Characterization and Modeling of Long-Range Dependent Telecommunication Traffic

Sponsor: Sprint

Yong-Qing Lu
David W. Petr
Victor Frost

Technical Report TISL-10230-4

Telecommunications and Information Sciences Laboratory
Department of Electrical Engineering and Computer Science
University of Kansas

August 1994

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Analytical modeling of traffic sources in a packet-switched communications network is the basis of performance analysis and network design. Along with the development of high speed networks, more types of traffic are being applied to the network, such as LAN interconnection, real time video traffic and multimedia traffic with voice, data, image and video. Since the ultimate goal of a high speed network is to support quality service for various types of traffic, it is very important to understand the traffic characteristics, especially their burstiness, and provide accurate analytical models that well describe the properties. Only with a thorough understanding of the traffic and with appropriate models, is the study of network performance and effective design of networks possible. For decades, traffic modeling has been an active research area.

Since the majority of real traffic possesses correlated interarrival sequences which have a major effect on queue performance, the major issue of traffic modeling has been to build up models which best capture the correlation characteristics of different traffic.

General traffic modeling involves the following procedure:

(1) Set up a mathematical stochastic process as an approximation model for the target traffic based on its properties observed from the empirical data.

(2) Determine the parameters of the model, such that it captures the traffic properties best. This step usually involves many analytical techniques. The choice of parameter values is also critical to the accuracy of the model.
(3) Use the model to analyze the network performance (queue length, queue delay, loss probability etc.) analytically or by simulation.
(4) If possible, test the model by comparing the simulated result to the result obtained from real traffic and find out the limitations of the model.

Many traffic models with different approaches have been proposed in the literature. A detailed review of traffic models is given in [1]. All the models can be approximately grouped as follows.

Markovian Type Models: These include Markov chains and Markov Modulated Poisson Process (MMPP) [4]. The former can be either two-state Markov chain or multiple two-state Markov chain. They have been used as simple models for voice and on-off type traffic [3][4]. The multiple two-state Markov chain was considered for the variable bit rate video traffic model [4]. MMPPs have been used to model multiple on-off sources such as aggregated voice and data as well as the superposed multimedia traffic [5]. A typical feature for Markovian type models is that they become analytically very involved as the number of states gets large.

Fluid Flow Models: These are simpler models than Markovian type models. In a fluid flow model, arrivals of discrete packets (or cells in an ATM network) are treated as a continuous arrival of a liquid flow [6]. The arrivals of individual data units are ignored. Because of this, the analytical simplification of the performance analysis is significant. Its applications include aggregated on-off source and variable bit rate sources. It is an attractive and useful model for ATM network traffic. In an ATM high speed network, the individual data units are numerous relative to a chosen time scale where the arrival rate changes can be observed. Therefore the impact of individual arrivals on the network performance is negligible.

Models for variable bit rate video: These include Autoregressive Moving Average models (ARMA) [7][8] and Transform-Expand-Sample (TES) models [9][10]. The TES method can model a general autocorrelation function and simultaneously match the marginal distribution of the empirical counterpart. Models for video traffic are codec dependent.

Miscellaneous: These include chaotic map models and frequency domain approaches. Chaotic maps address the problem from a fundamentally different perspective by using deterministic, nonlinear chaotic maps as traffic models [11]. Frequency domain approaches are based on the fact that the correlation function of an input traffic stream which is a stationary random process, can be transformed into a power spectral distribution function in the frequency domain [12][13]. Queue response to input spectral properties can then be made by the spectral theory of random processes. This approach can lead to effective dynamic link allocation and is especially efficient for multimedia traffic.

These models differ from one to another in terms of application. Each one has some advantages and disadvantages. However, a common characteristic of these models (except for the chaotic map and frequency domain method) is that they are all *short-range dependent*, namely, the autocorrelation function of each of these models decays exponentially as the lag increases.

A totally different behavior of traffic has been recently revealed at Bellcore through rigorous statistical analysis of Ethernet LAN traffic and video traffic. The packets per time unit process of the Ethernet traffic was observed to be statistically **self-similar** and the bytes per frame process of the video traffic was observed to be **long-range dependent**. Both traffic are said to possess **long-range dependence**. None of the above existing models can capture this long-range dependenc.

Since this long-range dependence characterizes the burstiness of the traffic, the long-range dependent traffic has significant implication for the design, control and analysis of high speed, cell-based networks. Thus, the modeling of this type of traffic becomes necessary for the purpose of accurate network performance evaluation.

In statistics, the importance of modeling long-range dependence has attracted increasing attention in recent years [2], since long-range dependence has been observed in many areas besides telecommunication traffic. The Hurst effect that occurred in geophysics and hydrology can be explained by slowly decaying correlations. The record of the Nile river minima fits the model of stationary processes with correlations decaying hyperbolically which is the simplest model with long memory. Long-range dependence is also discovered in astronomical data, in agriculture, in chemistry, in biology, meteorology, linguistics, music, and in environmental data [2] . In statistics, the first mathematical model with long-range dependence was introduced by Mandelbrot in 1968. There is increasing awareness in statistical science that long-range dependence instead of short-range dependence exists in many data applications. If not taken into account, it can completely invalidate statistical inference.

This paper is devoted to the modeling of long-range dependent telecommunication traffic and a study of its characteristics. It consists of the following parts. Chapter 2 covers background information about the measurement of the long-range dependent and the self-similar traffic, their mathematical definitions and properties. Chapter 3 will present modeling issues of long-range dependent traffic and the approach that is used in this paper. Chapter 4 will discuss characteristics of the *ARIMA(0,d,0)* process, which is a fundamental self-similar process that possesses long-range dependence. Chapter 5 gives the algorithm of parameter estimation for modeling and will show our complete modeling approach using *ARIMA(p,d,q)* process based upon the *ARIMA(0,d,0)* process. Chapter 6 is about video trace data modeling using the method presented in the previous chapters. Chapter 7 will be a simulation performance study of the real video traffic and the simulated traffic. Chapter 8 concludes this paper.

Chapter 2

Understanding of Long-Range Dependent Traffic

2.1 Measurement of Long-Range Dependent Traffic

Motivated by understanding the interconnections between LANs and the proposed BISDN interconnection network, a high quality Ethernet traffic measurement was done at Bellcore [14][15]. Four sets of Ethernet local area network traffic measurements are considered. Each one represents between 20 and 40 consecutive hours of Ethernet traffic and each one consists of tens of millions of Ethernet packets. The data were collected on different intracompany LAN networks at different periods in time over the course of approximately 4 years (August '89, October '89, January '90, and February '92), corresponding to different network loads. A high quality Ethernet monitor was used to record the Ethernet packets without loss (irrespective of the traffic load). A high resolution timestamp was used to record the arrival time of the end of the packet and packet length. The data showed that the process of packet arrivals per time unit is statistically self-similar.

As mentioned in chapter 1, currently available video traffic models are coding specific. The desire for finding a more universal property inherent to variable bit rate (VBR) video traffic that is independent of scene and codec led to the discovery of the long-range dependence in video traffic [17]. Twenty different video sequences ranging from 15 seconds (a few hundred frames) to 2 hours (some 171,000 frames) worth of actual video were measured [17]. They represent all different kinds of scenes and are recorded using at least 4 different types of VBR video codecs. All data sets are collected with high resolution timestamp. A common predominant characteristic of all VBR video data is that at the frame level, the rate process bytes/frame or cells/frame (where ATM cells contain the compressed and coded information for the corresponding frame) presents long-range dependence with different degrees for different data sets. The intensity of long-range dependence depends on the activity level of the recorded scene, e.g. video conference and video phone are low activity scenes, while video TV and full motion pictures are high activity scenes. According to [17], video with high activity scenes are tested to have higher degree of long-range dependence than video with low activity scenes.

Note however, *long-range dependence* and *self-similar* are not mathematically exchangable terms. Definitions of both concepts will be given in section 2.3.

The empirical data are tested to be long-range dependent or self-similar by three statistical methods [15]: (1) analysis of the variances of the aggregated processes $X^{(m)}$, (2) time-domain analysis based on the R/S-statistics (*rescaled adjusted range*), and (3) periodogram-based analysis in the frequency-domain.

The first method is an application of a key feature of long-range dependence and leads to the so-called *variance-time plots.* It is used as a tool in this paper for identifying the long-range dependence and a method of estimating the degree of long-range dependence or self-similarity. It will be introduced in chapter 5.

Historically, the second method was used to provide an interpretation of an empirical law that is known as *Hurst's law* or the *Hurst effect* [15]. Briefly, for a given set of observations $(X_k : k = 1, 2, ..., n)$ whose sample mean is $\overline{X}(n)$ and sample variance $S^2(n)$, the *R/S statistic* is given by

$$R(n)/S(n) = 1/S(n)[max(0, W_1, W_2, ...W_n) - min(0, W_1, W_2, ..., W_n)],$$

with $W_k = (X_1 + X_2 + \cdots X_k) - k\overline{X}(n), k = 1, 2, ..., n$. It was found by Hurst that many naturally occuring time series that possess long-range dependence appear to be well represented by the relation $E[R(n)/S(n)] \sim cn^H, \quad as \quad n \to \infty$, with *Hurst parameter H* "typically" about 0.73, and c an arbitrary finite positive constant. It was also proved that if the observations X_k come from a short-range dependent model, then $E[R(n)/S(n)] \sim fn^{0.5}, \quad as \quad n \to \infty$ with f an arbitrary finite positive constant. This discovery is referred to as the *Hurst phenomenon.* How the Hurst parameter relates to the degree of the long-range dependence will be given in section 2.3.

The periodogram-based analysis, on the other hand, is maximum likehood type estimation [15]. It provides a more refined data analysis than variance-time plots and R/S analysis, since it has the robustness of dealing with commonly encountered problems for a given set of observations: (1) deviations from the marginal Gaussianity, (2) deviations from the assumed model spectrum. Both variance-time plots and R/S analysis are very useful tools for identifying self-similarity in a heuristic manner except for that they are inadequate when a more refined data analysis is required.

2.2 Visual Understanding of Long-Range Dependence

Visually, an aggregated long-range dependent time series looks different from an aggregated time sequence of a conventional model. Figure 2-1 [14] provides a pictorial proof of self-similarity of the Ethernet traffic.

The left column plots correspond to Ethernet traffic in packets per time unit ranging from 100 seconds to 10 milliseconds. Each subsequent plot is obtained from the previous one by increasing the time resolution by a factor of 10 and by concentration on a randomly chosen subinterval (the darker shade). The right column set of plots is the synthetic counterpart of the compound Poisson model. Obviously, Ethernet traffic plots seem to look very "similar" to one another from all time scales in a distributional sense. In contrast, Poisson data has the plots of packets per time unit which are indistinguishable from white noise after aggregation over a few hundred milliseconds. The presence of "burstiness" across an extremely wide range of time scales implies that there is no natural length of a "burst". This time scale invariant or "self-similar" feature of Ethernet traffic makes it drastically different from both conventional telephone traffic and packet traffic models.

2.3 Mathematical Definitions and Properties of Self- Similar Process and Long-Range Dependence

Let $X = (X_t : t = 0, 1, 2, ...)$ be a wide-sense stationary process, i.e., a process with constant mean $\mu = E[X_t]$, finite variance $\sigma^2 = E[(X_t - \mu)^2]$, and an autocorrelation function $r(k) = E[(X_t - \mu)(X_{t+k} - \mu)]/E[(X_t - \mu)^2] \quad (k = 0, 1, 2, ...)$

Figure 2-1: Pictorial proof of self-similarity: Ethernet traffic (packets per time unit) on 5 different time scales (a)-(e). For comparison, synthetic traffic from an appropriately chosen compound Poisson model on the same 5 different time scales (a1)-(e1) [1].

that depends only on k. Assume that X has an autocorrelation function which exhibits long-range dependence with the form,

$$r(k) \sim k^{-\beta} L_1(k), \qquad as \quad k \rightarrow \infty \tag{2.1}$$

where $0 < \beta < 1$ and L_1 is slowly varying at infinity, i.e., $\lim_{t\rightarrow\infty} L_1(tx)/L_1(t) = 1$, for all $x > 0$ (examples of such slowly varying functions are $L_1(t) = const$, $L_1(t) = log(t)$). For each $m = 1, 2, 3, ...$, let $X^{(m)} = (X_k^{(m)} : k = 1, 2, 3, ...)$ denote a new time series obtained by averaging (arithmetic mean) the original series X over nonoverlapping blocks of size m [14][15], i.e.,

$$X_k^{(m)} = (1/m)(X_{km-m+1} + ... + X_{km}), \quad k = 1, 2, 3, ... \tag{2.2}$$

Note that for each m, the aggregated time series $X^{(m)}$ defines a wide sense stationary process. Let $r^{(m)}$ denote the corresponding autocorrelation function. X is called *exactly self-similar* with self-similar parameter $H = 1 - \beta/2$ if for all $m = 1, 2, 3...$, $1/m^H(X_{km-m+1} + \cdots + X_{km})$, $k = 1, 2, 3, ...$, has the same finite-dimensional distributions as X [16].

The process X is called *exactly second-order self-similar* with self-similarity parameter $H = 1 - \beta/2$ [16][15] if for all m=1,2,3..., the corresponding aggregated process $X^{(m)}$ has a variance which is var$(X^{(m)}) = \sigma^2 m^{-\beta}$ and the same correlation structure as X, i.e.,

$$r^{(m)}(k) = r(k), \quad (k = 1, 2, 3, ...) \tag{2.3}$$

X is called *asymptotically second-order self-similar* with self- similarity parameter $H = 1 - \beta/2$ [16][15] if

$$r^{(m)}(k) \to r(k), \quad \text{as} \quad m \to \infty, \quad (k = 1, 2, 3, ...) \tag{2.4}$$

In other words, X is exactly or asymptotically second-order self-similar if the corresponding aggregated processes $X^{(m)}$ are the same as X or become indistinguishable from X at least with respect to their autocorrelation functions.

It is easy to see that exact self-similarity implies second-order self-similarity. It is the strongest definition of self-similarity. We are more interested in the second-order self-similar process in this paper.

More generally, a stochastic process satisfying relation (2.1) is said to exhibit *long-range dependence.* So long-range dependence is characterized by an hyperbolically decaying autocorrelation function. Obviously, long-range dependence is a necessary condition of self-similarity but not a sufficient condition.

Our interest in this paper is modeling the telecommunication traffic that is long-range dependent (may not necessary be self-similar) using self-similar stochastic processes that possess long-range dependence.

The parameter H is also called *Hurst parameter.* It is also quantity for measuring the degree of long-range dependence or equivalently the degree of the "burstiness" of the traffic, namely, the burstier the traffic the higher the H.

The basic mathematical differences between long-range dependent traffic and the conventional models are [14]:

1. Aggregated Process

The aggregated processes $X^{(m)}$ of a long-range dependent process possess a nondegenerate correlation structure as $m \to \infty$, while the conventional models all have the property that the autocorrelation functions of their aggregated processes $X^{(m)}$ tend to second order pure noise, i.e., for all $k \geq 1$,

$$r^{(m)}(k) \to 0, \quad as \quad m \to \infty$$

This is illustrated in the plots in Figure 2-1. If the original time series X represents the number of Ethernet packets per 10 milliseconds (plot (e)), then plots (a) to (e) depict segments of the aggregated time series $X^{(10000)}$, $X^{(1000)}$, $X^{(100)}$, and $X^{(10)}$, respectively. All the plots look " similar" and distinctively different from pure noise, which means they are not independently distributed random variables.

2. Autocorrelation Function

The autocorrelation function of a long-range dependent process in the form of equation (2.1) implies that $\sum_k r(k) = \infty$, which tells us that although high-lag correlations are all individually small, their cumulative effect is of importance and gives rise to features which are drastically different from those of the more conventional short-range dependent processes. In contrast, the latter are characterized by an exponential decay of the autocorrelations resulting in a summable autocorrelation function $0 < \sum_k r(k) < \infty$.

3. Frequency Domain

In the frequency domain, the spectral density of a long-range dependent process obeys a power-law behavior near the origin. i.e.,

$$f(\lambda) \sim \lambda^{-\gamma} L_2(\lambda), \quad \text{as} \;\; \lambda \to 0 \qquad (2.5)$$

where $0 < \gamma < 1$, L_2 is slowly varying at 0, and $f(\lambda) = \sum_k r(k) e^{ik\lambda}$ denotes the spectral density function. This implies that $f(0) = \sum_k r(k) = \infty$, that is, the spectral density tends to $+\infty$ as the frequency λ approaches 0. On the other hand, processes that only possess short-range dependence are characterized by a spectral density function $f(\lambda)$ which is positive and finite at $\lambda = 0$.

4. Variance

The variance of the aggregated time series $X^{(m)}$ of a long-range dependent process decreases more slowly than the reciprocal of the sample size, that is

$$var(X^{(m)}) \sim am^{-\beta}, \quad \text{as} \quad m \to \infty, \tag{2.6}$$

where a is a finite positive constant independent of m, and $0 < \beta < 1$. The β here is related to the γ in equation (2.5) by $\beta = 1 - \gamma$. Note that the self-similar parameter H is represented in terms of β by $H = 1 - \beta/2$, as in the definitions (2.3) and (2.4). In contrast, for stationary processes whose aggregated series $X^{(m)}$ tend to second-order pure noise (i.e., $r^{(m)}(k) \to 0, \text{for} k \geq 1$) the sequence $\text{var}(X^{(m)}) : m \geq 1$ satisfies

$$var(X^{(m)}) \sim bm^{-1}, \quad as \quad m \to \infty, \tag{2.7}$$

where b is a finite positive constant independent of m.

This feature leads to one of the methods of identifying self-similarity, variance-time plots, as mentioned in the previous section. It is later used in this paper for estimating the degree of self-similarity or the degree of long-range dependence, namely, parameter H.

It has been proved [15] that a specification of the autocorrelation function satisfying equation (2.1) is equivalent to the specification of the spectral density function satisfying equation (2.5), and further it is equivalent to the specification of the sequence $(var(X^{(m)})$ in equation (2.6). These are different manifestations of the long-range dependence of the underlying wide-sense stationary process X.

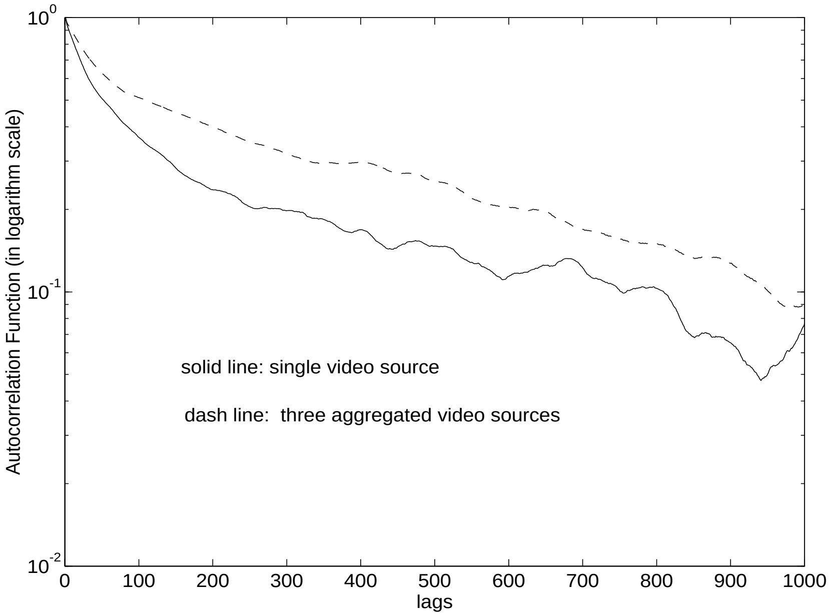


Figure 2-2: Autocorrelation Comparison of Single Video versus Aggregated Video Sources

2.4 Implication of Long-Range Dependent Process on Network Performance

A direct implication of self-similarity or long-range dependence is that the burstiness of the traffic typically intensifies as the number of active traffic sources increases. This is in contrast to the conventional idea that traffic becomes smoother (less bursty) as the number of traffic sources increases, which is the typical nature of aggregate traffic of the "Poisson-like" models.

The burstiness is proportional to the degree of the correlation of the traffic, so the correlation function can reflect the degree of burstiness. Therefore, the above statement can be easily proved visually by looking at the correlation function. Figure (2-2) shows a correlation function comparison between a single video source data that was collected at Bellcore against aggregated video source.

The lower curve is the autocorrelation of a single video file with 57000 frame samples. The upper curve is the autocorrelation function of three aggregated video sources each with 57000 samples. The aggregation is simply the summation of three video sources. Clearly, the autocorrelation of the aggregate traffic is bigger than that of the single traffic, which is equivalent to saying that the aggregated source is more bursty than the single source.

A congestion management study in the presence of self-similar traffic at Bellcore [18][19] suggested that the congestion phenomenon seen in the presence of self-similar traffic differs drastically from that predicted by the conventional traffic models. An access class scheme proposed for Switched Multimegabit Digital Service (SMDS) on a public B-ISDN was considered. Ethernet data in a trace-driven simulation of a LAN/B-ISDN interface was employed to observe the effect of the actual aggregate LAN traffic on the behavior of the SMDS service interface buffer, i.e. the relationship between packet delay, packet loss and the amount of buffering at the interface.

The results show that overall packet loss decreases very slowly with increasing buffer capacity, in sharp contrast to Poisson-based models where losses decrease exponentially fast with increasing buffer size [20]. Moreover, packet delay (95th percentile) always increases with buffer capacity, again in contrast to the conventional models where delay does not exceed a fixed limit regardless of buffer size [18]. This behavior, according to the author, is typical for self-similar traffic and can be readily explained using its properties.

In addition, a simple network that provides LAN interconnection was simulated at Bellcore to study the congestion behavior due to traffic access contention [19]. The combined traffic is modeled by the measured Ethernet traces. The results from their study presented below follows [19]:

(1) There exist large variations in the network traffic on time scales of hours, days, or months; this aggravates careful sizing of network components, since small errors in engineering can incur drastic penalties in loss or delay.

(2) Although some of the standard traffic models suggest that congestion problems essentially disappear with sufficient buffer capacity, realistic network traffic shows that such behavior cannot be expected; large buffers will not prevent congestion from occurring but introduce instead undesirable delay characteristics.

(3) During congestion periods, congestion persists long enough for the effects of user and protocol responses to be felt.

(4) A detailed examination of congestion periods shows that when congestion occurs, losses are severely concentrated and are far greater than the background loss rate; losses may exceed the long-run loss probability by an order of magnitude during the first second following the onset of congestion, while the losses are elevated by over two orders of magnitude during the first 100 milliseconds.

(5) Fortunately, many congestion episodes are preceded by signs of impending danger; whether detecting congestion or activating congestion avoidance responses can be done reliably far enough in advance of an actual congestion period requires further study.

These results provide convincing evidence for the significance of self- similar or long-range dependent network traffic for engineering future integrated high-speed data networks.

Chapter 3

Modeling of Long-Range Dependent Phenomena

An important requirement of practical traffic modeling is to generate synthetic data sequences that exhibit long-range dependent features corresponding to the measured data so that the corresponding network performance can be studied. Hence, we need to find out mathematical models that can be used for the above purpose. The simplest models are stationary stochastic processes that have hyperbolically decaying correlations, or equivalently, are long-range dependent.

The two best known classes of stationary processes with slowly decaying correlations/long-range dependence are *fractional Gaussian noise* and *fractional autoregressive integrated moving- average* (*ARIMA*) processes. The former is exactly second-order self-similar process, while the latter is asymptotically second-order self-similar process [15].

Fractional Gaussian noise $X = (X_k : k \geq 0)$ with a parameter $H \in (0,1)$, is a stationary Gaussian process with mean μ, variance σ^2, and autocorrelation function $r(k) = 1/2(|k+1|^{2H} - |k|^{2H} + |k-1|^{2H}), k > 0$. As $k \to \infty$, $r(k) \sim H(2H-1)|k|^{2H-2}$. It has been proved [15] that the resulting aggregated processes $X^{(m)}(m = 1, 2, 3, ..)$ all have the same distribution as X for $H \in (0,1)$. Thus, by definition of equation (2.3), fractional Gaussian noise is exactly second-order self-similar with self-similar parameter $1/2 < H < 1$ (since $H = 1 - \beta/2$ and $0 < \beta < 1$ from equation (2.1)). Methods for estimating the three unknown parameters μ, σ^2 and H are known.

A *fractional ARIMA(p,d,q)* process $Y = (Y_k : k \geq 0)$, where p and q are non-negative integers and d is real is a natural generalization of standard *ARIMA(p,d,q)* where d is integer [2]. It is defined to be,

$$\phi(B)\nabla^d Y_k = \theta(B) a_k \tag{3.1}$$

where $\phi(B) = 1 - \phi_1 B - \cdots - \phi_p B^p$ and $\theta(B) = 1 - \theta_1 B - \cdots - \theta_q B^q$ are polynomials in the backward-shift operator B where $By_k = y_{k-1}$. a_k with $k = 1, 2, 3, \ldots$ is a white noise process. $\nabla = 1 - B$ denotes the differencing operator, and the ∇^d is the fractional differencing operator defined by

$$\begin{aligned} \nabla^d &= (1-B)^d \\ &= \sum_{k=0}^{\infty} \binom{d}{k} (-B)^k \\ &= 1 - dB - \frac{1}{2}d(1-d)B^2 - \frac{1}{6}d(1-d)(2-d)B^3 - \cdots \end{aligned} \tag{3.2}$$

It has been shown [15] that for $d \in (-1/2, 1/2)$, Y is stationary and invertible. Its autocorrelations satisfy $r(k) \sim \alpha k^{2d-1}$ as $k \to \infty$, where α is a finite positive constant independent of k. Moreover, the aggregated time series $X^{(m)}$ of a fractional *ARIMA(p,d,q)* process satisfy (2.4) for $-1/2 < d < 1/2$. Thus, relation (2.1) and (2.4) hold and Y is asymptotically second order self-similar with self-similarity parameter $H = 1 - \beta/2 = 1 - (1-2d)/2 = d + 1/2$, for all $0 < d < 1/2$ $(0 < \beta < 1)$.

Since d is in proportion to H, its value represents the degree of long-range dependence. The higher the degree, the bigger the value.

Compared with the fractional *ARIMA(p,d,q)*, fractional Gaussian noise has only three parameters. It may not be capable of capturing the wide range of low lag correlation structure. On the other hand, fractional ARIMA(p,d,q) which has more parameters (including $\phi's$ and $\theta's$) has the advantage in that respect over fractional Gaussian noise. In particular, fractional *ARIMA* can be expressed by its two basic forms: *ARIMA(0,d,0)* and *ARIMA(p,0,q)* (equivalent to *ARMA(p,q)*), where the former describes the long-term correlation and the latter describes the short-term correlation behavior. Therefore the modeling can be done by two steps, i.e., first build *ARIMA(0,d,0)* and then build *ARIMA(p,d,q)* on top of *ARIMA(0,d,0)*. This simplifies the modeling to a certain extent. Because of this feature, fractional *ARIMA* processes are much more flexible and capable of simultaneous modeling of short- term and long-term behavior of a time series than fractional Gaussian noise does.

An alternative way for constructing a self-similar process is based on aggregating many simple renewal reward processes exhibiting inter-renewal times with infinite variance [15]. In contrast to the fact that the two formal mathematical models do not provide any physical interpretation to self-similar phenomena, this method represents some underlying physical process for forming the self-similar process, which can be appealing in the context of high-speed packet traffic.

Two approaches in the above regard were used at Bellcore for constructing self-similar processes [14]. The first method simulates the buffer occupancy in an M/G/∞ queue, where the service time distribution G has infinite variance. It results in an asymptotically second-order self-similar buffer occupancy process. The second method is aggregation of many simple AR(1)-processes and the resultant superposition process is asymptotically second-order self-similar. This methods require massively parallel computers with a large amount of processors.

The fractional *ARIMA(p,d,q)* process is chosen for modeling the long -range dependent traffic in this paper, mainly because of its good features and the fact that it does not require massively parallel computers. We believe it is important that the self-similar phenomena be observed and be processed in a less demanding environment such as workstations.

Modeling using *ARIMA(p,d,q)* involves parameter estimation of p, d and q as well as the corresponding θ and ϕ parameters. The effect of the d parameter on distant observations decays hyperbolically as the lag increases while the effect of ϕ and θ parameters decay exponentially. So d is to describe the high-lag correlation structure and the θ and ϕ are to describe the low-lag correlation structure. Provided that *ARIMA(0,d,0)* and *ARIMA(p,0,q)* are available, the combination of the two will result in *ARIMA(p,d,q)*.

3.1 ARIMA(0,d,0)

ARIMA(0,d,0) is the simpliest and the most fundamental of the fractionally differenced *ARIMA* processes. Its correlation at high lags is supposed to be similar to those of an *ARIMA(p,d,q)* with the same value of d, since for very distant observations the effects of the ϕ and θ parameters will be negligible. With the intention of understanding the long-range dependence, we started with *ARIMA(0,d,0)*. This section is a review on *ARIMA(0,d,0)* process from [21].

ARIMA(0,d,0) is defined to be a discrete-time stochastic process x_t which may be represented as

$$\nabla^d x_t = a_t, \tag{3.3}$$

where the operator ∇^d is defined in equation (3.2) and the white noise a_t consists of independent identically distributed random variables with mean zero and variance σ_a^2.

The spectral density of x_t is

$$s(\omega) = (2sin\frac{1}{2}\omega)^{-2d}, \qquad 0 < \omega \leq \pi \tag{3.4}$$

and

$$s(\omega) \sim \omega^{-2d}, \qquad \omega \to 0. \tag{3.5}$$

The covariance function of x_t is

$$\gamma_k = E(x_t x_{t-k}) = \frac{(-1)^k(-2d)!}{(k-d)!(-k-d)!} \tag{3.6}$$

and the correlation function of x_t is

$$\rho_k = \gamma_k/\gamma_0 = \frac{(-d)!(k+d-1)!}{(d-1)!(k-d)!}, \qquad (k = 0, \pm 1 ...), \tag{3.7}$$

$$\rho_k = \frac{d(1+d)...(k-1+d)}{(1-d)(2-d)...(k-d)} \qquad (k = 1, 2, ...). \tag{3.8}$$

In particular $\gamma_0 = (-2d)!/(-d)!^2$ and $\rho_1 = d/(1-d)$.

As $k \to \infty$

$$\rho_k \sim \frac{(-d)!}{(d-1)!} k^{2d-1} \tag{3.9}$$

Factorial of a fractional number r that we used in the above several equations is defined as

$$r! = \Gamma(r+1)$$

where the *Gamma Function* is defined as

$$\Gamma(r) = \int_0^\infty t^{r-1} e^{-t} dt$$

However, just like factorial of an integer number, $r!$ can be expressed as $r(r-1)(r-2)\cdots$. Because of this, equation (3.7) can be reduced to equation (3.8).

Obviously, equation (3.9) indicates hyperbolic decay of the correlation function as it should.

When $0 < d < \frac{1}{2}$, the *ARIMA(0,d,0)* process is a stationary process with long memory. The correlations of x_t are all positive as in equation (3.8) and decay monotonically and hyperbolically to zero as the lag increases as in equation (3.9). The spectral density of x_t is concentrated at low frequencies: $s(\omega)$ is a decreasing function of ω and $s(\omega) \rightarrow \infty \quad as \quad \omega \rightarrow 0$, but $s(\omega)$ is integrable as in equation (3.5).

When $d = \frac{1}{2}$ the *ARIMA(0,d,0)* process just fails to converge. Its spectral density has the form

$$s(\omega) = 1/2sin(\frac{1}{2}\omega) \sim \omega^{-1}$$

When $d = 0$, the *ARIMA(0,d,0)* process is white noise, with zero correlations and constant spectral density.

3.2 Generating ARIMA(0,d,0) Time Series

We now consider how to generate *ARIMA(0,d,0)* time sequences for a given *d*. According to equation (3.3), an *ARIMA(0,d,0)* sequence can be obtained if d is known and if white noise is available. However, an important assumption we need to make is that $t \geq 0$, namely, all the x values are equal to zero for negative time index, i.e., $x = 0, \quad \text{for} \quad t \leq 0$. Under this assumption, we can get a finite time series starting from x_0 to x_n, where n is any integer number.

$$
\begin{aligned}
x_0 &= a_0 \\
x_1 &= a_1 + dx_0 \\
x_2 &= a_2 + dx_1 + \frac{1}{2}d(1-d)x_0 \\
\ldots \\
x_n &= a_n + dx_{n-1} + \frac{1}{2}d(1-d)x_{n-2} \ldots + \frac{1}{n!}d(1-d)(2-d)\ldots(d-n+1)x_0
\end{aligned}
$$

In general.

$$x_n = \sum_{k=1}^{n} \binom{d}{k} x_{n-k} + a_n \qquad n \geq 0 \tag{3.10}$$

For the n*th* sample, n multiplications/additions are involved.

Since this method implements the definition of *ARIMA(0,d,0)*, we call it the direct *ARIMA* method.

Another way of generating *ARIMA(0,d,0)* was introduced by Hosking and was originally used for hydrological self-similarity studies [22]. This method yields a stationary process with a normal marginal distribution which is not guaranteed in the direct method. The algorithm is as follows:

(1) Generate a starting value x_0 from the stationary distribution of the process $N(0, v)$, where v_0 is the required variance of the x_t.

(2) For $t = 1, 2, ...n-1$, calculate partial correlation coefficient ϕ_{tj}, $j = 1, ...t$ recursively via the equations

$$\begin{aligned} \phi_{tt} &= d/(t-d) \\ \phi_{tj} &= \phi_{t-1,j} - \phi_{tt}\phi_{t-1,t-j} \qquad j = 1, 2, ...t-1 \end{aligned}$$

Calculate $m_t = \sum_{j=1}^{t} \phi_{tj} x_{t-j}$ and $v_t = (1 - \phi_{tt}^2) v_{t-1}$. Then generate x_t from the distribution $N(m_t, v_t)$.

For the n*th* sample, n+(n-1) multiplications/additions are required. This is double the amount of computation compared to the direct method.

Both of the above methods have been implemented in this paper. Because of the uniqueness of the long-range dependence, the processing time required for generating one data point increases as the time index increases. This is because, first, the number of multiplications and additions increases as the time index increases, and secondly, all the previous x values need to be stored for calculating the current x. Note that in order to be able to test the long-range dependence, a large number of sample points is definitely required. When the number of sample points is large, the amount of memory needed to store the data can be huge and the processing time required can be very long. In order to generate tens of thousands or even several millions of data points on a Sun workstation (sparc 10) or a Dec Alpha machine, which have limited memory, proper programming with efficient usage of memory is crucial. A trade off between more processing and less memory occupation (to avoid using a huge array for storing the data, for example) is necessary to be able to generate hundreds of thousands or millions of data points. If we calculate all the coefficients $\binom{d}{k}$ in equation (3.10) in advance and store them in a file for later use of generating *ARIMA(0,d,0)*, we can increase considerably the number of *ARIMA(0,d,0)* sample points that a machine can generate. The maximum amount of data that our available Sun Sparc 10 can generate in this case is $120 * 10^6$, while the available DEC Alphas's capacity is around $20 * 10^6$, since the Sparc 10 has 128 MB memory while the DEC Alpha has 64 MB memory.

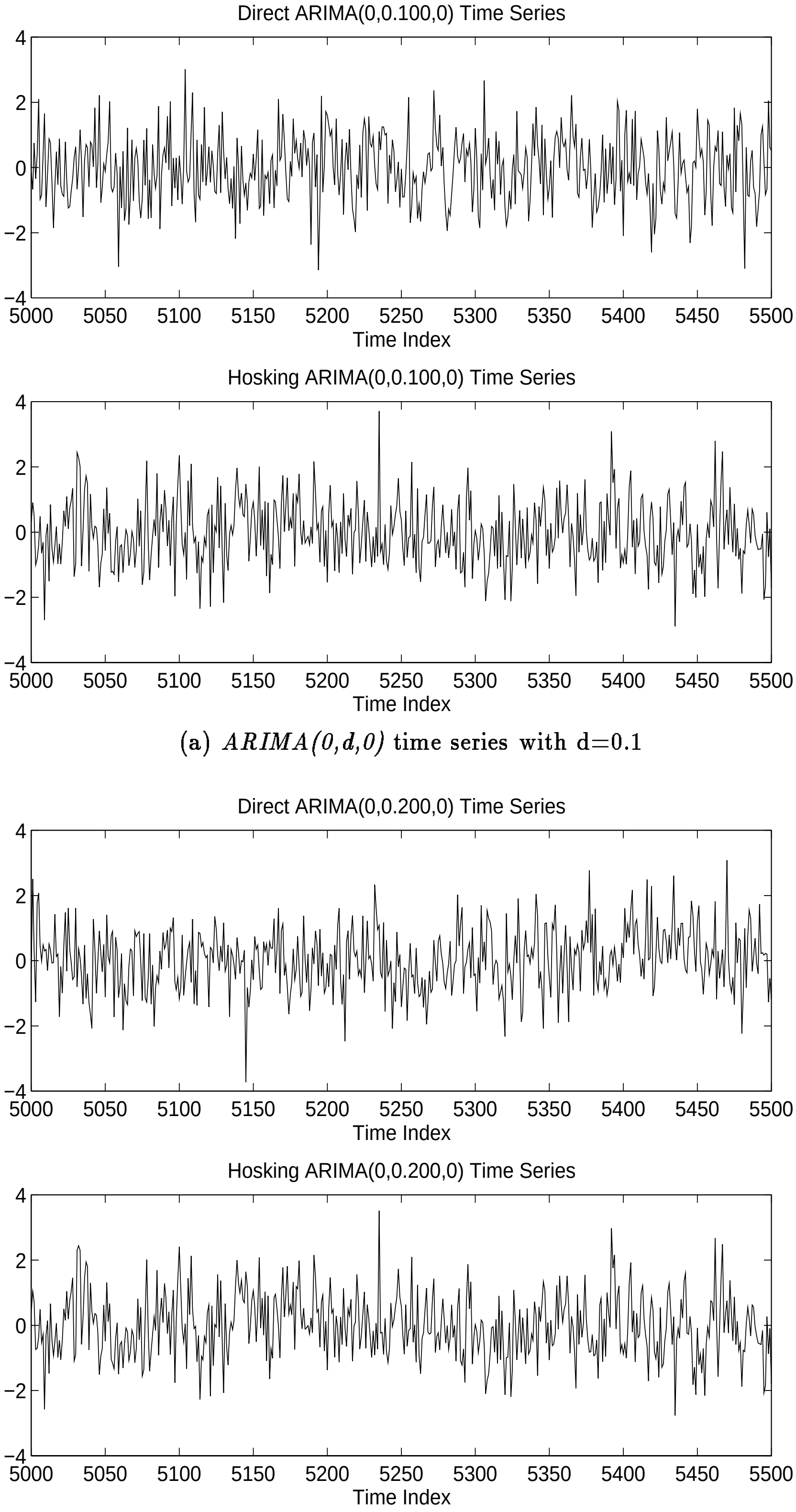
In general, the Hosking algorithm requires more time (by approximately a factor of 2) than the direct *ARIMA* method for the same amount of data. However, proper use of the "malloc" function and array pointers can reduce the time for Hosking method. So there is no significant disadvantage for using the Hosking method in terms of time required.

For 500,000 sample data, the Hosking method requires about 3 days of running time on a DEC Alphas. However, it only takes about 18 hours to run for 250,000 samples. Moreover, it takes less than one hour to generate 50,000 samples. Clearly, the number of sample points and time have a nonlinear relationship. With the direct *ARIMA(0,d,0)* method, the time required is about 75% to 80% of the total time required for the Hosking method.

Plots 3-1 show *ARIMA(0,d,0)* time sequences generated by the direct *ARIMA* method and the Hosking method for different d's. All the time series are scaled to have zero mean and variance 1 (see below for explanations).

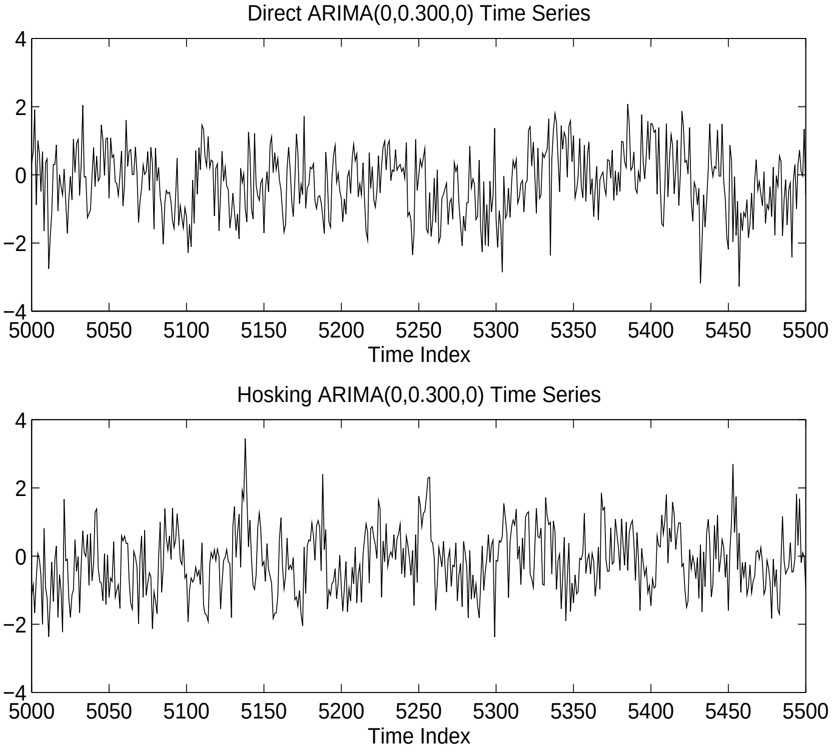
Note first that for all values of d, two *ARIMA(0,d,0)* processes generated by different methods look similar. Further more, the changes in their shapes with increasing d look similar as well. In fact, the change in shape with increasing d corresponds to the increasing of the degree of long-range dependence. We will see this in chapter 4. A closer look shows that both data seem to have local trends in particular parts and the expected value seems to be changing slowly (more obvious for big d) even though the mean over the whole series seems to be constant. Such behavior are referred by [2] as typical for stationary processes with long memory.

An explanation for scaling the time series to have zero mean and variance 1 is given as follows. Originally, both methods of generating the *ARIMA(0,d,0)* process are supposed to produce zero mean and variance 1. Due to limited number

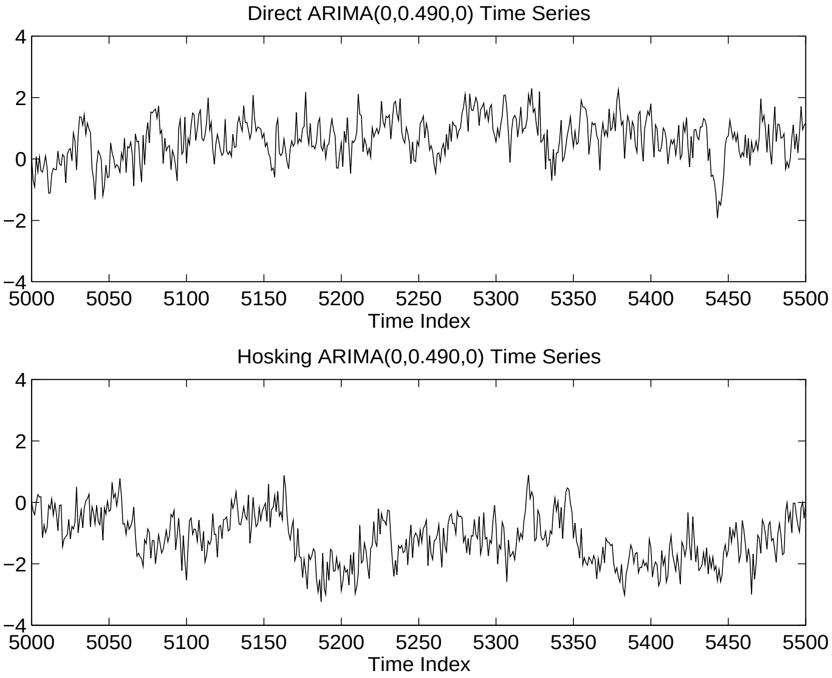


(a) *ARIMA(0,d,0)* time series with d=0.1

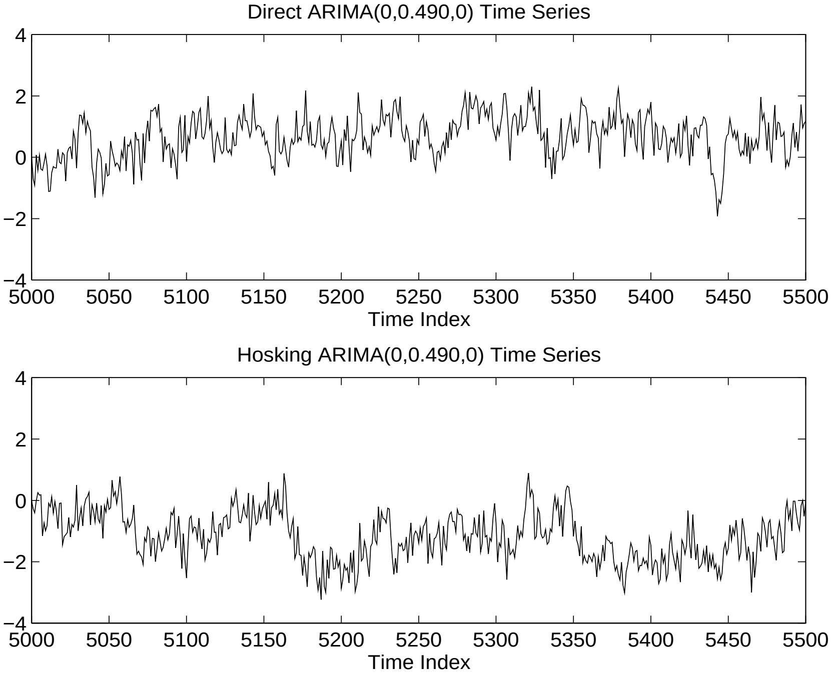
(b) *ARIMA(0,d,0)* time series with d=0.2



(c) *ARIMA(0,d,0)* time series with d=0.3



(d) *ARIMA(0,d,0)* time series with d=0.415



(e) *ARIMA(0,d,0)* time series with d=0.49

Figure 3-1: *ARIMA(0,d,0)* processes with zero mean and variance 1 generated by the direct *ARIMA* and the Hosking methods . Value of d range from 0.1 to 0.49.

of sample points (50,000 in this case) that we generate, the resultant process have a nonzero mean and a nonunit variance. For purpose of comparison, we scale all the time series to have zero mean and variance 1. In real traffic modeling, no matter what mean and variance the simulated *ARIMA* process has, it needs to be eventually scaled so that it has the same mean and variance of the real traffic data.

A detailed comparison of *ARIMA(0,d,0)* time sequences generated by the direct *ARIMA(0,d,0)* method and the Hosking method will be given in the next chapter.

3.3 Estimation of d from ARIMA(0,d,0)

A primary question now is how can we have a method to test the generated *ARIMA(0,d,0)* sequences so that we know it does indeed have the long-term correlation characteristics that we wanted. In other words, does it still have the same value d as we set it to be? This is especially important in view of the positive time index constraint in generating the time series.

From equation (3.6) we can get an expression for d in terms of correlations by taking the ratio of r_{k+1} to r_k, i.e.,

$$\begin{aligned} \frac{r_{k+1}}{r_k} &= \frac{(-1)(k-d)!(-k-d)!}{(k+1-d)!(-k-1-d)!} \\ &= \frac{k+d}{k+1-d} \qquad (3.11) \end{aligned}$$

Solving this equation for d we get,

$$d = \frac{r_{k+1}(k+1) - r_k k}{r_k + r_{k+1}} \qquad (3.12)$$

Since r_k differs from autocorrelation ρ_k only by a factor of $1/r_0$, the above equation holds for ρ_k as well

$$d = \frac{\rho_{k+1}(k+1) - \rho_k k}{\rho_k + \rho_{k+1}} \tag{3.13}$$

For a given time series of *ARIMA(0,d,0)*, it is always possible to calculate its correlation functions. Hence its parameter d can be in principle calculated using equation (3.13). Equation (3.13) provides a simple means of measuring the parameter d of a given *ARIMA(0,d,0)* sample time series. Therefore we can know the difference of the theoretical *ARIMA(0,d,0)* and the simulated *ARIMA(0,d,0)*. This method is used in chapter 4 as a testing tool.

A more general method for estimating d of an *ARIMA(p,d,q)* time sequence and real traffic trace data is needed. It is the variance time plot method, which is introduced in the next chapter.

Chapter 4

Characterizations of Simulated ARIMA(0,d,0)

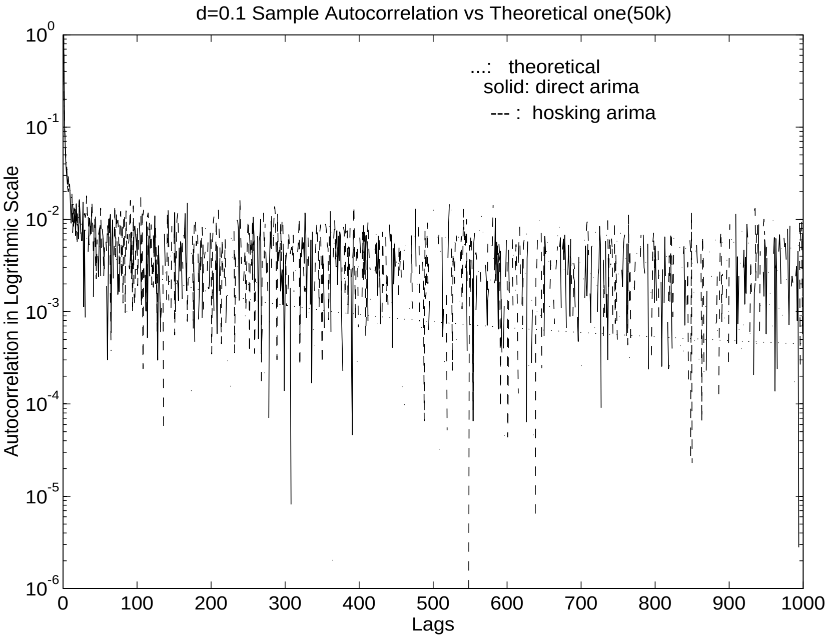
An embedded limitation of implementing practical *ARIMA(0,d,0)* time series is that only a finite number of samples can be generated, i.e., the time index of the sequence has to start from zero then goes to a finite positive integer number. With time index zero, x_0 is not dependent upon any previous value, x_1 can depend only upon x_0, and so on. In the case of the theoretical formula, x_0 and all the is other x_t are supposed to be dependent upon all the other previous x values with negative time index. The limitation on the time index in the positive direction brings a startup problem in practical implementation, that is, the long-range dependence can only be correctly observed with a time index greater than some value. This value is usually very large and it differs with different *d*. The bigger the *d*, the bigger this value, because the bigger the *d*, the higher is the degree of long-range dependence.

However, the number of data that we can generate is always finite and is limited by the computer facilities available. Therefore, a difference between the simulated *ARIMA(0,d,0)* process and the theoretical *ARIMA(0,d,0)* process will always exist.

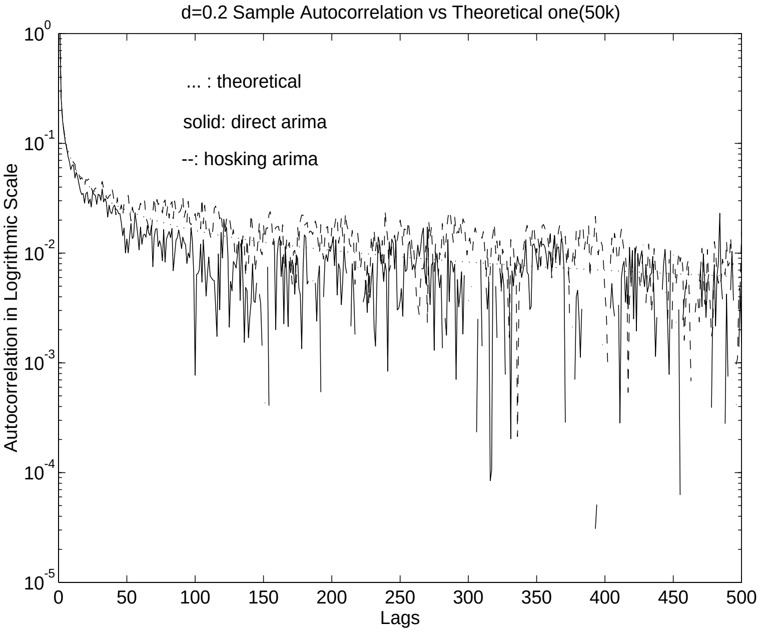
This chapter is a study of the characteristics of the simulated *ARIMA(0,d,0)* processes, both for the direct *ARIMA* method and the Hosking method. We will compare the simulated *ARIMA(0,d,0)* series with its theoretical counterpart with different values for d. This study reveals some very interesting characteristics of *ARIMA(0,d,0)* that are exceptional for long-range dependence, that we usually do not expect to see. It helps us greatly to understand long-range dependence. It provides as well the foundation to build the *ARIMA(p,d,q)* on top of *ARIMA(0,d,0)*

Since a hyperbolic decay of the correlation function of an *ARIMA(0,d,0)* indicates long-range dependence [21], the correlation function is used throughout this paper as a characterization of the long-range dependence and an estimator to characterize the difference between the theoretical *ARIMA(0,d,0)* and the simulated *ARIMA(0,d,0)*.

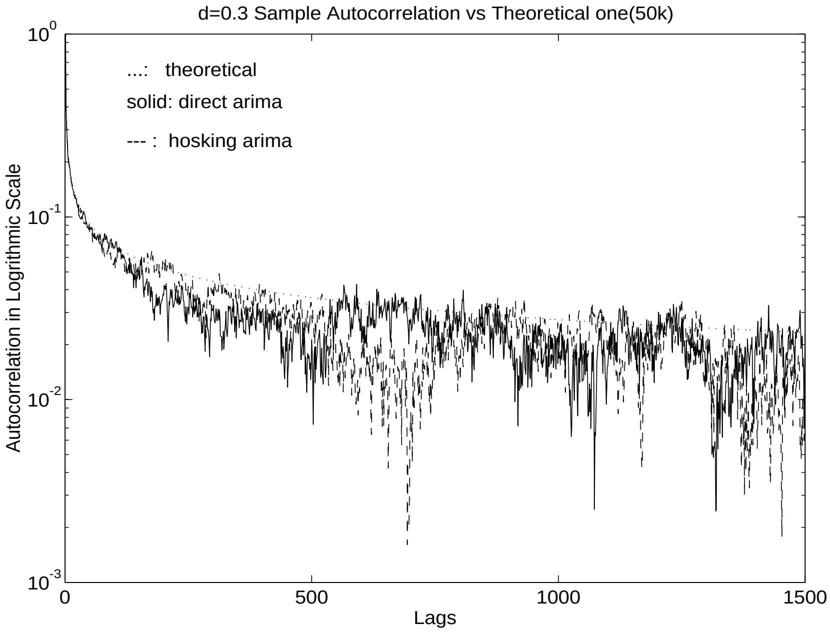
Before we go into a detailed discussion, let us first look at the plots that present evidence of long-range dependence. Figures 4-1 (a)-(f) show comparisons of the theoretical autocorrelation function of *ARIMA(0,d,0)* process to the simulated *ARIMA(0,d,0)* counterpart, generated both by the direct *ARIMA* method and the Hosking method. Values of d are 0.1, 0.2, 0.3, 0.35, 0.415, and 0.49 for plots (a) to (f). The dotted line in each plot is the theoretical autocorrelation function of *ARIMA(0,d,0)*. It is calculated according to equation (3.7). We can see clearly that the correlations of all the plots decay very slowly towards a large lags and remain persistent through the long term, although the values are small. As d increases, the correlations increase as well, implying larger degree of self-similarity or long-range dependence. These plots are discussed more fully in the next section.



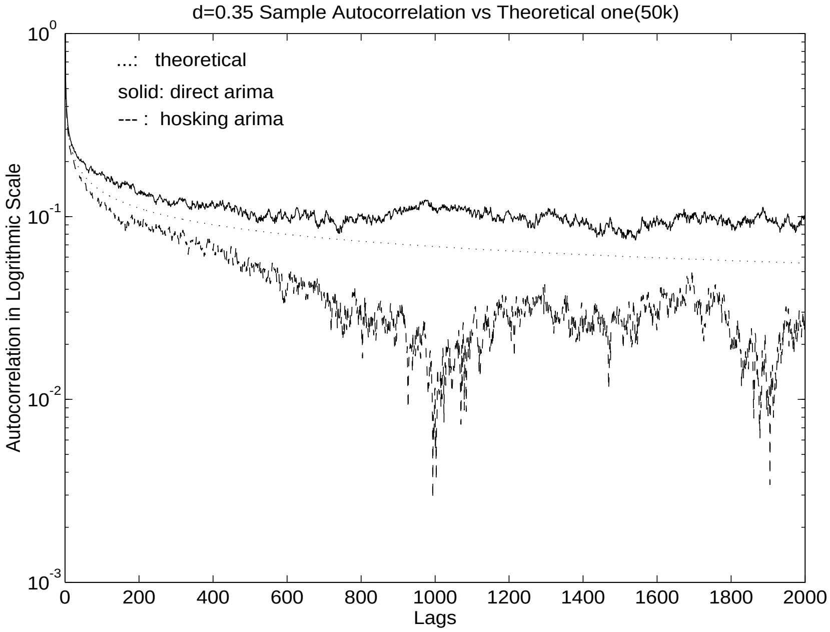
(a) Correlation functions of *ARIMA(0,0.1,0)*



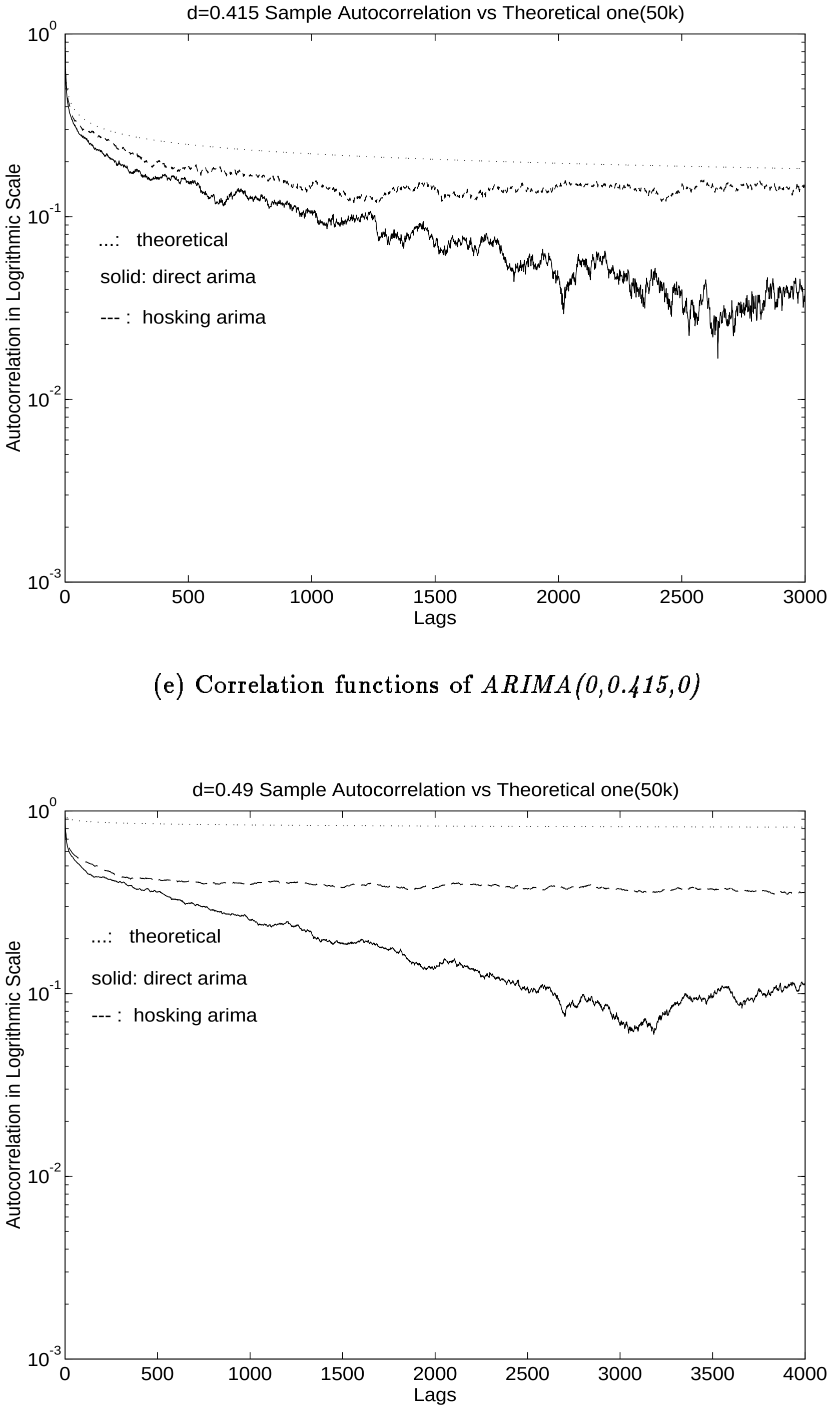
(b) Correlation functions of *ARIMA(0,0.2,0)*



(c) Correlation functions of *ARIMA(0,0.3,0)*



(d) Correlation functions of *ARIMA(0,0.35,0)*



(e) Correlation functions of *ARIMA(0,0.415,0)*

(f) Correlation functions of *ARIMA(0,0.419,0)*

Figure 4-1: Sample autocorrelation functions of simulated and theoretical *ARIMA(0,d,0)*. Value of d are 0.1, 0.2, 0.3, 0.35, 0.415, and 0.49 respectively

4.1 Correlation Behavior of the Simulated ARIMA(0,d,0) For Fixed Amount of Sample Data

In this section we will compare the correlation function of the simulated *ARIMA(0,d,0)* to the theoretical correlation function and get a sense of how the difference is related to the d in terms of correlation. From our earlier discussion, a bigger d represents a larger degree of long-range dependence. So a larger amount of data will be required in order to closely represent the theoretical characteristics. If we are generating a fixed amount of *ARIMA(0,d,0)* samples for different d ranging from 0.1 to 0.49, we should expect to see that the difference between the theoretical autocorrelation and the simulated counterpart becomes bigger as d increases. Figure 4-1 is a demonstration of this. Plot (a) to (f) of figure 4-1 show correlation functions of simulated *ARIMA(0,d,0)* with fixed 50,000 sample points for all cases, both generated by the direct *ARIMA* method and by the Hosking method versus the theoretical correlation. Each plot corresponds to a specific d.

In general, we can see that the correlation difference between the simulated time series and the theoretical ones increases as d increases, regardless of the method used to generate the sequences. Secondly, the correlation is very noisy or random for small d. It becomes smoother as d increases. The reason is that for small d, the coefficients $\binom{d}{k}$ decay relatively fast, the correlation between sample points is very weak. The process is thus close to white noise process. So, its correlation function are of very small values and look random under logrithmic scale.

Moreover, the difference between correlations of the direct *ARIMA* series and the Hosking method seems to increase as d increases. Note however, that different sets of random white noise processes a_t were used for generating every *ARIMA(0,d,0)* time series in figure 4-1 Since only 50k sample points were generated, the *ARIMA(0,d,0)* process won't be a good ergodic process as we assumed to be for such small sample set. Hence, different white noise processes with different seeds may cause differences in the resulting correlations of the underlying *ARIMA(0,d,0)* processes. Sample average statistics over several random *ARIMA(0,d,0)* processes are needed to draw correct conclusions.

Therefore, ten samples for each of the two methods for each *d* were collected. The correlation function was then averaged over the ten correlation functions of the individual samples. Figure 4-2 shows the error of the averaged correlation function with respect to the theoretical correlation function. The error itself is the difference of the two (theoretical versus average simulated) averaged from lag 1000 to lag 3000 (i,e., E(abs(difference over lags 1000 to 3000))) This range of lags covers the major difference of the simulated *ARIMA(0,d,0)* and the theoretical *ARIMA(0,d,0)*, so it is representative.

Notice that for each *d*, ten samples of the direct *ARIMA* method and ten samples of the Hosking method are generated by the same ten random noise sequences. For different *d*, ten different sets of random sequences are used. All of the time series have 50,000 data points.

From the plot we can draw two conclusions. First, for fixed amount of data the error of the correlation of the simulated *ARIMA(0,d,0)* increases as *d* increases. This is exactly what we expected. Hence in order to reduce the error of correlation for bigger *d*, we need to generate a larger amount of data. Secondly, there is no obvious difference between the direct *ARIMA* method and the Hosking method. This is a very important result. If there is no statistical difference between the two, we can use either one to generate simulated *ARIMA(0,d,0)*. For example, we can use the simpler one, which is the direct *ARIMA* method.

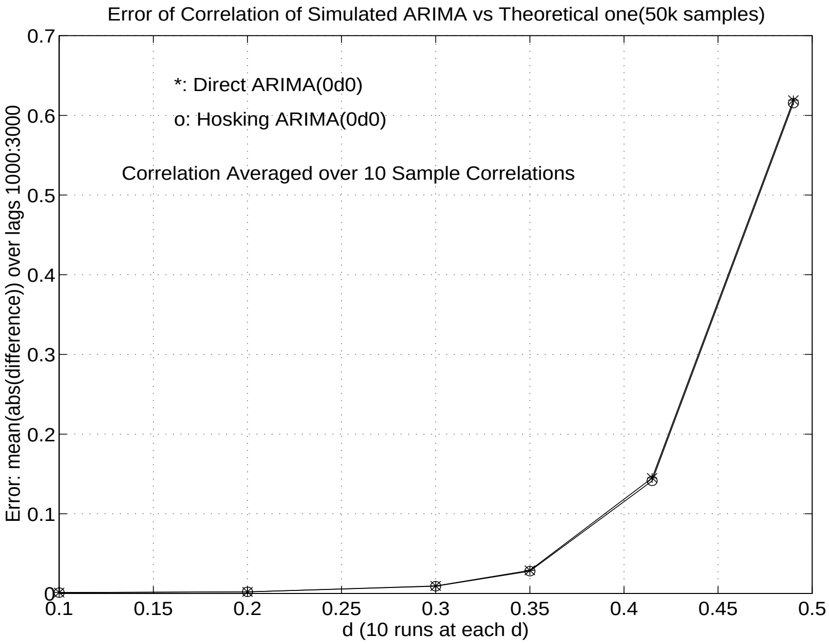
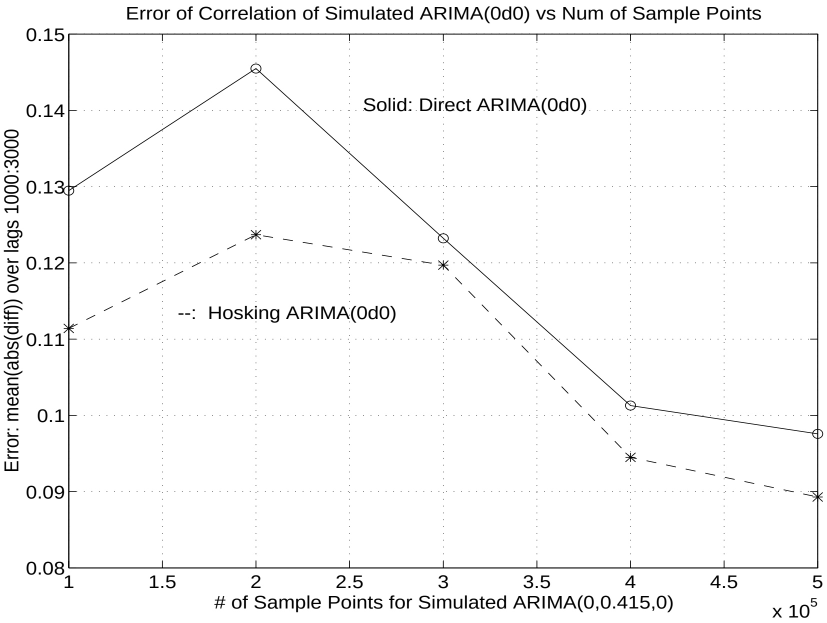


Figure 4-2: Error of the correlation function of the simulated *ARIMA(0,d,0)* (generated by two methods) with respect to the theoretical correlation function. Lags of comparison: 1000 to 3000. Value of d are 0.1,0.2,0.3,0.35, 0.415,0.49

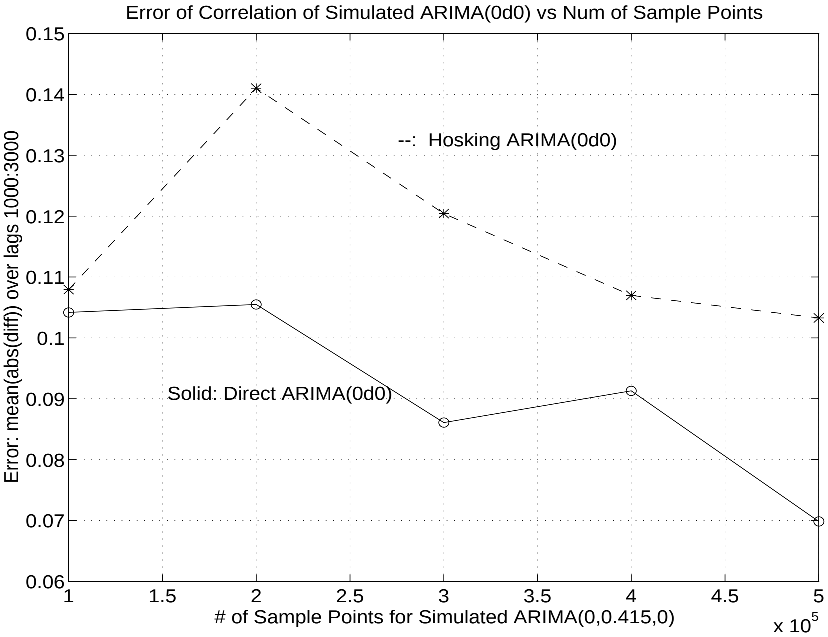
4.2 Correlation Behavior of the Simulated ARIMA(0,d,0) For Changing Amount of Sample Data

In this section, we will fix the value of *d* and compare first the correlations of the simulated *ARIMA(0,d,0)* with different amounts of sample data with respect to the corresponding theoretical correlation functions. The *d* value is chosen to be 0.415. The sample points ranges from 100k to 500k in a step size of 100k. They are drawn from the same *ARIMA(0,0,415,0)* time sequence with 500k sample points. The Hosking method is compared to the direct *ARIMA* method.

Figure 4-3 (a) and (b) show two cases of comparison, where four different random sequences were used to generate all four *ARIMA(0,0.415,0)* time sequences with 500k sample points for each. The error is measured as the mean difference

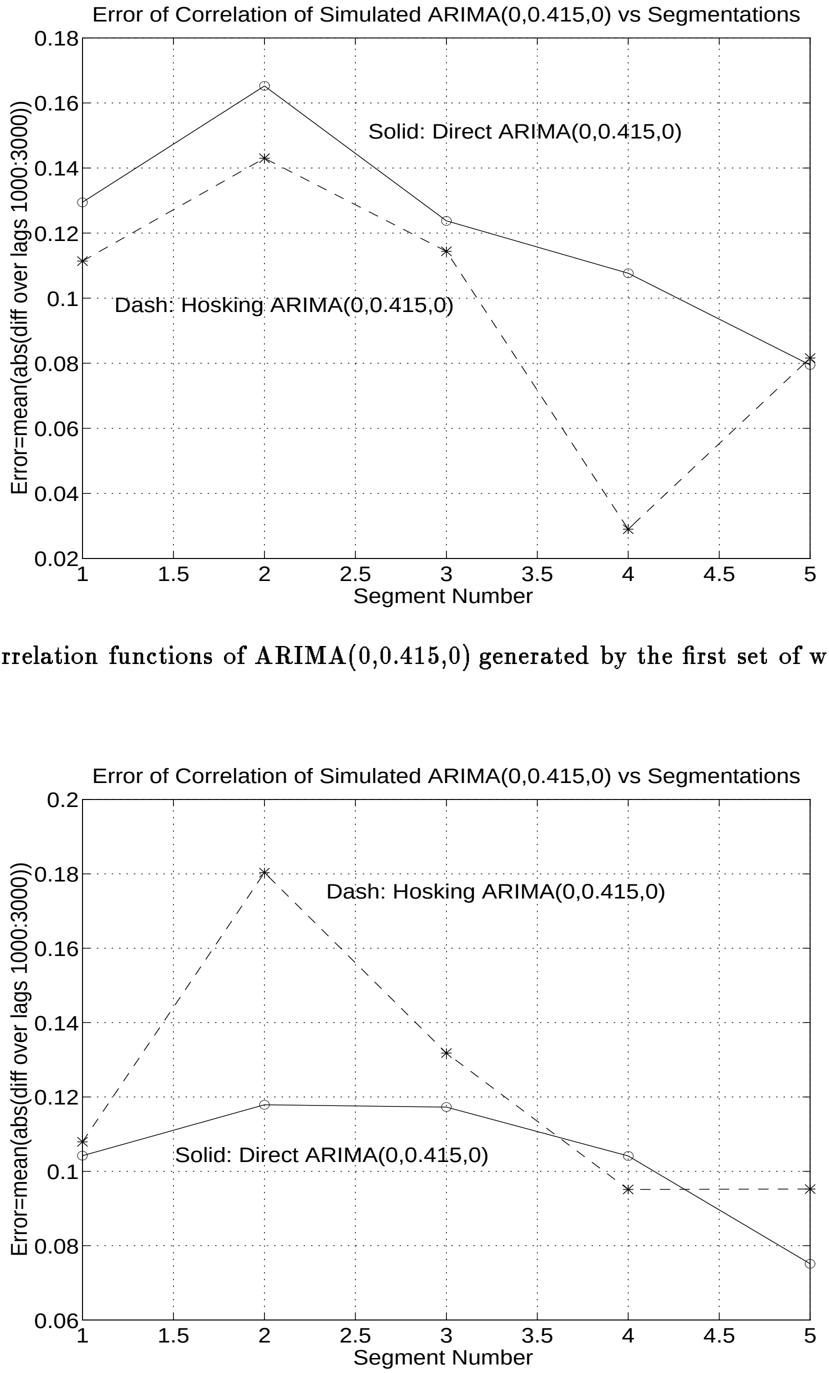


(a) Error of correlation functions of ARIMA(0,0.415,0) generated by the first set of white noise processes



(b) Error of correlation functions of ARIMA(0,0.415,0) generated by the second set of white noise processes

Figure 4-3: Error of the correlation functions of the simulated *ARIMA(0,d,0)* processes (generated by two methods) versus sample numbers of the time series. Value of d is 0.415



(a) Error of correlation functions of ARIMA(0,0.415,0) generated by the first set of white noise processes

(b) Error of correlation functions of ARIMA(0,0.415,0) generated by the second set of white noise processes

Figure 4-4: Error of the correlation function of the simulated *ARIMA(0,d,0)* versus different segmentations. Total sample number 500k. Segmentation size: 100k. Value of d is 0.415

of the correlation function of the simulated one to the theoretical one over lags from 1000 to 3000.

Clearly, for both methods, the error tends to decrease as the number of sample points increases. Again, this matches what we expected. Because of the randomness, plot (a) and plot (b) differ in that, in plot (a) the direct *ARIMA* method has a better performance than Hosking method, while the result is just the opposite in plot (b). So no conclusions about which one is better can be made.

In parallel, the correlations of different segments of an entire data set with size 500k are compared in figure 4-4. The segment size is 100k. Same four *ARIMA(0,0.415,0)* serieses used in figure 4-3 are used here. Results are shown in figure 4-4.

Interestingly, compared with figure 4-3 a similar tendency of error decrease is observed, but with respect to increasing segment number. That is, as more data is chopped off from the beginning, the error in the correlation decreases. This is a typical characteristic of the simulated *ARIMA(0,d,0)*, due to embedded limitation on infinite sample points as the long-range dependence requires. Figure 4-4 demonstrates that an effective way to compensate for the startup problem is to chop off the beginning part of the time series.

4.3 Measurement of d of the Simulated ARIMA(0,d,0)

So far, we have studied qualitatively the characteristics of the simulated *ARIMA(0,d,0)* in regard to the sample number, segmentation and different *d*.

In section 3.3, a method to estimate *d* of the simulated *ARIMA(0,d,0)* was discussed. In this section we are going to apply this method to characterize the degree of the long-range dependence of the simulated *ARIMA* and compare it to the intended *d*. The considered simulated *ARIMA(0,d,0)* sequences are generated by both the direct and the Hosking methods. The parameter d value ranges from 0.1 to 0.49.

The equation for calculating d is as stated in section 3.3,

$$d = \frac{\rho_{k+1}(k+1) - \rho_k k}{\rho_k + \rho_{k+1}} \tag{4.1}$$

Theoretically, the lag k can be any number in the range of [0, N], where N is the number of the data sample, as long as the correlation function is available. So presumably, a good measure of d is to get a sequences of d values by using different index k of the correlation, then take the resultant mean value. However, we need to be cautious at this point. We have observed that the accuracy of this method largely depends on what value d is and upon what lag of the correlation to use. In short, correct usage of this method for measuring d, depends upon the knowledge of the measured *ARIMA(0,d,0)* sequence.

The following figure 4-5 shows the first 50 consecutive measured d (in regard to index k) of 6 *ARIMA(0,d,0)* sequences with 50,000 sample points generated by the direct method. The first 50 d's are calculated by the first consecutive 25 correlations, where the first d is calculated by ρ_0 and ρ_1 and the second d is calculated by ρ_1 and ρ_2 and so on.

Obviously, the measured d in each case varies from the first d value as the correlation index k increases. The smaller the parameter d the bigger the variation. This result is consistent with sample correlation plots in figure 4-1, where correlation is smoother for larger d and noisier for smaller d.

Figure 4-6 is the corresponding variance of the measured 50 d for parameter d equals 0.1, 0.2, 0.3, 0.35, 0.415, 0.49 respectively. Sequences generated by both methods are measured. From what is shown in figure 4-6, the variance of the simulated *ARIMA(0,d,0)* sequences decreases as d increases. Both methods of generation show similar results.

An explanation of this is the following. When parameter d is small, the value

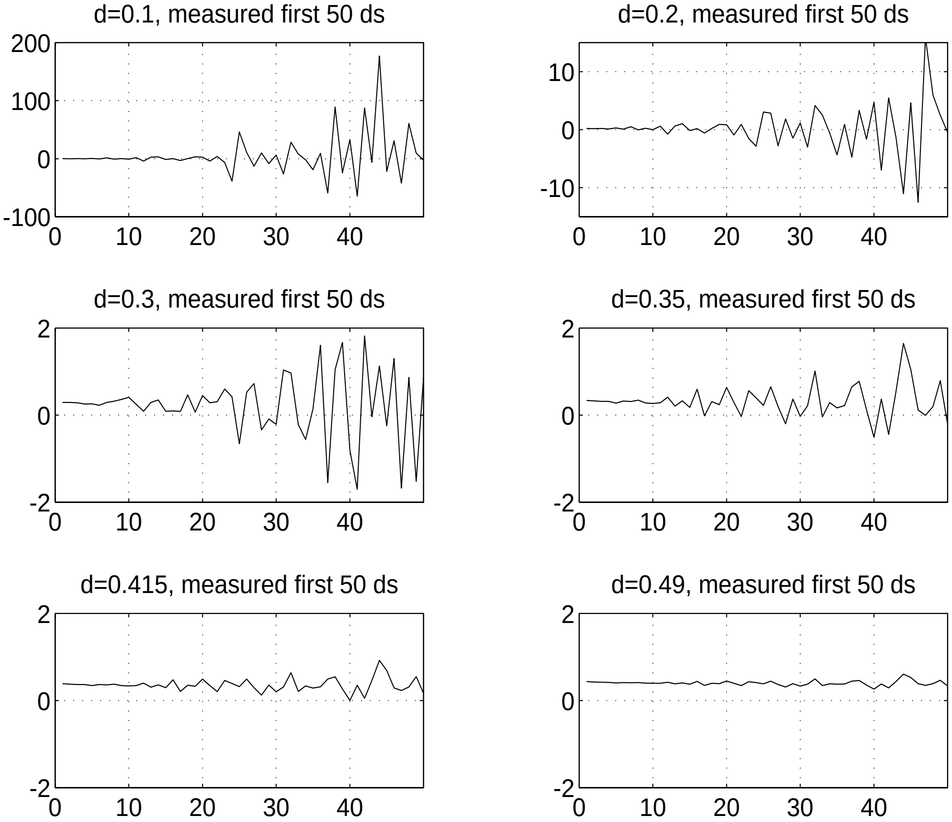


Figure 4-5: Measured first 50 d of *ARIMA(0,d,0)* generated by the direct method. The corresponding parameter of d is 0.1,0.2,0.3,0.0.35,415,0.49

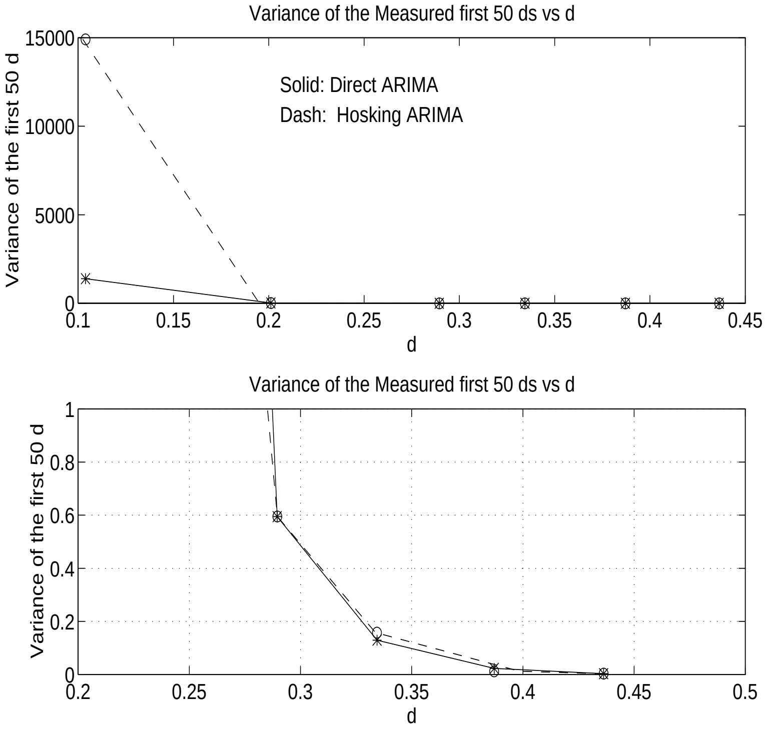


Figure 4-6: Variance of the measured first 50 d of simulated *ARIMA(0,d,0)* generated by direct method. The corresponding parameter d is 0.1,0.2,0.3,0.35,0.415,0.49 respectively

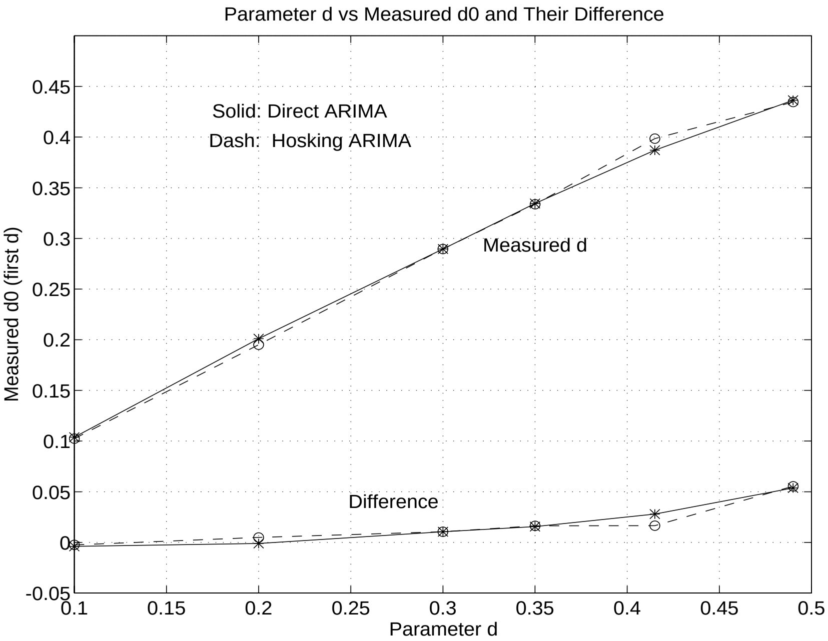


Figure 4-7: Measured first d versus the parameter d (originally set) which is set to be 0.1,0.2,0.3,0.35,0.415,0.49

of correlation is rather noisy as shown in figure 4-1. The randomness increases as the lag index k increases. As d increases the correlation becomes more smoother. So ρ_0 and ρ_1 are the most reliable values for calculating the d. In addition, ρ_0 and ρ_1 as statistical quantities encompass the maximum number of samples. Hence a measurement d based on ρ_0 and ρ_1 should be most reliable, especially when d is small.

Figure 4-7 and figure 4-8 show the measured value of d from ρ_0 and ρ_1 and the average of the first 10 d measurements (from ρ_0 and ρ_1, ρ_1 and ρ_2 through ρ_4 and ρ_5) as a function of the parameter (intended) d for the purpose of comparison. Again, the direct method is compared to the Hosking method. All six *ARIMA(0,d,0)* processes generated by each method have 50k sample points and are generated by six different random noise processes. However, for each d value, same random noise process is used for both methods. The difference between the parameter (intended) d and the measured d are the bottom curves in both figures.

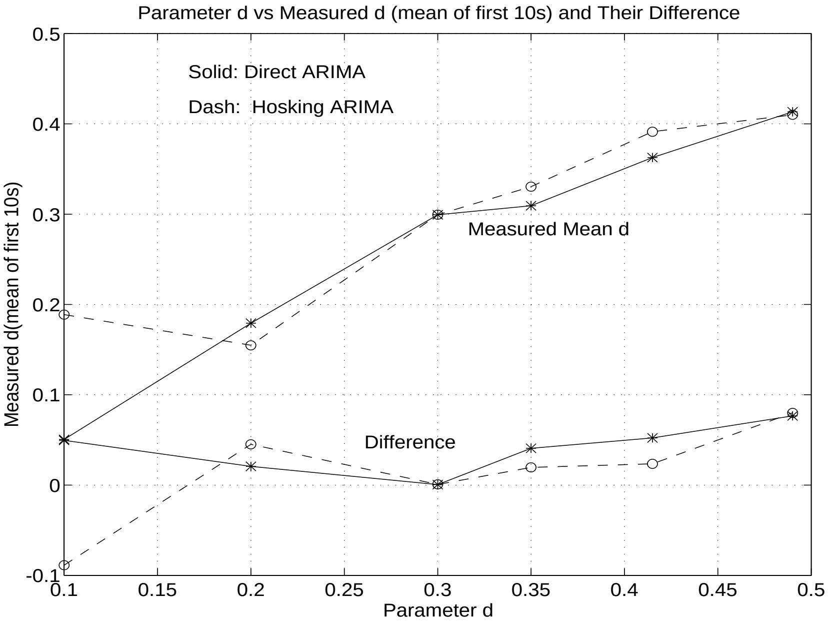


Figure 4-8: Measured *d*: average of the first 10 d versus the parameter *d* which is set to be 0.1,0.2,0.3,0.35,0.415,0.49

Note that the measurement based only on ρ_0 and ρ_1 matches the intended *d* better and is less noisy than the measurement based on averaging. Note also that the difference between the direct method and the Hosking method for both cases is small and negligible.

The method of measuring the *d* of a simulated *ARIMA(0,d,0)* discussed in the section may seem to have a disadvantage since it depends upon correlation, a statistical quantity, which is not reliable for small *d* when the number of sample points is not large enough. However this disadvantage reflects again the special characteristics of long-range dependence, i.e., its measurement requires a large number of sample points.

Despite the disadvantage of this method and the fact that it can only be used for *ARIMA(0,d,0)*, it is still a simple and direct way of testing the *d* of the simulated *ARIMA(0,d,0)* which was given a certain intended *d*. The difference between the two will provide enough information on the accuracy of the simulated sequence.

A general method of measuring the *d* or equivalently the degree of long-range dependence should be a method which is derived from the characteristics of the long-range dependence. One of such method is the *variance time plot* and will be introduced in the next chapter. Variance time plot is able to estimate *d* for any kind of self-similar process including *ARIMA(0,d,0)* and *ARIMA(p,d,q)*, though we will see that it is not perfect either.

Chapter 5

Modeling and Characterization of ARIMA(p,d,q)

5.1 Generating *ARIMA(p,d,q)*

In chapter 3, the basic definition of *ARIMA(p,d,q)* has been given in equation (3.1). In order to generate *ARIMA(p,d,q)*, we can directly implement equation (3.1) using white noise. However, since we have a way of generating *ARIMA(0,d,0)* from the previous chapter, we can generate *ARIMA(p,d,q)* from *ARIMA(0,d,0)*. This just requires some reformation of equation (3.1).

Equation (3.1) has the form:

$$\phi(B)\nabla^d Y_t = \theta(B)a_t \tag{5.1}$$

Let us move the differencing operator from left side to right side, we get:

$$\phi(B)Y_t = \theta(B)\nabla^{-d}a_t \tag{5.2}$$

Define $x_t = \nabla^{-d}a_t$, then move the operator to the left, which leads to the form of

$$\nabla^d x_t = a_t \tag{5.3}$$

This is exactly the same form of equation (3.3). Hence x_t is *ARIMA(0,d,0)*. Thus an alternative form of equation (3.1) is

$$\phi(B)Y_t = \theta(B)x_t \qquad t \geq 0, \tag{5.4}$$

This is just an *ARMA* filter, with an *ARMA(0,d,0)* process as the input.

From this equation we can get *ARIMA(p,d,q)* from *ARIMA(0,d,0)*. If $p = 2$ and $q = 2$, the equation (5.4) turns into

$$(1 - \phi_1 B - \phi_2 B^2)Y_t = (1 - \theta_1 B - \theta_2 B^2)x_t \qquad t \geq 0 \tag{5.5}$$

or

$$Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} - \theta_1 x_{t-1} - \theta_2 x_{t-2} + x_t \qquad t \geq 0 \tag{5.6}$$

Likewise, a similar format can be obtained for any p and q values.

5.2 Estimation of d

There are several formal mathematical methods of estimating the parameter d of a self-similar process, as we discussed in chapter 1. One of them is the *variance time plot.*

A variance time plot is a method that directly applies the characteristics of a self-similar or long-range dependent process. In section 2.3 we have discussed three equivalent features of long-range dependent process, which are features of self-similar process as well. One of them is that the variance of the arithmetic mean of an aggregated process $X^{(m)}$ decreases more slowly than the reciprocal of the sample size, that is,

$$var(X^{(m)}) \sim am^{-\beta}, \quad as \quad m \rightarrow \infty, \tag{5.7}$$

Therefore, given a self-similar process or a long-range dependent process, we can test its d via equation (5.7).

Take the log base 10 on both sides of equation (5.7), we get

$$log_{10}(var(X^{(m)}) \sim log_{10}a - \beta log_{10}m \tag{5.8}$$

where $X^{(m)}$ is defined in chapter 2 equation (2.2).

The so-called *variance time plot* is obtained by plotting $log_{10}(var(X^{(m)})$ against $log_{10}(m)$. The slope at large m is the estimated β. Values of this asymptotic slope between -1 and 0 suggest self-similarity [15]. According to the definition of the Hurst paramter H, $H = 1 - \beta/2$ and $H = d + 1/2$, therefore in terms of β

$$d = 1/2 - \beta/2 \tag{5.9}$$

However, the variance time plot is not reliable for empirical records with small sample size (several hundred thousand is not considered to be big sample size). Otherwise it is highly useful and gives a rather accurate picture about the self-similar nature of the underlying time series and about the degree of self-similarity. It can be used for measuring parameter d of *ARIMA(p,d,q)* as well as for real trace data. In cases where not enough data samples are available for an accurate

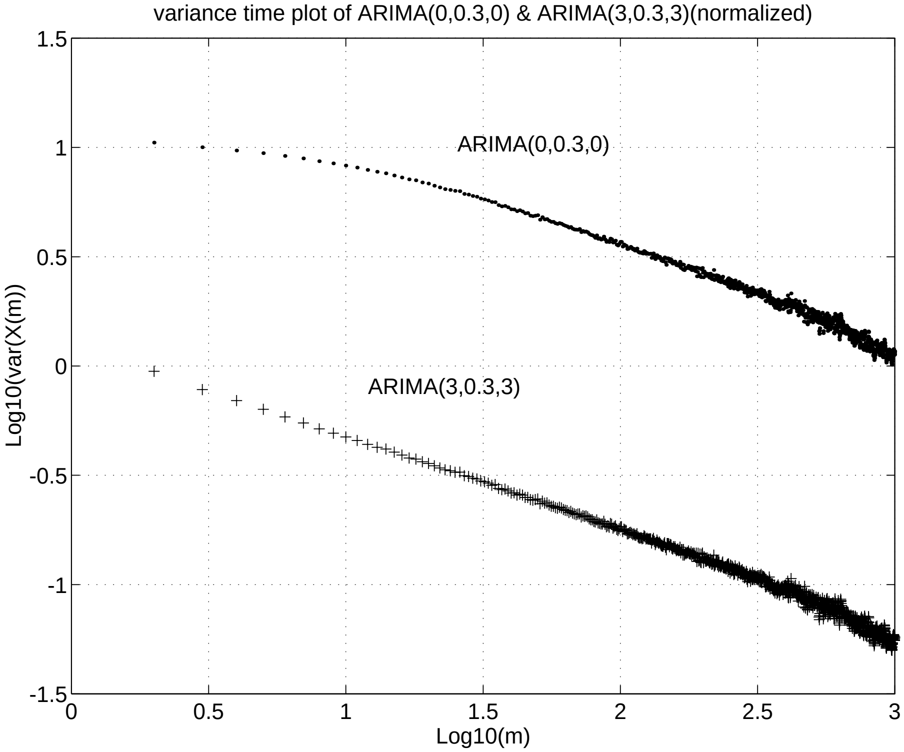


Figure 5-1: Variance time plot of *ARIMA(0,0.3,0)* and *ARIMA(3,0.3,03)* (both are normalized). Maximum m=1000

variance time plot, we still can use the variance time plot to get a rough estimate of *d* and the fine tuning can be done by matching the correlation function.

A direct use of the variance time plot is to check whether the *ARIMA(p,d,q)* generated by the *ARIMA(0,d,0)* retains the same *d*, i.e., the same slope as the generating *ARIMA(0,d,0)* for big *m*.

Figure 5-1 shows the variance time plots of *ARIMA(0,0.3,0)* and the corresponding *ARIMA(2,0.3,2)* that is generated using equation (5.4).

Clearly, the two curves are well parallel to each other and tend to keep in parallel as m increases, which implies that they have the same degree of self-similarity.

5.3 Estimation of Parameter θ and ϕ

Having the method of generating *ARIMA(p,d,q)* and the method of estimate parameter d available, the last and final question is: how do we estimate parameters θ and ϕ that describe the short range dependence? In this section, the method of estimating parameters θ and ϕ of *ARMA(p,q)* will be discussed. It will be shown how this method can be applied for the same parameter estimation but of *ARIMA(p,d,q)*.

The *pseudo linear regression algorithm* (PLR) [24] has been used for parameter estimation of *ARMA(p,q)*. It works in the following way. Assuming that values of p and q are given, an *ARMA (p,q)* process z_t can be expressed as

$$z_t = b_1 z_{t-1} + b_2 z_{t-2} + \cdots + b_p z_{t-p} + c_1 w_{t-1} + \cdots + c_q w_{t-q} + w_t \tag{5.10}$$

b_i's and c_i's are the corresponding parameters that need to be estimated and w is white noise.

Define the following matrix,

$$\theta_t = (b_1 \quad b_2 \quad \cdots \quad b_p \quad c_1 \quad c_2 \quad \cdots \quad c_q)^T \tag{5.11}$$

which is a column matrix of size $(p+q)$x1 that contains all the parameters. It can be initialized to any value. Then define

$$\phi_t = (z_{t-1} \quad z_{t-2} \quad \cdots \quad z_{t-p} \quad \varepsilon_{t-1} \quad \cdots \quad \varepsilon_{t-q})^T \tag{5.12}$$

which is also a column vector of size $(p+q)$x1 and contains the previous feedback values. ε can be initialized to any values.

Further define matrix P_n of size $(p+q)$x$(p+q)$ and initialized to $P_0 = I$, the unit diagonal matrix.

At time t, update the coefficient vector θ and ϕ and P_t according to the following rules:

$$\begin{aligned}
\theta_t &= \theta_{t-1} + P_{t-1}\phi_{t-1}(z_t - \phi_{t-1}^T\theta_{t-1}) && (5.13)\\
\varepsilon_t &= z_t - \phi_{t-1}^T\theta_t && (5.14)\\
\phi_t &= (z_t \quad \cdots \quad z_{t-p} \quad \varepsilon_t \quad \cdots \quad \varepsilon_{t-q})^T && (5.15)\\
& && (5.16)\\
P_t &= P_{t-1} - \frac{P_{t-1}\phi_t\phi_t^T P_{t-1}}{1 + \phi_t^T P_{t-1}\phi_t} && (5.17)
\end{aligned}$$

The outline of using this algorithm is

(1) Set the correct values of p and q.

(2) Initialize all the matrices to some random values.

(3) Update the parameters according to equation (5.14) through equation (5.17).

(4) Check the parameter vector θ for convergence; if it has not converged, then repeat the procedure.

However, this method only applies for the parameter estimation of an *ARMA* process. The only way to use this method for parameter estimation of an *ARIMA(p,d,q)* is to find the relation between the two.

From the definition of *ARIMA(p,d,q)* in equation (3.1), we define

$$u_t = \nabla^d y_t \tag{5.18}$$

such that (3.1) can be expressed in terms of u_t in the following way,

$$\phi(B)u_t = \theta(B)a_t \tag{5.19}$$

Compare equation (5.19) with equation (5.10), we can see that they have the same structure. In fact, equation (5.19) is the exact definition of an *ARMA(p,q)* process. Therefore equation (5.18) provides the transform from *ARIMA(p,d,q)* to *ARMA(p,q)*, namely, given *ARIMA(p,d,q)* sequence, we can get the corresponding *ARMA(p,q)* via equation (5.18). Then we can use the PLR algorithm to estimate the θ and ϕ parameters.

Thus we have established a method of estimating d by the method of variance time plot for any type of self-similar process and the method of estimating parameters θ and ϕ by equation (5.18) and the PLR algorithm, assuming that values of p and q are known.

The following is an experiment of applying the PLR algorithm to *ARMA(p,q,)* and *ARIMA(p,d,q)* processes whose parameters are all known to get estimated parameters. Thus by comparing the original parameters to the estimated ones, we can show how well this algorithm works.

The first example is *ARMA* time series generated by white noise that has a zero mean and variance one. The p and q are set to be 3 for both. The coefficients of $\theta_1, \theta_2, \theta_3, \phi_1, \phi_2$ and ϕ_3 are (0.5, -0.2, ,0.3, 0.9, -0.2, 0.1). We input this time series to the PLR algorithm as if its coefficients are unknown, set p=3, q=3, and initialize all parameters to 0. Then we execute the above procedure step by step. The estimated coefficients converge slowly towards their true values after 50,000 iterations. The coefficients of the last 5 iterations are listed in Table 5.1.

Among the four coefficients, only ϕ_1 has 34% error, all the others remain within 15% error.

In the second example, we check the θ and ϕ coefficients of an *ARIMA(3,0.3,3)*. To do that, it is necessary to transform first the *ARIMA(p,d,q)* sequences into the

Table 5.1: *ARMA(3,3)* coefficients estimation

	θ_1	θ_2	θ_3	ϕ_1	ϕ_2	ϕ_3
Original Value	0.5	-0.2	0.3	0.9	-0.2	0.1
Iterations	Estimated Coefficients					
27527	0.5281	-0.2294	0.3020	0.5938	-0.1459	0.0868
27528	0.5281	-0.2294	0.3020	0.5938	-0.1459	0.0868
27529	0.5280	-0.2294	0.3020	0.5939	-0.1456	0.0869
27530	0.5280	-0.2294	0.3020	0.5939	-0.1456	0.0868
27531	0.5280	-0.2294	0.3020	0.5939	-0.1457	0.0868

Table 5.2: *ARIMA(3,0.3,3)* coefficient estimation

	θ_1	θ_2	θ_3	ϕ_1	ϕ_2	ϕ_3
Original Values	0.5	-0.2	0.3	0.9	-0.2	0.1
Iterations	Estimated Coefficients					
39996	0.5454	-0.0329	0.3573	0.8807	0.0334	0.0117
39997	0.5453	-0.0329	0.3574	0.8808	0.0335	0.0116
39998	0.5453	-0.0329	0.3574	0.8808	0.0335	0.0117
39999	0.5453	-0.0329	0.3574	0.8808	0.0335	0.0116
40000	0.5453	-0.0329	0.3574	0.8808	0.0335	0.0116

Table 5.3: *ARIMA(2,0.3,1)* coefficient estimation

	θ_1	θ_2	ϕ_1
Original Values	1.2	-0.5	0.3
Iterations	Estimated Coefficients		
29996	1.3264	-0.4680	0.4982
29997	1.3264	-0.4680	0.4982
29998	1.3264	-0.4680	0.4982
30000	1.3264	-0.4680	0.4981
30001	1.3264	-0.4680	0.4981

corresponding *ARMA(3,3)*. The rest of the procedure is then the same as the first example. The results are shown in Table 5.2.

Comparing the two cases, obviously the former has better results. This is because noise information is not included in the PLR algorithm for parameter estimation. So the randomness of noise that generate the *ARMA* and the *ARIMA(0,0.3,0)* contribute to the error. Moreover, some information is lost during the transformation of *ARIMA(p,d,q)* to *ARMA(p,q)*. They are the short-term correlation contributed by *d*.

Table 5.3 gives a last example of coefficient estimation of *ARIMA(2,0.3,1)*, where the original short term coefficients are ((1.2,-0.5,0.3). Again, we can see the results agree reasonably well.

5.4 Comparison of ARIMA(p,d,q) with ARIMA(0,d,0) in Terms of Correlation

In the previous chapter, we have stated that parameters θ_i and ϕ_i determine the short-term correlation, while the value of *d* determines the long-term correlation.

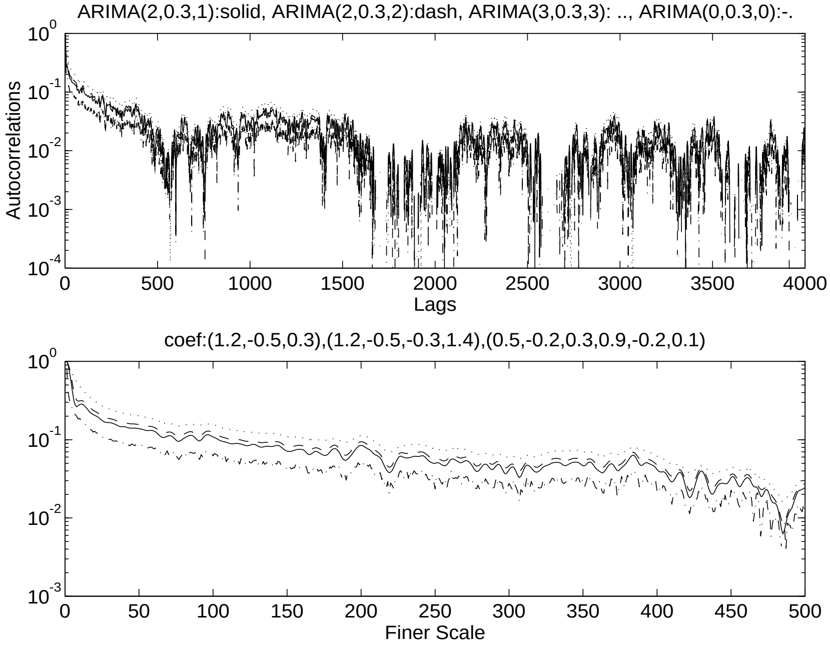


Figure 5-2: Correlation Comparison of *ARIMA(2,0.3,1)* with coefficients (1.2,-0.5,0.3) (solid), *ARIMA(2,0.3,2)* with coefficients (1.2,-0.5,-0.3,1,4) (dash), *ARIMA(3,0.3,3)* with coefficients (0.5,-0.2,0.3,0.9,-0.2,0.1) (dot) *and* *ARIMA(0,0.3,0)* (dash dot). 50,000 sample size for all sequences

Since now we can generate *ARIMA(p,d,q)* using *ARIMA(0,d,0)* with the same *d*, let us look at the correlation behaviour of the two.

The following *ARIMA(p,d,q)* are considered:

(1) *ARIMA(2,0.3,1)* with coefficients $\theta_1 = 1.2, \quad \theta_2 = -0.5, \quad \phi_1 = 0.3$;

(2) *ARIMA(2,0.3,2)* with coefficients $\theta_1 = 1.2, \quad \theta_2 = -0.5, \quad \phi_1 = -0.3, \quad and \quad \phi_2 = 1.4$

(3) *ARIMA(3,0.3,3)* with coefficients $\theta_1 = 0.5, \quad \theta_2 = -0.2, \quad \theta_3 = 0.3, \quad \phi_1 = 0.9, \quad \phi_2 = -0.2, \quad \phi_3 = 0.1$

All of them are generated by the same *ARIMA(0,0.3,0)* sequence according to equation (5.4). All the coefficients are chosen such that stationary *ARIMA(p,d,q)* processes are obtained.

Figure 5-2 shows the correlation of the above three sequences together with correlation function of *ARIMA(0,0.3,0)*. Sample size is 50,000 for all cases. Interestingly, we see that the correlation functions of all three *ARIMA(p,0.3,q)* are a nearly parallel up shift from the correlation of the *ARIMA(0,0.3,0)* regardless of the different p and q values for each case. And the shift covers at least 500 lags, a much larger lag than we intuitively expected from short term contribution, since the values of p and q determine only the short term correlation.

However, this is again the special feature of long-range dependence. Unlike the *ARMA(p,q)* or equivalently *ARIMA(p,0,q)* where uncorrelated white noise is the input of moving average portion, the input of the moving average portion of *ARIMA(p,d,q)* is a long-range dependent *ARIMA(0,d,0)* time sequence. Hence, for a nonzero p or q the long-range dependent correlation tends to propagate along short-term lags of the underlying *ARIMA(p,d,q)* process. This is why the correlation curve shifts upward in parallel. On the other hand, the short-term correlation due to nonzero p and q runs only through small lags and therefore its change in the short-term correlation is not obvious compared with the effect of the strong long-range dependence. Note however, that what we see from figure 5-2 does not imply that any *ARIMA(p,d,q)* process generated from *ARIMA(0,d,0)* with the same d must have a correlation function that is in parallel to that of the *ARIMA(0,d,0)*. Actually, with proper θ_i and ϕ_i coefficients it is possible that the propagation of the *ARIMA(0,d,0)* in the correlation will diminish for large lags due to the compensation of short-term contribution and thus leave only changes in short term correlation observable. Such an example will be shown in chapter 6.

Based on this observation, we can draw some conclusions about the inaccuracy of θ_i and ϕ_i parameter estimation by the pseudo regression linear algorithm. Imagine that we are given a real *ARIMA(2,0.3,1)* process whose coefficients need to be estimated. Let its actual coefficients be exactly the same as those of the *ARIMA(2,0.3,1)* process that we used in figure 5-2, i.e., $\theta_1 = 1.2$, $\theta_2 = -0.5$,

$\phi_1 = 0.3$. Two possible effects on correlation exist due to estimation error. If we set the $p = 3$ and $q = 3$ for estimation and assume that the estimated coefficients turned to be $\theta_1 = 0.5$, $\theta_2 = -0.2$, $\theta_3 = 0.3$, $\phi_1 = 0.9$, $\phi_2 = -0.2$, $\phi_3 = 0.1$ which are the same as those of the *ARIMA(3,0.3,3)* process we used in figure 5-2. This will result in the first possible error on correlation function. That is, its correlation will differ from the real correlation by almost a positive constant value and the difference will cover a medium range of lags, just as shown in figure 5-2. Another different set the estimated coefficients can result in second possible effect on the real correlation function. That is, only short-term correlation will have error, no effect on large lag correlation. We will discuss this case more in chapter 6.

In both case, the error in correlation function is not caused by error of a single parameter estimation but rather due to the combination of all the short-term coefficients. Obviously, inproper values of p and q may give error. Unfortunately, we have not developed so far a better way of estimating p and q than guessing their values by observing the correlation function. However, as our understanding, no matter what estimated short-term coefficients we get, error in correlation function falls into the above two categories.

Although the error of parameter estimation is unavoidable, it is very important to realize that this error will not significantly effect the long-range dependence. Figures 5-2 and 5-1 are demonstrations of this. Therefore, a variance time plot with large m will eventually retain the same slope, namely, it will retain the same d.

Chapter 6

Video Trace Data Modeling

Having obtained methods of modeling self-similar/long-range dependent *ARIMA(p,d,q)*, we are now able to model real self-similar traffic. Among the Ethernet traffic and the video "Star Wars" source files that are observed to be self-similar or long-range dependent at Bellcore [15], we choose the latter to be our modeling target. This is because the available Ethernet trace file provides only packet arrival time and packet length. In order to make a process of byte/time unit which is claimed to be self-similar out of the raw data, we have to change the original data structure. Since we are not sure that this processing will not change the self-similar nature of the raw data, we rather use the video Starwars data which is recorded as bytes/frame with fixed frame rate for modeling and later for simulation. The real trace data of starwars has been acquired from Bellcore.

There are 60 files of "Star Wars" with a total of 171,000 frames. Real data in bits/frame is available. The frame period is 1/24 seconds.

Figure 6-1 gives the correlation function with 15,000 lags of the total bits/frame process with 171,000 samples points.

Compared with the correlation function curves of *ARIMA(p,d,q)* that we have seen so far, this curve is obviously not a typical *ARIMA(p,d,q)*. However, it clearly possesses slowly decaying correlation. So the goal of this trace data modeling is

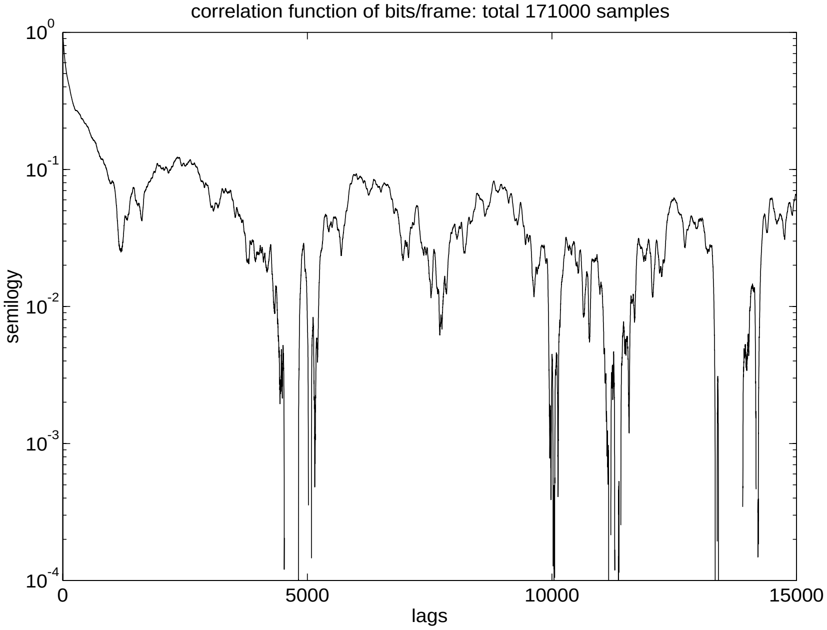


Figure 6-1: Correlation Function of Video Starwars in Bits/Frame. Total 171000 Samples.

to capture the long-range dependent characteristics such that the simulated time series has the closest match to its correlation function. Perfect modeling in terms of correlation function of a trace data with an irregular correlation structure is impossible.

The first step of modeling is to estimate the parameter d of the trace data by variance time plot. It is shown in figure 6-2. Note that the maximum m we chose is 10,000. Since we have 171,000 sample points in our trace data, the aggregated process $X^{(10000)}$ has only 17 sample points. This sample size is too small for calculating any statistics. That is why the curve becomes very noise when log(m) is greater than 3. The reliable part of measuring d by this variance time plot should exclude this portion.

According to our earlier discussion in section 5.2, we can calculate d from the slope of the curve β by using equation (5.9). From the plot, the slope at log(m) around 2.5 is 0.2, the resulting d is then 0.4.

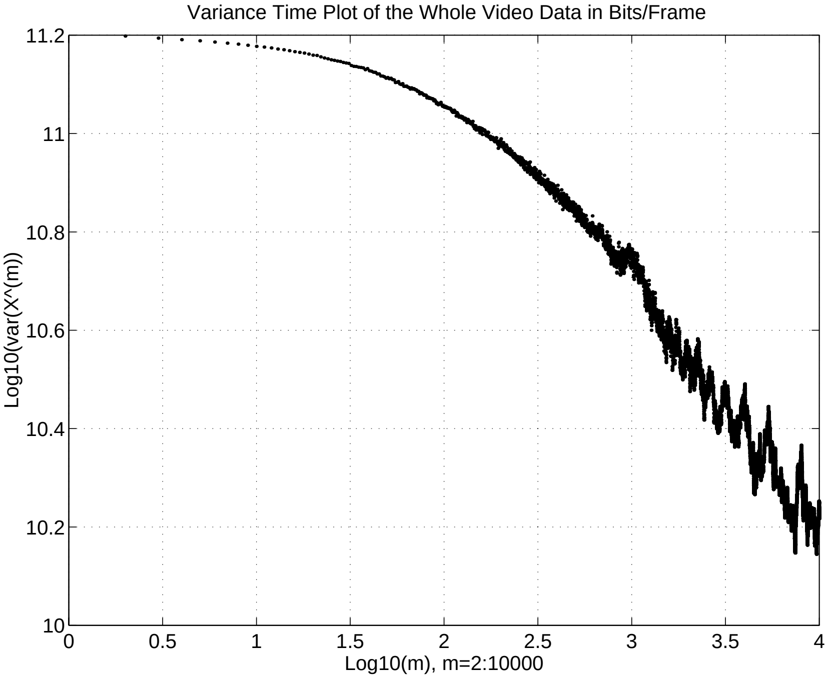


Figure 6-2: Variance time plot of video data in bits/frame. Number of aggregation m range from 2 to 1000 in a step size of 1.

Next, we generated *ARIMA(0,0.4,0)* sequence for 171,000 samples and plot the correlation function against the correlation plotted of the video trace data. The comparison showed that *d*=0.4 was too small. The variance time plot does not provide an accurate *d* due to the lack of sample points, so fine tuning is needed.

Basically the fine tuning is done by observing the correlation function of the trace data and choosing a proper *d* such that the correlation of the underlying *ARIMA(0,d,0)* has the best match to the counterpart of trace data. The best match found has a *d*=0.415 in this case.

Figure 6-3 shows the correlation functions of both the trace data and the simulated *ARIMA(0,0.415,0)*. It is generated by the direct *ARIMA* method. In general, the correlation of *ARIMA(0,0.415,0)* matches as well as can be expected for large scale of lags, but differs by a small gap in the short-term part. Note that this *ARIMA(0,0.415,0)* time sequence is chosen to be the best for matching among 10 other sequences that are generated using 10 different white noise processes with

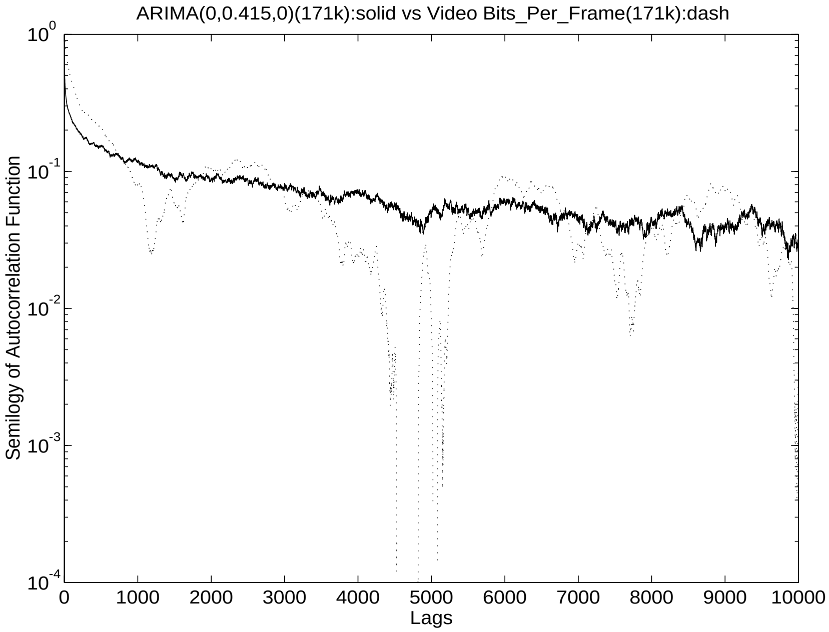


Figure 6-3: Correlation function of video trace data in bits/frame versus the correlation function of *ARIMA(0,4,15,0)*. Sample data is 171000 for both cases.

different seeds.

The next step of modeling is to determine values of p, q and estimate the corresponding parameters θ_i and ϕ_i using the method described in section 5.3, that is, transforming the trace data into an *ARMA(p,q)* process following equation (5.18) and then applying the pseudo regression linear algorithm for parameter estimation.

Unfortunately, the estimated coefficients of the transformed *ARMA* process does not converge for any value of p and q in this case. This is simply due to the fact that the trace data is clearly not a true *ARIMA* process, so the transformed process can no longer be *ARMA*.

If we definitely want to compensate the short-term difference, we can simply guess the coefficients for certain p and q. This will require that an *ARIMA(p,d,q)* process be generated from an *ARIMA(0,d,0)* process according to equation (5.4). Then the correlation function can be plotted to see whether it matches the trace

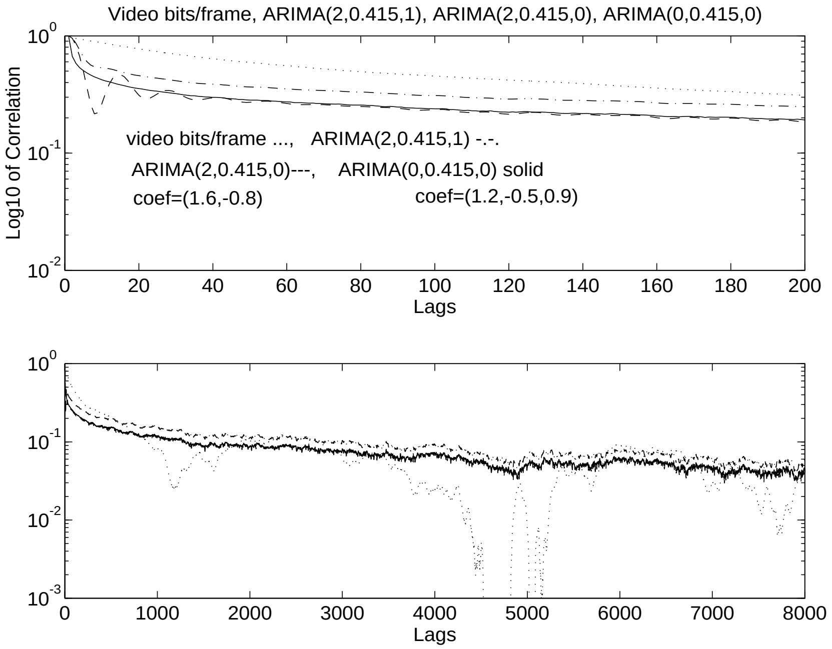


Figure 6-4: Correlation of video trace data (dotted line) versus that of *ARIMA(2,0.415,1)* with coefficients (1.2,-0.5,0.9) (dash dot line) and *ARIMA(2,0.415,0)* with coefficients (1.6,-0.8) (dash line) and *ARIMA(0,0.415,0)* (solid line)

data better or not.

Figure 6-3 provides some examples of the simulated *ARIMA(p,d,q)* with some randomly chosen short-term coefficients θ_i and ϕ_i It can be seen from the plots that the correlation function of *ARIMA(p,d,q)* is either parallel on top of the correlation function of the trace data or it oscillates in the short-term portion and has no effect on the long-term correlation. The particular effect depends totally on the combination of all the coefficients. These are the two cases we referred in section 5.4 that can happen to the correlation of the *ARIMA(p,d,q)*.

For the purpose of modeling, we want the second case to happen. However, the gap between video trace data and *ARIMA(0,0.415,0)* covers the range up to approximately 300 lags, which implies that value of p may be large. We are aware of no means to analytically estimate the short-term and it is hardly possible, if not impossible, to select proper coefficients by guessing that will result in a good match to the correlation of the trace data. On the other hand, our major goal of modeling is to capture the long-range dependence features of the trace data such that we can use the simulated sequence to study the network performance under this special traffic. Therefore, we can say that to some extent the *ARIMA(0,0.415,0)* provides a fair model for the trace data.

The *ARIMA(0,0.415,0)* time sequence at this point is generated by white noise whose mean is zero and variance is one. So the data are floating numbers, either positive or negative. The final version of simulated *ARIMA(0,0.415,0)* needs to be scaled to the mean and variance of the trace data. Moreover, since the simulated process is supposed to be bits/frame, the data should be rounded up to integer values and any negative number should be replaced by zero. Correlation of the final version of *ARIMA(0,0.415,0)* generated by direct *ARIMA* method is indistinguishable from that of unscaled *ARIMA(0,0.415,0)*.

A different view point of comparing the simulated sequences to the real trace data is to compare their average rate and peak rate. Figure 6-5 and figure 6-6 show average rate curve in bits/sec of the trace data and *ARIMA(0,0.415,0)* time series generated both by the direct *ARIMA* method and the Hosking method. Average rate is measured over a window of every four frames. Mean average rate is an average over the average rate of each window size. Peak rate is the maximum value of all the average rates. We can see from the plot that both simulated data matches well with the mean average rate but not the peak rate.

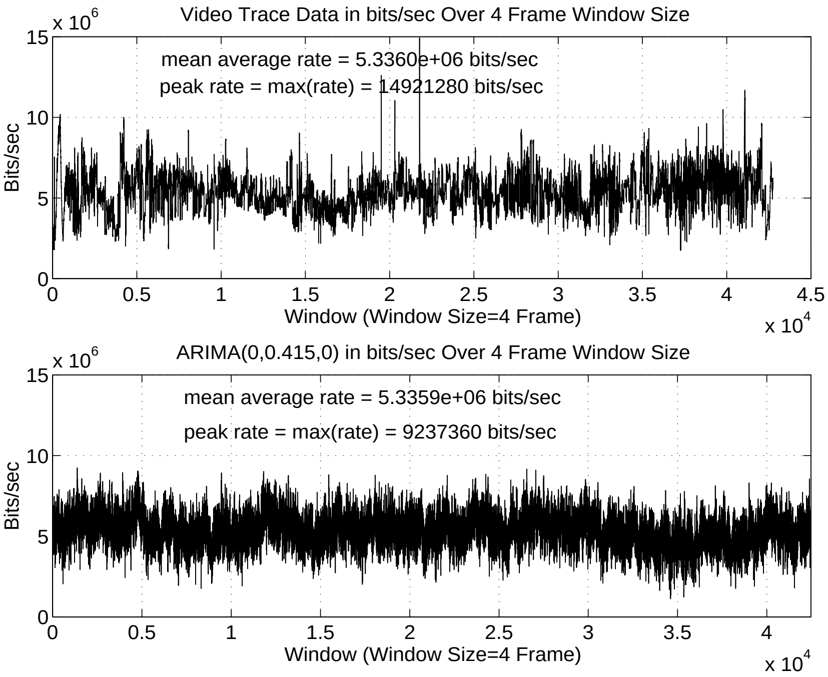


Figure 6-5: Average rate processes of video trace data versus *ARIMA(0,0.415,0)* generated by the direct *ARIMA* method. Window size is 4 frames

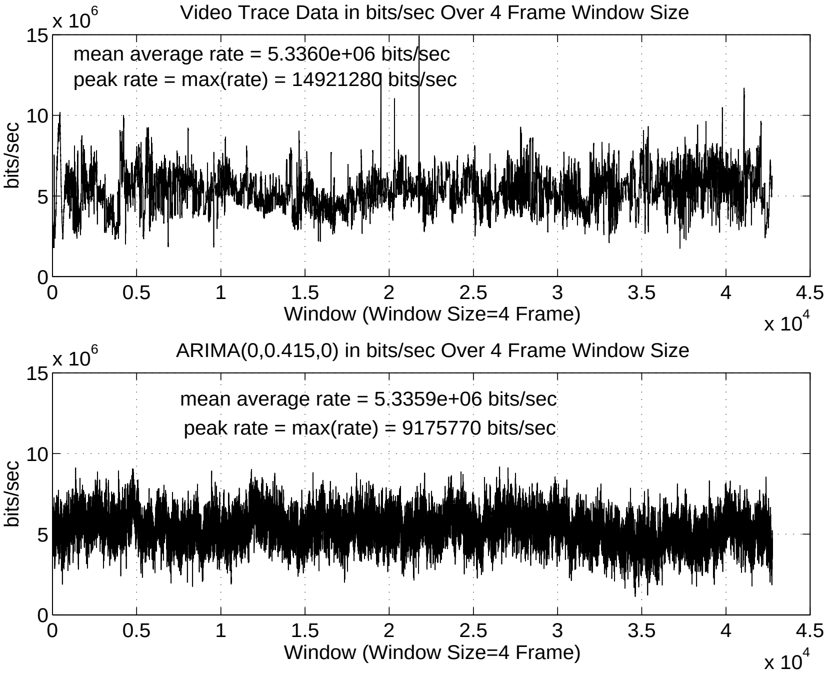


Figure 6-6: Average rate processes of video trace data versus *ARIMA(0,d,0)* generated by the Hosking method. Window size is 4 frames

Chapter 7

Performance Simulation of Long-Range Dependent Traffic

The ultimate goal of modeling of the self-similar/long-range dependent traffic is to study network performance using the simulated model and understand the behavior and effect on the network. Since the self-similar phenomenon is newly observed in telecommunication traffic and since such traffic does have very different statistical properties than the conventional models, it is of great interest to see what significant difference does it make on the network performance and under what conditions are these differences evident.

From our earlier discussion in section 2.5, simulation of aggregated LAN traffic in a SMDS environment conducted at Bellcore reveals that overall packet loss decreases very slowly with increasing buffer capacity, in sharp contrast to Poisson-based models where losses decrease exponentially fast with increasing buffer size. Packet delay (95 percentile) on the other hand, always increases with buffer capacity, again in contrast to the formal models where delay does not exceed a fixed limit regardless of buffer size.

With the "Star Wars" data aquired from Bellcore and our corresponding simulated *ARIMA(0,0.415,0)* process, we are able to simulate performance for both processes for loss, delay and delay variance using Block Orienated Network Simulator tools (BONeS) and hence test Bellcore's claim.

7.1 BONeS Simulation Model

The simulation performance measures considered are cell loss ratio (CLR) versus buffer size, cell delay versus buffer size and delay variance versus buffer size. The system environment is simply four video sources aggregated into a single FIFO queue.

One system uses "Star Wars" trace data in bits/frame as the video source traffic. The other system takes the corresponding simulated *ARIMA(0,0.415,0)* process in bits/frame as its video traffic source. Before the four sources merge into one queue, an ATM segmenter is used for each source to break the video packet into cells. Both the source module and the segmenter module are obtained from a previous project [23].

Figure 7-1 provides the diagram of the BONeS system model.

Source data are read from files. Although we have only one video trace data available that contains 171,000 frames, we can make four pseudo independent sources out of it by reading the same file beginning at a different point for each source. In our case, the first source reads from the beginning of the file. The second source starts from 25% offset of the beginning and wraps back to the beginning when the end is reached. The third and fourth sources start from 50% offset and 75% offset of the beginning respectively and wrap around if necessary. The same thing is done to the simulated *ARIMA(0,0.415,0)* process.

Note however, the resultant aggregated source is a periodic source. The period is 25% of the file. This is because by the time the last source reaches the end of the file, all the sources have gone through 25% of the file from their starting points. Subsequently, the first source will continue reading in data from 25% offset where

Figure 7-1: Simulation model of aggregated video source over a single queue

the second file started. The second source will go on from 50% offset, where the third source started, etc. Thus forms the second cycle of the period. In fact, this periodicity does not provide any more information than the first cycle. Therefore, the fourth source is made slightly different from all the other sources in such a way that the simulation will be terminated after it reaches the end of the file, which is also the end of the first cycle. This still provides sufficient data for testing. The start time of reading the file is set to be a uniform random number for each source so that simultaneous arrivals are avoided.

After each video packet in bits is broken into cells and leaves the segmenter, they are routed to the FIFO queue. Total number of cells that go into the queue is collected and written into a file and updated upon the issuing of a uniform pulse. Cells that are rejected by the queue are considered to be lost. Similarly, the number of losses is collected and written into a file upon the issuing of the uniform pulse.

Table 7.1: Parameters for BONeS Simulation

Source Parameters	Values
Video File 1 Start	1st frame
Video File 2 Start	42750th frame
Video File 3 Start	85500th frame
Video File 4 Start	128250th frame
Slice=0/Frame=1	1
Segmenter and FIFO Parameters	**Values**
Max Frame Size (bits)	627672(trace)/436047(ARIMA)
Segmenter Processing Delay	0.15 msec
Router to DSU Link Rate (Segmenter Input Rate)	13.25 Mbps
DSU to Access Line Link Rate	7.36 Mbps
Output Line Capacity(bps)	29.46 Mbps
Non-Cut Through Segmenter?	1 (Cut-through)
Maximum Queue Size	100 cells to 300 cells
Load	0.8

Two probes were used. A Generic Probe was used to record the history of the cell losses. A Batch Statistic Probe (F2-F1) was used to calculate the cell delay and delay variation statistics.

Buffer size is a simulation parameter in our simulation. The range of interest is from 100 cells to 300 cells with an increment of 50 cells. The other simulation parameters are listed in Table 7.1.

Since the segmenter was obtained directly from a previous project, the name of the parameters remained unchanged. "Router to DSU Link Rate" is in fact the rate at which video packets are input to the segmenter. This rate is set to be greater than the source arrival peak rate so that no loss should occur. "DSU to Access Line Link Rate" is actually the output rate of the segmenter after packets are segmented into cells. In order to let cell loss event occur, the link rate of the FIFO is set to be four times of the output rate of a single segmenter, that is, the total output rate of four segmenters are the same as the FIFO link rate, which is the "Output Link Rate" of value 29.46 Mbps. Once the source average arrival rate is known (provided in chapter 6, figure (6-5), (6-6)), with the desired load, the average output link rate is simply determined. The "Load" for our simulation is set to be 80%.

Another important parameter in our simulation is the simulation time, namely, TStop time. From a statistical point of view, we wish to run the simulation as long as possible subject to the 25% file limitation discussed above. However, we estimate that it would take about one week to finish the simulation even if the machines were under low load. Because of the limited resources and time, we reduced simulation time to be 500 simulation time units (500 seconds). Thus each source covers about 7.6% of the file. These simulations required about two days to finish. We may not obtain accurate statistics with this amount of samples, but we can expect reasonable results for comparing the system with video trace data to the system with simulated *ARIMA(0,0.415,0)*. Further, we can learn about the changing tendency of CLR, delay and variance versus buffer size.

7.2 Results and Discussions

Ten simulations were run in total, five for the video trace data and five for the simulated *ARIMA(0,0.415,0)*. They correspond to buffer size 100, 150, 200, 250 and 300 cells respectively. Simulation parameters are set to be the same for both cases except for different global seeds. Since TStop=500 time units is not a long time, results from different simulations with different global seed vary randomly. In order to be able to compare among different simulations, we set the global seed to be the same within each five simulations. Now, let us look at the cell loss history of both cases. Figure 7-2 shows the comparison of loss history of both cases.

Clearly, the history of the two cases are different. The trace data ((b)-1 and (a)-2. Loss history of *ARIMA(0,0.415,0)* process with buffer size=200 (b)-2. Loss history of video trace data with buffer size=200

Figure 7-2: Comparison of cell loss history of video "Star Wars" trace data and the simulated *ARIMA(0,0.415,0)* process

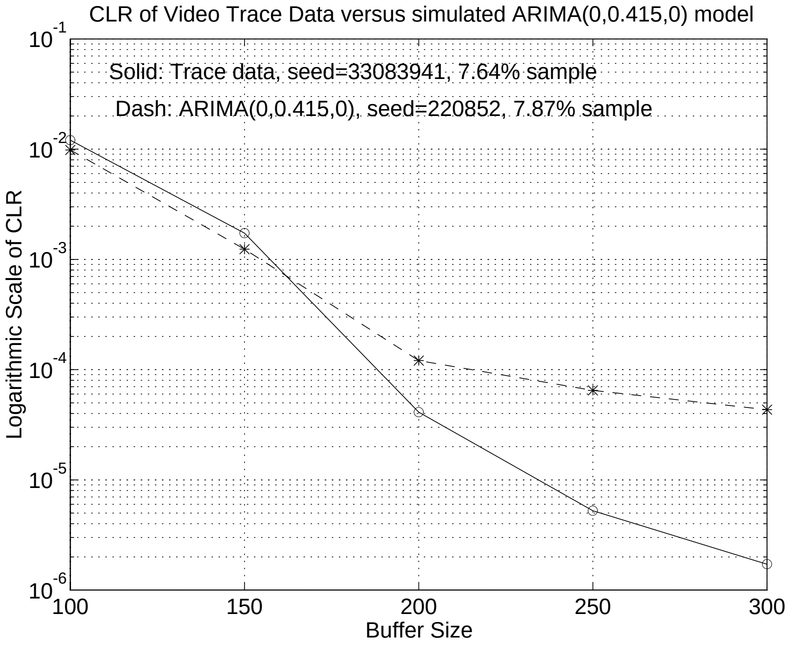


Figure 7-3: Cell Loss Ratio of video trace data and *ARIMA(0,0.415,0)* versus buffer size

(b)-2) is more burstier in the sense that loss events happen heavily in a short peorid of time, i.e., losses happen in a big burst followed by a blank gap where no loss event occurs. Loss events of the simulated *ARIMA(0,0.415,0)* process, on the other hand, occur in a less bursty manner, but more continuously across a large time duration. As the buffer size increases, losses tend to appear in a larger and more isolated chunks.

This is consistent with the correlation functions that is shown in figure 6-3 and the rate processes shown in figure 6-5, where we can see that the simulated *ARIMA(0,0.415,0)* process is smoother than the trace data. Therefore its data structure is more correlated throughout the whole file, while the trace data is more correlated in a piecewise manner.

Figure 7-3 shows the comparison of cell loss ratio (CLR) versus the buffer size for two cases in a logarithmic scale. From our discussion in the previous section, if cell decreases exponentially fast with the increase of buffer size, we should

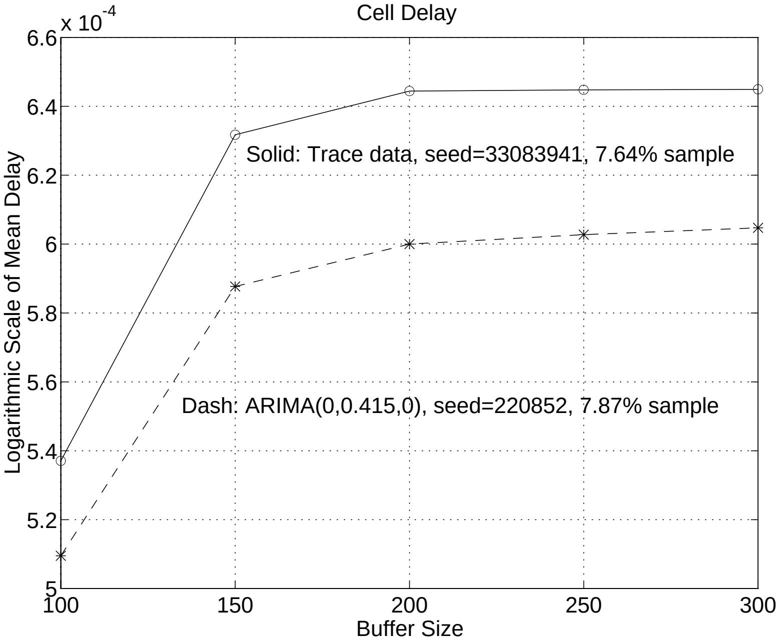


Figure 7-4: Delay of video trace data and *ARIMA(0,0.415,0)* versus buffer size

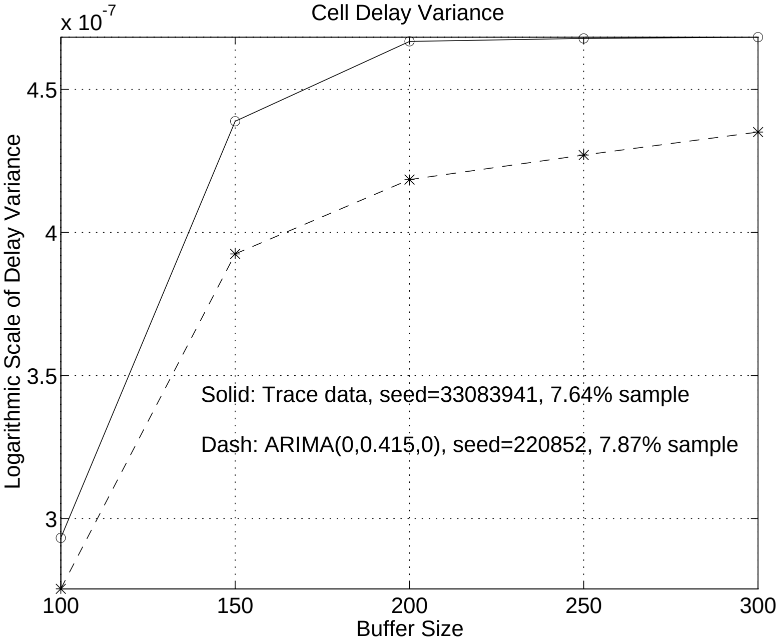


Figure 7-5: CDV of video trace data and *ARIMA(0,0.415,0)* versus buf fer size

see a linear curve in a logarithmic scale. Obviously, both curves are nonlinear, which means that CLR decreases more slowly than exponential with increasing buffer size as expected. This demonstrates that the CLR behavior of self-similar processes is indeed different from that of the conventional models.

Figure 7-4 and 7-5 show the delay and the delay variance for both cases. Clearly, both cell delay and its variance increase with increasing buffer size. From figure 7-4, it is not obvious that delay with increasing buffer size will not reach a limit. On the hand, the delay varince in figure 7-5 seems to increase with increasing buffer size. However, our simulation is relative short time and also the maximum buffer size is only 300. If we run longer time and get more points for the simulation, we may expect to see a different and more accurate results.

Again, the difference in the trace data and the simulated data is the result of imperfect modeling. In particular, the peak rate of the *ARIMA(0,0.415,0)* does not match that of the video trace data.

Chapter 8

Conclusions and Future Work

This thesis has made the following contributions.

(1) It presented a systematic way of modeling the most recently discovered self-similar/long-range dependent telecommunication traffic using the *ARIMA(p,d,q)* self-similar process. The approach includes modeling of *ARIMA(0,d,0)* as the first step and the modeling of *ARIMA(p,d,q)* process on top of that. This method allows simulated self-similar or long-range dependent traffic to be generated in a workstation based environment with a reasonable time duration. Therefore simulated sample data can be easily available for performance study.

(2) Basic understanding of the characteristics of long-range dependence was provided through the process of modeling. In particular, since long-range dependence in principle differs greatly from short-range dependence, their statistics and measurement depend greatly upon the amount of data being observed.

(3) Performance simulation of video trace data and the corresponding simulated *ARIMA(0,d,0)* process provided evidence of their difference from convention models in terms of cell loss versus buffer size, cell delay and variance versus buffer size.

The modeling method in this paper assumes that trace data can be modeled directly by *ARIMA(p,d,q)* process. However, for real traffic this assumption will not hold exactly in most cases. We can only expect that real traffic possesses long-range dependence, namely, very persistent correlations, but they may not be typical *ARIMA* process. The best example we have seen is the video Starwar's trace data in bits/frame, where the correlation structure appears to be irregular at many places.

Much work can be done in order to get a model that matches better than just the long-range dependent feature. Maximum likehood method can provide general purpose of modeling when specific feature is well defined. Filtering the noisy part of the trace data may help us focus on long-range dependent characteristics and get a clearer idea about the network effects contributed only by the long-range dependency. More attention should be given to the study of long-range dependent phenomenon in the performance of high speed network when good models are available. Simulation can provide essential information on that. Protocols and engineering of future high speed networks that provide service to self-similar traffic can be significantly affected by the knowledge obtained from simulations.

As a conclusion, Long-range dependence in telecommunication traffic is not a unique phenomenon. Long-range dependence is gaining more and more attention in statistics, since it has be observed in many areas and it has shown significant effect on statistical inference. Due to the vast number of examples from hydrology and geophysics, long-range dependence is recognized by most hydrologists and geophysicists to be the rule rather than the exception [2]. Whether it is also true in telecommunication traffic or not still need to be seen. Intuitively we believe that their different effect on network performance than our conventional idea is not negligible and yet to be studied and understood thoroughly.

Bibliography

[1] Yong-Qing Lu, David W. Petr, Victor S. Frost, "Survey of Source Modeling Techniques for ATM Networks," *TISL Technical Report TISL-10230-1*, The University of Kansas, September 1993

[2] Jan Beran, "Statistical Methods for Data with Long-Range Dependence," *Statistical Science*, 1992, vol.7, No.4, pp.404-427

[3] A. Baiocchi, N. Blefari Melazzi, M. Listanti, A. Roveri, R. Winkler, "Modeling Issues on an ATM Multiplexer Within a Bursty Traffic Environment", *INFOCOM'91*, vol.1, pp.2c.2.1.

[4] San-qi Li, Hong-Dah Sheng "Discrete Queuing Analysis of Multi-Media Traffic with Diversity of Correlation and Burstiness Properties", *INFOCOM'91*, vol.1, pp.4c.1.1.

[5] Harry Heffes, David M. Lucantoni, "Markov Modulated Characterization of Packetized and Data Traffic and Related Statistical Multiplexer Performance", *IEEE JSAC* vol.sac-4 No.6, Sept 1986, pp.856

[6] D. Anick, D. Mitra and M.M. Sondhi, "Stochastic theory of a data handling system with multiple resources," *Bell Syst. Tech. J.* vol.61, no.*, pp.1871-1894, 1982

[7] Reto Gruenenfelder, John P. Cosmas, Sam Manthorpe and Augustine Odinma-Okafor, "Characterzation of Video Codecs as Autoregressive Mov-

ing Average Processes and Related Queuing System Performance", *IEEE JSAC*, vol.9, No.3, April 1991

[8] G. Ramamurthy and B. Sengupta, "Modeling and Analysis of A Variable Bit Rate Video Multiplexer", *7th ITC Seminar*, Session 8, Oct 1990

[9] A. Baiocchi, N. Blefari Melazzi, M. Listanti, A. Roveri, R. Winkler, "Modeling Issues on an ATM Multiplexer Within a Bursty Traffic Environment", *INFOCOM'91*, vol.1, pp.2c.2.1.

[10] Benjamin Melamed, B. Sengupta,"TES Modeling of Video Traffic," IEICE Trans. Comm un., Vol. E75-b, No.12, December 1992

[11] A. Erramilli and R.P. Singh, "The Application of Deterministic Chaotic Maps to Characterize Traffic in Broadband Packet Networks," *7th ITC Seminar*, Session 8, Oct 1990

[12] Hong-Dah Sheng, San-qi, Li, "Second order effect of binary sources on characteristics of queue and loss rate", *IEEE INFOCOM'93*, pp 18-27

[13] San-qi Li, Chaia-Lin Hwang, "Queue Response to Input Correlation Functions: Discrete Spectral Analysis", *INFOCOM'92*, vol.3, pp.0382.

[14] Will E. Leland, Murad S. Taqqu, Walter Willinger, and Daniel V. Wilson, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)," to appear in *IEEE Trans on Networking*, 1994

[15] Will E. Leland, Murad S. Taqqu, Walter Willinger and Daniel V. Wilson, "Ethernet Traffic is Self-Similar: Stochastic Modeling of Packet Traffic Data,"

[16] Will E. Leland, Murad S. Taqqu, Walter Willinger, Daniel V. Wilson, "Self-Similarity in High-Speed Packet Traffic: Analysis and Modeling of Ethernet Traffic Measurements" to appear on *Statistical Science* 1994

[17] Jan Beran, Robert Sherman, Murad S. Taqqu and Walter Willinger, "Variable-Bit-Rate Video Traffic and Long-Range Dependence"

[18] Will E. Leland, Daciel V. Wilson, "High Time-Resolution Measurement and Analysis of LAN Traffic: Implication for LAN Interconnection," *INFOCOM'91* vol.3, pp.11d.3.1.

[19] Henry J. Fowler and Will E. Leland, " Local Area Network Traffic Characteristics, with Implications for Broadband Network Congestion Management", *IEEE JSAC*, vol.9, no.7, Sept 1991

[20] A. Baiocchi, N. Blefari Melazzi, M. Listanti, A. Roveri, R. Winkler, "Modeling Issues on an ATM Multiplexer Within a Bursty Traffic Environment", *INFOCOM'91*, vol.1, pp.2c.2.1.

[21] J. R. M. Hosing, "Fractional Differencing," *Biometrika*, vol. 68, No.1, pp.165-176 1981

[22] J. R. M. Hosking, "Modeling Persistence in Hydrological Time Series Using Fractional Differencing," *Water Resources Research*, vol. 20, No. 12, pp. 1898-1908, December 1984

[23] David W. Petr, Victor S. Frost, Ann Demirtjis, Cameron Braun, "Evaluation of Broadband Networking Technologies: Phase II Report," *TISL Technical Report TISL-9750-4*, The University of Kansas, July, 1993

[24] P.R. Kumar, Pravin Varaiya, "Stochastic Systems: Estimation, Identification and Adaptive Control," Prentice Hall Inc., Englewood Cliffs, New Jersey