorithm by showing that $\overline{l}(i_0, j_0, M, k)$ is uniformly bounded with respect to i_0, j_0, and k.

Lemma 4.8.4 *There exists a constant l_M such that for $k \geq K_2$*

$$\overline{l}(i_0, j_0, M, k) \leq l_M. \tag{4.8.60}$$

Proof: Throughout this proof, we denote $\overline{l}(i_0, j_0, M, k)$ as $\overline{l}$. Let's first consider $(i, j) \neq (i_0, j_0)$. Let $l_{i,j}$ denote the number of times side $v_{i,j}$ was successfully quantized in the interval $[k-\overline{l}, k-1]$. Then the update equations in our algorithm imply that

$$J_i v_{i,j}[k] = \frac{1}{Q^{l_{i,j}}} J_i^{\overline{l}}(J_i v_{i,j}[k-\overline{l}]) \tag{4.8.61}$$

By inequality 4.8.49, we obtain

$$\|J_i v_{i,j}[k]\|_2 \leq \frac{((1+\varepsilon_0)|\lambda_i|)^{\overline{l}}}{Q^{l_{i,j}}} \left\|J_i v_{i,j}[k-\overline{l}]\right\|_2 \tag{4.8.62}$$

When $(i, j) = (i_0, j_0)$, we know that side v_{i_0,j_0} was updated exactly M times during $[k-\overline{l}, k-1]$. So the algorithm's update equations imply that

$$J_{i_0} v_{i_0,j_0}[k] = \frac{1}{Q^M} J_{i_0}^{\overline{l}} \left(J_{i_0} v_{i_0,j_0}[k-\overline{l}]\right) \tag{4.8.63}$$

Using inequality 4.8.49 in equation 4.8.63 yields

$$\|J_{i_0} v_{i_0,j_0}[k]\|_2 \;\geq\; \frac{((1-\varepsilon_0)|\lambda_{i_0}|)^{\overline{l}}}{Q^M} \|J_{i_0} v_{i_0,j_0}[k-\overline{l}]\|_2 \tag{4.8.64}$$

From the definitions of (i_0, j_0) and $\overline{l}$, we also know that

$$\left\|J_{i_0} v_{i_0,j_0}[k-\overline{l}]\right\|_2 \;\geq\; \left\|J_i v_{i,j}[k-\overline{l}]\right\|_2 \tag{4.8.65}$$

$$\|J_{i_0} v_{i_0,j_0}[k]\|_2 \;\leq\; \|J_i v_{i,j}[k]\|_2 \tag{4.8.66}$$

Inserting equations 4.8.62, 4.8.64 and 4.8.65 into equation 4.8.66, yields,

$$((1-\varepsilon_0)|\lambda_{i_0}|)^{\overline{l}} \frac{1}{Q^M} \leq ((1+\varepsilon_0)|\lambda_i|)^{\overline{l}} \frac{1}{Q^{l_{i,j}}} \tag{4.8.67}$$

There are at most $\overline{l}(\varepsilon + \delta_0)$ dropouts during $[k-\overline{l}, k-1]$. So $l_{i,j}$ satisfies the inequality,

$$\sum_{(i,j)\neq(i_0,j_0)} l_{i,j} \;\geq\; \overline{l} - \overline{l}(\varepsilon+\delta_0) - M = (1-\varepsilon-\delta_0)\overline{l} - M \tag{4.8.68}$$

Multiply inequality 4.8.67 over all (i, j) not equal to (i_0, j_0) and use equation 4.8.68 to obtain

$$((1-\varepsilon_0)|\lambda_{i_0}|)^{\overline{l}(N-1)} \frac{1}{Q^{(N-1)M}} \leq \left((1+\varepsilon_0)^{N-1} \prod_{(i,j)\neq(i_0,j_0)} |\lambda_i|\right)^{\overline{l}} \frac{1}{Q^{\overline{l}(1-\varepsilon-\delta_0)-M}}$$

The above inequality may be solved with respect to $\overline{l}$ to show that $\overline{l} \leq l_{i_0}$ where

$$l_{i_0} = \frac{MN\ln(Q)}{(N-1)\ln\left(\frac{1-\varepsilon_0}{1+\varepsilon_0}\right) + N\ln(|\lambda_{i_0}|) + \ln\left(\frac{Q^{1-\varepsilon}}{\gamma(A)}\right) - \delta_0\ln(Q)}$$

Letting $l_M = \max_{i_0} l_{i_0}$ gives the desired bound. $\diamondsuit$

The following lemma establishes that the sides are *balanced* in the sense that the ratio $\|v_{i_1,j_1}[k]\|_2 / \|v_{i_2,j_2}[k]\|_2$ is uniformly bounded for all i_1, j_1, i_2, j_2, and $k \geq K_2$.

Lemma 4.8.5 *For $k \geq K_2$ and all i_1, j_1, i_2, and j_2, there exists a finite constant r such that*

$$\frac{\|v_{i_1,j_1}[k]\|_2}{\|v_{i_2,j_2}[k]\|_2} \leq r. \tag{4.8.69}$$

Proof: For any i_1, i_2, j_1, and j_2, equation 4.8.49 implies that

$$\frac{\|v_{i_1,j_1}[k]\|_2}{\|v_{i_2,j_2}[k]\|_2} \leq \alpha \frac{\|J_{i_1} v_{i_1,j_1}[k]\|_2}{\|J_{i_2} v_{i_2,j_2}[k]\|_2}, \tag{4.8.70}$$

where $\alpha = \frac{1+\varepsilon_0}{1-\varepsilon_0} \max_{i_1,i_2} \frac{|\lambda_{i_1}|}{|\lambda_{i_2}|}$.

Following the arguments used in the preceding lemma, we know that

$$\begin{aligned} \frac{\|J_i v_{i,j}[k]\|_2}{\|J_{i_0} v_{i_0,j_0}[k]\|_2} &\leq \frac{\frac{|\lambda_i|^{\overline{l}}}{Q^{l_{i,j}}}(1+\varepsilon_0)^{\overline{l}} \|J_i v_{i,j}[k-\overline{l}]\|_2}{\frac{|\lambda_{i_0}|^{\overline{l}}}{Q^M}(1-\varepsilon_0)^{\overline{l}} \|J_{i_0} v_{i_0,j_0}[k-\overline{l}]\|_2} \\ &\leq Q^M \left(\frac{|\lambda_i|}{|\lambda_{i_0}|}\right)^{\overline{l}} \left(\frac{1+\varepsilon_0}{1-\varepsilon_0}\right)^{\overline{l}} \leq r_0 \end{aligned}$$

where

$$r_0 = Q^M \left(\max_{i_1,i_2} \frac{|\lambda_{i_1}|}{|\lambda_{i_2}|}\right)^{l_M} \left(\frac{1+\varepsilon_0}{1-\varepsilon_0}\right)^{l_M},$$

and l_M is the bound in lemma 4.8.4.

At time k we know $\|J_{i_0} v_{i_0,j_0}[k]\|_2$ is the smallest among $\|J_i v_{i,j}[k]\|_2$, so

$$\frac{\|J_{i_1} v_{i_1,j_1}[k]\|_2}{\|J_{i_2} v_{i_2,j_2}[k]\|_2} \leq r_0, \tag{4.8.71}$$

for all i_1, i_2, j_1, and j_2. Let $r = r_0 \alpha$ to obtain the desired bound. $\diamondsuit$

Proof of theorem 4.4.1: This theorem follows from the direct application of lemmas 4.8.5 and 4.8.2. Let $K_0 = \max(K_2, K_3)$. At the beginning, we will limit our attention to $k \geq K_0$ so that lemmas 4.8.5 and 4.8.2 are true. Lemma 4.8.5 shows that $\frac{\|v_{i_1,j_1}[k]\|_2}{\|v_{i_2,j_2}[k]\|_2} \leq r$, for all i_1, j_1, i_2, and j_2. Choose v_{i_1,j_1} to be the longest side, to obtain $\frac{\max_{m,n} \|v_{m,n}[k]\|_2}{\|v_{i,j}[k]\|_2} \leq r$ which we may rewrite as

$$\|v_{i,j}[k]\|_2 \geq \frac{1}{r} \max_{m,n} \|v_{m,n}[k]\|_2 \tag{4.8.72}$$

The above relationship, the definition of $p[k]$, and lemma 4.8.2 yield

$$\max_{m,n} \|v_{m,n}[k]\|_2 \leq r \sqrt[N]{p_{\Delta\eta}} (\eta + \Delta\eta)^{\frac{k}{N}} \tag{4.8.73}$$

$U[k]$ is a parallelogram with sides $v_{i,j}[k]$. The triangle inequality implies

$$d_{\max}(U[k]) \leq \sum_{i=1}^{P} \sum_{j=1}^{n_i} \|v_{i,j}[k]\|_2 \leq N \max_{m,n} \|v_{m,n}[k]\|_2$$

Substituting equation 4.8.73 into the above bound on $d_{\max}(U[k])$ yields $d_{\max}(U[k]) \leq \lambda_0 (\eta + \Delta\eta)^{\frac{k}{N}}$ where $\lambda_0 = Nr \sqrt[N]{p_{\Delta\eta}}$. By choosing

$$\lambda_{\Delta\eta} = \max \left(\max_{m \in [1, K_0 - 1]} \left(d_{max} \left(U[m] \right) (\eta + \Delta\eta)^{-\frac{m}{N}} \right), \lambda_0 \right)$$

we can guarantee that equation 4.4.19 holds for all k. $\diamondsuit$

4.8.2 Proof to Theorem 4.4.1

The proof to theorem 4.5.1 is broken into several intermediate lemmas. Before the proof, we define two functions, $r_{i,j}[k]$ $(i = 1, \cdots, P; j = 1, \cdots, n_i)$ and $p[k]$. $r_{i,j}[k]$ is a modified version of the side length $L_{i,j}[k]$. $p[k]$ is the product of $r_{i,j}[k]$

$$p[k] = \prod_{i=1}^{P} \prod_{j=1}^{n_i} r_{i,j}[k]$$

When λ_i is real,

$$\begin{aligned} r_{i,n_i}[k] &= \max(L_{i,n_i}[k], \rho_0 \overline{M}) \\ r_{i,j}[k] &= \max(a_{i,j} L_{i,j}[k], r_{i,j+1}[k]), \\ &\quad \text{for } j = 1, 2, \cdots, n_i - 1 \end{aligned}$$

When λ_i is complex,

$$\begin{aligned} r_{i,j}[k] &= \max(L_{i,j}[k], \rho_0 \overline{M}), \\ &\quad \text{for } j = n_i - 1 \text{ and } n_i \\ r_{i,j}[k] &= \max(a_{i,j} L_{i,j}[k], r_{i,j+2}[k], r_{i,j+3}[k]), \\ &\quad \text{for } j = 1, 3, \cdots, n_i - 3 \\ r_{i,j}[k] &= max(a_{i,j} L_{i,j}[k], r_{i,j+1}[k], r_{i,j+2}[k]), \\ &\quad \text{for } j = 2, 4, \cdots, n_i - 2 \end{aligned}$$

The relationship between the growth-rate of $r_{i,j}[k]$ and the one of $p[k]$ is the key point to the proof of theorem 4.5.1. First we establish a general upper bound on the growth rate of $r_{i,j}[k]$.

Lemma 4.8.6

$$\frac{r_{i,j}[k+1]}{r_{i,j}[k]} \leq |\lambda_i| + \varepsilon_0, \forall i, j \tag{4.8.74}$$

where ε_0 is defined in equation 4.5.33.

Sketch of Proof: This lemma is proven by mathematical induction. There are two cases to consider, when λ_i is complex and when λ_i is real. If λ_i is real we first show that equation 4.8.74 holds for $j = n_i$. We then show that if equation 4.8.74 holds for $j = j_0$, it must also hold for $j_0 - 1$, thereby completing the induction. A similar inductive argument can be made when λ_i is complex. $\diamondsuit$

The bound in lemma 4.8.6 is quite loose. When $p[k]$ is larger than

$$p_1 = \prod_{i=1}^{P} \left(\frac{\rho_0 \overline{M}}{a_{i,1}} \right)^{n_i} \tag{4.8.75}$$

, we can place a tighter bound on the growth rate of $r_{I_k,J_k}[k]$.

Lemma 4.8.7 *If $p[k] > p_1$ (see eq. 4.8.75) and $d[k] = 0$ (there is no dropout at time k), then*

$$\frac{r_{I_k,J_k}[k+1]}{r_{I_k,J_k}[k]} \leq \frac{|\lambda_{I_k}|}{Q} + \varepsilon_0 \tag{4.8.76}$$

where ε_0 is defined in equation 4.5.33.

Sketch of proof: By the quantization policy, we know

$$a_{I_k,J_k}^2 L_{I_k,J_k} \geq a_{i,j}^2 L_{i,j}, \forall i, j \tag{4.8.77}$$

First prove that

$$a_{I_k,J_k}^2 L_{I_k,J_k} > \rho_0 \overline{M} \tag{4.8.78}$$

It can be achieved through contradiction method. If eq. 4.8.78 is violated, $a_{i,j}^2 L_{i,j} \leq \rho_0 \overline{M}$ for $\forall i, j$, which yields $p[k] \leq p_1$. Contradiction!

Second prove that

$$r_{I_k,J_k}[k] = a_{I_k,J_k} L_{I_k,J_k}[k] \tag{4.8.79}$$

It can be obtained through the definitions of $r_{i,j}[k]$ and equations 4.8.77 and 4.8.78.

Third prove two bounds on $L_{I_k,J_k}[k+1]$ for $d[k] = 0$

$$L_{I_k,J_k}[k+1] \geq \frac{|\lambda_{I_k}|}{Q} L_{I_k,J_k}[k] \tag{4.8.80}$$

$$L_{I_k,J_k}[k+1] \leq \left(\frac{|\lambda_{I_k}|}{Q} + \varepsilon_0 \right) L_{I_k,J_k}[k] \tag{4.8.81}$$

Those bounds can be obtained from the updating rule of $L_{I_k,J_k}[k]$.

Fourth prove that when $d[k] = 0$,

$$r_{I_k,J_k}[k+1] = a_{I_k,J_k} L_{I_k,J_k}[k+1] \tag{4.8.82}$$

It can be established from equations 4.8.77, 4.8.80 and lemma 4.8.6.

Finally equations 4.8.79, 4.8.81 and 4.8.82 yield the result. $\diamondsuit$

Lemma 4.8.8 *There exists finite k such that*

$$p[k] \leq p_1 \tag{4.8.83}$$

where p_1 is defined in equation 4.8.75.

Sketch of proof: Suppose equation 4.8.83 is violated for all k. When $d[k] = 0$, we can use lemmas 4.8.6 and 4.8.7 to show

$$p[k+1] \leq \eta_0 p[k] \tag{4.8.84}$$

where $\eta_0 = \frac{(1+Q\varepsilon_0)^N \rho(A)}{Q}$.

When $d[k] = 1$, lemma 4.8.6 yields

$$p[k+1] \leq \eta_0 Q p[k] \tag{4.8.85}$$

Consider a window $[k, k+l)$. Denote the dropout rate in the window as $\varepsilon_{l,k} = \frac{\sum_{m=k}^{k+l-1} d[k]}{l}$. Define the maximum difference between $\varepsilon_{l,k}$ and ε (the average in the long run) as $\varepsilon_l = max\,|\varepsilon_{l,k} - \varepsilon|$. By the dropout model in equation 4.2.7, we know $\lim_{l \longrightarrow \infty} \varepsilon_l = 0$. Therefore $\varepsilon_l < \varepsilon_1$ for large enough l.

From time k to $k+l$, equations 4.8.84 and 4.8.85 have been implemented for $l(1-\varepsilon_{l,k})$ and $l\varepsilon_{l,k}$ times respectively. Thus

$$\begin{aligned} \frac{p[k+l]}{p[k]} &= \left(\frac{\rho(A)(1+Q\varepsilon_0)^N}{Q^{1-\varepsilon_{l,k}}} \right)^l \\ &\leq \left(\frac{\rho(A)(1+Q\varepsilon_0)^N}{Q^{1-\varepsilon-\varepsilon_l}} \right)^l \end{aligned} \tag{4.8.86}$$

Let $\eta_l = \frac{\rho(A)(1+Q\varepsilon_0)^N}{Q^{1-\varepsilon-\varepsilon_l}}$ and $f(l) = \eta_l^l$. For large enough l, $\eta_l < 1$. So $\lim_{l \longrightarrow \infty} f(l) = 0$. From equation 4.8.86, we know

$$\lim_{l \longrightarrow \infty} p[k+l] \leq p[k] \lim_{l \longrightarrow \infty} f(l) = 0$$

The above result contradicts our assumption that $p[k] > p_1$ for all k. Therefore this lemma must be true. $\diamondsuit$

In the proof of lemma 4.8.8, we define a sequence $f(l)$. It was shown that $\lim_{l \longrightarrow \infty} f(l) = 0$. Therefore the maximum of $f(l)$ is finite and achieved by finite l. The maximum of $f(l)$ is defined as

$$f_0 = \max_l f(l) \tag{4.8.87}$$

Lemma 4.8.9 *$p[k]$ is eventually bounded, i.e. there exists k_1 such that*

$$p[k] \leq p_1 f_0 \tag{4.8.88}$$

for all $k \geq k_1$.

Sketch of proof: By lemma 4.8.8, there exists $k = k_1$ such that $p[k_1] \leq p_1$.

For any $k > k_1$, define $l = k - k_1$. Equation 4.8.86 yields

$$p[k] \leq f(l)p[k_1] \leq f_0 p[k_1] \leq f_0 p_1 \diamondsuit$$

Final proof to theorem 4.5.1: By the definition of $r_{i,j}[k]$, we know $r_{i,j}[k] \geq a_{i,j} L_{i,j}[k]$, $r_{i,j}[k] \geq \rho_0 \overline{M}$. The two lower bounds, in addition to the the definition $p[k] = \prod_{i=1}^{P} \prod_{j=1}^{n_i} r_{i,j}[k]$, yield

$$L_{i,j}[k] \leq \frac{1}{a_{i,j}} \frac{p[k]}{\left(\rho_0 \overline{M}\right)^{N-1}} \tag{4.8.89}$$

By the definition of $a_{i,j}$, we know that $\frac{1}{a_{i,j}} \leq \rho_0^N$. Substituting this bound into

equation 4.8.89 yields

$$L_{i,j}[k] \leq \rho_0^N \frac{p[k]}{\left(\rho_0 \overline{M}\right)^{N-1}} \tag{4.8.90}$$

From lemma 4.8.9, we know $p[k]$ is eventually bounded by $f_0 p_1$. Considering equation 4.8.90, we get the following eventual bound

$$L_{i,j}[k] \leq \rho_0^N \frac{f_0 p_1}{\left(\rho_0 \overline{M}\right)^{N-1}} \tag{4.8.91}$$

As argued before, the eventual boundedness of $L_{i,j}[k]$ implies that of $z[k]$ and $x[k]$.

CHAPTER 5

PERFORMANCE OF QUANTIZED LINEAR CONTROL SYSTEMS

5.1 Introduction

As a control system, we require the quantized system not only to be stable but also to perform well. In chapter 4, the stability of quantized systems is studied. It is shown that when the available bit rate is above the *"minimum"* bit rate, the stability of quantized systems can be guaranteed. By stability, we mean asymptotic stability for noise-free quantized systems, i.e. the state converges to 0 and BIBO stability for quantized systems with bounded noise, i.e. the state is eventually bounded. Although several quantization policies can achieve the above stability, they may perform quite differently. For example different policies may yield different state convergence rates for noise-free systems and different state bounds for systems with bounded noise. It is natural to pursue the quantization policy with the "best" performance, which is exactly the topic of this chapter.

The feedback system studied in this paper is shown in figure 5.1. In this figure, $x[k] \in \Re^n$ is the system state at time k. This state is generated by a discrete-time linear time-invariant system called the *plant*. The system state is transformed into a discrete-valued symbol, $s[k]$, by the *encoder*. The symbol is transmitted over a *channel* and is then transformed by the *decoder* into a quantized version of the state, $x^q[k] \in \Re^n$. The plant uses this quantized state and a noise signal, $w[k]$, as

inputs. The noise signal $w[k] \in \Re^n$ is assumed to be bounded as

$$\|w\|_\infty \leq M$$

where M is some known constant. A noise free quantized system is a special case of the system in figure 5.1 with $M = 0$.

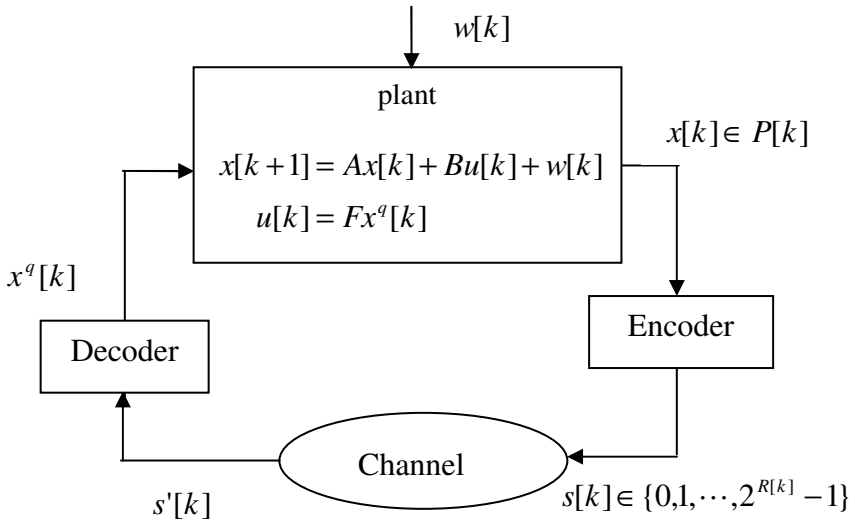


Figure 5.1. Quantized control systems

The *encoder* first assumes that $x[k] \in \Re^n$ lies in a known subset $P[k] \subset \Re^n$ which we call the *state's scope*. The encoder then partitions $P[k]$ into a finite number of subsets. The index of the subset containing $x[k]$ is taken as the *symbol*, $s[k]$ representing the state. This index takes values in the set $\mathcal{S}_R = \{0, 1, 2, \ldots, 2^{R[k]} - 1\}$ where $R[k]$ is the number of bits transmitted over the chan-

nel per measurement at time k. The encoder can therefore be represented as a potentially time-varying map $\text{Enc} : \Re^n \to \mathcal{S}_R$. The channel is assumed to be error free with a constant delay of d steps, so that $s'[k] = s[k-d]$. The *decoder* uses the received symbol $s'[k]$ to compute a single point estimate of the state. This estimate is denoted as $x^q[k]$ and the decoder can also be represented as a potentially time-varying map $\text{Dec} : \mathcal{S}_R \to \Re^n$. The triple $(P[k], \text{Enc}, \text{Dec})$ is referred to as a *quantization policy*.

For a given bit sequence $\{R[k]\}$, we can identify several different quantization policies. Policies whose state scope $P[k]$ is time-invariant are said to be *static*. *Dynamic policies* adjust the state scope $P[k]$ (zooming-in and zooming-out) at every time step to obtain a tight bound on $x[k]$. Dynamic quantization policies may be further classified by how they assign their $R[k]$ bits. In particular, let $b_i[k]$ denote the number of bits the encoder allocates to represent the ith component of the state $x[k]$. The total number of bits must equal $R[k]$

$$R[k] = \sum_{i=1}^{n} b_i[k] \tag{5.1.1}$$

The policy used for deciding $b_i[k]$ $(i = 1, \cdots, n)$ is called a *bit assignment policy*. A static bit assignment assumes $b_i[k]$ is a constant for all k. A dynamic bit assignment allows $b_i[k]$ to be time-varying. This thesis pursues the "best" bit assignment policy which minimizes the specified performance under the bit resource given in eq. 5.1.1. *What performance index should we choose?* In order to answer that question, we study the effect of quantization on the system state $\{x[k]\}$.

It can be seen from figure 5.1 that the quantization policy takes $x[k]$ as input and $x^q[k]$ as output. Their difference, i.e. the quantization error, is defined as $e[k] = x[k] - x^q[k]$. The quantized system in figure 5.1 can be equivalently

transformed into the following linear system with $\{w[k]\}$ and $\{e[k]\}$ as inputs.

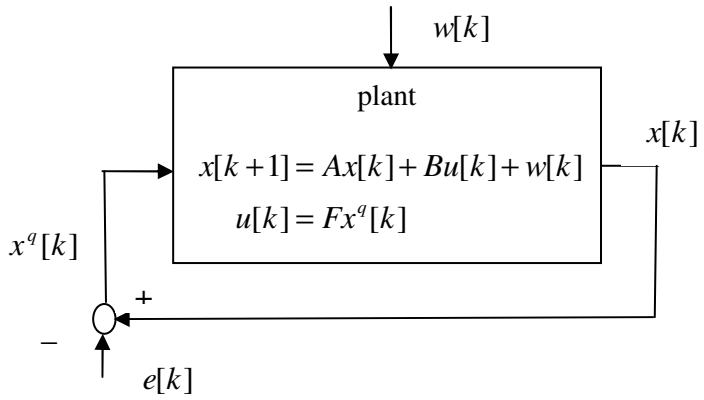


Figure 5.2. An equivalent linear control system

The system in figure 5.2 is linear, so its state $x[k]$ can be decomposed into 2 terms.

$$x[k] = \overline{x}[k] + x_e[k] \tag{5.1.2}$$

where $\overline{x}[k]$ is incurred by the initial condition $x[0]$ and the exogeneous noise $\{w[k]\}$ and $x_e[k]$ comes directly from the quantization error $\{e[k]\}$. We may say that the effect of quantization error is to introduce the term $x_e[k]$. For noise free quantized systems, the size of $\{x_e[k]\}$ can be measured by L_2 norm ($\|x_e\|_2 = \left(\sum_{k=1}^{\infty} x_e^T[k]x_e[k]\right)^{\frac{1}{2}}$) [39]. *The aim in designing a quantization policy is to make* $\|x_e\|_2$ *as small as possible,* i.e. to minimize the effect due to quantization. Denote

the transfer function from $\{e[k]\}$ to $\{x_e[k]\}$ as $G_{x_e,e}(z)$. Then we get [39]

$$\|x_e\|_2 \leq \|G_{x_e,e}(z)\|_\infty \|e\|_2 \tag{5.1.3}$$

where $\|G_{x_e,e}(z)\|_\infty$ denotes the H_∞ norm of $G_{x_e,e}(z)$. $G_{x_e,e}(z)$ is determined by controller design and is out of our control. So minimizing $\|x_e\|_2$ is equivalent to minimizing $\|e\|_2$ in eq. 5.1.3, i.e. **the optimal quantization policy is to minimize $\|e\|_2$.** The quantization error $\{e[k]\}$ is effected by the initial condition $x[0]$ (for noise-free quantized systems). So we measure performance by the worst case of $\|e\|_2$,

$$\sup_{x[0]\in P[0]} \|e\|_2 \tag{5.1.4}$$

For quantized systems with bounded noise, the quantization error $\{e[k]\}$ is eventually bounded [45] but not summable. We can measure the size of $\{e[k]\}$ with its eventual bound $\lim_{N\longrightarrow\infty} \sup\{e^T[k]e[k]\}_{k=N}^{\infty}$. By repeating the above arguments on noise-free quantized systems, we can see that in order to mitigate the effect of $\{e[k]\}$ on $\{x_e[k]\}$, the optimal quantization policy is to minimize $\{e[k]\}$ with the following defined performance.

$$\begin{array}{c} \sup \\ x[0] \in P[0] \\ \{w[k]\}_{k=0}^{\infty} \end{array} \quad \lim_{N\longrightarrow\infty} \sup\{e^T[k]e[k]\}_{k=N}^{\infty} \tag{5.1.5}$$

Note that we consider the worst case for all possible $x[0]$ and $\{w[k]\}_{k=0}^{\infty}$ in eq. 5.1.5.

Compared to the rich literature on stability of quantized systems, the literature

on performance of quantized systems is relatively scarce. In [30], the perturbation noise $\{w[k]\}$ is assmed to be Gaussian and the performance is defined as a linear quadratic form of the system state $\{x[k]\}$. Simulations were made to compare the performance under different quantization policies. In [63], the configuration similar to [30] is considered. It is shown that the concerned optimal performance decomposes into two terms. One is determined by the perfect (without quantization) state feedback . The other comes from the quantization error. The optimal quantization policy is the one to minimize the term due to quantization error, which agrees with our discussion on minimizing the performance specified in eq. 5.1.4 and 5.1.5. Although the results in [63] are intuitively pleasing, they are less realistic because of their estimation policies. In [63], the state is estimated as the conditional mean based on the received symbols $\{s'[k]\}$. It is not trivial to compute this conditional mean because the quantization operation is non-linear and it is difficult to find a recursive way to compute that mean. The only possible way is to compute it with the probability density function (pdf) of the state. So the state's pdf has to be updated every step which may be unrealistically complicated. In [40], a scalar quantized system is studied, the input noise is assumed to be bounded and performance is measured by the eventual upper bound of quantization error which is really the L_∞ norm of $\{e[k]\}$. The effect of the number of quantization levels and the packet dropout pattern on performance is studied. Note that a simple estimation policy is chosen in [40] and the proposed quantization policy is realistic. The results in [40] will be extended to 2-dimensional systems in this chapter. In [22], a noise-free quantized system is considered and the convergence rate of quantization error is used to measure system performance. It proves that the quantization policy proposed in [48] achieves the best (fastest)

convergence rate.

This chapter is organized as follows. Section 5.2 presents the mathematical model of the quantized system, the used quantization alogrithm, one special bit assignment policy named as **Dynamic Bit Assignment Policy (DBAP)** and some preliminary results. Section 5.3 studies a noise-free quantized system and proves that the performance defined in eq. 5.1.4 is *optimized by the proposed DBAP*. Section 5.4 studies a quantized system with bounded noise and comes up with both a lower and an upper bounds on the optimal performance defined in eq. 5.1.5. It deserves special attention that the obtained upper performance bound can be achieved by the proposed DBAP. So DBAP may be treated as a sub-optimal quantization policy with the known performance gap. Section 5.5 includes a summary of achieved results in this chapter and some remarks on relaxing the assumptions. All proofs are placed in the appendix, section 5.6.

5.2 System model and preliminary results

The quantized system in figure 5.1 can be modeled as

$$\begin{cases} x[k+1] &= Ax[k] + Bu[k] + w[k] \\ u[k] &= Fx^q[k] \end{cases} \tag{5.2.6}$$

We assume that

1. (A, B) is controllable and F is a stabilizing state feedback gain.

2. A is diagonalizable. For simplicity, we focus on the two-dimensional case of $A = diag(\lambda_1, \lambda_2)$ and $\lambda_i > 1$ $(i = 1, 2)$.

3. The system is driven by bounded process noise, $\|w\|_\infty \leq M$.

4. Fixed bit rate is assumed, i.e. $R[k] = R$.

5. The network is error-free with 1 step delay, i.e. $s'[k] = s[k-1]$.

The quantization method used in this paper originates from the uncertainty set evolution method introduced in [14] and [61]. This approach presumes that the encoder and the decoder agree that the state lies within the set $P[k]$ which has the form of

$$x[k] \in P[k] = x^q[k] + U[k], \forall k \geq 0. \tag{5.2.7}$$

where $x^q[k]$ is the center of $P[k]$. By assumption 2, we restrict our attention to a two-dimensional system. So the uncertainty set $U[k]$ may be characterized as

$$\begin{aligned} U[k] &= \text{rect}(L_1[k], L_2[k]) \\ &= [-L_1[k], L_1[k]] \times [-L_2[k], L_2[k]]. \end{aligned}$$

In this equation $L_1[k]$ and $L_2[k]$ are non-negative and they represent the half-length of the sides of the rectangular set $U[k]$. We define the *quantization error* as $e[k] = x[k] - x^q[k]$. Just prior to time k we know that $e[k] \in U[k]$ where we refer to $U[k]$ as the *uncertainty set* at time k. We then partition both sides of $U[k]$. The first side, $L_1[k]$, is partitioned into $2^{b_1[k]}$ equal parts and the second side, $L_2[k]$, is partitioned into $2^{b_2[k]}$ equal parts. We impose a constant bit rate constraint on our bit assignment which requires that

$$b_1[k] + b_2[k] = R \tag{5.2.8}$$

for all k. After a new measurement of the state $x[k]$ is made, then the encoder knows that

$$x[k] \in x^q_{s[k]}[k] + U_{s[k]}[k]$$

where $x^q_{s[k]}[k]$ is the center of the smaller subset and the set

$$U_{s[k]}[k] = \text{rect}\left(\frac{L_1[k]}{2^{b_1[k]}}, \frac{L_2[k]}{2^{b_2[k]}}\right)$$

The index of this smaller subset, $s[k]$, is transmitted across the channel and the decoder reconstructs the state at time $k+1$ using the equations

$$\begin{cases} x[k+1] & \in & x^q[k+1] + U[k+1] \\ U[k+1] & = & \text{rect}(L_1[k+1], L_2[k+1]) \\ x^q[k+1] & = & Ax^q_{s[k]}[k] + BFx^q[k] \\ L_1[k+1] & = & \frac{\lambda_1}{2^{b_1[k]}} L_1[k] + M \\ L_2[k+1] & = & \frac{\lambda_2}{2^{b_2[k]}} L_2[k] + M \end{cases} \tag{5.2.9}$$

The choice for $b_i[k]$ $(i = 1, 2)$ represents a *bit assignment policy*. With the requirement that $b_1[k] + b_2[k] = R$, we're confining our attention to constant bit rate quantization schemes. The motivation for doing this is that many communication systems work best under a constant bit rate [29].

This chapter is interested in the performance achievable under various bit assignment policies. In particular, we measure performance using equations 5.1.4 and 5.1.5. Note that by definition

$$|e_i[k]| \leq L_i[k], i = 1, 2 \tag{5.2.10}$$

This inequality becomes equality for the specific choice of $x[0]$ and $\{w[k]\}$, e.g. $x[0] = [L_1[0], L_2[0]]^T$ and $w[j] = [M, M]^T, \forall j$, that maximizes $\|e\|_2$ defined in eq. 5.1.4 or $\lim_{N\longrightarrow\infty} \sup\{e_1^2[k] + e_2^2[k]\}_{k=N}^{\infty}$ in eq. 5.1.5.

One bit assignment policy used in this chapter is a variation of the bit assignment policy found in chapter 4 We still call it *dynamic bit assignment policy* or **DBAP**. DBAP is a recursive algorithm that generates $b_i[k]$ as follows.

Algorithm 5.2.1 Dynamic Bit Assignment Policy

1. *Initialize* $b_1[k] = 0$ *and* $b_2[k] = 0$,
   and set $L_1 = \lambda_1 L_1[k]$ *and* $L_2 = \lambda_2 L_2[k]$.

2. *For* $q = 1$ *to* R
   $I = argmax_{i\in\{1,2\}} L_i$.
   $b_I[k] := b_I[k] + 1$ *and* $L_I = L_I/2$.

The following lemma provides a closed-form characterization of $b_i[k]$ $(i = 1, 2)$ generated by DBAP.

Lemma 5.2.1 *The bit assignment algorithm, DBAP, generates the following bit assignments*

$$b_1[k] = R - b_2[k] \tag{5.2.11}$$

$$b_2[k] = \begin{cases} 0, & \frac{1}{2^{R+1}}\lambda_1 L_1[k] \geq \lambda_2 L_2[k] \\ R, & \frac{1}{2^{-R-1}}\lambda_1 L_1[k] \leq \lambda_2 L_2[k] \\ \left[\frac{1}{2}\left(R - \log_2\left(\frac{\lambda_1 L_1[k]}{\lambda_2 L_2[k]}\right)\right)\right], & otherwise \end{cases} \tag{5.2.12}$$

where $[\cdot]$ *is the downward rounding function in which* $[1.5] = 1$.

The fundamental insight behind DBAP may be interpreted as *"keeping* $L_1[k+1]$ *and* $L_2[k+1]$ *as balanced as possible"*. This balancing idea is demonstrated through the following lemma.

Lemma 5.2.2 *Under DBAP, there exists a finite* $k_0 > 0$ *such that*

$$0.5 \leq \frac{L_1[k]}{L_2[k]} < 2, \forall k \geq k_0 \tag{5.2.13}$$

5.3 Optimal bit assignment policy in noise free quantized linear control systems

For a noise free quantized system, we measure its performance with the index defined in eq. 5.1.4. Specifically we consider a N-step version of that performance index.

$$J_N = \max_{x[0] \in P[0]} \sum_{k=1}^{N} \left(e_1^2[k] + e_2^2[k] \right) \tag{5.3.14}$$

The inequality in eq. 5.2.10 ($|e_i[k]| \leq L_i[k]$) becomes equality for the specific choices of $x[0]$, e.g. $x[0] = [L_1[0], L_2[0]]^T$, that maximize $e_1^2[k] + e_2^2[k]$ for all k. This means that J_N in equation 5.3.14 may be rewritten as

$$J_N = \sum_{k=1}^{N} \left(L_1^2[k] + L_2^2[k] \right) \tag{5.3.15}$$

This section characterizes the bit assignment policy that minimizes the performance index, J_N, in equation 5.3.15. Our optimization problem is formally stated as follows,

$$\begin{array}{ll} \min_{\{b_1[k], b_2[k]\}_{k=0}^{N-1}} & \sum_{k=1}^{N} (L_1^2[k] + L_2^2[k]) \\ \text{subject to} & b_1[k] + b_2[k] = R, \end{array} \tag{5.3.16}$$

where $b_1[k], b_2[k] \in \mathcal{N}$. Let $\mathbf{b} = \{b_1[j], b_2[j]\}_{j=0}^{N-1}$ denote the optimal solution to this problem. Optimization problem 5.3.16 is an integer programming. The general integer programming is, unfortunately, $\mathcal{N}P$ hard [67]. We will circumvent the $\mathcal{N}P$ complexity by first considering a sequence of simpler problems and then showing that the solutions to these simpler problems also solve the original problem and furthermore that they are generated by the proposed DBAP.

Consider the following sequence of minimization problems indexed by k for $k = 1, \ldots, N$.

$$\begin{array}{ll} \min_{\{b_1[j], b_2[j]\}_{j=0}^{k-1}} & (L_1^2[k] + L_2^2[k]) \\ \text{subject to} & b_1[j] + b_2[j] = R, \end{array} \tag{5.3.17}$$

where $b_1[j], b_2[j] \in \mathcal{N}$. The solution to the kth subproblem will be denoted as $\mathbf{b}^{(k)} = \{b_1^{(k)}[j], b_2^{(k)}[j]\}_{j=0}^{k-1}$. The following lemma establishes the basic relationship between subproblems 5.3.17 and the original problem 5.3.16. In the following lemma, we say $\mathbf{b}^{(k-1)} \subset \mathbf{b}^{(k)}$ if and only if $b_i^{(k-1)}[j] = b_i^{(k)}[j]$ for $j < k-1$. Essentially this means that $\mathbf{b}^{(k-1)}$ is a prefix of $\mathbf{b}^{(k)}$.

Lemma 5.3.1 *If $\{\mathbf{b}^{(k)}\}_{k=1}^{N}$ solves the sequence of subproblems 5.3.17 such that $\mathbf{b}^{(k-1)} \subset \mathbf{b}^{(k)}$ for $k = 2, \ldots, N$, then $\mathbf{b}^{(N)}$ solves the original problem 5.3.16.*

Rather than directly solving subproblem 5.3.17, we consider a *relaxed* problem of the form

$$\begin{array}{ll} \min_{s_1[k], s_2[k]} & \left(\frac{\lambda_1^k}{2^{s_1[k]}} L_1[0]\right)^2 + \left(\frac{\lambda_2^k}{2^{s_2[k]}} L_2[0]\right)^2 \\ \text{subject to} & s_1[k] + s_2[k] = kR \end{array} \tag{5.3.18}$$

where $s_1[k], s_2[k] \in \mathcal{N}$. In these relaxed problems, we interpret $s_i[k]$ as the number of bits used to represent the ith component of the state up to time k. In other

words, we let $s_i[k] = \sum_{j=0}^{k-1} b_i[j]$. Let $\mathbf{s}^{(k)} = \{s_1[k], s_2[k]\}$ denote the solution to the kth relaxed subproblem. Note that $L_i[k] = \frac{\lambda_i^k}{2^{s_i[k]}} L_i[0]$ $(i = 1, 2)$ by eq. 5.2.9. So subproblems 5.3.17 and 5.3.18 have the same performance index. In subproblems 5.3.18, the constant bit rate constraint (equation 5.2.8) implies that the summed numbers of bits satisfy,

$$s_1[k] + s_2[k] \;=\; kR \tag{5.3.19}$$

So this problem relaxes problem 5.3.17 by only minimizing the cost index with respect to the bit sum, rather than the individual history of assigned bits, i.e. the feasible set is extended. The following lemma states the solution for problem 5.3.18.

Lemma 5.3.2 *The solution to the kth problem in equation 5.3.18 is*

$$s_1[k] \;=\; kR - s_2[k] \tag{5.3.20}$$

$$s_2[k] \;=\; \begin{cases} 0, & \frac{\lambda_1^k}{2^{kR+1}} L_1[0] \geq \lambda_2^k L_2[0] \\ kR, & \frac{\lambda_1^k}{2^{-kR-1}} L_1[0] \leq \lambda_2^k L_2[0] \\ \left[\frac{1}{2}\left(kR - \log_2\left(\frac{\lambda_1^k L_1[0]}{\lambda_2^k L_2[0]}\right)\right)\right], & \textit{otherwise} \end{cases} \tag{5.3.21}$$

It is important to note a similarity between equation 5.3.21 in lemma 5.3.2 and the characterization of the bit assignment generated by DBAP in equation 5.2.12 in lemma 5.2.1. The following lemma formalizes this relationship by asserting that the sequence of summed bits, $\mathbf{s}^{(k)}$, generated by DBAP indeed solve the relaxed problem 5.3.18 while enforcing the additional reqirements that $b_1[k] + b_2[k] = R$ and $s_i[k] = \sum_{i=1}^{k-1} b_i[k]$. These additional constraints are precisely those that were

relaxed in going from problem 5.3.17 to 5.3.18, so DBAP also solves the original sequence of subproblems in equation 5.3.17.

Lemma 5.3.3 *Let* $\{b_i[k]\}$ *denote the bit sequence generated by the proposed DBAP. If we let*

$$s_i[k] = \sum_{j=0}^{k-1} b_i[j],\ i = 1, 2$$

then $\mathbf{s}^{(k)} = \{s_1[k], s_2[k]\}$ *also solves the* $k-th$ *relaxed minimization problem in equation 5.3.18.*

Based on Lemmas 5.3.1, 5.3.2 and 5.3.3, we establish the optimality of our proposed DBAP for noise-free quantized linear systems.

Theorem 5.3.1 *Dynammic bit assignment (DBAP) generates a bit assignment that solves optimization 5.3.16.*

Example: The plant with system matrices

$$A = \begin{bmatrix} 1.1 & 0 \\ 0 & 1.8 \end{bmatrix}, B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Let the feedback gain matrix be $F = [1.7, -4.6]$ and let $R = 2$. By static bit assignment policy [41] [62],

$$b_1[k] = 1, b_2[k] = 1 \tag{5.3.22}$$

From figure 5.3 it is clear that DBAP performs better than the static bit assignment policy.

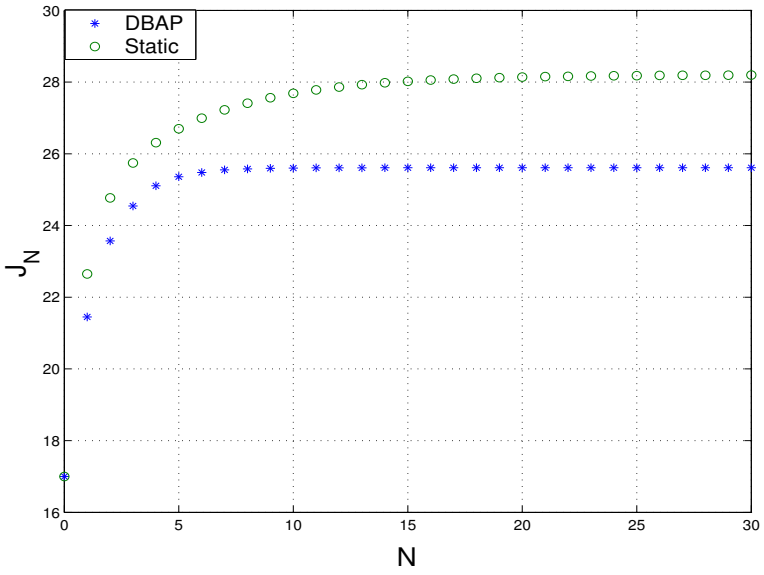


Figure 5.3. Performance of a quantized system

5.4 Performance of quantized linear systems with bounded noise under dynamic bit assignment

From [45], we know the quantization error $\{e[k]\}$ for a quantized system with bounded noise is bounded. By eq. 5.1.5, the concerned performance is defined as

$$J = \sup_{\substack{x[0] \in P[0], \\ \{w[k]\}_{k=0}^{\infty}}} \lim_{N \longrightarrow \infty} \frac{1}{N} \sum_{k=1}^{N} e_1^2[k] + e_2^2[k] \tag{5.4.23}$$

The inequality in eq. 5.2.10 ($|e_i[k]| \leq L_i[k]$) becomes equality for the specific choice of $x[0]$ and $\{w[k]\}$, e.g. $x[0] = [L_1[0], L_2[0]]^T$ and $w[j] = [M, M]^T, \forall j$, that maximizes $e_1^2[k] + e_2^2[k]$ for all k. This means that J in equation 5.4.23 may be

rewritten as

$$J = \lim_{N \longrightarrow \infty} \sup\{L_1^2[k] + L_2^2[k]\}_{k=N}^{\infty} \tag{5.4.24}$$

Different bit assignments will achieve various performance levels, J (as defined in equation 5.4.24), for the same constant bit rate, R. We may therefore consider the *optimal achievable performance*, J^*, as generated by the following optimization problem,

$$\begin{array}{ll} \text{minimize} & \lim_{N \longrightarrow \infty} \sup\{L_1^2[k] + L_2^2[k]\}_{k=N}^{\infty} \\ \text{with respect to} & \{b_1[k]\}_{k=0}^{\infty}, \{b_2[k]\}_{k=0}^{\infty} \\ \text{subject to} & b_1[k] \in \mathcal{N}, b_2[k] \in \mathcal{N} \\ & b_1[k] + b_2[k] = R \end{array} \tag{5.4.25}$$

The optimization problem 5.4.25 is an integer programming, which is known $\mathcal{NP}$ hard [67]. Furthermore there are an infinite number of decision variables $\{b_1[k], b_2[k]\}_{k=0}^{\infty}$ in that optimization problem. So it is very difficult to solve it. Instead of searching for the optimal performance in eq. 5.4.25, we try to bound the optimal performance J^* under the assumption that J^* exists. The main results stated below identify a lower bound, $\underline{J}$, and an upper bound, $\overline{J}$, on J^*. Moreover, we demonstrate that these bounds are sometimes achievable by various bit assignment policies.

Proposition 5.4.1 *J^* is bounded below by*

$$J^* \geq \underline{J} = 2p_0 \tag{5.4.26}$$

where J^* satisfies optimization problem 5.4.25 and p_0 is the positive solution to

$$p_0 = \rho p_0 + 2\sqrt{\rho} M \sqrt{p_0} + M^2 \tag{5.4.27}$$

where $\rho = \lambda_1 \lambda_2 / 2^R < 1$

Furthermore, if

$$\frac{1}{2}\left(R + \log_2\left(\frac{\lambda_1}{\lambda_2}\right)\right) \text{ is an integer.} \tag{5.4.28}$$

then the inequality in equation 5.4.26 holds with equality.

Proof: See section 5.6

One important consequence of the above proposition is that if the condition in equation 5.4.28 holds, then the optimal performance level, J^*, is achieved by a static bit assignment policy. This result is stated in the following proposition.

Proposition 5.4.2 *If the condition in equation 5.4.28 holds, we construct a static quantization policy with*

$$\begin{aligned} b_1[k] = b_1 &= \frac{1}{2}\left(R + \log_2\left(\frac{\lambda_1}{\lambda_2}\right)\right) \\ b_2[k] = b_2 &= R - b_1 \end{aligned}$$

The performance under the above static policy achieves the lower bound in proposition 5.4.1.

$$\begin{aligned} J_{static} &= \underline{J} \\ &= \left(\frac{M}{1 - \frac{\lambda_1}{2^{b_1}}}\right)^2 + \left(\frac{M}{1 - \frac{\lambda_2}{2^{b_2}}}\right)^2 \end{aligned} \tag{5.4.29}$$

Proof: See section 5.6.

An upper bound on J^* is obtained using algorithm 5.2.1 (DBAP) in section 5.2. The following proposition formally states this bound.

Proposition 5.4.3 *The optimal performance J^* is bounded above by*

$$\begin{aligned} J^* &\leq \overline{J} \\ &= 2\left(\rho p_{\alpha_u} g(2) + 2M\sqrt{\rho}\sqrt{p_{\alpha_u}} g(1) + M^2\right) \end{aligned}$$

where $\rho = \lambda_1\lambda_2/2^R$, $g(\alpha) = 0.5\,(2^{0.5\alpha} + 2^{-0.5\alpha})$, and p_{α_u} satisfies the equation

$$p_{\alpha_u} = \rho p_{\alpha_u} + 2\sqrt{\rho} M \sqrt{p_{\alpha_u}} g(1) + M^2 \tag{5.4.30}$$

Proof: See section 5.6.

Remark: Let J_{DBAP} denote the performance level achieved by our proposed DBAP algorithm. The proof for proposition 5.4.3 actually shows that the performance achieved by DBAP is bounded above by $\overline{J}$ (i.e. $J_{DBAP} \leq \overline{J}$). By the optimality nature of J^*, this means that $J^* \leq J_{DBAP} \leq \overline{J}$.

Example: The quantized plant has the system matrices

$$A = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

with state feedback matrix $F = [1.7, -4.6]$ and noise bound $M = 1$. It is assumed that $R = 2$.

We first study the effect of the eigenvalue ratio λ_2/λ_1 on the achievable performance J under various bit assignment policies. In particular, we fixed $\lambda_1 = 1.1$ and varied λ_2 subject to the constraint that $\lambda_2 > \lambda_1$. Simulations were used to

evaluate the performance, J, achieved using the dynamic bit assignment described in algorithm 5.2.1 assuming $R = 2$.

Figure 5.4 plots the lower bound, $\underline{J}$ (J_l), of proposition 5.4.1 and the upper bound, $\overline{J}$ (J_u), of proposition 5.4.3 as a function of λ_2/λ_1. This figure also plots the performance level, J_{static}, achieved using the aforementioned static bit assignment policy as well as the performance level, J_{DBAP}, using DBAP.

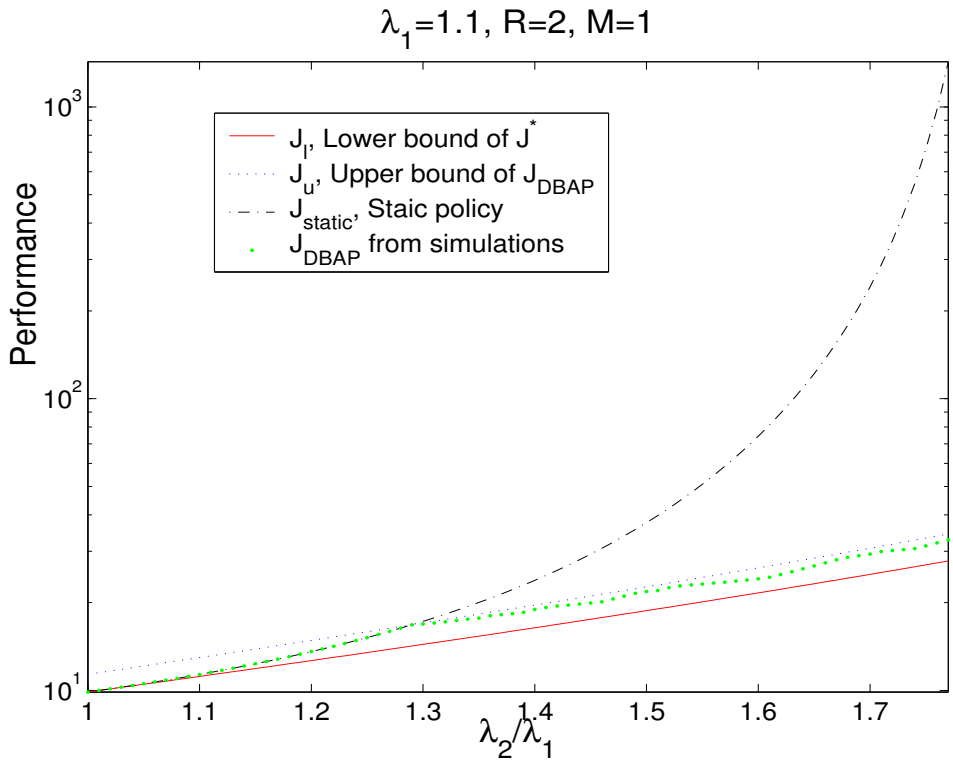


Figure 5.4. Performance of a quantized system with bounded noise

These results show that the gap, $\Delta J = \overline{J} - \underline{J}$ is small and almost constant for

varying $\frac{\lambda_1}{\lambda_2}$ ($\frac{\Delta J}{\underline{J}} \approx 10\%$). The results show that performance achieved by the static policy lies within the specified bounds until $\lambda_2/\lambda_1 \geq 1.3$. At this point, the static policy no longer appears to be optimal. For the DBAP, we see that our proposed algorithm actually generates the same static bit assignment for $\lambda_2/\lambda_1 < 1.3$. For $\lambda_2/\lambda_1 \geq 1.3$, the performance of the DBAP appears to closely follow the upper bound $\overline{J}$ from proposition 5.4.3. These results appear to support the preliminary conclusion that our proposed bit assignment policy (DBAP) is nearly optimal with respect to the performance measure defined in eq. 5.1.5 This observation has led us to conjecture that perhaps the DBAP can indeed be considered an optimal bit assignment policy.

We then study the effect of the bit rate, R, on the gap $\Delta J = \overline{J} - \underline{J}$. In this particular case we chose $\lambda_1 = 1.1$ and $\lambda_2 = 1.6$ and allowed R to vary between 1 and 10. Figure 5.5 plots the upper and lower performance bounds, $\overline{J}$ (J_u) and $\underline{J}$ (J_l), as a function of R. It can be seen that the gap, ΔJ appears to be relatively tight for this example and that the gap asymptotically approaches zero as R goes to infinity. Both observations are not that surprising. The equation characterizing the lower bound (eq. 5.4.27) and the upper bound (eq. 5.4.30) have nearly identical forms in which the size of the gap is governed by ρ. Numerical evaluation shows that this gap is small and asymptotically approaches zero when R increases.

The results shown above can be interpreted as bounds on a type of *rate-distortion* function for feedback control systems. This observation is emphasized in figure 5.5 where we plot the effective distortion measure for a control system (its achieved performance level) as a function of the bit rate, R. The fact that there are bit assignment policies (or what we might call coding policies) that

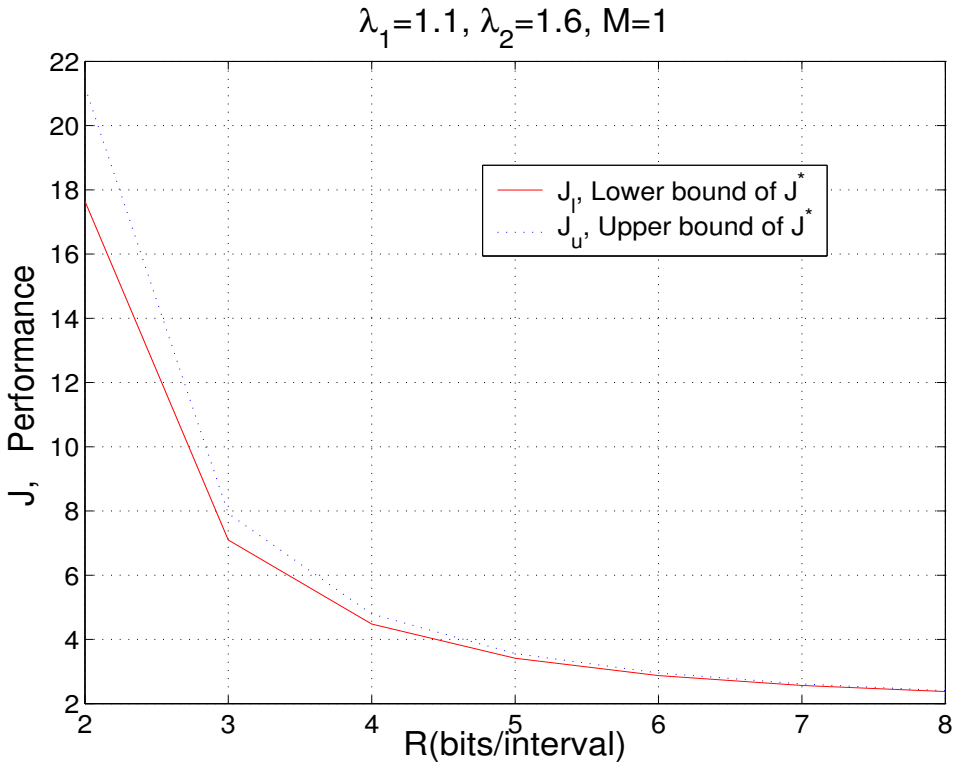


Figure 5.5. Performance bounds for a quantized system with bounded noise

appear to achieve J^* suggests that assignment policies such as DBAP may be extremely useful in developing source codes for control systems implemented over communication networks.

5.5 Conclusions

One of the two primary results in this chapter is that the optimal performance of a noise-free quantized system can be achieved by the proposed **dynamic bit assignment policy (DBAP)**, i.e. DBAP is the pursued optimal quantization policy. The other primary result shows that the optimal performance of a quantized system with bounded noise can be tightly bounded above and below. The

upper bound can be achieved by our proposed dynamic bit assignment policy (DBAP). Under certain situations, the lower bound can be achieved by a static bit assignment policy. In our opinion, the fact that these bounds are achievable has great significance for the feedback control of dynamic systems over communication networks.

While the derivation in this chapter was restricted to a 2-dimensional system, the results can be extended for n-dimensional diagonalizable systems. When the system matrix is n-dimensional real and diagonal, $U[k]$ may be overbounded by a rectangular set whose side lengths are $L_1[k]$, $\cdots$, $L_n[k]$. The bit assignment strategy used in algorithm 5.2.1 may be used to allocate $b_i[k]$ $(i = 1, \cdots, n)$ in a similar manner to that described in this paper. If the system matrix is 2-dimensional and diagonalizable with complex eigenvalues $\rho e^{j\theta}$, then we can rotate the coordinate axes by θ every step as in [61]. The new system matrix under the rotated coordinates will be real and diagonal, so algorithm 5.2.1 can still be used.

5.6 Appendix: proofs

This section uses the following notational conventions.

$$\gamma[k] = \frac{L_1[k]}{L_2[k]} \tag{5.6.31}$$

$$p[k] = L_1[k]L_2[k] \tag{5.6.32}$$

$$\rho = \frac{\lambda_1\lambda_2}{2^R} \tag{5.6.33}$$

$$g(\alpha) = 0.5\left(\sqrt{2^\alpha} + \frac{1}{\sqrt{2^\alpha}}\right). \tag{5.6.34}$$

Note that $g(\alpha)$ is a strictly increasing function of α for $\alpha \geq 0$. The stability condition in [45] requires

$$\rho < 1 \tag{5.6.35}$$

The following technical lemma is frequently used. Its proof is straightforward and omitted here.

Lemma 5.6.1 *If $x, y > 0$ and $xy = \beta$, then*

$$x + y = 2\sqrt{\beta} g(|\log_2(x/y)|). \tag{5.6.36}$$

5.6.1 Proofs of the lemmas in section 5.2

Proof of Lemma 5.2.1: We prove this lemma using mathematical induction on R. When $R = 1$, Lemma 5.2.1 trivially holds.

Suppose Lemma 5.2.1 holds for $R = R_1$. We try to prove that it also holds for $R = R_1 + 1$. By the assumption, we know

$$\begin{aligned} & b_2[k](R_1, L_1[k], L_2[k]) \\ = & \begin{cases} 0, & \frac{1}{2^{R_1+1}}\lambda_1 L_1[k] \geq \lambda_2 L_2[k] \\ R_1, & \frac{1}{2^{-R_1-1}}\lambda_1 L_1[k] \leq \lambda_2 L_2[k] \\ \left[\frac{1}{2}\left(R_1 - \log_2\left(\frac{\lambda_1 L_1[k]}{\lambda_2 L_2[k]}\right)\right)\right], & otherwise \end{cases} \end{aligned} \tag{5.6.37}$$

where the inclusion of arguments R_1, $L_1[k]$ and $L_2[k]$ is to emphasize the dependence of $b_2[k]$ on R_1, $L_1[k]$ and $L_2[k]$. We now compute $b_2[k](R_1 + 1, L_1[k], L_2[k])$. There are 3 different cases to consider based on $\gamma[k]$.

Case 1: In this case,

$$\gamma[k] \geq \frac{\lambda_2}{\lambda_1} 2^{(R_1+1)+1}$$

The procedure in algorithm 5.2.1, selects $b_2[k] = 0$, which satisfies eq. 5.2.12. So lemma 5.2.1 holds for this case.

Case 2: In this case

$$\gamma[k] \leq \frac{\lambda_2}{\lambda_1} 2^{-(R_1+1)-1}$$

Algorithm 5.2.1 assigns $b_2[k] = R_1 + 1$, which also satisfies eq. 5.2.12. So lemma 5.2.1 also holds for this case.

Case 3: In this case,

$$\frac{\lambda_2}{\lambda_1} 2^{-(R_1+1)-1} < \gamma[k] < \frac{\lambda_2}{\lambda_1} 2^{(R_1+1)+1}$$

This case can be further decomposed into two subcases; $\lambda_1 L_1[k] \geq \lambda_2 L_2[k]$ and $\lambda_1 L_1[k] < \lambda_2 L_2[k]$.

- If $\lambda_1 L_1[k] \geq \lambda_2 L_2[k]$, then algorithm 5.2.1 assigns the first bit to $L_1[k]$. So

$$\begin{aligned} & b_2[k]\,(R_1 + 1, L_1[k], L_2[k]) \\ = \;& b_2[k] \left(R_1, \frac{L_1[k]}{2}, L_2[k] \right) \end{aligned} \tag{5.6.38}$$

Since $\gamma[k] < \frac{\lambda_2}{\lambda_1} 2^{(R_1+1)+1}$ and $\lambda_1 L_1[k] \geq \lambda_2 L_2[k]$, we can infer that

$$2^{-R_1-1} < 2^{-1} < \frac{\lambda_1 \frac{L_1[k]}{2}}{\lambda_2 L_2[k]} < 2^{R_1+1}. \tag{5.6.39}$$

By the assumption that lemma 5.2.1 holds for $R = R_1$, we therefore see that

$$\begin{aligned} & b_2[k]\left(R_1, \frac{L_1[k]}{2}, L_2[k]\right) \\ = & \left[\frac{1}{2}\left(R_1 - \log_2\left(\frac{\lambda_1 \frac{L_1[k]}{2}}{\lambda_2 L_2[k]}\right)\right)\right] \\ = & \left[\frac{1}{2}\left((R_1+1) - \log_2\left(\frac{\lambda_1 L_1[k]}{\lambda_2 L_2[k]}\right)\right)\right] \end{aligned} \tag{5.6.40}$$

Substituting eq. 5.6.40 into eq. 5.6.38 yields

$$\begin{aligned} & b_2[k](R_1+1, L_1[k], L_2[k]) \\ = & \left[\frac{1}{2}\left((R_1+1) - \log_2\left(\frac{\lambda_1 L_1[k]}{\lambda_2 L_2[k]}\right)\right)\right] \end{aligned}$$

The above expression for $b_2[k](R_1+1, L_1[k], L_2[k])$ agrees with eq. 5.2.12 for $R = R_1 + 1$. So Lemma 5.2.1 holds for this sub-case.

- If $\lambda_1 L_1[k] < \lambda_2 L_2[k]$, algorithm 5.2.1 assigns the first bit to $L_2[k]$. So

$$\begin{aligned} & b_2[k]\left(R_1+1, L_1[k], L_2[k]\right) \\ = & 1 + b_2[k]\left(R_1, L_1[k], \frac{L_2[k]}{2}\right) \end{aligned}$$

We can compute $b_2[k]\left(R_1, L_1[k], \frac{L_2[k]}{2}\right)$ in a similar manner to show that $b_2[k](R_1+1, L_1[k], L_2[k])$ satisfies eq. 5.2.12 with $R = R_1 + 1$.

Because Lemma 5.2.1 holds for both sub-cases, it holds for $\frac{\lambda_2}{\lambda_1} 2^{-(R_1+1)-1} < \gamma[k] < \frac{\lambda_2}{\lambda_1} 2^{(R_1+1)+1}$. Since Lemma 5.2.1 holds for all three cases on $\gamma[k]$, it must hold for $R = R_1 + 1$. So by mathematical induction we know the lemma is true for all $R \geq 1$. $\diamondsuit$

Proof of Lemma 5.2.2: From equation 5.2.12, there are three kinds of bit

assignment decisions. We now consider the evolution of the side length ratio $\gamma[k] = \frac{L_1[k]}{L_2[k]}$ under each decision.

- **Case 1:** In this case,

$$\gamma[k] \geq \frac{\lambda_2}{\lambda_1} 2^{R+1}.$$

By the decision in eq. 5.2.12, the side length ratio is updated as

$$\begin{aligned} \gamma[k+1] &= \frac{\frac{\lambda_1}{2^R} L_1[k] + M}{\lambda_2 L_2[k] + M} \\ &\leq \frac{\frac{\lambda_1}{2^R} L_1[k]}{\lambda_2 L_2[k]} \\ &= \frac{\lambda_1}{\lambda_2 2^R} \gamma[k] \end{aligned}$$

where the inequality comes from $\frac{\lambda_1}{2^R} L_1[k] > \lambda_2 L_2[k]$ and $M \geq 0$. Because $\rho = \frac{\lambda_1 \lambda_2}{2^R} < 1$ and $\lambda_2 > 1$, we know $\frac{\lambda_1}{\lambda_2 2^R} < 1$ and $\gamma[k]$ exponentially decreases. Note that although $\gamma[k]$ decreases, it is still true that $\gamma[k+1] > \frac{\lambda_2}{\lambda_1} 2^{-R-1}$.

- **Case 2:** In this case,

$$\gamma[k] \leq \frac{\lambda_2}{\lambda_1} 2^{-R-1}.$$

Similarly to the above case, we see that

$$\begin{aligned} \gamma[k+1] &= \frac{\lambda_1 L_1[k] + M}{\frac{\lambda_2}{2^R} L_2[k] + M} \\ &\geq \frac{\lambda_1 2^R}{\lambda_2} \gamma[k] \end{aligned}$$

$\gamma[k]$ therefore increases exponentially. It is also true that $\gamma[k+1] < \frac{\lambda_2}{\lambda_1} 2^{R+1}$.

- **Case 3:** In this case

$$\frac{\lambda_2}{\lambda_1}2^{-R-1} < \gamma[k] < \frac{\lambda_2}{\lambda_1}2^{R+1}$$

By the bit assignment policy in eq. 5.2.12, we obtain

$$0.5 \leq \frac{\frac{\lambda_1}{2^{b_1[k]}}L_1[k]}{\frac{\lambda_2}{2^{b_2[k]}}L_2[k]} < 2 \tag{5.6.41}$$

Considering the update rule $L_i[k+1] = \frac{\lambda_i}{2^{b_i[k]}}L_i[k] + M$ $(i = 1, 2)$, eq. 5.6.41 yields

$$0.5 < \gamma[k+1] < 2 \tag{5.6.42}$$

The assumptions on λ_1, λ_2 and R guarantee that

$$\frac{\lambda_2}{\lambda_1}2^{-R-1} < 0.5 < 2 < \frac{\lambda_2}{\lambda_1}2^{R+1} \tag{5.6.43}$$

So if the quantized system enters into case 3, it will stay there forever.

Because $\gamma[k]$ exponentially increases in case 2 and decreases in case 1, it must eventually enter into case 3 after a finite number of (k_0) steps. As noted above, the system remains in case 3 once it enters it, thereby assuring that the bound in equation 5.6.42 is eventually achieved. The proof is completed. $\diamondsuit$

5.6.2 Proofs of the lemmas and the theorems in section 5.3

Proof of Lemma 5.3.1:

We use P^* and $P^{(k)*}$ to denote the optimal performance of problem 5.3.16 and the kth subproblem in equation 5.3.17 respectively.

It is straightforward to see that

$$\begin{aligned}
&\min_{\{b_1[k],b_2[k]\}_{k=0}^{N-1}} \sum_{k=1}^{N} (L_1^2[k] + L_2^2[k]) \\
\geq\ & \sum_{k=1}^{N} \min_{\{b_1[j],b_2[j]\}_{j=0}^{N-1}} (L_1^2[k] + L_2^2[k]) \qquad (5.6.44) \\
=\ & \sum_{k=1}^{N} \min_{\{b_1[j],b_2[j]\}_{j=0}^{k-1}} (L_1^2[k] + L_2^2[k]) \qquad (5.6.45)
\end{aligned}$$

The equality in eq. 5.6.45 comes from the fact that $L_1[k]$ and $L_2[k]$ are independent of $\{b_1[j], b_2[j]\}_{j=k}^{N-1}$ due to the causal updating rule in eq. 5.2.9. Note that all *min* operations in the above equations are performed under the constraint of $b_1[j] + b_2[j] = R$ $(j = 0, \cdots, N-1)$. Considering the definitions of P^* and $P^{(k)*}$, eq. 5.6.44 and 5.6.45 can be rewritten into

$$P^* \geq \sum_{k=1}^{N} P^{(k)*} \qquad (5.6.46)$$

As stated in Lemma 5.3.1, $\mathbf{b}^{(k-1)} \subset \mathbf{b}^{(k)}$ $(k = 2, \cdots, N)$. So the performance of the kth problem in eq. 5.3.17 under $\mathbf{b}^{(N)}$ is

$$L_1^2[k] + L_2^2[k] = P^{(k)*} \qquad (5.6.47)$$

Summing eq. 5.6.47 for $k = 1, \cdots, N$ yields

$$\sum_{k=1}^{N} L_1^2[k] + L_2^2[k] = \sum_{k=1}^{N} P^{(k)*} \qquad (5.6.48)$$

Because $\mathbf{b}^{(N)}$ satisfies the constraint of problem 5.3.16, i.e. $b_1^{(N)}[k] + b_2^{(N)}[k] = R$ $(k = 0, \cdots, N-1)$, $\mathbf{b}^{(N)}$ is a feasible solution to problem 5.3.16. By eq. 5.6.48,

the performance of problem 5.3.16 under $\mathbf{b}^{(N)}$ is $\sum_{k=1}^{N} P^{(k)*}$. By the optimality of P^*, we obtain

$$P^* \leq \sum_{k=1}^{N} P^{(k)*} \tag{5.6.49}$$

Combining eq. 5.6.46 and 5.6.49 yields

$$P^* = \sum_{k=1}^{N} P^{(k)*} \tag{5.6.50}$$

By the feasibility of $\mathbf{b}^{(N)}$ and eq. 5.6.48 and 5.6.50, we know $\mathbf{b}^{(N)}$ solves the original problem 5.3.16. $\diamondsuit$

Proof of Lemma 5.3.2:

The performance index in problem 5.3.18 is the summation of two terms, $\left(\frac{\lambda_1^k}{2^{s_1[k]}} L_1[0]\right)^2$ $(= L_1^2[k])$ and $\left(\frac{\lambda_2^k}{2^{s_2[k]}} L_2[0]\right)^2$ $(= L_2^2[k])$. We know the product of the two terms is independent of $s_1[k], s_2[k]$ due to the constraint $s_1[k] + s_2[k] = kR$.

$$L_1^2[k] L_2^2[k] = \left(\frac{\lambda_1^k \lambda_2^k}{2^{kR}} L_1[0] L_2[0]\right)^2 \tag{5.6.51}$$

Apply lemma 5.6.1 to $L_1^2[k] + L_2^2[k]$ with eq. 5.6.51 considered, we get

$$L_1^2[k] + L_2^2[k] \quad = \quad 2Cg(2|\log_2(L_1[k]/L_2[k])|) \tag{5.6.52}$$

where $C = \frac{\lambda_1^k \lambda_2^k}{2^{kR}} L_1[0] L_2[0]$. In order to minimize $L_1^2[k] + L_2^2[k]$, we have to minimize $|\log_2(L_1[k]/L_2[k])|$, i.e. keeping $L_1[k]$ and $L_2[k]$ as balanced as possible. By the

expression of $L_i[k] = \frac{\lambda_i^k}{2^{s_i[k]}} L_i[0]$ $(i = 1, 2)$, we know

$$\begin{aligned} & \log_2(L_1[k]/L_2[k]) \\ = \; & \log_2\left(\frac{\lambda_1^k L_1[0]}{\lambda_2^k L_2[0]}\right) - (s_1[k] - s_2[k]) \\ = \; & \log_2\left(\frac{\lambda_1^k L_1[0]}{\lambda_2^k L_2[0]}\right) - kR + 2s_2[k] \end{aligned}$$

The second equality shown above comes from the constraint $s_1[k] + s_2[k] = kR$. $s_2[k]$ is an integer between 0 and Rk. The minimization of $|\log_2(L_1[k]/L_2[k])|$ may be formally expressed as

$$\begin{aligned} & \min_{s_2[k]} \left|\log_2\left(\frac{\lambda_1^k L_1[0]}{\lambda_2^k L_2[0]}\right) - kR + 2s_2[k])\right| \\ & s.t. \qquad s_2[k] \in \{0, 1, \cdots, kR\} \end{aligned} \tag{5.6.53}$$

It is straightforward to show that the solution to optimization 5.6.53 is exactly eq. 5.3.21. By the strictly increasing property of $g(\alpha)$ $(\alpha \geq 0)$ and eq. 5.6.52, we know $s_2[k]$ in eq. 5.3.21, together with $s_1[k]$ in eq. 5.3.20, solves problem 5.3.18. $\diamondsuit$

Proof of Lemma 5.3.3:

$\{b_1[k], b_2[k]\}_{k=0}^{N-1}$ is generated by DBAP and $s_i[k]$ is defined as

$$s_i[k] = \sum_{j=0}^{k-1} b_i[j], i = 1, 2 \tag{5.6.54}$$

We will prove Lemma 5.3.3 by showing that $s_2[k]$ defined in eq. 5.6.54 satisfies eq. 5.3.21. This result will be established by using mathematial induction on k.

When $k = 1$, $s_2[k] = b_2[k-1]$ by the definition of $s_2[k]$. Eq. 5.3.21 (for $s_2[k]$) and 5.2.12 (for $b_2[k-1]$) are really the same. So Lemma 5.3.3 holds for $k = 1$.

Suppose $s_2[k-1]$ satisfies eq. 5.3.21. We will prove $s_2[k]$ also satisfies eq. 5.3.21.

By eq. 5.3.21, the decision on $s_2[k]$ is categorized into three cases based on $\gamma[0] = \frac{L_1[0]}{L_2[0]}$.

Case 1: $\gamma[0] \geq \frac{\lambda_2^k}{\lambda_1^k} 2^{kR+1}$

Under this situation, we get

$$\begin{aligned} \frac{\lambda_1^{k-1}}{2^{(k-1)R+1}} L_1[0] &\geq \lambda_2^{k-1} L_2[0] \frac{2^R \lambda_2}{\lambda_1} \\ &> \lambda_2^{k-1} L_2[0] \end{aligned}$$

where the last inequality comes from $\frac{2^R \lambda_2}{\lambda_1} > 1$. By assumption, $s[k-1]$ satisfies eq. 5.3.21. So

$$s_2[k-1] = 0 \tag{5.6.55}$$

Then we obtain

$$L_1[k-1] = \frac{\lambda_1^{k-1}}{2^{(k-1)R}} L_1[0] \tag{5.6.56}$$

$$L_2[k-1] = \lambda_2^{k-1} L_2[0] \tag{5.6.57}$$

We can verify that $\frac{\lambda_1}{2^{R+1}} L_1[k-1] \geq \lambda_2 L_2[k-1]$. Therefore DBAP yields $b_2[k-1] = 0$ and

$$s_2[k] = s_2[k-1] + b_2[k-1] = 0 \tag{5.6.58}$$

The above result on $s_2[k]$ satisfies eq. 5.3.21.

Case 2: $\gamma[0] \leq \frac{\lambda_2^k}{\lambda_1^k} 2^{-kR-1}$

We can similarly prove $s[k]$ satisfies eq. 5.3.21 as we did for the case $\gamma[0] \geq \frac{\lambda_2^k}{\lambda_1^k} 2^{kR+1}$.

Case 3: $\frac{\lambda_2^k}{\lambda_1^k} 2^{-kR-1} < \gamma[0] < \frac{\lambda_2^k}{\lambda_1^k} 2^{kR+1}$

First we prove it is **impossible** that

$$\frac{\lambda_1}{2^{R+1}} L_1[k-1] \geq \lambda_2 L_2[k-1] \tag{5.6.59}$$

Suppose eq. 5.6.59 holds. Substituting the expressions of $L_1[k-1]$ ($L_1[k-1] = \frac{\lambda_1^{k-1}}{2^{s_1[k-1]}} L_1[0]$) and $L_2[k-1]$ ($L_2[k-1] = \frac{\lambda_2^{k-1}}{2^{s_2[k-1]}} L_2[0]$) into eq. 5.6.59 yields

$$\gamma[0] = \frac{L_1[0]}{L_2[0]} \geq \frac{\lambda_2^k}{\lambda_1^k} 2^{R+1+s_1[k-1]-s_2[k-1]} \tag{5.6.60}$$

Combining the requirement $\gamma[0] < \frac{\lambda_2^k}{\lambda_1^k} 2^{kR+1}$ with the above bound produces

$$R + 1 + s_1[k-1] - s_2[k-1] < kR + 1 \tag{5.6.61}$$

Considering $s_1[k-1] + s_2[k-1] = (k-1)R$, we get

$$s_2[k-1] > 0 \tag{5.6.62}$$

i.e. side L_2 gets at least one bit among the total of $(k-1)R$ ones. Suppose side L_2 gets the first bit at $k = k_1$ ($k_1 \leq k-1$). By algorithm 5.2.1, the decision on $b_1[j]$ and $b_2[j]$ aims to balance $L_1[j+1]$ and $L_2[j+1]$, which guarantees that

$$\frac{L_1[j]}{L_2[j]} \leq 2, \forall j \geq k_1 \tag{5.6.63}$$

The above equation certainly holds for $j = k-1$, i.e.

$$\frac{L_1[k-1]}{L_2[k-1]} \leq 2 \tag{5.6.64}$$

Thus

$$\begin{aligned} \frac{\lambda_1 L_1[k-1]}{\lambda_2 L_2[k-1]} &\leq 2\frac{\lambda_1}{\lambda_2} && (5.6.65) \\ &< 2^{R+1} && (5.6.66) \end{aligned}$$

The above result contradicts eq. 5.6.59 ! So eq. 5.6.59 is **impossible**.

Second we can similarly prove it is also **impossible** that

$$\frac{\lambda_1}{2^{-R-1}} L_1[k-1] \leq \lambda_2 L_2[k-1] \tag{5.6.67}$$

Based on the impossibility of eq. 5.6.59 and 5.6.67 and the decision rule in eq. 5.2.12, we get

$$b_2[k-1] = \left[\frac{1}{2}\left(R - \log_2\left(\frac{\lambda_1 L_1[k-1]}{\lambda_2 L_2[k-1]}\right)\right)\right] \tag{5.6.68}$$

Substituting the expressions of $L_1[k-1]$ and $L_2[k-1]$ into the above equation yields

$$\begin{aligned} b_2[k-1] &= \left[\frac{1}{2}\left(R - \log_2\left(\frac{\lambda_1 \frac{\lambda_1^{k-1}}{2^{s_1[k-1]}} L_1[0]}{\lambda_2 \frac{\lambda_2^{k-1}}{2^{s_2[k-1]}} L_2[0]}\right)\right)\right] \\ &= \left[\frac{1}{2}\left(R + s_1[k-1] - s_2[k-1] - \log_2\left(\frac{\lambda_1^k L_1[0]}{\lambda_2^k L_2[0]}\right)\right)\right] \end{aligned}$$

By the identity $s_1[k-1] = (k-1)R - s_2[k-1]$, the above result can be simplified

into

$$
\begin{aligned}
& b_2[k-1] \\
&= \left[0.5\left(kR - \log_2\left(\frac{\lambda_1^k L_1[0]}{\lambda_2^k L_2[0]}\right)\right)\right] - s_2[k-1]
\end{aligned}
$$

Considering the definition of $s_2[k]$ in eq. 5.6.54, we obtain

$$
\begin{aligned}
s_2[k] &= s_2[k-1] + b_2[k-1] \\
&= \left[\frac{1}{2}\left(kR - \log_2\left(\frac{\lambda_1^k L_1[0]}{\lambda_2^k L_2[0]}\right)\right)\right]
\end{aligned}
$$

Therefore $s_2[k]$ satisfies eq. 5.3.21.

In summary, $s_2[0]$ satisfies eq. 5.3.21. If $s_2[k-1]$ satisfies eq. 5.3.21, then $s_2[k]$ also satisfies equation 5.3.21. So by mathematical induction method, we can guarantee that $s_2[k]$ satisfies eq. 5.3.21 for all k and the proof is complete. $\diamondsuit$

Proof of Theorem 5.3.1:

Denote the optimal performance of problems 5.3.16, 5.3.17 and 5.3.18 as P^*, $P^{(k)*}$ and $P_s^{(k)*}$ respectively. By the relaxation relationship among them, P^*, $P^{(k)*}$ and $P_s^{(k)*}$ satisfy the following equations.

$$
P^* \geq \sum_{k=1}^{N} P^{(k)*} \tag{5.6.69}
$$

$$
P^{(k)*} \geq P_s^{(k)*} \tag{5.6.70}
$$

We will prove the equalities in eq. 5.6.69 and 5.6.70 hold.

By Lemmas 5.3.2 and 5.3.3, we know the solution to problem 5.3.18, $\mathbf{s}^{(k)}$,

satisfies

$$s_i[k] = \sum_{j=0}^{k-1} b_i^{(k)}[j], i = 1, 2 \tag{5.6.71}$$

where $\mathbf{b}^{(k)} = \{b_1^{(k)}[j], b_2^{(k)}[j]\}_{j=0}^{k-1}$ is generated by DBAP. Of course $\mathbf{b}^{(k)}$ satisfies the constraint $b_1^{(k)}[j] + b_2^{(k)}[j] = R$ $(j = 0, \cdots, k-1)$. So $\mathbf{b}^{(k)}$ is a feasible solution to problem 5.3.17. The performance of problem 5.3.17 under $\mathbf{b}^{(k)}$ is $P_s^{(k)*}$ because problems 5.3.17 and 5.3.18 have the same performance index and the optimal performance of problem 5.3.18, $P_s^{(k)*}$, is achieved by $s_i[k]$ given in eq. 5.6.71. Therefore the equality in eq. 5.6.70 holds. The solution to problem 5.3.17 is $\mathbf{b}^{(k)}$ which is generated by DBAP.

By DBAP algorithm in 5.2.1, we know $b_i^{(k)}[j]$ $(i = 1, 2)$ is totally determined by the initial condition, $L_1[0]$ and $L_2[0]$, and the time j. $b_i^{(k)}[j]$ is independent of k. So

$$\mathbf{b}^{(k-1)} \subset \mathbf{b}^{(k)}, k = 2, \cdots, N \tag{5.6.72}$$

By Lemma 5.3.1, the equality in eq. 5.6.69 holds and the solution to problem 5.3.16 is $\mathbf{b}^{(N)}$. Because $\mathbf{b}^{(N)}$ is generated by DBAP, DBAP is the optimal policy. $\diamondsuit$

5.6.3 Proofs of the propositions in section 5.4

From the proof of Lemma 5.2.2 and especially in light of equation 5.6.41, we can directly obtain the following balancing result. Its proof is omitted here.

Corollary 5.6.1 *Under DBAP, there exists a finite $k_0 > 0$ such that*

$$0.5 \leq \frac{\frac{\lambda_1}{2^{b_1[k]}} L_1[k]}{\frac{\lambda_2}{2^{b_2[k]}} L_2[k]} < 2, \forall k \geq k_0 \tag{5.6.73}$$

Preliminary lemmas: The following lemmas are used in proving the three main propositions in this paper. These lemmas characterize some important properties of $p[k] = L_1[k]L_2[k]$ that are used in the following proofs.

Lemma 5.6.2 *$p[k]$ is governed by*

$$p[k+1] = \rho p[k] + 2\sqrt{\rho} M \sqrt{p[k]} g(\alpha[k]) + M^2 \tag{5.6.74}$$

where

$$\alpha[k] = \left| \log_2 \left(\frac{\lambda_1 L_1[k]}{2^{b_1[k]}} \frac{2^{b_2[k]}}{\lambda_2 L_2[k]} \right) \right| \tag{5.6.75}$$

Lemma 5.6.2 comes from the definition of $p[k]$ in eq. 5.6.32 and the updating rule of $L_i[k]$ in eq. 5.2.9. Its proof is omitted.

By eq. 5.6.74, $\{p[k]\}$ is determined by the sequence $\{\alpha[k]\}$. We now consider a special sequence $\{\alpha[k] = c\}$, where c is a positive constant. The corresponding solution to eq. 5.6.74 is denoted as $\{p_c[k]\}$. We have the following limit on $p_c[k]$.

Lemma 5.6.3

$$\lim_{k \longrightarrow \infty} p_c[k] = p_c \tag{5.6.76}$$

where p_c is the positive solution to the following equation with respect to z.

$$z = \rho z + 2\sqrt{\rho} M g(c) \sqrt{z} + M^2 \tag{5.6.77}$$

Proof: Because $\rho < 1$, eq. 5.6.77 has a unique positive solution, denoted as z_0.

Define a function

$$f(z) = \rho z + 2\sqrt{\rho} M g(c) \sqrt{z} + M^2 \tag{5.6.78}$$

The updating rule of $p[k]$ under the give constant $\alpha[k]$ can be rewritten into

$$p_c[k+1] = f(p_c[k]) \tag{5.6.79}$$

Because $f(z) - z > 0$ for $z < z_0$, $f(z_0) = z_0$ and $f(z) - z < 0$ for $z > z_0$, we know $\lim_{k \longrightarrow \infty} p_c[k]$ exists and is equal to z_0. $\diamondsuit$

For any finite c, the limit in eq. 5.6.76, p_c is finite. Such finiteness can be used to establish the following ordering relationship.

Lemma 5.6.4 *If $c > \alpha_1[k] \geq \alpha_2[k] \geq 0, \forall k$, then*

$$\lim_{N \longrightarrow \infty} \sup\{p_{\alpha_1}[k] - p_{\alpha_2}[k]\}_{k=N}^{\infty} \geq 0 \tag{5.6.80}$$

Proof: By the updating rule in eq. 5.6.74, we can easily establish the following bounds.

$$M^2 \leq p_{\alpha_2}[k] \leq p_{\alpha_1}[k] < p_c[k], k \geq 1 \tag{5.6.81}$$

where $p_c[k]$ is similarly defined as in Lemma 5.6.3. By Lemma 5.6.3, we know $\lim_{k \longrightarrow \infty} p_c[k]$ exists and is finite. By eq. 5.6.81, we know $\lim_{N \longrightarrow \infty} \sup\{p_{\alpha_1}[k] - p_{\alpha_2}[k]\}_{k=N}^{\infty}$ exists. Because $p_{\alpha_1}[k] \geq p_{\alpha_2}[k]$,

$$\lim_{N \longrightarrow \infty} \sup\{p_{\alpha_1}[k] - p_{\alpha_2}[k]\}_{k=N}^{\infty} \geq 0. \qquad \diamondsuit$$

We now use Lemma 5.6.4 to derive lower and upper bounds on $p[k]$ for bounded $\alpha[k]$.

Corollary 5.6.2 *If* $0 \leq \alpha[k] \leq 1$, *then*

$$\lim_{N \longrightarrow \infty} \sup\{p[k]\}_{k=N}^{\infty} \geq p_l \tag{5.6.82}$$

$$\lim_{N \longrightarrow \infty} \sup\{p[k]\}_{k=N}^{\infty} \leq p_u \tag{5.6.83}$$

where p_l *and* p_u *are the positive solutions to the following equations.*

$$p_l = \rho p_l + 2\sqrt{\rho} M \sqrt{p_l} + M^2 \tag{5.6.84}$$

$$p_u = \rho p_u + 2\sqrt{\rho} M g(1) \sqrt{p_u} + M^2 \tag{5.6.85}$$

Corollary 5.6.2 comes directly from Lemma 5.6.4 with Lemma 5.6.3 considered. Its proof is omitted.

Proof of Proposition 5.4.1: Applying Lemma 5.6.1 to $L_1^2[k] + L_2^2[k]$ yields

$$\begin{aligned} & L_1^2[k] + L_2^2[k] \\ = \; & 2p[k] g\left(2\left|\log_2\left(\frac{L_1[k]}{L_2[k]}\right)\right|\right) \\ \geq \; & 2p[k] \end{aligned} \tag{5.6.86}$$

The fact $g\left(2\left|\log_2\left(\frac{L_1[k]}{L_2[k]}\right)\right|\right) \geq 1$ is utilized to derive the above inequality. Compute the ultimate bound on eq. 5.6.86, we get

$$J = \lim_{N \longrightarrow \infty} \sup\{L_1^2[k] + L_2^2[k]\}_{k=N}^{\infty} \tag{5.6.87}$$

$$\geq 2 \lim_{N \longrightarrow \infty} \sup\{p[k]\}_{k=N}^{\infty} \tag{5.6.88}$$

where J is the performance achieved by some bit assignment policy. Applying Corollary 5.6.2, especially eq. 5.6.82, to eq. 5.6.88 yields

$$J \geq 2p_l \tag{5.6.89}$$

where p_l is defined in eq. 5.6.84. Note that eq. 5.6.84 and 5.4.27 are the same. So

$$p_l = p_0 \tag{5.6.90}$$

Eq. 5.6.89 holds for J under any bit assignment policy. It certainly holds for the optimal bit assignment policy which yields the performance of J^*. So

$$\begin{aligned} J^* &\geq 2p_l \\ &= 2p_0 \\ &= \underline{J} \quad \diamondsuit \end{aligned}$$

Proof of Proposition 5.4.2: Under the specified static policy, $L_i[k]$ $(i = 1, 2)$ is evolved as

$$\begin{aligned} L_i[k+1] &= \frac{\lambda_i}{2^{b_i[k]}} L_i[k] + M \\ &= \frac{\lambda_i}{2^{b_i}} L_i[k] + M \end{aligned} \tag{5.6.91}$$

From eq. 5.6.91, we know $\lim_{k \longrightarrow \infty} L_i[k]$ $(i = 1, 2)$ exists and is equal to L_i, which is defined as

$$L_i = \frac{M}{1 - \frac{\lambda_i}{2^{b_i}}} \tag{5.6.92}$$

Due to the configuration of b_1 and b_2, we can verify that $\frac{\lambda_1}{2^{b_1}} = \frac{\lambda_2}{2^{b_2}}$. Denote $c = \frac{\lambda_1}{2^{b_1}}$. Then $\rho = \lambda_1\lambda_2/2^R = c^2$. Because $\rho < 1$, $c < 1$. By eq. 5.6.92, we obtain that

$$L_1 = L_2 = \frac{M}{1-c} \tag{5.6.93}$$

By eq. 5.6.93 and $\rho = c^2$, we can verify that L_1^2 is a solution to eq. 5.4.27. Because p_0 is also the solution to eq. 5.4.27 and eq. 5.4.27 has a unique solution due to $\rho < 1$, we get

$$L_1^2 = p_0 \tag{5.6.94}$$

The performance under the specified static policy is

$$\begin{aligned} J_{\text{static}} &= \lim_{N \longrightarrow \infty} \sup\{L_1^2[k] + L_2^2[k]\}_{k=N}^{\infty} \\ &= L_1^2 + L_2^2 \\ &= 2L_1^2 \end{aligned} \tag{5.6.95}$$

Substituting eq. 5.6.94 into eq. 5.6.95 yields

$$J_{\text{static}} = 2p_0 = \underline{J} \tag{5.6.96}$$

So the lower bound $\underline{J}$ is achieved by the given static bit assignment policy. Clearly, $\underline{J} = J_{\text{static}} \geq J^*$ (otherwise J^* is not optimal). From proposition 5.4.1, we know that $J^* \geq \underline{J}$, so we can conlude that $J^* = \underline{J} = J_{\text{static}}$. $\diamondsuit$

Proof of Proposition 5.4.3: By the updating rule of $L_i[k]$ in eq. 5.2.9, we obtain

$$\begin{aligned} & L_1^2[k+1] + L_2^2[k+1] \\ = & \left(\frac{\lambda_1}{2^{b_1[k]}} L_1[k]\right)^2 + \left(\frac{\lambda_2}{2^{b_2[k]}} L_2[k]\right)^2 \\ & +2M\left(\frac{\lambda_1}{2^{b_1[k]}} L_1[k] + \frac{\lambda_2}{2^{b_2[k]}} L_2[k]\right) + 2M^2 \\ = & 2\rho p[k] g\left(2\alpha[k]\right) + 4M\sqrt{\rho p[k]}\, g(\alpha[k]) + 2M^2 \end{aligned} \tag{5.6.97}$$

where $\alpha[k] = \left|\log_2\left(\frac{\lambda_1 L_1[k]}{2^{b_1[k]}} \frac{2^{b_2[k]}}{\lambda_2 L_2[k]}\right)\right|$. By Corollary 5.6.1, we know $\alpha[k] \leq 1$. By the strictly increasing property of $g(\alpha)$ for $\alpha \geq 0$, we get

$$\begin{cases} g(\alpha[k]) & \leq & g(1) \\ g(2\alpha[k]) & \leq & g(2) \end{cases} \tag{5.6.98}$$

Substituting eq. 5.6.98 into eq. 5.6.97 yields

$$\begin{aligned} & L_1^2[k+1] + L_2^2[k+1] \\ \leq & 2\rho p[k] g(2) + 4M\sqrt{\rho}\sqrt{p[k]}\, g(1) + 2M^2 \end{aligned} \tag{5.6.99}$$

Under DBAP, the performance achieved is J_{DBAP}. Computing the ultimate bound on both sides of eq. 5.6.99 yields

$$\begin{aligned} J_{\text{DBAP}} & = \lim_{N\longrightarrow\infty} \sup\{L_1^2[k+1] + L_2^2[k+1]\}_{k=N}^{\infty} \\ & \leq 2\rho g(2) \lim_{N\longrightarrow\infty} \sup\{p[k]\}_{k=N}^{\infty} \\ & \quad +4M\sqrt{\rho} g(1) \sqrt{\lim_{N\longrightarrow\infty} \sup\{p[k]\}_{k=N}^{\infty}} + 2M^2 \end{aligned}$$

Applying Corollary 5.6.2 to the above equality yields

$$J_{\text{DBAP}} \leq \overline{J} = 2\rho g(2)p_u + 4M\sqrt{\rho}g(1)\sqrt{p_u} + 2M^2$$

Clearly $J^* \leq J_{DBAP}$, so we can conclude $J^* \leq \overline{J}$. $\diamondsuit$

CHAPTER 6

CONCLUSIONS

This thesis studies control systems with limited feedback information. Two types of feedback limitations, dropout and quantization, are studied. We analyze the effect of dropout and quantization on stability and performance of control systems and develop synthesis methods to improve system performance.

We consider two dropout models, independent and identically distributed (i.i.d.) processes and Markov chains. For a control system with i.i.d. dropouts, we

1. Obtained the closed-form expression of the output's power spectral density (PSD).

2. Proposed a linear time-invariant (LTI) system which is equivalent to the original control system in the sense of the same stability condition and the same output's PSD. We show that we can do synthesis through the equivalent LTI system.

3. Designed the optimal linear dropout compensator which takes a form of optimization among a group of LQG controllers.

For a control system with dropouts governed by a Markov chain, we

1. Proved the necessity of the stability condition in theorem 3.2.1. That condition is more convenient than the other testing conditions.

2. Computed the output power.

3. Formulated the optimal dropout policy problem. That problem is sovled through a gradient method and a branch-and-bound method. The achieved optimal dropout policy is implemented as a new QoS (quality of service) constraint in real-time resource scheduling. Simulations show that the new QoS constrant can yield good real-time control performance [49] [50]. We did some hardware experiments to verify the achieved optimal dropout policy.

4. Extended the achieved performance computation and optimal dropout policy results to a distributed control system.

For a control system with quantization, we

1. Derived the minimum constant bit rate to guarantee stability (asymptotic stability for noise-free quantized systems and BIBO stability for quantized systems with bounded noise). We proposed a *dynamic bit assignment policy* (DBAP), which can achieve the minimum constant bit rate for stability.

2. Formulated the optimal quantization policy problem. For a noise-free quantized system, the optimal quantization policy is obtained, which is the previously proposed DBAP with slight modification. For a quantized system with bounded noise, both a lower bound and an upper bound on the optimal performance are proposed. The upper bound can always be achieved by the proposed DBAP. So DBAP may be viewed as a sub-optimal quantization policy with known performance gap.

APPENDIX A

MATHEMATICAL PRELIMINARIES

A.1 Convergence of stochastic processes

Let x be a random vector and let $\mathbf{E}[x]$ denote the expectation of x. A real-valued discrete-time stochastic process $x = \{x[n]\}$ is convergent in the mean square sense if there exists a random vector $\overline{x}$ such that

$$\lim_{n\to\infty} \mathbf{E}\left[(x[n]-\overline{x})^T(x[n]-\overline{x})\right] = 0.$$

It can be shown [68] that a random process $x = \{x[n]\}$ is convergent in the mean square sense if and only if

$$\lim_{n\to\infty} \sup_{m\geq n} \mathbf{E}\left[(x[m]-x[n])^T(x[m]-x[n])\right] = 0.$$

A.2 Wide sense stationary stochastic processes

A random process $x = \{x[n]\}$ is said to be *wide sense stationary* (WSS) if its mean is constant and its covariance is shift invariant. In other words, $\{x[n]\}$ is WSS if and only if $\mathbf{E}\left[x[n]\right] = \text{constant} = \mu_x$ and $\mathbf{E}\left[(x[k]-\mu_x)(x[l]-\mu_x)^T\right] = \mathbf{E}\left[(x[k+n]-\mu_x)(x[l+n]-\mu_x)^T\right]$ for arbitrary n. Obviously if $\{x[n]\}$ is WSS, $\mathbf{E}\left[x[k]x^T[l]\right] = \mathbf{E}\left[x[k+n]x^T[l+n]\right]$ for arbitrary n. The mean of the WSS process $x = \{x[n]\}$ is denoted as μ_x and the correlation matrix of this process is

denoted as $R_{xx}(m) = \mathbf{E}\left[x[n+m]x[n]^T\right]$, where n can be arbitrarily choosen because of the wide sense stationarity of x.

If $x = \{x[n]\}$ is WSS, then the *power spectral density* (PSD) of x is the Fourier transform of its covariance function [1],

$$S_{xx}(e^{j\omega}) = \sum_{k=-\infty}^{\infty} R_{xx}(m)e^{-j\omega k}, \tag{A.2.1}$$

Given two WSS processes $x = \{x[n]\}$ and $y = \{y[n]\}$, the cross-correlation function $R_{xy}(m) = \mathbf{E}\left[x[n+m]y[n]^T\right]$ is also shift invariant with respect to n. The cross spectral density $S_{xy}(e^{j\omega})$ is obtained by taking the Fourier transform of the cross-correlation function. In this proposal, we often drop the explicit dependence on $e^{j\omega}$ to improve the readability. In this case, $S_{xx}(e^{j\omega})$ will be simply denoted as S_{xx}. The complex conjugate transpose of S_{xx} is denoted as S_{xx}^* and satisfies the equation $S_{xx}^*(e^{j\omega}) = S_{xx}^T(e^{-j\omega})$.

Some of the technical proofs in this thesis use a single-sided Fourier transform. Given a WSS process x, the positive and negative single-sided power spectral densities are defined by the equations

$$\begin{aligned} S_{xx}^+(e^{j\omega}) &= \sum_{m=1}^{\infty} R_{xx}[m]e^{-jm\omega}, \\ S_{xx}^-(e^{j\omega}) &= \sum_{m=-\infty}^{-1} R_{xx}[m]e^{-jm\omega}. \end{aligned}$$

Since $R_{xx}(m) = R_{xx}^T(-m)$, it is straightforward to see that

$$S_{xx}^+ = \left[S_{xx}^-\right]^*. \tag{A.2.2}$$

[1] Most of the WSS processes in this thesis are zero mean, so their covariances (cross-covariances) and correlations (cross-correlations) are equal. Therefore we interchangeably use these terms.

So the power spectral density can be expressed as

$$S_{xx} = S_{xx}^{+} + S_{xx}^{-} + R_{xx}[0]. \tag{A.2.3}$$

The power of a WSS process $x = \{x[n]\}$ is

$$\mathbf{E}[x^T x] = \text{Trace}(R_{xx}[0]). \tag{A.2.4}$$

By equation A.2.1, we obtain

$$\mathbf{E}[x^T x] = \text{Trace}\left(\frac{1}{2\pi}\int_{-\pi}^{\pi} S_{xx}(e^{j\omega})d\omega\right).$$

Under the assumption that the input is white noise with unit variance, the power of an LTI system's output is the square of the H_2 norm of the system [39]. So the output's power can be used to measure the noise attenuation property of a system with noise input.

Ergodicity is an important property of a WSS process. The mean of x, $\mathbf{E}[x[n]]$, is the ensemble average of $x[n]$ (because x is WSS, $\mathbf{E}[x[n]]$ is constant for any n.). The time average of x for n steps is defined as $\hat{\mathbf{E}}[x[n]] = \frac{1}{n}\sum_{i=1}^{n} x[i]$. x is *ergodic* if

$$\lim_{n\longrightarrow\infty} \hat{\mathbf{E}}[x[n]] = \mathbf{E}[x[n]] \quad \text{in the mean square sense.} \tag{A.2.5}$$

It can be shown [60] that a WSS process $x = \{x[n]\}$ is ergodic if

$$\lim_{m\longrightarrow\infty} \mathbf{E}\left[(x[n+m] - \mu_x)(x[n] - \mu_x)^T\right] = 0,$$

where $\mu_x = \mathbf{E}[x[n]]$. When $\mu_x = 0$, the above condition is equivalent to $\lim_{m\longrightarrow\infty} R_{xx}[m] = 0$.

A.3 Kronecker product

Some of the technical proofs in this thesis make use of the *Kronecker product*, $\otimes$ [8]. The Kronecker product of two matrices $A = (a_{ij})_{M\times N}$, $B = (b_{pq})_{P\times Q}$ is defined as

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1N}B \\ a_{21}B & a_{22}B & \cdots & a_{2N}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1}B & a_{M2}B & \cdots & a_{MN}B \end{bmatrix}_{MP\times NQ}.$$

For simplicity, $A \otimes A$ is denoted as $A^{[2]}$ and $A \otimes A^{[n]}$ is denoted as $A^{[n+1]}$ $(n \geq 2)$.

For two vectors x and y, $x \otimes y$ simply rearranges the columns of xy^T into a vector. So for two WSS processes $\{x[n]\}$ and $\{y[n]\}$, $\lim_{n\longrightarrow\infty} \mathbf{E}[x[n] \otimes y[n]] = 0$ if and only if $\lim_{n\longrightarrow\infty} \mathbf{E}\left[x[n]y^T[n]\right] = 0$. It then follows that a zero-mean WSS process $x = \{x[n]\}$ is ergodic if

$$\lim_{m\longrightarrow\infty} \mathbf{E}[x[n+m] \otimes x[n]] = 0.$$

The following property of Kronecker product will be frequently used in the technical proofs,

$$(A_1\ A_2 \cdots A_n) \otimes (B_1\ B_2 \cdots B_n) = (A_1 \otimes B_1)\ (A_2 \otimes B_2) \cdots (A_n \otimes B_n), \quad \text{(A.3.6)}$$

where $A_i, B_i (i = 1, 2, \cdots, n)$ are all matrices with appropriate dimensions.

In computations using Kronecker product, two linear operators, **vec** and **devec**, are used. The **vec** operator transforms a matrix $A = (a_{ij})_{M\times N}$ into the vector

$$\text{vec}(A) = [a_{11}\ a_{21}\cdots a_{M1}\ a_{12}\cdots a_{M2}\cdots a_{1N}\cdots a_{MN}]^T.$$

The **devec** operator inverts the **vec** operator for a square matrix, i.e.

$$devec\,(vec\,(A)) = A \tag{A.3.7}$$

where A is a square matrix.

BIBLIOGRAPHY

1. Example: Modeling an inverted pendulum. *http://www.engin.umich.edu /group/ctm/examples/pend/invpen.html*.

2. B. Anderson and J. Moore, *Optimal control : linear quadratic methods*. Prentice Hall, Englewood Cliffs, NJ (1990).

3. K. Astrom, *Computer-controlled systems : theory and design (third edition)*. Upper Saddle River, N.J., Prentice Hall (1997).

4. K. Astrom and B. Bernhardsson, Comparison of riemann and lebesgue sampling for first order stochastic systems. In *the 41st IEEE Conference on Decision and Control*, pages 2011–2016 (2002).

5. J. Baillieul, Feedback coding for information-based control - operating near the data-rate limit. In *IEEE Conference on Decision and Control*, pages 3229–3236 (2002).

6. P. Bauer, M. Sichitiu and K. Premaratne, Controlling an integrator through data networks: stability in the presence of unknown time-variant delays. In *IEEE International Symposium on Circuits and Systems*, pages 491–494 (1999).

7. O. Beldiman, G. Walsh and L. Bushnell, Predictors for networked control systems. In *American Control Conference*, pages 2347–2351 (2000).

8. R. Bellman, *Introduction to Matrix Analysis*. McGraw-Hill (1960).

9. G. Bernat and A. Burns, Combining (n/m)-hard deadlines and dual priority scheduling. In *the 18th IEEE Real-Time Systems Symposium*, pages 46–57, San Francisco, CA, USA (1997).

10. G. Bernat and R. Cayssials, Guaranteed on-line weakly-hard real-time systems. In *Real-Time Systems Symposium*, pages 25–34 (2001).

11. B. Bhaurucha, *On the stability of randomly varying systems*. Ph.D. thesis, Univ. of California, Berkeley (1961).

12. M. Branicky, S. Phillips and W. Zhang, Scheduling and feedback co-design for networked control systems. In *IEEE Conference on Decision and Control*, Las Vegas, Nevada, USA (2002).

13. R. Brockett, Minimal attention control. In *IEEE Conference on Decision and Control* (1997).

14. R. Brockett and D. Liberzon, Quantized feedback stabilization of linear systems. *IEEE Transactions on Automatic Control*, 45(7): 1279 –1289 (2000).

15. O. Costa, Necessity proof to the stability condition of jump linear systems. In *Personal email communication* (Mar. 2004).

16. O. Costa and M. Fragoso, Stability results for discrete-time linear systems with markovian jumping parameters. *Journal of Mathematical Analysis and Applications*, 179: 154–178 (1993).

17. O. Costa and R. P. Marques, Comments on "stochastic stability of jump linear systems". *IEEE transactions on Automatic Control*, 49(8): 1414 – 1416 (2004).

18. R. D'Andrea, Linear matrix inequalities, multidimensional system optimization, and control of spatially distributed systems: An example. In *American Control Conference*, pages 2713–2717 (1999).

19. D. Delchamps, Stabilizing a linear system with quantized state feedback. *IEEE Transactions on Automatic Control*, 35(8): 916–924 (1990).

20. N. Elia and S. Mitter, Stabilization of linear systems with limited information. *IEEE Transactions on Automatic Control*, 46(9): 1384 –1400 (2001).

21. F. Fagnani and S. Zampieri, Stability analysis and synthesis for scalar linear systems with a quantized feedback. *IEEE Transactions on Automatic Control*, 48(9): 1569–1584 (2003).

22. H. Fang and P. Antsaklis, Convergence rate of quantization error. In *submitted to IEEE Conference on Decision and Control*, Seville, Spain (2005).

23. Y. Fang and K. Loparo, Stochastic stability of jump linear systems. *IEEE Transactions on Automatic Control*, 47(7): 1204–1208 (2002).

24. X. Feng, K. Loparo, Y. Ji and H. Chizeck, Stochastic stability properties of jump linear systems. *IEEE Transactions on Automatic Control*, 37(1): 38–53 (1992).

25. M. Fu, Robust stabilization of linear uncertain systems via quantized feedback. In *IEEE Control and Decision Conference 2003* (2003).

26. P. Gahinet, A. Nemirovski, A. Laub and M. Chilali, *LMI control toolbox: for use with MATLAB*. The MATH works Inc. (1995).

27. R. Gray and D. Neuhoff, Quantization. *IEEE Transactions on Information Theory*, 44(6): 2325–2383 (1998).

28. C. Hadjicostis and R. Touri, Feedback control utilizing packet dropping network links. In *IEEE Conference on Decision and Control*, Las Vegas, Nevada, USA (2002).

29. S. Haykin and M. Moher, *Modern wireless communications*. Pearson Prentice Hall, Upper Saddle Reiver, NJ 07458 (2003).

30. J. Hespanha, A. Ortega and L. Vasudevan, Towards the control of linear systems with minimum bit-rate. In *Proc. of the Int. Symp. on the Mathematical Theory of Networks and Systems* (2002).

31. R. Horn and C. Johnson, *Matrix analysis*. Cambridge University Press (1985).

32. D. Hristu-Varsakelis and P. Kumar, Interrupt-based feedback control over a shared communication medium. In *IEEE Conference on Decision and Control*, Las Vegas, Nevada, USA (2002).

33. H. Ishii and B. Francis, Quadratic stabilization of sampled-data systems with quantization. *Automatica*, 39(10): 1793–1800 (2003).

34. Y. Ji and H. Chizeck, controllability, stabilizability, and continuous-time markovian jump linear quadratic control. *IEEE Transactions on Automatic Control*, 35(7): 777–788 (1990).

35. Y. Ji and H. Chizeck, Jump linear quadratic gaussian control in continuous time. *IEEE Transactions on Automatic Control*, 37(12): 1884–1892 (1992).

36. Y. Ji, H. Chizeck, X. Feng and K. Loparo, Stability and control of discrete-time jump linear systems. *Control Theory and Advanced Technology*, 7(2): 247–270 (1991).

37. A. Kalavade and P. Mogh, A tool for performance estimation of networked embedded end-systems. In *Design Automation Conference*, pages 257–262, San Francisco, CA, USA (1998).

38. G. Koren and D. Shasha, Skip-over: algorithms and complexity for overloaded systems that allow skips. In *Real-Time Systems Symposium*, pages 110–117 (1995).

39. M. Lemmon, *Lecture Notes on Robust Optimal Control*. http://www.nd.edu / ˜ lemmon/courses/ee555/robust-control.pdf (2004).

40. M. Lemmon and Q. Ling, Control system performance under dynamic quantization: the scalar case. In *IEEE Conference on Decision and Control*, Atlantis, Paradise Island, Bahamas (2004).

41. D. Liberzon, On stabilization of linear systems with limited information. *IEEE Transactions on Automatic Control*, 48(2): 304–307 (2003).

42. B. Lincoln and A. Cervin, Jitterbug: a tool for analysis of real-time control performance. In *IEEE Conference on Decision and Control* (2002).

43. Q. Ling and M. Lemmon, Optimal dropout compensator in networked control systems. In *IEEE Conference on Decision and Control*, Hyatt Regency Maui, Hawaii, USA (2003).

44. Q. Ling and M. Lemmon, Soft real-time scheduling of networked control systems with dropouts governed by a markov chain. In *American Control Conference*, Denver, Colorado (2003).

45. Q. Ling and M. Lemmon, Stability of quantized linear systems with bounded noise under dynamic bit assignment. In *IEEE Conference on Decision and Control*, Atlantis, Paradise Island, Bahamas (2004).

46. Q. Ling and M. Lemmon, Optimal dynamic bit assignment in noise-free quantized linear control systems. In *submitted to IEEE Conference on Decision and Control*, Seville, Spain (2005).

47. Q. Ling and M. Lemmon, Performance of quantized linear systems with bounded noise under dynamic bit assignment. In *submitted to IEEE Conference on Decision and Control*, Seville, Spain (2005).

48. Q. Ling and M. Lemmon, Stability of quantized control systems under dynamicc bit assignment. *IEEE Trans. on Automatic Control, also appeared in American Control Conference (2004)*, 50(5) (2005).

49. D. Liu, X. Hu, M. Lemmon and Q. Ling, Firm real-time system scheduling based on a novel qos constraint. In *Real Time Systems Symposium* (2003).

50. D. Liu, X. Hu, M. Lemmon and Q. Ling, Scheduling tasks with markov-chain based constraints. In *17th Euromicro Conference on Real-time Systems* (2005).

51. M. Mariton, *Jump Linear Systems in Automatic Control*. Marcel Dekker, Inc. (1990).

52. L. Montestrue and P. Antsaklis, On the model-based control of networked systems. *Automatica*, 39(10): 1837–1843 (2003).

53. G. Nair and R. Evans, Exponential stabilisability of finite-dimensional linear systems with limited data rates. *Automatica*, 39: 585–593 (2003).

54. G. Nair and R. Evans, Stabilizability of stochastic linear systems with finite feedback data rates. *SIAM Journal of Control and Optimization*, 43(2): 413–436 (2004).

55. J. Nilsson, *Real-time control systems with delays*. Ph.D. thesis, Lund Institute of Technology (1998).

56. G. Quan and X. Hu, Enhanced fixed-priority scheduling with (m,k)-firm guarantee. In *IEEE Real-Time Systems Symposium*, pages 79–88 (2000).

57. P. Ramanathan, Overload management in real-time control applications using (m, k)-firm guarantee. *IEEE Transactions on Parallel and Distributed Systems*, 10: 549–559 (1999).

58. S. Sarma, M. Dahleh and S. Salapaka, On time-varying bit-allocation maintaining input-output stability: a convex parameterization. In *IEEE Conference on Decision and Control*, pages 1430–1435 (2004).

59. P. Seiler and R. Sengupta, Analysis of communication losses in vehicle control problems. In *American Control Conference* (2001).

60. H. Stark and J. Woods, *Probability, Random Processes, and Estimation Theory for Engineers*. Prentice Hall, Upper Saddle River, NJ, second edition (1994).

61. S. Tatikonda, *Control under communication constraints*. Ph.D. thesis, M.I.T. (2000).

62. S. Tatikonda and S. Mitter, Control under communication constraints. *IEEE Transactions on Automatic Control*, 49(7): 1056–1068 (2004).

63. S. Tatikonda, A. Sahai and S. Mitter, Stochastic linear control over a communication channel. *IEEE Transactions on Automatic Control*, 49(9): 1549–1561 (2004).

64. H. Tuan, P. Apkarian and Y. Nakashima, A new lagrangian dual global optimization algorithm for solving bilinear matrix inequalities. *International Journal of Robust and Nonlinear Control*, 10: 561–578 (2000).

65. G. Walsh, O. Beldiman and L. Bushnell, Asymptotic behavior of nonlinear networked control systems. *IEEE Transactions on Automatic Control*, 46: 1093–1097 (2001).

66. G. Walsh, Y. Hong and L. Bushnell, Stability analysis of networked control systems. In *American Control Conference*, pages 2876–2880 (1999).

67. L. Wolsey, *Integer programming*. John Wiley & Sons, Inc. (1998).

68. E. Wong and B. Hajek, *Stochastic Processes in Engineering Systems*. Springer-Verlag (1984).

69. W. Wong and R. Brockett, Systems with finite communication bandwidth constraints- part ii: stabilization with limited information feedback. *IEEE Transactions on Automatic Control*, 44(5): 1049–1053 (1999).

70. W. Zhang, *Stability analysis of networked control systems*. Ph.D. thesis, Case Western Reserve University (2001).

This document was prepared & typeset with LaTeX 2_ε*, and formatted with* NDdiss2_ε *classfile (v1.0[2004/06/15]) provided by Sameer Vijay.*