A Practical Controller for Explicit Rate Congestion Control

Kenneth P. Laberteaux, Charles E. Rohrs, *Senior Member, IEEE*, and Panos J. Antsaklis

***Abstract*—This paper examines congestion control for explicit rate data networks. The available bit rate (ABR) service category of asynchronous transfer mode (ATM) networks serves as an example system, however, the results of this paper are applicable to other explicit rate systems as well. After a plant model is established, an adaptive control strategy is presented. Several algorithm enhancements are then introduced. These enhancements reduce convergence time, improve queue depth management, and reduce parameter bias. This work differentiates itself from the other contributions in the area of rate-based congestion control in its balanced approach of retaining enough complexity as to afford attractive performance properties, but not so much complexity as to make implementation prohibitively expensive.**

***Index Terms*—Adaptive control, asynchronous transfer mode (ATM), available bit rate (ABR), coefficient bias, congestion control, convergence rate, data network, explicit rate, Internet, normalize least mean square (NLMS), queue control.**

I. Introduction

IN 1984, the Consultative Committee on International Telecommunications and Telegraph (CCITT), a United Nations organization responsible for telecommunications standards, selected asynchronous transfer mode (ATM) as the paradigm for broadband integrated service digital networks (B-ISDN) [2]. ATM networks provide six service categories. Each category of service is customized for a particular type of traffic. Of these five categories, only one, available bit rate (ABR), uses a feedback mechanism to create a closed-loop congestion control. The creation of a control mechanism for a switch that can work with the closed-loop congestion control mechanism such as the one specified by the ATM Forum [1] is the focus of this paper.

Congestion control is a process by which networks use feedback to adjust the influx of data such that the customer's quality of service (QoS) requirements are met while simultaneously attempting to maximize the utilization of the network's resources. Networks that attempt to deliver more data than their capacity will experience congestion, leading to undesirable data loss, excessive delays, or both. The closed-loop nature of congestion control implies communication between the network and customer throughout the life of the connection. Generally this communication comes in the form of instructions to the customer to increase or decrease its sending rate. Closed-loop congestion control is well suited for data that is not strongly delay sensitive. Closed-loop congestion control uses a feedback mechanism and thus can draw heavily on the feedback control theory.

The complete ABR congestion control mechanism is described in [1] and [2]. This paper focuses on explicit rate congestion control. The plant description of Section II-A is an approximation to the mechanisms specified in [1]. The present challenge is to devise a controller that resides at the output queue of an ATM switch port and produces a single *explicit rate* to be sent to all ABR sources passing through the queue. The explicit rate must be chosen such that the incoming ABR bandwidth matches the available ABR bandwidth in some appropriate sense. Specifying a single explicit rate at time n for all sources ensures fairness. Matching the incoming ABR bandwidth to the available ABR bandwidth attains efficiency.

This paper's treatment of ATM ABR congestion control is quite general. Issues studied by this paper are likely to arise in future networking protocols and should not be considered applicable only to ATM ABR. Given the rate at which bandwidth consumption is increasing and computational costs are decreasing, it seems likely that future data networks will employ a high-performance explicit rate congestion control mechanism.

A. ABR Congestion Control

The standard for ABR traffic [1] states that "the ABR service category provides a low cell loss ratio," and that "no numeric commitment is made about cell transfer delay," but both should be minimized. Key to this goal is avoiding congestion at any switching node in the ATM network; cells that arrive to a nearly full switch buffer will experience excessive delay, while cells arriving to a completely full buffer are lost.

ATM ABR explicit rate congestion control for a single source/destination pair [*virtual circuit* (VC)] is illustrated in Fig. 1 and occurs as follows: congestion control for ABR traffic utilizes a feedback mechanism, namely *resource management* (RM) cells. ABR source S periodically inserts (at least every N_{RM} cells) RM cells into its stream of data cells. These RM cells contain an explicit rate (ER) field that is initialized to the maximum possible sending rate of the source, its peak cell rate (PCR). Upon arrival at the destination, the RM cell returns to the source (usually along the same path of switches).

Each switch i at time n has $y_i^*(n)$ cells/sec of bandwidth to allocate to ABR traffic and therefore (independent of other switches) chooses a maximum explicit rate $u_i(n)$. As the RM

Manuscript received October 3, 2000; revised June 8, 2001 and November 6, 2001. Recommended by the Guest Editors W. Gong and T. Başar. This work was supported in part by the Army Research Office under Grant DAAG55-98-1-0199, and in part by the National Science Foundation under Grant NSF ECS99-12 458.

K. P. Laberteaux is with the Tellabs Research Center, Tellabs Operations, Inc., Mishawaka, IN, 46545 USA (e-mail: klanerte@yahoo.com).

C. E. Rohrs is with the Tellabs Research Center, Tellabs Operations, Inc., Mishawaka, IN, 46545 USA, and also with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: crohrs@mit.edu).

P. J. Antsaklis is with the Department of Electrical Engineering, The University of Notre Dame, Notre Dame, IN 46556 USA (e-mail: antsaklis.1@nd.edu).

Publisher Item Identifier S 0018-9286(02)05502-2.

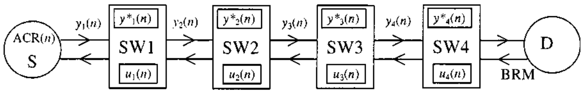


Fig. 1. Congestion control mechanism from perspective of source/destination pair.

cell moves from switch to switch, each switch i will reduce the rate[1] indicated by the ER field to $u_i(n)$ if $u_i(n)$ is less than the contents of the ER field. When the RM cell is returned to the source, the source is required to adjust its *allowed cell rate* (ACR), an upper bound on its sending rate, to be no greater than the rate indicated by the explicit rate field. Thus, the ACR of an ABR source equals the minimum rate allowed by the switches in the path of the flow as indicated by the most recently received RM cell.

1) Value of ATM ABR Congestion Control: For many years, ATM ABR received considerable attention, not in small part due to its extensive support of sophisticated congestion control. ATM ABR, despite being well suited for the explosively popular applications of web browsing, e-mail, and data backup, is yet to be widely utilized. Instead, the dominant protocol of today's Internet remains TCP/IP, with its comparatively less sophisticated congestion control [27].

Despite the ambiguities of the marketplace, there are at least two reasons to continue research in ATM ABR congestion control. The first reason is that ABR may yet see wide-scale adoption. Although no longer the newest technology, ATM ABR has yet to be outperformed by newer technologies in its stated task of providing efficient, fair, and reliable transport for nonreal-time, large bandwidth data applications. In fact, ABRs critics contend that its high-performance-through-high-complexity approach exceeds, both in capability and cost, the needs of the network marketplace of tomorrow. These critics claim that cheaper and simpler solutions, albeit less robust, are possible, most likely by extending the TCP/IP paradigm. Examples of these innovations include [29]–[35].

The outcome of the current ATM verses TCP/IP battle remains uncertain. ATM ABR has become a well-defined technology. The onus is on the new TCP/IP enhancements to prove their claims of doing well enough with less.

B. Related Work

Congestion control has been and continues to be a topic of active research. Significant contributions to the understanding of congestion control in ATM ABR networks have been made in the past decade. Contributors include [2]–[24]. Benmohamed and Meerkov made a significant early contribution in plant modeling with [4] and [5]. The assumptions developed in [4] are widely employed. Reference [4] treats the single-node case, while [5] treats the multiple-node case. In the end, through careful reasoning and imposing judicious assumptions, [5] essentially arrives back at the single bottleneck node case described in [4]. Reference [5] makes a strong case for simplifying the congestion control problem to a single node study; few investigators have deviated from this since. Computationally, the controller must solve dB $+ 3$ simultaneous equations each time action delays change (dB is the maximum action delay). References [4] and [5] place the closed-loop poles. No effort is made to cancel the plant (and thus closed-loop) zeros. Importantly, the number of responsive sources and their action delays are assumed known, thereby avoiding computational complexity usually associated with congestion controllers.[2]

Altman *et al.* make several contributions [7]–[9]. Of particular relevance to this paper, [7] discusses how a pure rate-matching algorithm, i.e., where the bandwidth available to ABR traffic is completely apportioned without regard of the current queue depth, will produce unacceptably long queues. However, [8] shows that under fairly general restrictions, under-allocating the available bandwidth, using either an additive or multiplicative constant, will ensure stability in the queue length. This gives some credibility to the rate matching schemes proposed here and elsewhere. Throughout [7]–[9] (like [4], [5]), the number of sources and their action delays are assumed to be known. Also note that their models do not include the presence of ABR traffic which is controlled by other switches.

Raj Jain has made the best know contributions to the field of ATM ABR congestion control. His implementation-friendly explicit rate indication for congestion avoidance (ERICA) algorithm [10] and its successor, ERICA+ [12], work well in a large number of situations and appear to be favored by ATM switch designers. ERICA is computationally inexpensive to implement (as compared to the other contributions mentioned above) and has been shown, via simulations, to rapidly achieve max–min fairness in many cases. However, further study discovered various scenarios where max–min fairness was not achieved. In a 1998 contribution [14], persisting fairness concerns of ERICA+ prompted a new approach. The switch determines an effective number of sources. This effective number of sources, or N_{eff}, assigned a specific fractional value to sources unable to use their fair share allocation. This approach is very similar to that suggested by Fulton and Li in 1997 [17], and marks an intersection in these two bodies of work. Imer also proposes a controller in the same vein [15]. A comparison of these related schemes to that proposed by this paper occurs in Section II-C2.

In addition, there has been significant contributions made in the ATM Forum. The ATM Forum has approved what has become the *de facto* guidelines for the operation of ABR congestion control by defining the required behaviors and properties of ABR sources, destinations and resource management (RM) cells [1].

This work differentiates itself from the other contributions in the area of rate-based congestion control in its balanced approach of retaining enough complexity as to afford attractive performance properties, but not so much complexity as to make implementation prohibitively expensive.

[1] To minimize reaction delay, switches generally mark backward (returning-to-source) RM cells. Networks can also use the same congestion control techniques while marking forward RM cells (if, for example, backward RM cells do not take the same path as forward RM cells), but with much longer action delays, with the expected performance degradation.

[2] The controller presented in Section II-B does not assume that the number of sources and delays $(\mathbf{B})$ is given. This accounts for a significant amount of the controller's complexity.

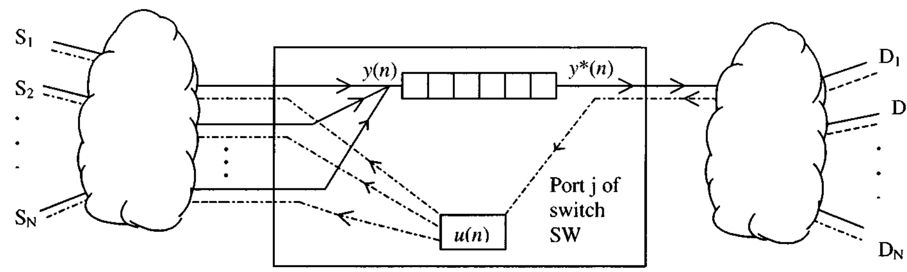


Fig. 2. Plant from perspective of switch output port.

Another vein of congestion control research focuses on lower complexity-lower performance solutions for the Internet. Floyd proposed random early detection (RED) [30] and explicit congestion notification (ECN) [31], two methods for routers to signal congestion by probabilistically dropping RED or marking ECN packets. These two concepts have given rise to several suggestions for Internet congestion control using one-bit marking strategies (see [34] and [36] and references therein). This group of one-bit, Internet-specific algorithms, and our ATM ABR algorithm, occupy very different places on the performance-cost curve, specifically our algorithm gives more performance with more cost. At best, the one-bit marking algorithms can match bandwidth to capacity only in the mean, requiring large buffers (as discussed in Section II-C.2).

The general flow control problem has also received extensive investigation (see [26] and the references within). This general approach is helpful in framing specific flow control problems, but required communication and computational resources are not generally available in ATM and TCP applications.

C. Outline of Paper

The remainder of this paper is as follows. Section II specifies an appropriate plant and controller for the congestion control problem. Section III presents three algorithm enhancements. The first, described in Section III-A, dramatically improves the convergence time of the controller. The second, described in Section III-B, extends the purely rate-matching control scheme to provide queue depth management. The third, described in Section III-C, extends the plant model to include a noise disturbance, corresponding to nonresponsive sources with varying rates. Then, a strategy for reducing the resulting coefficient bias is introduced. Conclusions are made in Section IV.

II. The Congestion Control System

A. Plant Definition

Reference [1] defines the mechanism used for congestion control for ATM ABR networks. In this section, the important features of [1] are distilled into a plant model (Fig. 2). The following description augments the material in Section I-A.

Since each switch implements its own, independent controller, one may consider the plant from the perspective of a single switch SW. A discrete-time model is used, where sample intervals correspond to control intervals, i.e., a new control action $u(n)$ is calculated for each n.

The present challenge is to devise a controller that resides at output port j of switch SW and produces a single explicit rate u to be sent to all ABR sources passing through the port. The explicit rate u must be chosen such that the incoming ABR bandwidth y matches the available ABR bandwidth y^* in some appropriate sense. Specifying a single explicit rate at time n for all sources ensures fairness. Matching y to y^* attains efficiency.

It is assumed that for each VC, at least one RM cell passes j during each sample interval. Rates $u(n), y(n)$, and $y^*(n)$ are in units of cells/s.

Output port j will observe changes to its input rate $y(n)$ as various sources (S_i) react to previously specified explicit rates $u(n-m)$. The *reaction delay*, m, as viewed by j for source S_i, is the time between j's adjustment of its explicit rate to the time j measures this explicit rate as its input rate from S_i. These reaction delays will vary for different sources. Assume that there are b_0 sources that respond with reaction delay d, b_i sources that respond with delay $d+1, \ldots$, and b_{dB} with delay $d+\text{dB}$, where dB is a known upper bound on j's reaction delay. There is also an unknown subset of the N sources that are unresponsive to port j's explicit rates. In the first part of this paper, the rate of this unresponsive traffic is assumed to be a constant C cells/s. This assumption is justified if each unresponsive source has a guaranteed minimum cell rate that exceeds port j's explicit rates. In Section III-C, the unresponsive traffic is instead modeled as a random process.

It is assumed that $b_0, b_1, \ldots, b_{\text{dB}}$ (and C until Section III-C) remain constant for periods of time long enough for adaptive identification to occur. Faster convergence speed of the adaptive algorithm results in better tracking of these time-varying parameters. The plant is therefore given by

$$y(n) = b_0 u(n-d) + \cdots + b_{\text{dB}} u(n-d-\text{dB}) + C \quad (1)$$

$$y(n) = B(z^{-1}) u(n-d) + C \quad (2)$$

$$y(n) = \mathbf{B}^T \mathbf{u}(n-d) + C \quad (3)$$

$$\mathbf{B} \equiv [b_0, b_1, \ldots, b_{\text{dB}}]^T \quad \text{and}$$

$$\mathbf{u}(n) \equiv [u(n), u(n-1), \ldots, u(n-\text{dB})]^T.$$

Note that for convenience, filters in z^{-1} (denoting unit time delay) and time sequences in n are mixed in expressions, e.g., (2). Matrix notation is also used. Equations (1), (2), and (3) are equivalent.

Since the minimum delay in the plant is d, adjustments in $u(n)$ will not be observed until $n+d$. Therefore, to generate $u(n)$, it must be decided at time n what the desired value of $y(n+d)$ should be. This desired bandwidth, which is notated as $y^*(n+d \mid n)$, may reflect both bandwidth and buffer measurements made up to time n (this may be generated by a prediction filter as in [9]). By extension, in many cases, the input of the algorithm will be $y^*(n+d+V\,|\,n)$ (for some nonnegative V), i.e., the desired value of $y(n+d+V)$ chosen at time n. The goal of the congestion control mechanism of SW is to choose the control signal $u(n)$ at time n so as to minimize $E[(y(n+d+V)-y^*(n+d+V\,|\,n))^2]$.

This plant model was introduced in [16]. It is a direct generalization of the plant models implicit in the work of Fahmy, Jain *et al.* [14] and Fulton and Li [17] (see [16]), which have been extensively simulated under realistic conditions.

B. Controller Definition

The plant (3) is a finite impulse response (FIR) filter and is thus bounded-input–bounded-output (BIBO) stable. However, it is quite possible that plant (3) is nonminimum phase (NMP), necessitating a controller that performs adequately with NMP plants. The large phase lags inherent in NMP plants generally make them difficult to control. However, the challenge is greatly simplified if the NMP plant is known to be stable. Specifically, the adaptive controller, first proposed in [21], approximately inverts the stable FIR plant (3) with another FIR filter. The concept of approximately inverting one FIR filter with another adaptive FIR filter is not new, e.g., [37], [39]. Yet this concept of *adaptive approximate inverse control* seems to have gained relatively little attention despite its attractive characteristics, perhaps due to limited convergence and stability analysis. Analysis of this controller, found in the Appendix, demonstrates attractive performance for the current plant and may increase the adoption of this control scheme in other applications.

To understand adaptive approximate inverse control, consider that the plant $B(z^{-1})$ can have zeros inside and outside the unit circle. The ideal inverting IIR filter is then $B^{-1}(z^{-1}) \equiv 1/B(z^{-1})$. The time-domain realization $b^{-1}(n) \equiv \mathcal{Z}^{-1}\{B^{-1}(z^{-1})\}$, where $\mathcal{Z}^{-1}\{x(z^{-1})\}$ is the inverse Z-transform of $x(z^{-1})$ [38], is not specified until a region of convergence is specified. If the region of convergence is chosen to include the unit circle, the impulse response is generally two-sided, i.e., nonzero for both positive and negative n. However, unless there is a root of $B(z^{-1})$ on the unit circle, thereby preventing a region of convergence that includes the unit circle, $|b^{-1}(n)|$ converges to zero exponentially as $n \to \pm\infty$ [38]. By delaying $b^{1}(n)$ by V samples, the resulting $b^{-1}(n-V)$ can be truncated to form an FIR filter if $b^{-1}(n-V)\approx 0$ for $n<0$ and $n>dQ$ $(V\geq 0)$. The resulting causal $(dQ+1)$ tap FIR filter $q(n)$ approximates $b(n-V)^{-1}$ increasing well with increasing choices of V and dQ (if $B(z)$ has no roots on the unit circle)

$$Q(z^{-1}) = q_0 + q_1 z^{-1} + \cdots + q_{dQ} z^{-dQ} \approx \frac{z^{-V}}{B(z^{-1})}. \tag{4}$$

Note that adding delay is a common characteristic of nonminimum phase plant control, given the large phase lags inherent in nonminimum-phase plants. (The above explanation does not appear in [37] or [39], although the more recent [40] makes brief, similar comments).

To illustrate, consider as an example $B(z^{-1}) = 2 + 9z^{-1} + 8z^{-2} + 3z^{-3}$, which has a pair of complex minimum-phase zeros and one nonminimum-phase zero, as shown in Fig. 3(a). Fig. 3(b) and 3(c) show $b^{-1}(n)$ and $q(n)$ (with $V=10$), respectively. Fig. 3(d) shows the accuracy of the approximation (4) for this example.

Since the plant $\mathbf{B}$ is not known *a priori*, the controller must be determined adaptively. Various methods and architectures for adaptively discovering $\hat{\mathbf{Q}}(n)$ are explored in [21]. Fig. 4 specifies the structure for controller identification recommended by [21]. The controller $\hat{\mathbf{Q}}(n)$ can be adaptively determined using the normalize least mean square (NLMS) algorithm [45]. Faster convergence speed of the adaptive algorithm results in better tracking of the time-varying parameters of $\mathbf{B}$ resulting from the opening and closing of ABR connections. At time n, calculate

$$\hat{u}(n-d-V) = \hat{\mathbf{Q}}(n)^T\mathbf{y}(n) \tag{5}$$

$$\begin{aligned} \hat{\mathbf{Q}}(n) &= [\hat{q}_0(n), \hat{q}_1(n), \ldots, \hat{q}_{dQ}(n), \hat{q}_{\mathrm{DC}}(n)]^T \\ \mathbf{y}^*(n+d+V\,|\,n) &\equiv [y^*(n+d+V\,|\,n), \ldots, \\ &\qquad y^*(n+d+V-dQ\,|\,n-dQ), y_{\mathrm{DC}}]^T \\ \mathbf{y}(n) &= [y(n), y(n-1), \ldots, y(n-dQ), y_{\mathrm{DC}}]^T \end{aligned} \tag{6}$$

$$\begin{aligned} e(n) \equiv e_u(n-d-V) &\equiv u(n-d-V) \\ &\quad - \hat{u}(n-d-V) \end{aligned} \tag{7}$$

$$\hat{\mathbf{Q}}(n+1) = \hat{\mathbf{Q}}(n) + \frac{\mu\mathbf{y}(n)}{\mathbf{y}(n)^T\mathbf{y}(n)} e(n), \quad 0<\mu<2 \tag{8}$$

$$u(n) = \hat{\mathbf{Q}}(n+1)^T\mathbf{y}^*(n+d+V\,|\,n). \tag{9}$$

The scalar d is the minimum plant delay, V is an operator chosen (nonnegative) inversion polynomial delay (previously discussed), and μ is the adaptive gain chosen such that $0 < \mu < 2$. The constant y_{DC} is operator-chosen, appended to the delay-chain values of $\{y\}$ in (6) so that the final tap of $\hat{Q}(n)$ becomes a DC tap $\hat{q}_{\mathrm{DC}}(n)$ (see [20]), $\hat{\mathbf{Q}}(n) \equiv [\hat{\mathbf{Q}}_{\mathbf{lin}}(n)^T, \hat{q}_{\mathrm{DC}}(n)]^T$.

Fig. 5 shows the complete control architecture. The identification section uses NLMS adaptation to determine $\hat{\mathbf{Q}}(n+1)$ (shown with $\hat{q}_{\mathrm{DC}}$ separated from the remaining linear taps, $\hat{\mathbf{Q}}_{\mathbf{lin}}$, and with $y_{\mathrm{DC}}=1$) by creating estimate $\hat{u}(n-V-d)$ using (5). $\hat{\mathbf{Q}}(n+1)$ is copied into the Controller, which produces $u(n)$ from the set point $y^*(n+V+d\,|\,n)$. The plant is represented by (3).

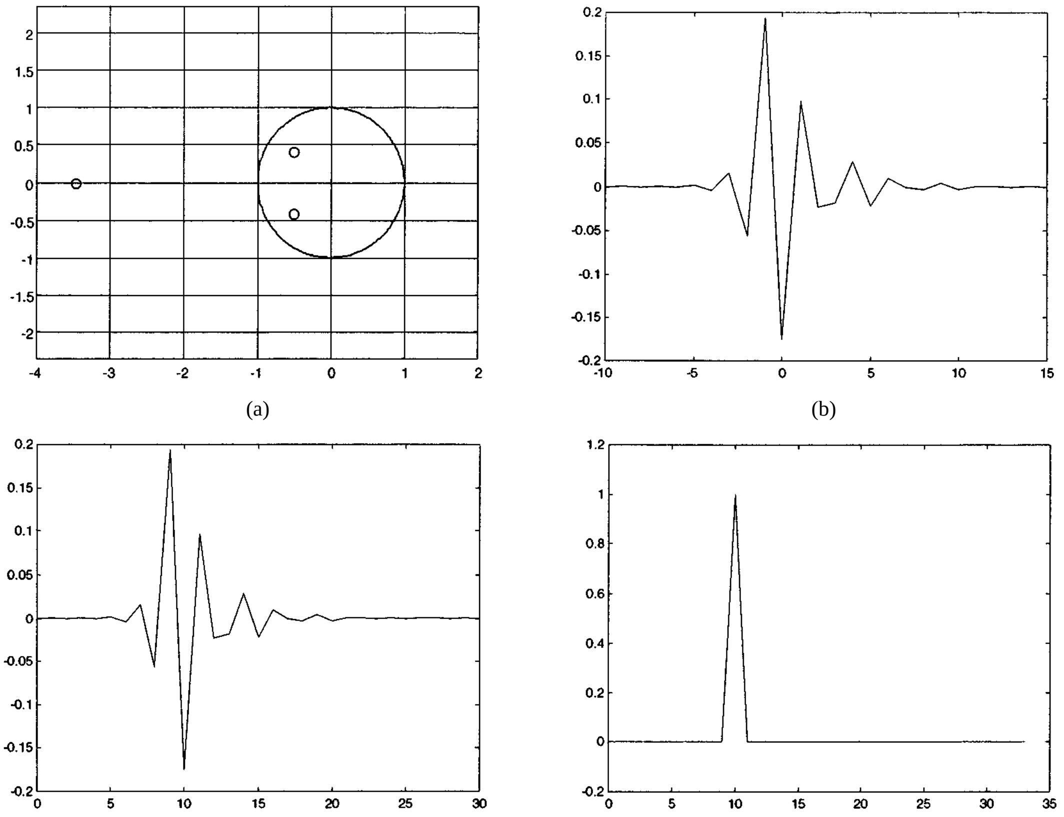
C. Discussion and Comparisons

The control algorithm (5)–(9) has many attractive features, as described in the Appendix. Specifically, the Appendix shows that, under a set of reasonable assumptions, the adaptive, FIR controller filter $\hat{\mathbf{Q}}(n)$ converges to its optimal Weiner solution

$$\mathbf{Q}_0 \equiv \{E[\mathbf{y}(n)\mathbf{y}(n)^T]\}^{-1}E[\mathbf{y}(n)u(n-d-V)] \tag{10}$$

in the mean and mean-square. Further, the Appendix gives conditions that guarantees that $\lim_{n\to\infty} y(n) - y^*(n\,|\,n-d-V) = 0$.

This plant-controller formulation limits itself to controlling rates, with no explicit queue control. This strategy, supported by [8], requires that the bandwidth available for ABR traffic be slightly under-utilized, thus creating extremely short (or zero)



(a)

(b)

(c)

(d)

Fig. 3. $B(z^{-1}) = 2 + 9z^{-1} + 8z^{-2} + 3z^{-3}$ (a) The zeros of $B(z^{-1})$. (b) A two-sided, causal impulse response $b^{-1}(n)$ if the region of convergence is chosen to include the unit circle. (c) The impulse response of $q(n)$, a delayed, truncated version of $b^{-1}(n)$. (d) The convolution $b(n)^{*}q(n) \approx z^{-10}$.

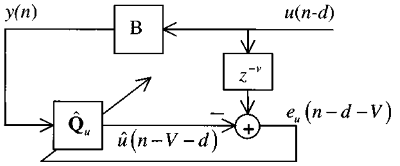


Fig. 4. Direct inverse plant modeling.

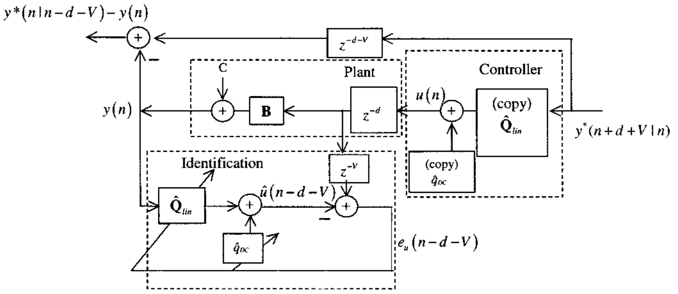


Fig. 5. Complete control architecture ($y_{\rm DC} = 1$).

queue lengths in steady state. By avoiding explicit queue modeling, the plant is reasonably modeled as an FIR filter and thus open-loop stable. The proposed controller, lacking the ability to modify closed-loop eigenvalues, can only be employed with known stable plants. Its greatest asset is its effective and intuitive control of stable NMP plants. In Section III-B, explicit queue control is proposed. The integrating action of a queue poses stability issues. Control parameters must be chosen with care to ensure stability of the enhanced system.

1) Simulation Framework: To provide a baseline for comparisons in this paper, a common simulation framework is now defined. These simulations use the Matlab [46] simulation tool.

The plant, defined in Section II-A and shown in Fig. 2, envisions a switch SW having an output port j containing a congestion controller. For the purpose of a common simulative framework, the output port rate of port j is 2488 Mbps (million bits per second) $= 5.869$ Mcps (million cells per second), i.e., an OC48, which is a realistic port speed for ATM switches currently under development. Of that, some subset (10–20% seems reasonable) of the bandwidth will be allocated for ABR traffic, and in the current framework, 1 Mcps is used as the average ABR rate for port j. Let $C = 200$ Kcps (thousand cells per second) of this 1 Mcps constitute ABR traffic controlled by other ports, leaving on average 800 Kcps of ABR traffic responsive to the port j. The set-point y^* is therefore chosen to be a white Gaussian process with mean $E[y^*] = 1$ Mcps and a standard deviation σ_{y^*} of 22 Kcps.[3]

It seems plausible that the complexity of ABR will discourage its use for short-lived connections (e.g., domain name server queries, individual e-mail deliveries, etc.). Instead ABR connections in a single port will likely constitute a small number of large bandwidth aggregations of traffic, e.g., connecting sites of a college or industrial campus. Therefore, for these simulations,

[3] These deviations about the mean of the desired ABR rate are determined by the extent that the port measures and reallocates bandwidth from higher level service category flows. It is somewhat uncertain how aggressively ports will attempt to reallocate unused bandwidth. Very small variances are possible.

let the 800 Kcps of responsive ABR traffic be comprised of 22 high-capacity, greedy sources, each averaging 15.4 Mbps $=$ 36.4 Kcps. If the number of ABR cells that must include one RM cell, $N_{\rm RM}$, is 32, then the per-connection rate of RM cells corresponding to responsive ABR sources is 1.14 Kcps, or one RM cell every 880 microseconds. The measurement and control sample time is $T_s = 1$ ms.

The minimum response is chosen to be delay $d = 10$ msec. The distribution of the delays of the 22 sources is given by $B(z^{-1}) = z^{-10}(2 + 9z^{-1} + 8z^{-2} + 3z^{-3})$. This corresponds to a plant with one nonminimum phase zero and a pair of complex minimum phase zeros [see Fig. 3(a)]. The number of taps in the controller is $dQ = 30$, with $V = 10$. The adaptation gain is set at its optimal value $\mu = 1$. Cell rates are not strictly limited to be nonnegative, although manual inspections reveal that this rarely occurs after an initial transient.

2) Comparisons to Less Complex Schemes: Before proceeding to the main contribution of this paper, the algorithm enhancements, it is appropriate to evaluate the merits of the general control scheme proposed here. Other approaches to congestion control have been outlined in Section I-B. Included in this list are approaches that claim to provide satisfactory performance with a lower computational cost than (5)–(9). In what follows, it is shown that the added computational cost of (5)–(9) provides better performance than less computationally complex schemes, specifically [14], [15], [17]. Also, (5)–(9), in its simplest $(dQ = 0)$ case, is essentially equivalent in performance and complexity to the simpler schemes.

Consider the proposed controller (5)–(9), specifically the identification (8), with only one adaptive tap, i.e., $dQ = 0, V = 0, \mu = 1$

$$\hat{q}(n+1) = \hat{q}(n) + \frac{1}{y(n)}(u(n-d) - \hat{q}(n)y(n)) = \frac{u(n-d)}{y(n)} \tag{11}$$

Note that in the $dQ = 0$ case, the NLMS adaptation devolves into a single division. Compare this to Fulton's identification [17]

$$\hat{N}_{\text{eff,Fulton}}(n+1) = \frac{y(n)}{\bar{u}(n-1)}$$

where $\bar{u}(n-1)$ is the time average of a sequence of previous values of u. Fulton does not explicitly estimate d, therefore requires the averaging on u for convergence (unsurprisingly, the recommended time interval for averaging is d samples). Similarly, Imer [15] calculates

$$\hat{N}_{\text{eff,Imer}} = \frac{y}{u}.$$

every d' samples, $d' \geq d$, where u is kept constant over the past d' samples. The Fahmy parameter *effective number of active VCs*, or $\hat{N}_{\text{eff,Fahmy}}$, is defined similarly [14], albeit in an indirect manner.

Clearly $\hat{N}_{\text{eff,Laberteaux}} = 1/\hat{q}, \hat{N}_{\text{eff,Fulton}}, \hat{N}_{\text{eff,Imer}}$, and $\hat{N}_{\text{eff,Fahmy}}$ are adaptive estimates that attempt to capture the same information. In each controller, this value is used to divide the amount of available bandwidth y^* so that a future y will match a future y^* in some appropriate way. Thus the controller (5)–(9), in its simplest version $(dQ = 0)$, is essentially equivalent, in performance and complexity, to the suggestions made by Fulton, Imer, and Fahmy.

Consider how these one-tap controllers perform. The controller in each case consists of dividing a future estimate of $y^*(n)$ by the associated $\hat{N}_{\text{eff}}$. All provide fair and efficient allocation of y^* in the long-term. The best such a controller could accomplish is that the incoming ABR bandwidth matches the available ABR traffic in the mean, i.e., $E[\chi(n)] = 0, \chi(n) \equiv y^*(n) - y(n)$.

While the authors of [14], [15], [17] make a fair and stable allocation their performance goal, here fair and stable allocation is taken to be a minimum acceptable performance objective. This difference in performance objective may be based on a modeling assumption. Clearly for ATM ABR congestion control systems, two quantities change with time: the amount of bandwidth allocated to ABR and the number of competing ABR connections vying for this bandwidth. Since operational experience with ABR is limited, it is difficult to know with certainty the time-scales over which these two quantities change. However, this paper assumes that an ABR controller is likely to see its available bandwidth change more rapidly than the number of connections.

If the available bandwidth $y^*(n)$ remains constant for long periods (e.g., multiples of the maximum round trip time, d), or $\sigma^2_{y^*} \approx 0$, then the single-tap schemes discussed above work effectively. Note that Imer, both in his development and simulations, assumes that $y^*(n)$ is constant. Fulton uses $\overline{y^*(n)}$, the sample mean of $y^*(n)$, in her calculation of the explicit rate $u(n)$.

Since this paper assumes that $y^*(n)$ changes more quickly than changes in the number of ABR connections, $y^*(n)$ is modeled as a noise source. Using this paper's notation, in the one-tap case, $y(n)$ can be modeled as

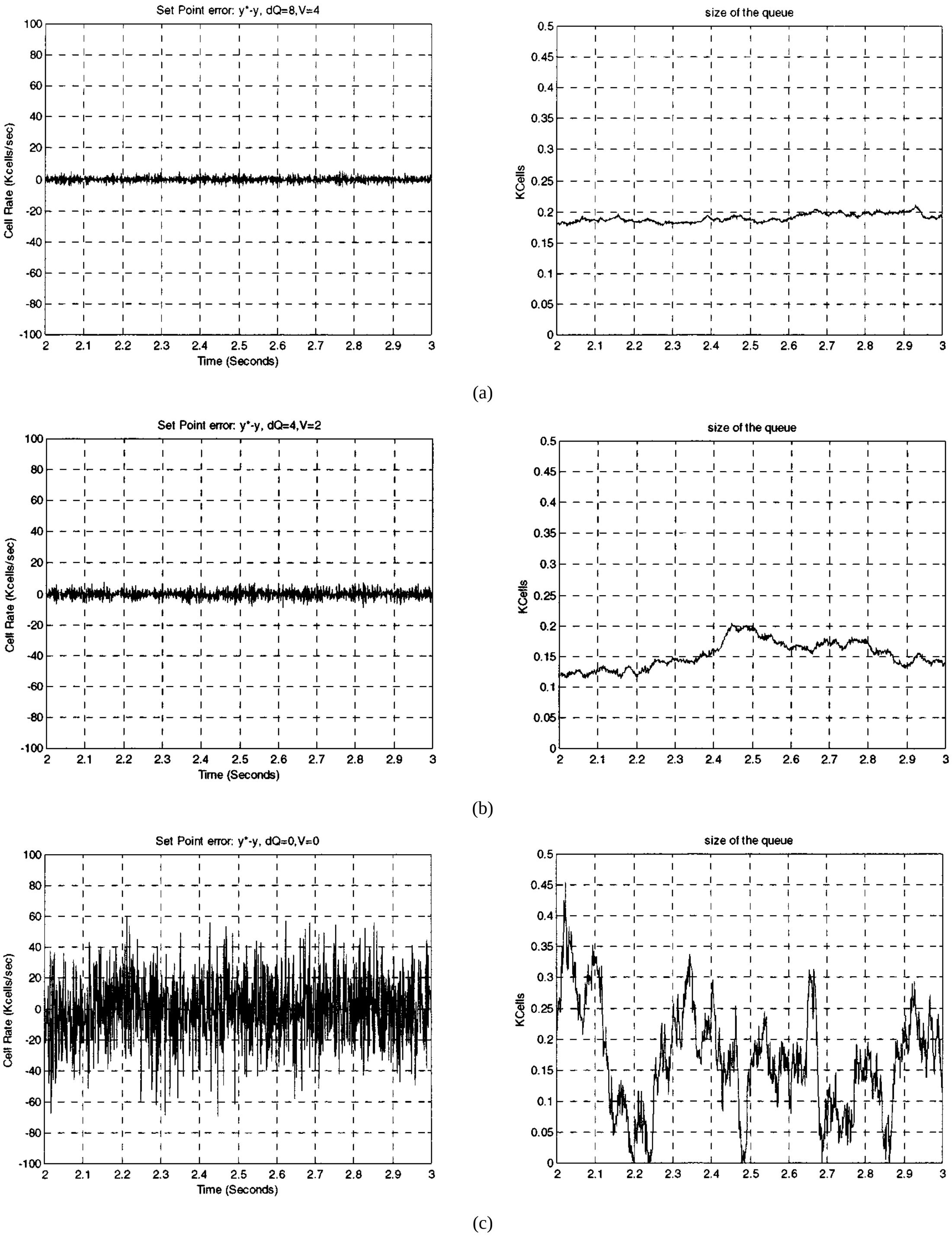
$$y(n) = y^*(n + d + V \,|\, n)\frac{B(z^{-1}) + C}{\hat{N}_{\text{eff}}} \tag{12}$$

i.e., a noise source filtered through the FIR filter $(B(z^{-1}) + C)/\hat{N}_{\text{eff}}$. From (12)

$$\begin{aligned}\chi(n) &= y^*(n \,|\, n - d - V) - y(n)\\ &= y^*(n \,|\, n - d - V)\left(1 - \frac{B(z^{-1}) + C}{\hat{N}_{\text{eff}}}\right).\end{aligned}$$

Unless, $B(z^{-1}) = b_o$, the variance of $\chi(n), \sigma^2_\chi$, increases as $\sigma^2_{y^*}$ increases. The queue size, $\text{queue}(n)$, is the integral of $\chi(n)$. From the definition of variance, for a one-tap controller N_{eff}, the variance of $\text{queue}(n), \sigma^2_{\text{queue}}$, also increases as $\sigma^2_{y^*}$ increases (these observations will be supported by simulations below). This increases the necessary buffer size if overflow is to be avoided. Also, if buffer underflow is to be avoided, a larger average queue size must be targeted as σ^2_{queue} increases.

Since larger queue sizes require a larger memory cost and also increases the delay through the switch, both of which are preferably avoided, this paper views minimizing σ^2_χ, and thus



(a)

(b)

(c)

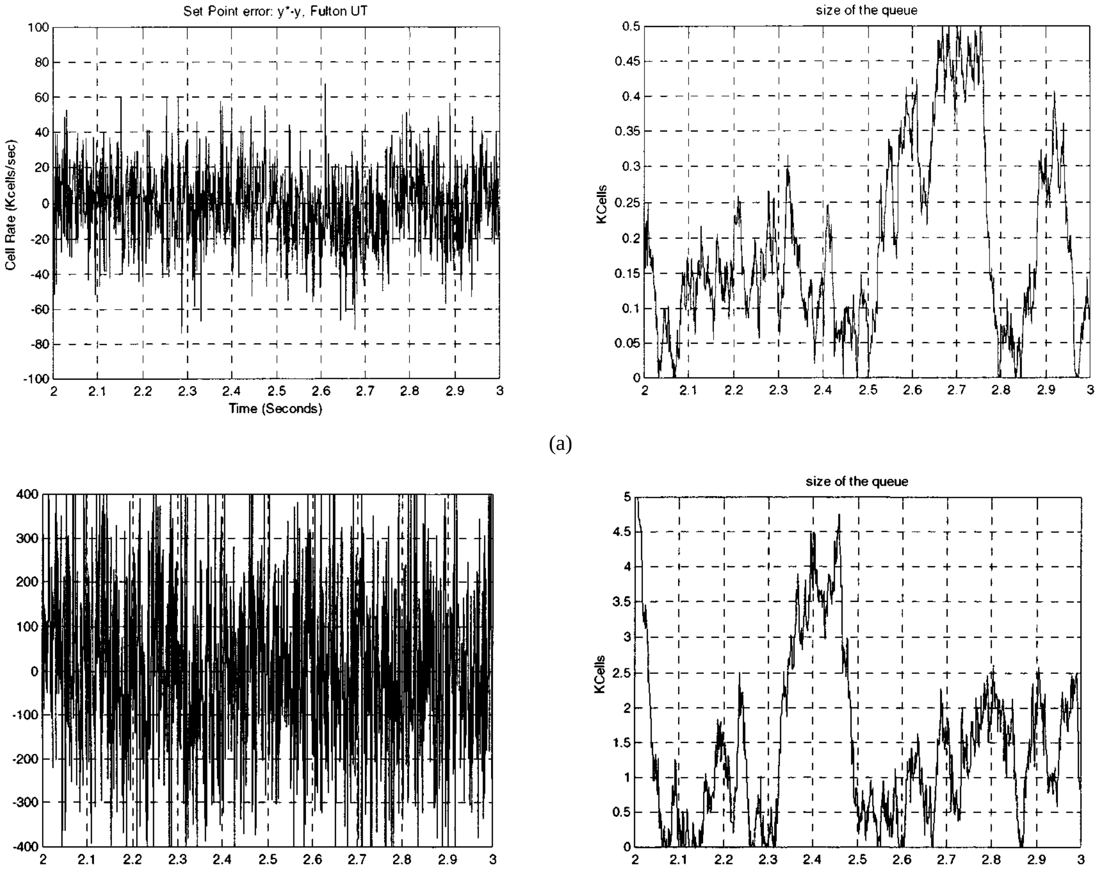
Fig. 6. Set-point error (left), and size of queue (right), (same scaling as Fig. 7(a)), with $\sigma_{y^*} = 22$. (a) $dQ = 8, V = 4$. (b) $dQ = 4, V = 2$. (c) $dQ = 0, V = 0$.

$\mathrm{queue}(n)$, as a desirable performance goal. Using the simulation environment established in Section II-C.1, Fig. 17 shows[4] how effectively σ_χ^2 and queue size can be minimized by using 31 $(dQ = 0)$ taps in the controller (5)–(9). As dQ is decreased to the limiting one-tap $(dQ = 0)$ case, performance gracefully degrades to that of the one-tap solutions discussed above.

Fig. 6(a)–(c) show the performance, both in terms of set-point error, $y^*(n) - y(n)$, and the size of the queue, $\mathrm{queue}(n)$, gradually degrading as the number of taps dQ decreases. In the limiting one-tap case $(dQ = 0)$, the performance is essentially equal to the performance of Fulton's UT algorithm, shown in Fig. 7(a). This supports the near-equivalence of performance predicted in the discussion above. (These simulations are identical to those shown in Fig. 17-where the desired queue size is 100–200 cells-except for changes in dQ and V as noted).

Some may be satisfied with the performance of the simple, one-tap controllers shown in Fig. 6(c) and 7(a). However, it

[4] For the sake of space economy, the simulations of (5)–(9) shown in this section are in fact using the algorithm enhancements of Sections III-A and III-B. However, the basic complexity/performance comparisons are essentially the same for the nonenhanced controller of (5)–(9).



(a)

(b)

Fig. 7. Set-point error (left) and size of queue (right) with Fulton's UT algorithm. (a) $\sigma_{y^*} = 22$. (b) $\sigma_{y^*} = 223$.

is important to note that performance of one-tap controllers is highly dependent on the standard deviation of the set-point, σ_{y^*}. When σ_{y^*} is increased an order of magnitude from 22 to 223, the performance is observed to degrade an order of magnitude [compare Fig. 6(c) to 8(c) and 7(a) to (b)]. In contrast, the multitap controllers $(dQ > 0)$ improve performance as the number of taps increase, with the original $dQ = 30$ case showing no performance impairment due to increased σ_{y^*} (compare Figs. 8(a) and 17).

It is well known that the convergence time for LMS type algorithms, including NLMS, decreases as the number of taps increases [45]. The plots above begin after 2 s, as all cases converge within this time. However, convergence rates of the adaptive estimates increase as the number of taps decreases, revealing a short-term versus long-term performance tradeoff.

To summarize, the added complexity of (5)–(9) provides much improved performance over those of [14], [15], [17]. Further, (5)–(9) can be simplified in implementation (by reducing dQ), thereby gradually reducing its performance and complexity to that of the popular one-tap solutions [14], [15], [17]. For example, if the complexity budget for a specific available bit rate application allows five taps $(dQ = 4)$, then the added complexity of these five taps appears justified.

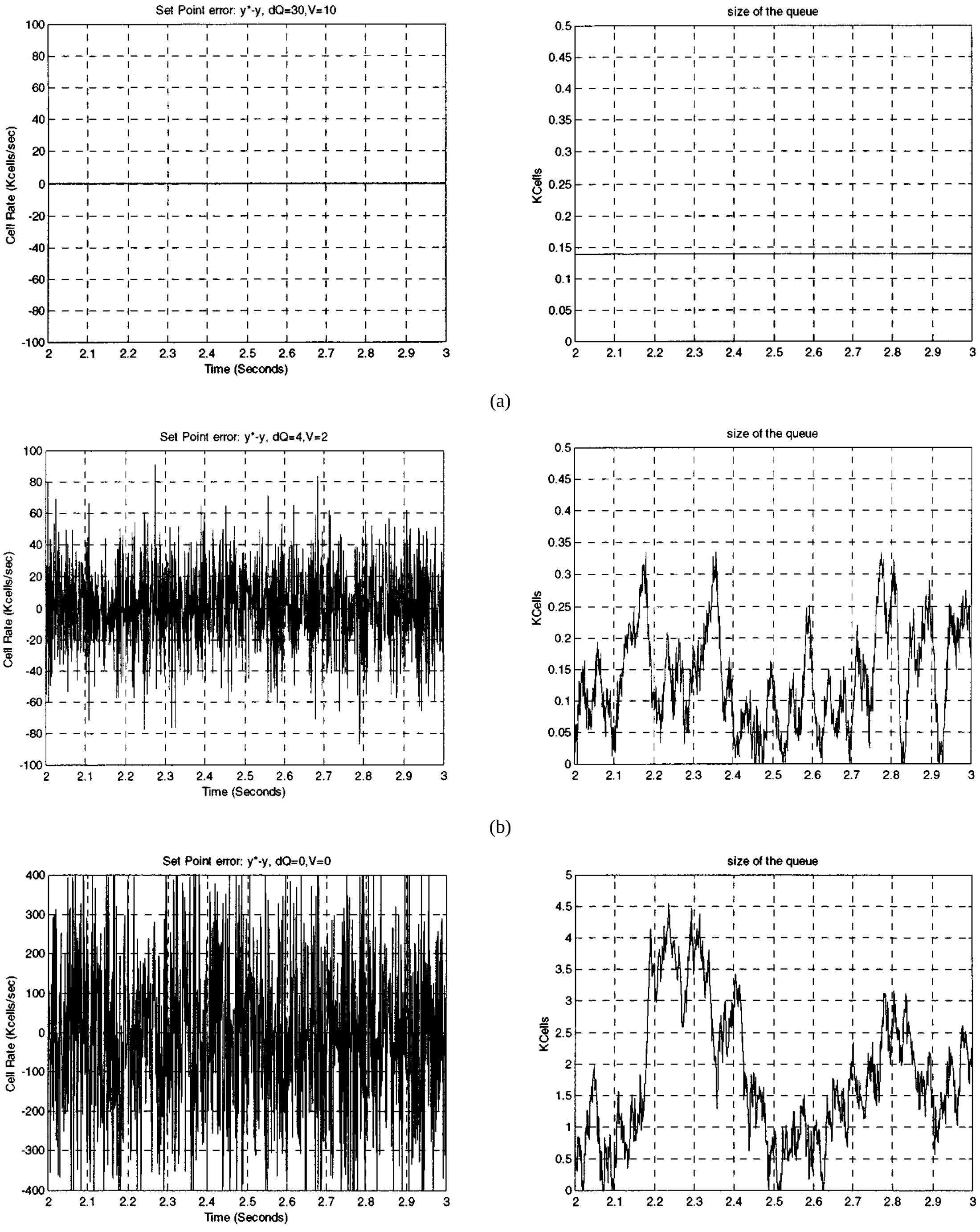
Other, even computationally simpler, congestion control schemes have been presented for the Internet (see Section I-B). Generally these schemes are one-bit marking approaches. These approaches occupy a very different location on the performance/complexity curve of congestion control. At best, these one-bit schemes will match the arriving bandwidth to the available bandwidth in the mean, with even greater error variance σ_{χ}^2. The aforementioned comparisons can therefore be extended to the one-bit Internet proposals.

III. Algorithm Enhancements

In this section, three additions to the congestion control mechanism are introduced and discussed. Each addition provides necessary mortar in cementing together theoretical analysis and practical design. These three modifications are singled out for attention here since each addresses a general issue likely to appear in many complex congestion control schemes, not just that of ATM ABR congestion control. The simulations use the framework is described in Section II-C.1. Material from this section was first published in [22].

A. Convergence Rate Improvements

The first algorithm enhancement addresses the convergence rate of the controller. The results of the Appendix ensure that the originally proposed congestion controller converges. However, without the modifications presented in Section III-A, convergence rates are unnecessarily, and possibly unacceptably, slow. Significant speedup is obtained with the following modifications.



(a)

(b)

(c)

Fig. 8. Set-point error (left), and size of queue (right), with $\sigma_{y^*} = 223$. (a) $dQ = 30, V = 10$. (b) $dQ = 4, V = 2$. (c) $dQ = 0, V = 0$.

1) Unmodified Convergence: Fig. 9 shows the results of simulating the system without any modifications to improve the rate of convergence. After 8 s, the convergence of the controller is so poor that it appears to be admitting over twice the desired rate of traffic[5] . This is clearly an unacceptable performance.

2) Managing the Eigenvalue Spread: The least mean square (LMS) algorithm has the property that the mean of the coefficient error vector, $E[\tilde{\mathbf{Q}}(n)]$, converges to zero at a rate inversely proportional to the eigenvalue spread $\lambda_{\max}/\lambda_{\min}$ of $\mathbf{R} = E[\mathbf{y}(n)\mathbf{y}(n)^T]$ [45]. Note that the eigenvalue spread is a measure of the conditionality of a matrix. It is more difficult to specify the convergence trajectory of $E[\tilde{\mathbf{Q}}(n)]$ for normalized least mean square (NLMS) adaptation in all but the simplest cases [43]. Practical experience shows that speed of convergence is still a strong function of eigenvalue spread, with larger spread results in slower convergence.

What follows are three proposals for reducing the eigenvalue spread of $\mathbf{R}$, thereby increasing convergence times, followed by a comparative discussion.

[5] Note that the results from the Appendix ensures that $y(n)$ will eventually coincide with $y^*(n)$.

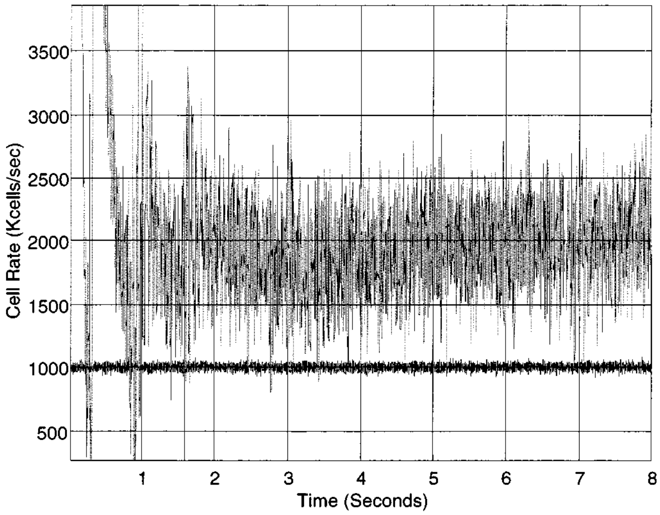


Fig. 9. Comparing the set point (lower curve centered at 1000) and port Input Rates (higher curve approximately centered at 2000)—unmodified case. The lower set point y^* plot remains around 1000 Kcps while the port input rate y plot has a mean value around 2000 Kcps.

a) Reducing Means via Constant Estimates: Several strategies to improve convergence time of the system defined in Section II have been proposed and evaluated. For this application, the best strategy is as follows: Provide the identification algorithm with zero-mean signals by estimating and removing the signal means. Then perform "DC correction" in the controller by an additive term.

The basic concept is illustrated by Fig. 10. Let α and β be fixed estimates of $E[u(n)]$ and $E[y(n)]$ respectively. Subtract α and β from their corresponding signals to perform identification. Constants α and β are then added to the controller to perform DC correction. It is easily shown (see [24]) that for the architecture of Fig. 10, once $\hat{\mathbf{Q}}(n)$ converges to its optimal $\mathbf{Q}_0, E[y(n)] = E[y^*(n \,|\, n - V - d)]$.

There are several methods for choosing α and β. One possibility is to set β equal to the sample mean of $y^*(n + d + V \,|\, n)$. Then, if N is the total number of ABR flows supported by the port (including bottle-necked flows), set $\alpha = \beta/N$. The intuitive proposal of using sample means of y and u leads to instability, as will be shown in Section III-A2b.

Simulations show the method depicted in Fig. 10 has the potential to make a significant improvement in convergence rate, but that performance is quite sensitive to the accuracy of the mean estimates. For example, Fig. 11(a) shows the case when $a = 0.99E[u(n)]$ and $\beta = 1.01E[y(n)]$. The measured eigenvalue spread of $\mathbf{R}$ is 50. The convergence is very fast. However, when $a = 0.9E[u(n)]$ and $\beta = 1.1E[y(n)]$, as shown in Fig. 11(b), the performance is noticeably slower, albeit much better than shown in Fig. 9 (where essentially $a = 0$ and $\beta = 0$). The measured eigenvalue spread is 1.7×10^5.

In summary, if an offline method can be found to estimate $E[u(n)]$ and $E[y(n)]$ accurately, this method holds promise, but its effectiveness decreases rapidly as the estimates a and β become less accurate.

b) Reducing Means via Constantly Updating Estimates: One obvious method for estimating $E[u(n)]$ and $E[y(n)]$ is by directly calculating sample means. The most common method is using a single-pole filter. If the sample means of $u(n)$ and $y(n)$ are notated $u_{\mathrm{SM}}(n)$ and $y_{\mathrm{SM}}(n)$ respectively, then

$$u_{\mathrm{SM}}(n) = u_{\mathrm{SM}}(n-1)(1-\delta) + \delta u(n) \tag{13}$$

$$y_{\mathrm{SM}}(n) = y_{\mathrm{SM}}(n-1)(1-\delta) + \delta y(n) \tag{14}$$

where $0 < \delta < 1$.

The sample means $u_{\mathrm{SM}}(n)$ and $y_{\mathrm{SM}}(n)$ then replace α and β in Fig. 10. As shown in Fig. 12. Generally no DC tap is needed (y_{DC} is eliminated from $\mathbf{y}$ and $\mathbf{y}*$). From Fig. 12, the necessary and sufficient condition that $E[y^*(n \,|\, n - V - d)] = E[y(n)]$ is that $q * (n)$ is given by ([24])

$$q^*(n) = E[u(n)] - E[y(n)] \sum_{i=0}^{dQ} \hat{q}_i(n). \tag{15}$$

However, there is a problem. Signal $q^*(n)$ creates feedback paths not readily observable in Fig. 12. With $\breve{u}(n) \equiv u(n) - u_{\mathrm{SM}}(n)$, and $\breve{y}(n)$ similarly defined, redrawing Fig. 12 gives Fig. 13, where the feedback path is plainly shown.

Several simulations expose the unstable behavior suggested by Fig. 13. When the closed-loop poles and zeros of this system are periodically plotted during system convergence, it is clear that unstable performance occurs when the closed-loop poles fall outside the unit-circle during the convergence interval. Unsurprisingly, stable performance is more likely as δ is decreased, e.g., below .001. This has the effect of nearly breaking the feedback path shown in Fig. 13. However, as δ is decreased, the sample-mean estimates $u_{\mathrm{SM}}(n)$ and $y_{\mathrm{SM}}(n)$ take much longer to converge to good estimates of $E[u(n)]$ and $E[y(n)]$. As a result $\breve{u}(n)$ and $\breve{y}(n)$ take longer to become approximately zero-mean signals, thus the eigenvalue spread of $E[\breve{\mathbf{y}}(n)\breve{\mathbf{y}}(n)^T]$ remains large and $\hat{\mathbf{Q}}_{\mathbf{lin}}(n)$ converges very slowly.

c) Reducing Means via Downsampled Estimates: The optimal strategy is now presented. To break the feedback path shown in Fig. 13 and thus avoid instability, use significantly down-sampled versions of $u_{\mathrm{SM}}(n)$ and $y_{\mathrm{SM}}(n)$ for DC correction. Specifically, run the identification process as shown in Fig. 12, but update $q^*(n)$ at a down-sampled rate

$$\begin{aligned} q^*(n) &= u_{\mathrm{SM},q*}(n) - y_{\mathrm{SM},q*}(n) \\ &\qquad \times \sum_{i=0}^{dQ} \hat{q}_i \left(\left\lfloor \frac{n}{\mathrm{dsInterval}} \right\rfloor \mathrm{dsInterval} \right) \\ u_{\mathrm{SM},q*}(n) &\equiv u_{\mathrm{SM}} \left(\left\lfloor \frac{n}{\mathrm{dsInterval}} \right\rfloor \mathrm{dsInterval} \right) \\ y_{SM,q*}(n) &\equiv y_{\mathrm{SM}} \left(\left\lfloor \frac{n}{\mathrm{dsInterval}} \right\rfloor \mathrm{dsInterval} \right) \end{aligned} \tag{16}$$

where $\lfloor x \rfloor$ is the integer part of x and dsInterval is an integer down-sample interval.

By infrequently latching the values of $u_{\mathrm{SM},q*}(n)$ and $y_{\mathrm{SM},q^*}(n)$ used for determining $q^*(n)$, the feedback paths of Fig. 13 are essentially broken. For example, Fig. 14 shows the case when $\mathrm{dsInterval} = 500$, i.e., $q^*(n)$ is updated once per 500 ms. The final measured eigenvalue spread is 6. The convergence rate is satisfactorily fast.

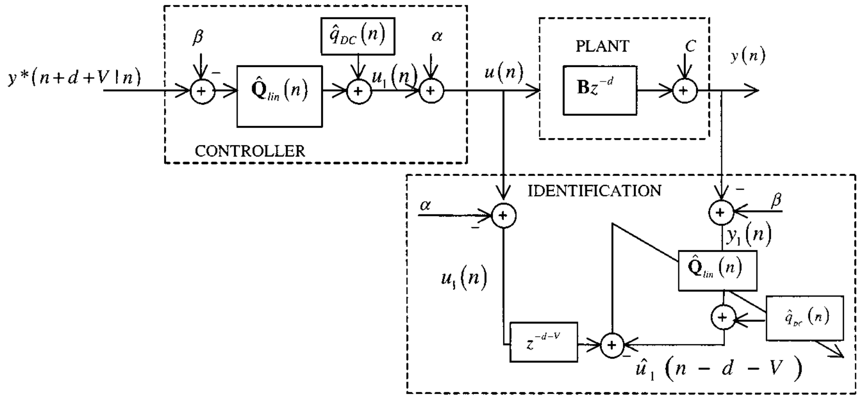
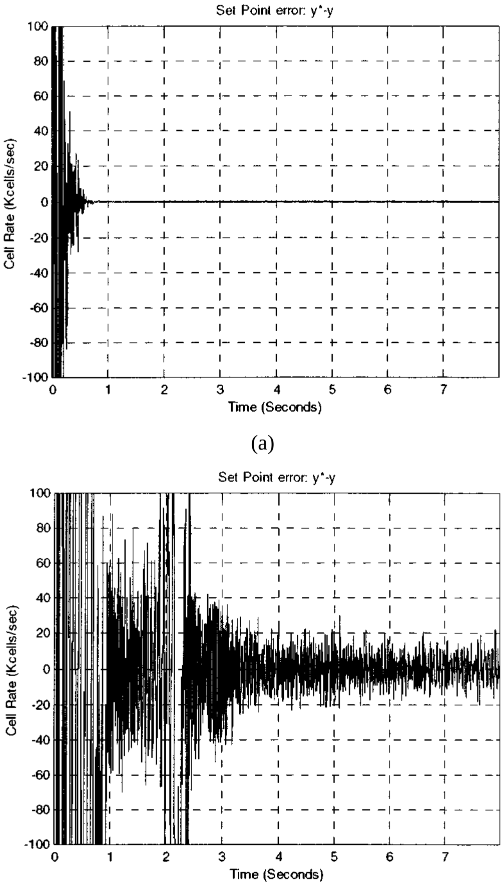


Fig. 10. Architecture for adding and subtracting fixed estimates of the means.



(a)

(b)

Fig. 11. (a) Set-Point error when estimates α and β are within 1% of their correct values. (b) Set-Point error when estimates α and β are within 10% of their correct values.

3) Discussion: Convergence rate is a serious issue for the proposed explicit rate congestion controller. Without modifications, performance is unacceptable (Fig. 9). If accurate estimates of $E[u(n)]$ and $E[y(n)]$ can be obtained *a priori*, fixed estimates provide excellent performance [Fig. 11(a)], but if these fixed estimates are less accurate, performance degrades severely [Fig. 11(b)]. An online sample mean calculation works quite well, as long as the feedback path of Fig. 13 is broken by downsampling the DC correction update (Fig. 14).

B. Control of Queue Size

Congestion Control work done by control theorists, e.g., [4]–[7], [9], often explicitly include queue matching in addition to rate matching in their cost functions, no doubt in part a response to [7]. In contrast, Section II-B presents a pure rate-matching controller, a strategy supported by [8]. This strategy requires that the bandwidth available for ABR traffic be slightly under-utilized, thus creating extremely short (or zero) queue lengths in steady state. While this has advantages, e.g., shorter end-to-end delay and smaller memory requirements, it may be more desirable to have, on average, longer queue depths. Since ABR is not designed for delay-sensitive traffic, it may be preferable to target a nonzero buffer size in order to ensure network efficiency. The scheme presented thus far does not allow for a desired queue depth greater than zero.

Queue control is fairly easily incorporated into rate-matching schemes. The basic idea, suggested by [13], is to use any preferred rate-matching scheme to determine an explicit rate. This explicit rate is then increased if the present queue depth is below its target, or decreased if the present queue depth is above the target.

The proposal of this section is distinct from [13] in that it scales the set point, $y^*(n+d+V \mid n)$, not the explicit rate $u(n)$ directly. Specifically, decide at time n the target input rate for time $n+d+V$, but notate this as $\Theta(n+d+V \mid n)$ instead of $y^*(n+d+V \mid n)$. The target input rate $\Theta(n+d+V \mid n)$ is chosen without regard of the queue size. Further, for simplicity of presentation, assume that $\Theta(n+d+V \mid n)$ is the actual service capacity for ABR traffic at time $n+d+V$. Define a scalar $\eta(n)$ that is monotonically decreasing function of the queue size $\mathrm{queue}(n)$. Control of this queue size is accomplished by multiplying $\Theta(n+d+V \mid n)$ by $\eta(n)$ to form $y^*(n+d+V \mid n)$, i.e.,

$$y^*(n+d+V \mid n) = \eta(n)\Theta(n+d+V \mid n). \qquad (17)$$

This queue-aware set-point $y^*(n+d+V \mid n)$ is used in exactly the same way as outline in Sections III-A2a and III-A2c. The

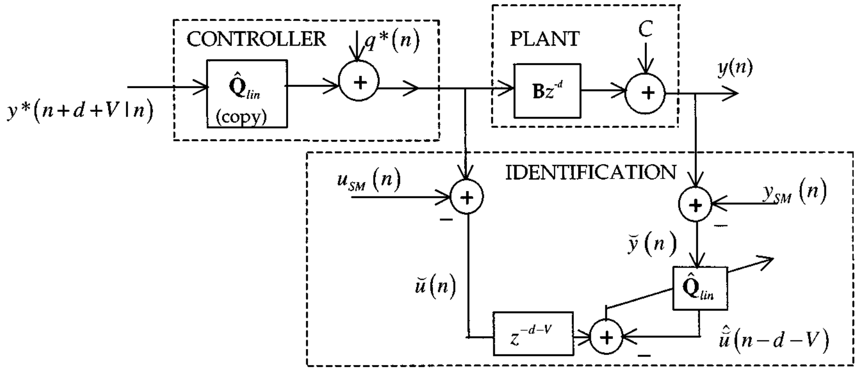


Fig. 12. Architecture for subtracting sample mean estimates.

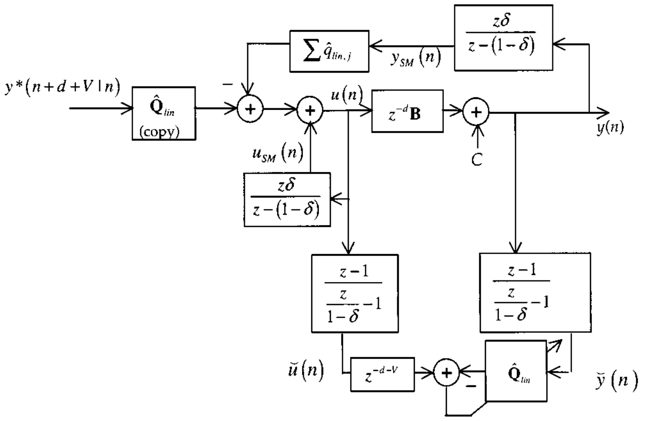


Fig. 13. Architecture for subtracting sample mean estimates, as shown in Fig. 12 with feedback explicitly shown.

plant model now includes the queue-depth $\text{queue}(n)$, which progresses[6] as

$$\text{queue}(n+1) = \text{queue}(n) + y(n) - \Theta(n\,|\,n-d-V). \quad (18)$$

Taking the constant mean estimate method of Section III-A2a (shown by Fig. 10) and incorporating (17) and (18) produces Fig. 15.

To illustrate the queue control provided by (17), reconsider the example discussed in Section III-A2a, where accurate constant estimates $a = 0.99E[u(n)]$ and $\beta = 1.01E[y(n)]$ are used to reduce the convergence rate. The set-point error is shown in Fig. 11(a). In one example (not shown here), with no attempt to control the size of the queue, i.e, $\eta(n) = 1$, the queue grows to just over 5000 cells. To target a nonzero queue-depth, use a $\eta(n)$ function that decreases monotonically with $\text{queue}(n)$. A sample function is shown in Fig. 16.

Using the function shown in Fig. 16, with $\text{queue\_scale\_bound} = 0.01, Q_1 = 100$ cells, $Q_2 = 200$ cells, $Q_3 = 300$ cells, the target-queue-depth is achieved the without perceptibly affecting the convergence rate, as shown in Fig. 17.

Comparing Figs. 10–15, clearly potentially destabilizing feedback is created by performing queue control with (17). As discussed in Section II-C, the controller $\hat{\mathbf{Q}}(n)$ lacks the ability to stabalize unstable behavior in the system. The integral action produced by explicitly modeling the queue

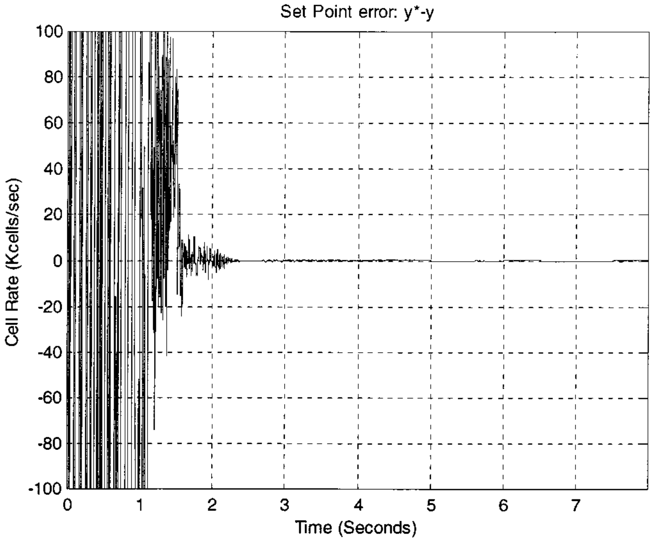


Fig. 14. Set-point error when $Q^*(n)$ is updated twice a second.

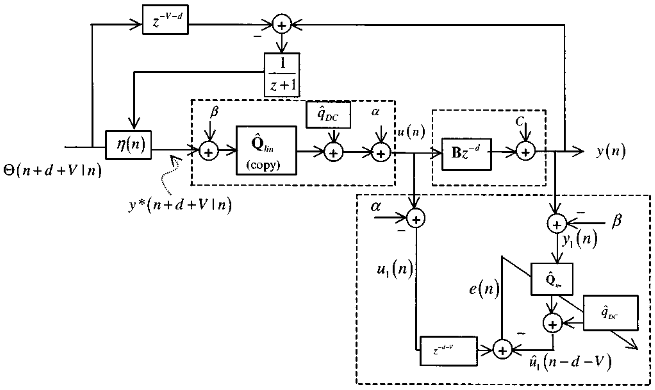


Fig. 15. Queue control added to controller of III-A.2.A. Compare to Fig. 10.

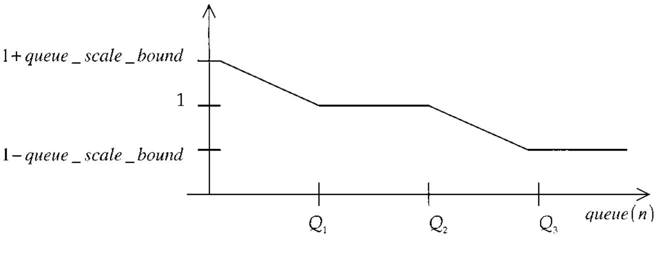
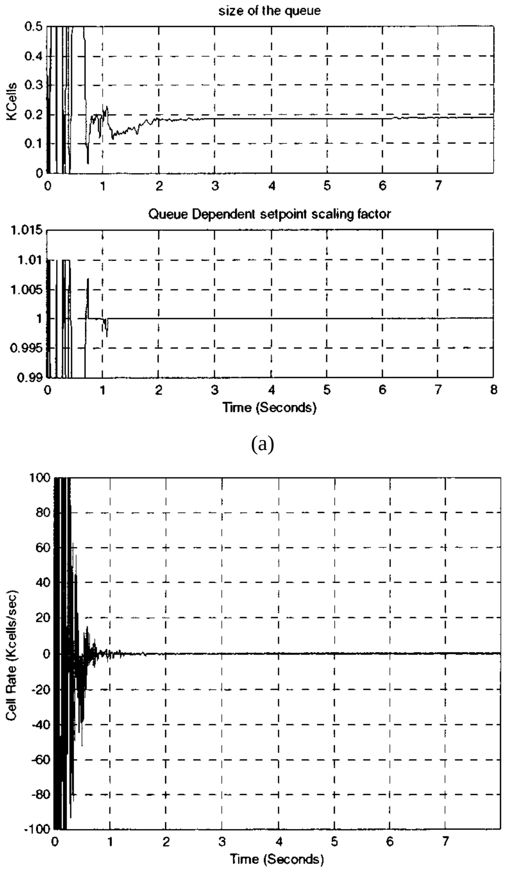


Fig. 16. Sample $\eta(n)$ Function.

[6] For ease of presentation, we ignore the saturation nonlinearity of the queue.



(a)

(b)

Fig. 17. (a) Queue depth (upper plot) and Set-point scaling factor $\eta(n)$ (lower plot) when queue target is 100–200 cells, $\sigma_{y^*} = 22$. (b) Set point error when queue depth is actively controlled. Estimates a and β are within 1% of their correct values. Note that this is comparable to Fig. 11(a).

in the plant, then incorporating queue size in the controller, makes unstable behavior a possibility. Intuition suggests, and simulations confirm, that stability is only in jeopardy when the scaling of $\eta(n)$ is aggressive. Stability is maintained, using the $\eta(n)$ shown in Fig. 16, with queue_scale_bound equal to 0.01. However, if we change queue_scale_bound from 0.01 to 0.1, simulations show [24] that the oscillations introduced significantly impact overall performance. It seems intuitive that using a small queue_scale_bound can make the impact of $\eta(n)$ on $y^*(n+d+V \mid n)$ nearly negligible, yet still effect the desired behavior.

C. Biasing Issues

The third algorithm enhancement responds to an enhancement in the plant model. The enhanced model generalizes the behavior of the nonresponsive ABR sources, allowing them nonconstant rates. This is modeled as a noise source in the plant model. This noise causes biasing in the parameter estimates used for the controller. A novel method to minimize the bias is introduced. Unlike previously published remedies for bias, this solution requires only a trivial amount of added calculations. Further, unlike other methods, this new method does not jeopardize convergence.

1) Generalizing the Plant by Incorporating Noise: Until this point, ABR traffic that is nonresponsive to the explicit rate $u(n)$ of port j has been characterized as a constant C (see (3)). This characterization is plausible if the nonresponsiveness is due to certain source characteristics. For example, a source may be entitled to a minimum cell rate (MCR) that exceeds the explicit rates proposed by port j, or the source provides data at a fixed rate below the offered explicit rate of port j. However, an ABR source may be nonresponsive to port j because it is responsive to another port $i(\neq j)$ of another switch. The explicit rates of port i are no more likely to be constant than those of port j. Therefore a more realistic traffic model for port j has nonzero variance in its nonresponsive traffic. Specifically, a zero-mean, white Gaussian noise signal $\varpi(n)$ which is uncorrelated to $u(n)$, is added to the plant (3)

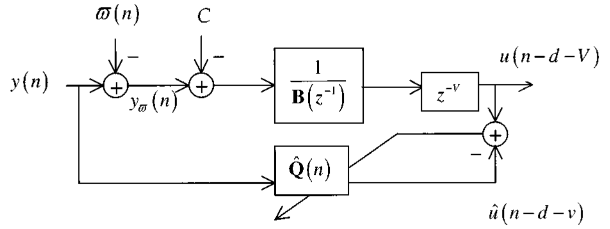


Fig. 18. Identification process incorporating plant noise $\varpi(n)$.

$$y(n) = \mathbf{B}^T\mathbf{u}(n-d) + C + \varpi(n) \tag{19}$$

The signal $(C + \varpi(n))$ can be viewed as the nonresponsive traffic having mean C and variance σ_ϖ^2. Let $y_\varpi(n) \equiv y(n) - \varpi(n)$ be the plant output without noise. Fig. 18 shows the modified identification process incorporating the plant noise.

A parameter estimation process is said to be biased if the mean of the estimates are not equal to the parameters being estimated. In the Appendix, the controller identification process of (5)–(9) is shown to converge to its Weiner solution. For the noiseless case $(\varpi(n) = 0), y(n) = y_\varpi(n)$ and the unbiased Weiner solution $\mathbf{Q}_{\mathrm{UB}}$ is

$$\begin{aligned}\mathbf{Q}_{\mathrm{UB}} &\equiv \mathbf{Q}_0\\ &= \{E[\mathbf{y}(n)\mathbf{y}(n)^T]\}^{-1}E[\mathbf{y}(n)u(n-d-V)]\\ &= \left\{E\left[\mathbf{y}_\varpi(n)\mathbf{y}_\varpi(n)^T\right]\right\}^{-1}E[\mathbf{y}_\varpi(n)u(n-d-V)].\end{aligned} \tag{20}$$

When $\varpi(n) \neq 0$, the biased Weiner solution $\mathbf{Q}_B$ is

$$\begin{aligned}\mathbf{Q}_B &= \{E[\mathbf{y}(n)\mathbf{y}(n)^T]\}^{-1}E[\mathbf{y}(n)u(n-d-V)]\\ &= \left\{E\left[\mathbf{y}_\varpi(n)\mathbf{y}_\varpi(n)^T\right] + \sigma_\varpi^2\mathbf{I}\right\}^{-1}\\ &\quad\times E[\mathbf{y}_\varpi(n)u(n-d-V)].\end{aligned} \tag{21}$$

Clearly $\mathbf{Q}_B \neq \mathbf{Q}_{\mathrm{UB}}$ when $\varpi(n) \neq 0$.

2) Related Work: The biasing effect of $\varpi(n) \neq 0$ on adaptive approximate inverse control was previously reported [39]–[42]. The accompanying recommendations focus on adding a second adaptive filter $\hat{\mathbf{B}}(n)$, which includes a DC tap, to estimate the plant. This estimate will be unbiased, as the noise $\varpi(n)$ occurs on the output of the estimated plant $(\mathbf{B})$.

Fig. 19 shows a controller estimation process that identifies $\hat{\mathbf{Q}}(n)$ with $\hat{\mathbf{B}}(n)$ in place of the true plant $\mathbf{B}$. The scheme of

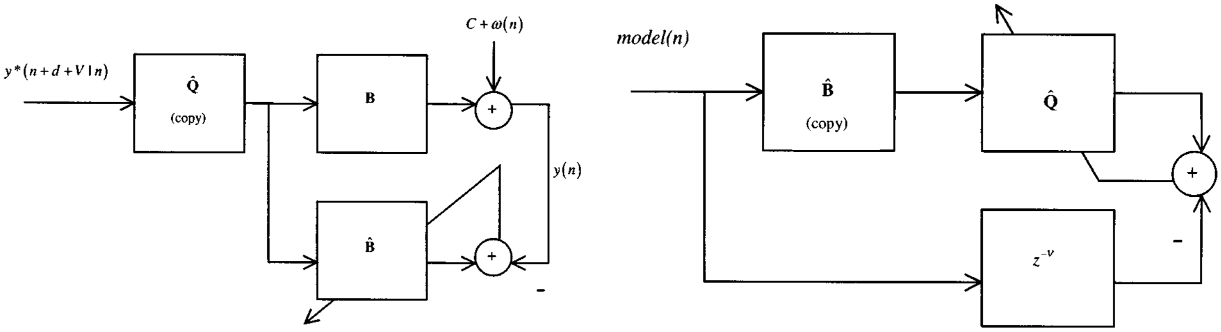


Fig. 19. A method for removing bias from $\hat{\mathbf{Q}}(n)$ [39]. The left figure is used to estimate $\hat{\mathbf{B}}$. The right figure depicts an offline process to estimate $\hat{\mathbf{Q}}$ using the estimate $\hat{\mathbf{B}}$ from the left figure.

Fig. 19 has intuitive merit, yet lacks complete analysis. Preliminary, often heuristic, results are presented in [39]. The possibility of poor estimates of $\hat{\mathbf{B}}(n)$ motivates yet another architecture (see [39, Ch. 7]) to reduce the sensitivity of $\hat{\mathbf{Q}}(n)$ to the parameter errors in $\hat{\mathbf{B}}(n)$. However, this new architecture filters its adaptation error, and thus cannot be assured to converge. Other possible solution to the biasing problem exist in the literature including the pseudolinear regression algorithm [44] and the simple hyperstable adaptive recursive filter (SHARF) algorithm [28]. Both require added computational complexity and care to ensure convergence.

3) Reducing Estimation Bias: This section presents a novel method for reducing the biasing effect of plant noise described in Section III-C1. Unlike the previous suggestions of Section III-C2, this strategy does not require additional adaptive filter coefficients, e.g., $\hat{\mathbf{B}}(n)$ in Fig. 19 (or $\hat{\mathbf{C}}(n)$ of the pseudolinear regression algorithm), and is thereby computationally less expensive. Further, this bias-reducing strategy poses no threat to global stability, as was the case with the methods of Section III-C2.

The strategy employed is reparameterization. Instead of adaptively finding $\hat{\mathbf{Q}}(n)$ by estimating $\hat{u}(n-d-V)$ as in (5), repeated here

$$\begin{aligned}\hat{u}(n-d-V) &= \mathbf{y}(n)^T\hat{\mathbf{Q}}(n)\\ &= \hat{q}_0(n)y(n) + \hat{q}_1(n)y(n-1)\\ &\quad + \cdots + \hat{q}_{dQ}(n)y(n-dQ) \qquad (22)\end{aligned}$$

use the following reparameterized adaptive model to estimate $y(n-\pi)$:

$$\begin{aligned}\hat{y}(n-\pi) &= \hat{\theta}_0(n)u(n-d-V)\mathrm{scale}_{yu} + \hat{\theta}_1(n)y(n)\\ &\quad + \hat{\theta}_2(n)y(n-1) + \cdots + \theta_\pi(n)y(n-(\pi-1))\\ &\quad + \hat{\theta}_{\pi+1}(n)y(n-(\pi+1))\\ &\quad + \cdots + \hat{\theta}_{dQ}(n)y(n-dQ)\\ &= \boldsymbol{\varphi}(n)^T\hat{\boldsymbol{\theta}}(n) \qquad (23)\end{aligned}$$

for some appropriately chosen integer $\pi, 0 \leq \pi \leq dQ$, and

$$\begin{aligned}\boldsymbol{\varphi}(n) = [&u(n-d-V)\mathrm{scale}_{yu}, y(n), \ldots, y(n-(\pi-1)),\\ &y(n-(\pi+1)), \ldots, y(n-dQ)]^T\end{aligned}$$

where scale_{yu} is an operator chosen constant (discussed below). NLMS adaptation is performed using

$$e_{y_\pi}(n) = y(n-\pi) - \hat{y}(n-\pi), \qquad (24)$$

$$\hat{\boldsymbol{\theta}}(n+1) = \hat{\boldsymbol{\theta}}(n) + \frac{\mu\boldsymbol{\varphi}(n)e_{y_\pi}(n)}{\boldsymbol{\varphi}(n)^T\boldsymbol{\varphi}(n)}. \qquad (25)$$

For each $n, \hat{\boldsymbol{\theta}}(n+1)$ is translated into the controller FIR $\hat{\mathbf{Q}}^\theta(n+1)$ using

$$\begin{aligned}\hat{\mathbf{Q}}^\theta(n+1) = &\frac{1}{\hat{\theta}_0(n+1)\mathrm{scale}_{yu}}[-\hat{\theta}_1(n+1), -\hat{\theta}_2(n+1), \ldots,\\ &-\hat{\theta}_\pi(n+1), 1, -\hat{\theta}_{\pi+1}(n+1), \ldots, -\hat{\theta}_{dQ}(n+1)]^T \qquad (26)\end{aligned}$$

Note that (23)–(25) do not attempt to include a characterization of the noise, nor attempt to otherwise filter the adaptation error. Such techniques, including those of [44] and [39], often require strictly positive-real (SPR) assumptions on the "noise filter" or some other plant aspect. Violation of such an assumption compromises convergence, both theoretically and practically. By avoiding any adaptation error filtering, the reparameterized adaptation of (23)–(25) will converge to its Weiner Solution. This Weiner Solution will be biased, but as shown in the following, the biasing is decreased for the reparameterized case as compared to the nonreparameterized case.

Choosing $\pi = 0$ creates numerical problems in calculating (23) and (26). This is the reason for choosing a nonzero π for the purpose of estimating $Q_0(z^{-1})$. Ideally π is chosen as $\pi = \arg\max_j |q_{0j}|$, although any π such that $|q_{0,\pi}|$ is "relatively large" will avoid numerical problems.

For the noiseless case, i.e., $\varpi(n) = 0$ both the original nonreparameterized adaptation scheme (5)–(8) and the reparameterized scheme (23)–(25) have unbiased Weiner Solutions. Let the unbiased Weiner Solution for the nonreparameterized case and reparameterized case be Q_{UB} and $\boldsymbol{\theta}_{\mathrm{UB}}$, respectively

$$\mathbf{Q}_{\mathrm{UB}} = \{E[\mathbf{y}_\varpi(n)\mathbf{y}_\varpi(n)^T]\}^{-1}E[\mathbf{y}_\varpi(n)u(n-d-V)] \qquad (27)$$

$$\boldsymbol{\theta}_{\mathrm{UB}} = \{E[\boldsymbol{\varphi}_\varpi(n)\boldsymbol{\varphi}_\varpi(n)^T]\}^{-1}E[\boldsymbol{\varphi}_\varpi(n)y_\varpi(n-\pi)]. \qquad (28)$$

Further, define the transformation of $\boldsymbol{\theta}_{\mathrm{UB}}$ to $\mathbf{Q}^{\theta}_{\mathrm{UB}}$ as

$$\mathbf{Q}^{\theta}_{\mathrm{UB}} \equiv \frac{1}{\mathrm{scale}_{yu}\theta_{\mathrm{UB},0}}[-\theta_{\mathrm{UB},1}, -\theta_{\mathrm{UB},2}, \ldots \\ -\theta_{\mathrm{UB},\pi}, 1, -\theta_{\mathrm{UB},\pi+1}, \ldots, -\theta_{\mathrm{UB},dQ}]^T. \quad (29)$$

Note that $\mathbf{Q}^{\theta}_{\mathrm{UB}} = \mathbf{Q}_{\mathrm{UB}}$ if perfect inversion of $\mathbf{B}$ by $\mathbf{Q}_0$ is assumed.

When $\varpi(n) \neq 0$, the Weiner solutions for both the nonreparameterized case $\mathbf{Q}_B$ and reparameterized case $\boldsymbol{\theta}_B$ are biased.

$$\mathbf{Q}_B = \left\{E[\mathbf{y}_{\varpi}(n)\mathbf{y}_{\varpi}(n)^T] \\ +\sigma^2_{\varpi}\mathbf{I}\right\}^{-1} E[\mathbf{y}_{\varpi}(n)u(n-d-V)] \quad (30)$$

$$\boldsymbol{\theta}_B = \left\{E[\boldsymbol{\varphi}_{\varpi}(n)\boldsymbol{\varphi}_{\varpi}(n)^T] \\ +\sigma^2_{\varpi}\mathrm{diag}\{0, 1, 1, \ldots, 1\}\right\}^{-1} \\ \times E[\boldsymbol{\varphi}_{\varpi}(n)y_{\varpi}(n-\pi)] \quad (31)$$

where $\boldsymbol{\varphi}_{\varpi}(n)$ is defined

$$\boldsymbol{\varphi}_{\varpi}(n) \equiv [u(n-d-V)\mathrm{scale}_{yu}, y_{\varpi}(n), \ldots, \\ y_{\varpi}(n-(\pi-1)), y_{\varpi}(n-(\pi+1)), \ldots, y_{\varpi}(n-dQ)]^T.$$

Noting the bias error in $\mathbf{Q}_B$ and $\boldsymbol{\theta}_B$ as vectors $\mathbf{Q}_{\mathrm{BE}} \equiv \mathbf{Q}_B - \mathbf{Q}_{\mathrm{UB}}$ and $\boldsymbol{\theta}_{\mathrm{BE}} \equiv \boldsymbol{\theta}_B - \boldsymbol{\theta}_{\mathrm{UB}}$, then from (30) and (31)

$$\mathbf{Q}_{\mathrm{BE}} = -\left\{E[\mathbf{y}_{\varpi}(n)\mathbf{y}_{\varpi}(n)^T] + \sigma^2_{\varpi}\mathbf{I}\right\}^{-1} \sigma^2_{\varpi} \begin{bmatrix} q_{\mathrm{UB},0} \\ \vdots \\ q_{\mathrm{UB},\pi} \\ \vdots \\ q_{\mathrm{UB},dQ} \end{bmatrix} \quad (32)$$

and

$$\boldsymbol{\theta}_{\mathrm{BE}} = \left\{E[\boldsymbol{\varphi}_{\varpi}(n)\boldsymbol{\varphi}_{\varpi}(n)^T] + \sigma^2_{\varpi} \begin{bmatrix} 0 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}\right\}^{-1} \\ \times \frac{\sigma^2_{\varpi}}{q^{\theta}_{\mathrm{UB},\pi}} \begin{bmatrix} 0 \\ q^{\theta}_{\mathrm{UB},0} \\ \vdots \\ q^{\theta}_{\mathrm{UB},\pi-1} \\ q^{\theta}_{\mathrm{UB},\pi+1} \\ \vdots \\ q^{\theta}_{\mathrm{UB},dQ} \end{bmatrix}. \quad (33)$$

It is possible to translate $\boldsymbol{\theta}_{\mathrm{BE}}$ into an analytical expression for the bias error $\mathbf{Q}^{\theta}_{\mathrm{BE}}$. However, the nonlinearity of the translation (26) and (29) obscures any added intuition provided by such an analytical expression. Instead, what follows are heuristic arguments claiming that σ^2_{ϖ} has a larger biasing effect on $\mathbf{Q}_B$ than on $\boldsymbol{\theta}_B$, and thus $\mathbf{Q}^{\theta}_B$.

Consider the large σ^2_{ϖ} case. As σ^2_{ϖ} increases, (32) indicates that $\mathbf{Q}_{\mathrm{BE}} \to -\mathbf{Q}_{\mathrm{UB}}$, or $\mathbf{Q}_B \to 0$. Such a controller produces an all-zero control signal, i.e., doing nothing is better than attempting any nontrivial control, the biasing effect is so great. In contrast, as σ^2_{ϖ} becomes large in (33), the matrix

$$\left\{E[\boldsymbol{\varphi}_{\varpi}(n)\boldsymbol{\varphi}_{\varpi}(n)^T] + \sigma^2_{\varpi}\mathrm{diag}\{0, 1, 1, \ldots, 1\}\right\} \quad (34)$$

becomes increasingly diagonal. (It also becomes increasingly ill-conditioned, but avoids singularity since $E[u(n-d-V)^2] \neq 0$.) As (34) becomes more diagonal, from (33), $\theta_{\mathrm{BE},0}$, the first term of $\boldsymbol{\theta}_{\mathrm{BE}}$, becomes close to zero. Surprisingly, as the noise increases, $\theta_{B,0} \approx \theta_{\mathrm{UB},0}$ and thus $Q^0_{B,\pi}$ is only slightly biased (and not equal to zero, as in the nonreparameterized case). By construction, the π'th tap of the controller is one of its most significant taps.

Before presenting the simulation results, a few comments on scale_{yu} are in order. The constant scale_{yu} should be chosen to reduce eigenvalue spread of the auto-correlation matrix $E[\boldsymbol{\varphi}(n)\boldsymbol{\varphi}(n)^T]$. As discussed in Section III-A.2, reducing the eigenvalue spread of the auto-correlation matrix is desirable as this reduces the convergence time. One possible measurement-based scheme is $\mathrm{scale}_{yu} = \sqrt{\tilde{\sigma}^2_y/\tilde{\sigma}^2_y}$, where $\tilde{\sigma}^2_y$ and $\tilde{\sigma}^2_u$ are sample-mean estimates of the variance of y and u respectively.

The simulation experiments presented below demonstrate the reduction of bias that occurs with reparameterization. As in Section II-C.1, $B(z^{-1}) = z^{-10}(2 + 9z^{-1} + 8z^{-2} + 3z^{-3}), C = 200, dQ = 30, V = 10$. The sample time is $T_s = 1$ msec. The bandwidth available for explicit rate traffic, $y^*(n\,|\,n-d-V)$, is modeled as a Gaussian random process with $E[y^*(n\,|\,n-d-V)] = 1$ Mcps, $\sigma^2_{y^*} = 484$ Kcps. When reparameterization is performed, $\pi = 9$, as this is the largest magnitude tap of $\mathbf{Q}_{\mathrm{UB}}$ (Fig. 20). To reduce the eigenvalue spread of the autocorrelation matrix, the method of *reducing means via downsampled estimates* (Section III-A.2.C) is used.

When the plant output noise $\varpi(n)$ is a zero-mean, Gaussian random process with variance $\sigma^2_{\varpi} = 120$ Kcps, without reparameterization, biasing is pronounced. Fig. 20(a) shows the impulse response the parameter estimate $\hat{\mathbf{Q}}$ and the optimal, unbiased $\mathbf{Q}_{\mathrm{UB}}$ after 8 s (8000 samples) of convergence. The estimate $\hat{\mathbf{Q}}$ bears a poor resemblance to $\mathbf{Q}_{\mathrm{UB}}$. When $\hat{\mathbf{Q}}$ is convolved with $\mathbf{B}$, instead of the expected impulse at $V = 10$ Fig. 20(b) demonstrates that $\hat{\mathbf{Q}}$ poorly inverts $\mathbf{B}$. Comparing bode plots of $\mathbf{Q}_{\mathrm{UB}}, \mathbf{Q}_B$, and $\hat{\mathbf{Q}}$ in Fig. 20(c) shows that $\hat{\mathbf{Q}}$ does indeed closely resemble $\mathbf{Q}_B$ and poorly resembles $\mathbf{Q}_{\mathrm{UB}}$. The set point error, $y^*(n\,|\,n-d-V) - y_{\varpi}(n)$, is shown in Fig. 20(d).

In contrast, the reparameterized case, with $\pi = 9$, shows much less bias. The impulse response of $\hat{\mathbf{Q}}$ is much closer to $\mathbf{Q}^{\theta}_{\mathrm{UB}}$ as shown in Fig. 21(a). Convolution of $\hat{\mathbf{Q}}$ and $\mathbf{B}$ reveals an impulse at delay $V = 10$, as shown in Fig. 21(b). In Fig. 21(c), the bode plots of $\mathbf{Q}^{\theta}_{\mathrm{UB}}, \mathbf{Q}^{\theta}_B$, and $\hat{\mathbf{Q}}$ show that $\hat{\mathbf{Q}}$ well approximates $\mathbf{Q}^{\theta}_B$ and nearly well approximates $\mathbf{Q}^{\theta}_{\mathrm{UB}}$. The upward shift of $\hat{\mathbf{Q}}$ as compared to $\mathbf{Q}^{\theta}_{\mathrm{UB}}$ is consistent with the slight overshoot observed in the delayed impulse of Fig. 21(b). The set point error, as shown in Fig. 21(d), is noticeably superior as compared to the nonparameterized case shown in Fig. 20(d).

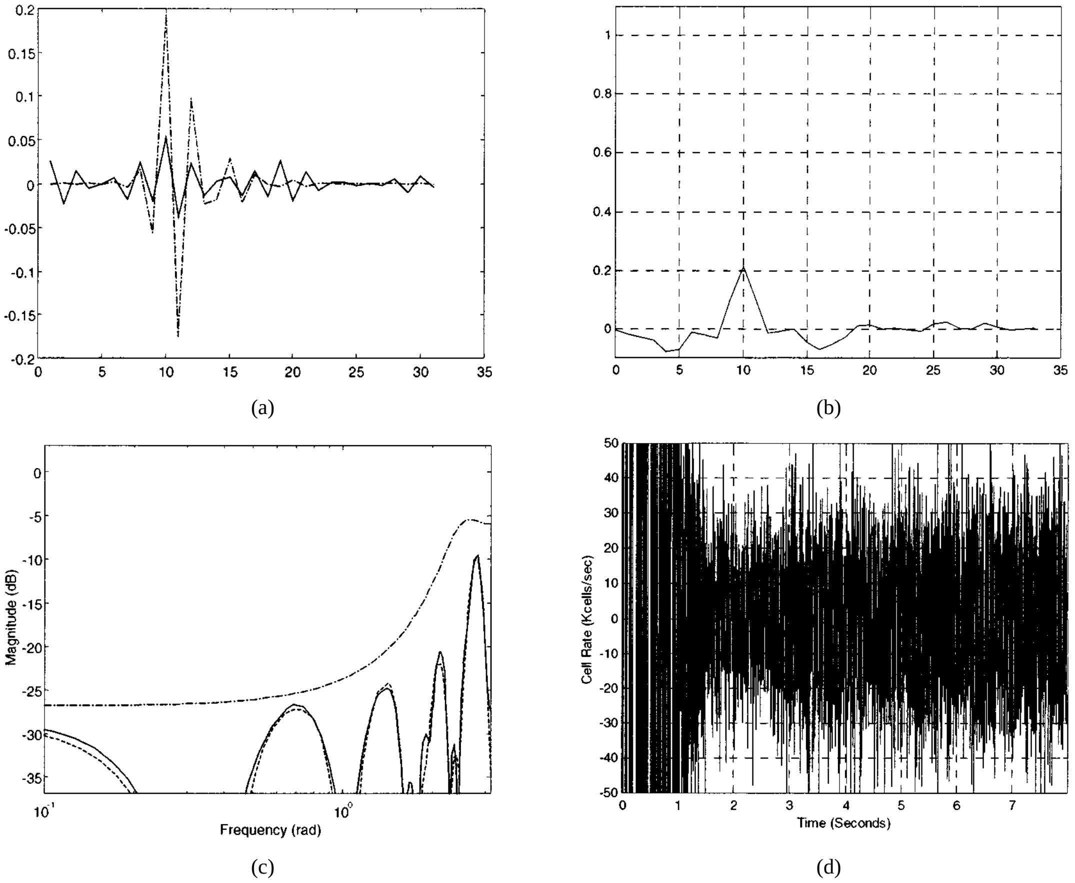


Fig. 20. (a) Impulse response of $\hat{\mathbf{Q}}$ (solid line) and $\mathbf{Q}_{\text{UB}}$ (dash-dot line). (b) Convolution of $\mathbf{B}$ and $\hat{\mathbf{Q}}$. (c) Bode plot of $\mathbf{Q}_{\text{UB}}$ (dash-dot line), $\mathbf{Q}_B$ (dashed line), and $\hat{\mathbf{Q}}$ (solid line). (d) Set-point error, $y^*(n \mid n-d-V) - y_{\varpi}(n)$, with $\sigma^2_{\varpi} = 120$ Kcps. No reparameterization.

IV. Summary and Concluding Remarks

This paper takes up the challenge of finding an effective control strategy for the explicit rate congestion controller. The problem is motivated in Section I, where other related work is summarized. The system under study is defined and compared to other schemes in Section II. The new contributions of this paper are presented in Section III. These contributions consist of algorithm enhancements to the system defined in Section II, and include convergence rate improvements, queue depth management, and a method to reduce coefficient bias without compromising convergence or significantly increasing computational complexity.

There are several potential directions for future research. One path would examine real-world protocols and networks in an attempt to improve the fidelity of the plant model. This will almost certainly create a more complex plant model. Modeling the blending effect introduced in [24] is but one possibility. Other modeling extensions include delayed or lost data (e.g., resource management cells), nonlinearities due to rate and buffer saturations, bursty sources, and other phenomena.

Appendix

This appendix provides hereto unpublished as well as a summary of published convergence and stability results for the proposed explicit-rate ATM ABR system given by (3)–(9) and shown in Fig. 5.

In [20], the adaptive process for the controller $\hat{\mathbf{Q}}(n)$ given by (5)–(9) is guaranteed to converge stably to its optimal solution $\mathbf{Q}_0$ (defined by (10)). Specifically, [20] defines $\Upsilon(n) \equiv [y(n), y(n-1), \ldots, y(n-dQ)]^T$ and $\tilde{\mathbf{Q}}(n) \equiv \hat{\mathbf{Q}}(n) - \mathbf{Q}_0$, and makes and justifies the following four assumptions: *Assumption 1* is Gaussian; *Assumption 2* $\mathbf{y}(n)$ and $\tilde{\mathbf{Q}}(n)$ are independent. Also $u(n-V-d)$ and $\tilde{\mathbf{Q}}(n)$ are independent; *Assumption 3* The auto-covariance matrix, $\boldsymbol{\sigma}^2 \equiv E[(\Upsilon(n) - E[\Upsilon(n)])(\Upsilon(n) - E[\Upsilon(n)])^T]$, is full rank; *Assumption 4* $\alpha_0 \le \|\mathbf{y}(n)\|^2, \alpha_0 > 0$

Then the following theorems are proved, thereby assuring that adaptive controller coefficients converge both in mean and mean-square:

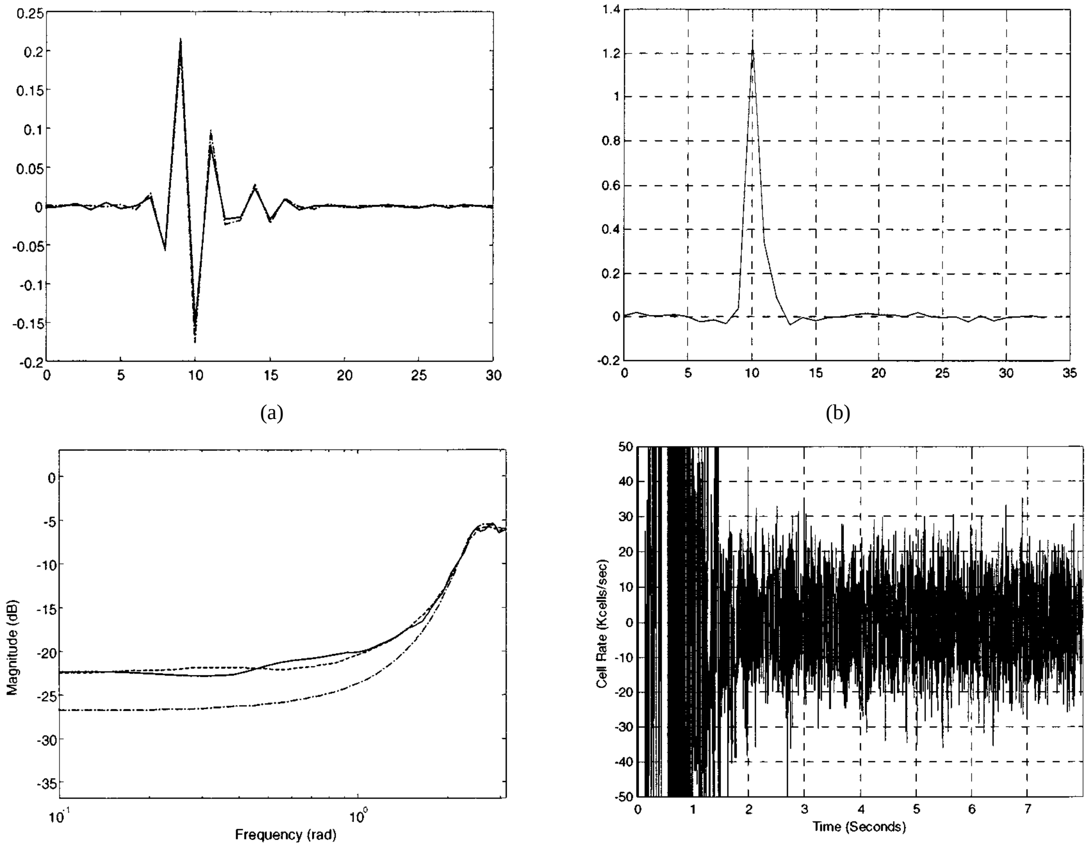
Theorem 1: Given Assumption 1–Assumption 4 and $0 < \mu < 2, \lim_{n\to\infty} E[\tilde{\mathbf{Q}}(n)] = \mathbf{0}$.

Theorem 2: Given Assumption 1–Assumption 4 and $0 < \mu < 2, \lim_{n\to\infty} E[\tilde{\mathbf{Q}}(n)\tilde{\mathbf{Q}}(n)^T] = \mu^2\varepsilon^*\mathbf{W}^T(\mathbf{I}-\mathbf{F})^{-1}\mathbf{HW}$, where W. F, and H are constant matrices, defined in [20].

The results of [20] assure that the controller converges in mean and mean-square to the optimal controller $\mathbf{Q}_0$ for any given dQ. However, given $\mathbf{B}$, for a large enough dQ, $\hat{\mathbf{Q}}(n)$ will nearly perfectly invert $\mathbf{B}$ (simulations in Section II-C.2 suggest that dQ is on the order of 30). Consider the realistic case when dQ is chosen large enough to make the following assumption.

Assumption 5: $B(z^{-1})$ has no zero on $|z| = 1$ and the plant (2) is equivalently expressed as

$$Q(z^{-1})y(n) = u(n-d-V). \quad \text{(A.1)}$$



(a) (b)

(c) (d)

Fig. 21. (a) Impulse response of $\hat{\mathbf{Q}}$ (solid line) and $\mathbf{Q}_{\rm UB}$ (dashed line). (b) Convolution of $\mathbf{B}$ and $\hat{\mathbf{Q}}$. (c) Bode plot of $\mathbf{Q}_{\rm UB}$ (dash-dot line), $\mathbf{Q}_B$ (dashed line), and $\hat{\mathbf{Q}}$ (solid line). (d) Set-point error, $y^*(n \,|\, n-d-V) - y_{\varpi}(n)$ with $\sigma_{\varpi}^2 = 120$ Kcps. Using reparameterization. $\pi = 9$, $\text{scale}_{yu} = 12.5$.

Here are two other assumptions.

Assumption 6: $\|\hat{\mathbf{Q}}(n)\| > 0$ for all n; *Assumption 7*: At each $n, z = e^{-j\omega'}$ is not a root of $\hat{\mathbf{Q}}(z^{-1}) = 0$ if $y^*(n)$ contains the frequency ω'.

Assumptions 6 and 7 prevent pathological cases; neither pose significant limitations in practice.

If we can make Assumption 5 as well as the minor Assumptions 6 and 7, then the more restrictive Assumptions 1–4 are not needed. This leads to a cleaner proof with stronger global stability results.

For this proof, substitute the update (8) with

$$\hat{\mathbf{Q}}(n+1) = \hat{\mathbf{Q}}(n) + \frac{\mu \mathbf{y}(n)}{\delta + \mathbf{y}(n)^T \mathbf{y}(n)} e(n), \quad \delta > 0. \quad \text{(A.2)}$$

The update (A.1) is identical to [44, (3.3.19)]. From (7), (A.1), and (5), $e(n) = -\tilde{\mathbf{Q}}(n)^T \mathbf{y}(n)$, and from [44, Lemma 3.3.2]

$$\lim_{n\to\infty} \frac{e(n)}{(\delta + \mathbf{y}(n)^T \mathbf{y}(n))^{1/2}} = 0$$
$$\lim_{n\to\infty} \|\hat{\mathbf{Q}}(n-k) - \hat{\mathbf{Q}}(n)\| = 0 \text{ for any finite } k. \quad \text{(A.3)}$$

From (7), (9), (5), and (A.3)

$$\lim_{n\to\infty} \frac{(\hat{\mathbf{Q}}(n)^T \boldsymbol{\chi}(n))^2}{\delta + \mathbf{y}(n)^T \mathbf{y}(n)} = 0 \quad \text{(A.4)}$$

where the set-point error is $\boldsymbol{\chi}(n) \equiv \mathbf{y}^*(n \,|\, n-d-V) - \mathbf{y}(n)$. Using Assumption 6

$$\|\mathbf{y}(n)\| \le \kappa_3 + \kappa_4 \max_{0 \le \tau \le n} |\hat{\mathbf{Q}}(\tau)^T \boldsymbol{\chi}(\tau)|, \quad 0 < \kappa_3, \kappa_4 < \infty \quad \text{(A.5)}$$

With (A.4) and (A.5), the key technical lemma [44] asserts that

$$\|\mathbf{y}(n)\| \text{ is bounded, and } \lim_{n\to\infty} (\hat{\mathbf{Q}}(n)^T \boldsymbol{\chi}(n))^2 = 0. \quad \text{(A.6)}$$

Theorem 3: Given Assumption 5–7, the plant (2), which is equivalent to (A.1), controlled by (5)–(7), (9) and (A.2), gives $\lim_{n\to\infty} \chi(n) = 0$.

Proof: Equation (A.3) gives (A.4). The key technical lemma gives (A.6), which, along with Assumption 7, gives the result. This completes the proof. Theorem 3 first appeared in [24]. □

References

[1] *Traffic Management Specification Version 4.1*, J. Kenney, Ed., ATM Forum.

[2] R. Jain, “Congestion control and traffic management in ATM networks: Recent advances and a survey,” *Comp. Net. ISDN Syst.*, vol. 28, pp. 1723–1738, Oct. 1996.

[3] C. Rohrs, R. Berry, and S. O’Halek, “Control engineer’s look at ATM congestion avoidance,” *Comp. Comm.*, vol. 19, no. 3, pp. 226–234, Mar. 1996.

[4] L. Benmohamed and S. M. Meerkov, "Feedback control of congestion in packet switching networks: The case of a single congested node," *IEEE/ACM Trans. Networking*, vol. 1, pp. 693–708, Dec. 1993.

[5] ——, "Feedback control of congestion in packet switching networks: The case of multiple congested nodes," *Int. J. Comm. Syst.*, vol. 10, no. 5, pp. 227–246, Sept.–Oct. 1997.

[6] J.-C. Bolot, "A self-tuning regulator for adaptive overload control in communication networks," presented at the *Proc. 31st Conf. Decision Control*, Tucson, AZ, Dec. 1992.

[7] E. Altman, F. Baccelli, and J.-C. Bolot, "Discrete-time analysis of adaptive rate control mechanisms," in *High Speed Networks and Their Performance*, H. G. Perros and Y. Viniotis, Eds. Amsterdam, The Netherlands: North Holland, 1994, pp. 121–140.

[8] O. Ait-Hellal, E. Altman, and T. Basar, "Rate based flow control with bandwidth information," in *Proc. 35th Conf. Decision Control*, Kobe, Japan, Dec. 1996.

[9] E. Altman, T. Basar, and R. Srikant, "Robust rate control for ABR sources," in *Proc. INFOCOM 98*, vol. 1, 1998, pp. 166–173.

[10] R. Jain, S. Kalyanaraman, and R. Viswanathan, "A sample switch algorithm", AF-TM 95-0178R1, Feb. 1995.

[11] ——, "The OSU scheme for congestion avoidance using explicit rate indication", AF-TM 94-0883, Sept. 1994.

[12] R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, and F. Lu, "ERICA+: Extensions to the ERICA switch algorithm", Tech. Rep. AF-TM 95-1145R1, Oct. 1995.

[13] B. Vandalore, R. Jain, R. Goyal, and S. Fahmy, "Design and analysis of queue control functions for explicit rate switch schemes," in *Proc. IC3N '98*, Lafayette, LA, Oct. 1998, pp. 780–786.

[14] S. Fahmy, R. Jain, S. Kalyanaraman, R. Goyal, and B. Vandalore, "On determining the fair bandwidth share for ABR connections in ATM networks," in *Proc. ICC 98*, vol. 3, 1998, pp. 1485–1491.

[15] O. Imer *et al.*, "ABR congestion control in ATM networks," *IEEE Control Syst. Mag.*, Feb. 2001.

[16] K. Laberteaux and C. Rohrs, "Application of adaptive control to ATM ABR congestion control," in *Proc. GLOBECOM '98*, Sydney, Australia, 1998.

[17] C. Fulton and S. Q. Li, "UT: ABR feedback control with tracking," in *Proc. INFOCOM'97*, Apr. 1997.

[18] Y. D. Zhao, S. Q. Li, and S. Sigarto, "A linear dynamic model for design of stable explicit-rate ABR control schemes", AF-TM 96-0606, Apr. 1996.

[19] S. Mascolo, "Smith principle for congestion control in high-speed data networks," *IEEE Trans. Automat. Contr.*, vol. 45, pp. 358–364, Feb. 2000.

[20] K. Laberteaux and C. Rohrs, "On the convergence of a direct adaptive controller for ATM ABR congestion control," presented at the *Proc. ICC 2000*, June 2000.

[21] ——, "A direct adaptive controller for ATM ABR congestion control," presented at the *Proc. American Control Conf.*, June 2000.

[22] K. Laberteaux, C. Rohrs, and P. Antsaklis, "A pragmatic controller for explicit rate congestion control," in *Proc. GLOBECOM*, 2001.

[23] K. Laberteaux and C. Rohrs, "A Proof of Convergence for a Direct Adaptive Controller for ATM ABR Congestion Control," Notre Dame ISIS Lab, Tech. Rep. ND-ISIS-2000-02, 2000.

[24] K. Laberteaux, "Explict Rate Congestion Control for Data Networks," Ph.D. dissertation, Dept. Elec. Eng., Univ. Notre Dame, Notre Dame, IN, 2000.

[25] ATM Forum [Online]. Available: http://www.atmforum.org

[26] S. Low and D. Lapsley, "Optimization flow control, I: Basic algorithm and convergence," *IEEE/ACM Trans. Networking*, vol. 7, pp. 861–874, Dec. 1999.

[27] R. Braden, "Requirements for internet hosts—communication layers", STD 3, Internet RFC 1122, Oct. 1989.

[28] L. Larimore, J. Treichler, and C. Johnson, "SHARF: An algorithm for adaptive IIR digital filters," *Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 428–440, Aug. 1980.

[29] S. Blake *et al.*, "An architecture for differentiated services", Internet RFC 2475, Dec. 1998.

[30] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, pp. 397–413, Aug. 1993.

[31] S. Floyd, "TCP and explicit congestion notification," *ACM Comp. Comm. Review*, vol. 24, no. 5, pp. 10–23, Oct. 1994.

[32] K. K. Ramakrishnan and S. Floyd, "A proposal to add explicit congestion notification (ECN) to IP", RFC 2481, Jan. 1999.

[33] D. Awduche, "Requirements for traffic engineering over MPLS", Internet RFC 2702, Sept. 1999.

[34] S. Athuraliya *et al.*, "REM: Active queue management," presented at the *Proc. 17th Int. Teletraffic Congress*, Sept. 2001.

[35] S. Karandikar, S. Kalyanaraman, P. Bagal, and B. Packer, "TCP rate control," *Comp. Comm. Review*, vol. 30, pp. 45–58, Jan. 2000.

[36] C. V. Hollot *et al.*, "On designing improved controllers for AQM routers supporting TCP flows," in *Proc. IEEE INFOCOM*, Apr. 2001.

[37] T. Yahagi and J. Lu, "On self-tuning control of nonminimum phase discrete-time systems using approximate inverse systems," *ASME J. Dyna. Syst., Meas, Control*, vol. 115, pp. 12–81, Mar. 1993.

[38] A. Oppenheim and R. Schafer, *Digital Signal Processing*. Upper Saddle River, NJ: Prentice-Hall, 1975.

[39] B. Widrow and E. Walach, *Adaptive Inverse Control*. Upper Saddle River, NJ: Prentice-Hall, 1996.

[40] B. Widrow and G. Plett, "Nonlinear adaptive inverse control," in *Proc. IEEE Conf. Decision Control*, vol. 2, San Diego, CA, Dec. 1997, pp. 1032–1037.

[41] ——, "'Intelligent' adaptive inverse control," in *Proc. IFAC 96*, San Francisco, CA, July 1996, pp. 104–105.

[42] ——, "Adaptive inverse control based on linear and nonlinear adaptive filtering," in *Proc. World Congress Neural Net.*, vol. 2, Dec. 1997, pp. 620–627.

[43] M. Tarrab and A. Feuer, "Convergence and performance analysis of the normalized LMS algorithm with uncorrelated gaussian data," *IEEE Trans. Inform. Theory*, vol. 34, pp. 680–691, July 1988.

[44] G. Goodwin and K. Sin, *Adaptive Filtering Prediction and Control*. Upper Saddle River, NJ: Prentice-Hall, 1984.

[45] S. Haykin, *Adaptive Filter Theory*. Upper Saddle River, NJ: Prentice-Hall, 1991.

[46] Matlab [Online]. Available: http://www.mathworks.com/

Kenneth P. Laberteaux received the B.S.E. degree in electrical engineering (*summa cum laude*) from the University of Michigan, Ann Arbor, and the M.S. and Ph.D. degrees from the University of Notre Dame, Notre Dame, IN, in 1992, 1996, and 2000, respectively.

Shortly thereafter, he joined the Tellabs Research Center, Mishawaka, IN, where he has investigated echo cancellation, equalization, data networking protocols, multicasting, switch architecture and scheduling, call admission control, and congestion control.

Dr. Laberteaux holds five patents and is a member of Tau Beta Pi.

Charles E. Rohrs (S'78–M'82–SM'88) received the B.S. degree from the University of Notre Dame, Notre Dame, IN, and the M.S and Ph.D. degrees from the Massachusetts Institute of Technology (MIT), Cambridge, MA, in 1976, 1978, and 1982, respectively.

He is a Principle Research Scientist at MIT, and is also affiliated with the Tellabs Research Center, Mishawaka, IN, which is the research arm of Tellabs Operations, Inc., a manufacturer of telecommunications equipment for public service network providers. Before becoming a Tellabs' Fellow in 1995, he was Director of the Tellabs Research Center for ten years. He also served on the faculty at the University of Notre Dame from 1982 to1997, and was a Visiting Professor at MIT from 1997 to 2000, at which time he began his current position. While best known for his early contributions to the study of robustness in adaptive control, he has also contributed work in adaptive control, adaptive signal processing, communication theory, and communication networks. He has been active in analyzing and designing traffic control schemes for communication networks. In this area, his work was among the first to apply the techniques of linear control theory to such schemes.

Panos J. Antsaklis (S'74–M'76–SM'86–F'91) is H.C. and E.A. Brosey Professor of Electrical Engineering and Director of the Center for Applied Mathematics at the University of Notre Dame, Notre Dame, IN. His work includes analysis of behavior and design of control strategies for complex autonomous, intelligent systems. His recent research focuses on networked embedded systems, and addresses problems in the interdisciplinary research area of control, computing and communication networks, and on hybrid and discrete-event dynamical systems. He has authored a number of publications in journals, conference proceedings, and books, and has edited four books on Intelligent Autonomous Control and Hybrid Systems. In addition, he has coauthored the research monograph *Supervisory Control of Discrete Event Systems Using Petri Nets* (Norwell, MA: Kluwer, 1998) and the graduate textbook *Linear Systems* (New York: McGraw-Hill, 1997). He serves on the editorial boards of several journals.

Dr. Antsaklis currently serves as Assistant Editor-at-Large for the IEEE TRANSACTIONS ON AUTOMATIC CONTROL, and has been Guest Editor of Special Issues on Hybrid Systems in the IEEE TRANSACTIONS ON AUTOMATIC CONTROL and the PROCEEDINGS OF THE IEEE. He has served as Program Chair and General Chair of major systems and control conferences, and was the 1997 President of the IEEE Control Systems Society (CSS). He is a Distinguished Lecturer of the IEEE Control Systems Society, a recipient of the IEEE Distinguished Member Award of the Control Systems Society, and an IEEE Third Millennium Medal recipient.