MODELLING OF SELF-SIMILAR TELETRAFFIC FOR SIMULATION

A thesis submitted in partial fulfilment

of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

University of Canterbury

by

Hae-Duck Joshua Jeong

University of Canterbury

July 2002

To my God who gave me all the things I needed for this

Contents

List of Figures

List of Tables

MODELLING OF SELF-SIMILAR TELETRAFFIC FOR SIMULATION

Hae-Duck Joshua Jeong
Department of Computer Science
University of Canterbury
Christchurch, New Zealand

ABSTRACT

Recent studies of real teletraffic data in modern computer networks have shown that teletraffic exhibits *self-similar* (or *fractal*) properties over a wide range of time scales. The properties of self-similar teletraffic are very different from the traditional models of teletraffic based on Poisson, Markov-modulated Poisson, and related processes. The use of traditional models in networks characterised by self-similar processes can lead to incorrect conclusions about the performance of analysed networks. These include serious over-estimations of the performance of computer networks, insufficient allocation of communication and data processing resources, and difficulties ensuring the quality of service expected by network users. Thus, full understanding of the self-similar nature in teletraffic is an important issue.

Due to the growing complexity of modern telecommunication networks, simulation has become the only feasible paradigm for their performance evaluation. In this thesis, we make some contributions to discrete-event simulation of networks with strongly-dependent, self-similar teletraffic.

First, we have evaluated the most commonly used methods for estimating the self-similarity parameter H using appropriately long sequences of data. After assessing properties of available H estimators, we identified the most efficient estimators for practical studies of self-similarity.

Next, the generation of arbitrarily long sequences of pseudo-random numbers possessing specific stochastic properties was considered. Various generators of pseudo-random self-similar sequences have been proposed. They differ in computational complexity and accuracy of the self-similar sequences they generate. In this thesis, we propose two new generators of self-similar teletraffic: (i) a generator based on Fractional Gaussian Noise and Daubechies Wavelets (FGN-DW), that is one of the fastest and the most accurate gener-

ators so far proposed; and (ii) a generator based on the Successive Random Addition (SRA) algorithm. Our comparative study of sequential and fixed-length self-similar pseudo-random teletraffic generators showed that the FFT, FGN-DW and SRP-FGN generators are the most efficient, both in the sense of accuracy and speed.

To conduct simulation studies of telecommunication networks, self-similar processes often need to be transformed into suitable self-similar processes with arbitrary marginal distributions. Thus, the next problem addressed was how well the self-similarity and autocorrelation function of an original self-similar process are preserved when the self-similar sequences are converted into suitable self-similar processes with arbitrary marginal distributions. We also show how pseudo-random self-similar sequences can be applied to produce a model of teletraffic associated with the transmission of VBR JPEG/MPEG video. A combined gamma/Pareto model based on the application of the FGN-DW generator was used to synthesise VBR JPEG/MPEG video traffic.

Finally, effects of self-similarity on the behaviour of queueing systems have been investigated. Using $M/M/1/\infty$ as a reference queueing system with no long-range dependence, we have investigated how self-similarity and long-range dependence in arrival processes affect the length of sequential simulations being executed for obtaining steady-state results with the required level of statistical error. Our results show that the finite buffer overflow probability of a queueing system with self-similar input is much greater than the equivalent queueing system with Poisson or a short-range dependent input process, and that the overflow probability increases as the self-similarity parameter approaches one.

Chapter 1

INTRODUCTION

The growth of broadband networks and the Internet has been exponential in recent years. These high-speed communication networks have had a tremendous impact on our civilisation. High-speed communication networks offer a range of multimedia applications, such as audio, video and computer data, which differ significantly in their traffic characteristics and performance requirements [100], [133], [150]. Current Internet protocol (IP)-based communication networks can barely meet current demands. The goal now for telecommunication developers is to develop a universal high-speed communication network platform capable of carrying diverse traffic and supporting diverse levels of quality of service (QoS), building on the strengths of the QoS record of asynchronous transfer mode (ATM) [34]. However, many different technologies are involved in the successful development of new networks. Some of the development difficulties relate to teletraffic engineering, in particular, the modelling of teletraffic for simulation studies of multimedia communication networks.

Recent studies of real teletraffic data in modern computer networks have shown that teletraffic exhibits *self-similar* (or *fractal*) properties over a wide range of time scales [89], [172]. The properties of self-similar teletraffic are very different from properties of traditional models based on Poisson, Markov-modulated Poisson, and related processes. The use of traditional models in networks characterised by self-similar processes can lead to incorrect conclusions about the performance of analysed networks. The use of traditional models leads to over-estimation of the performance quality of computer networks, insufficient allocation of communication and data processing resources,

and difficulties in ensuring the quality of service expected by network users. Thus, full understanding of the self-similar nature in teletraffic is an important issue.

1.1 Self-Similar Nature of Real-World Teletraffic

Self-similar processes are relevant not only in telecommunication networks, but also in such areas of scientific activity as climatology, economics, environmental sciences, geology, geophysics, hydrology, computer science and computer engineering. In networks, they provide good models of packet traffic, for example, in local area networks (LANs) [89], [172], and video traffic [8], [10].

High-speed computer networks should be designed to carry a range of telecommunication services, some requiring very special treatment. Thus, understanding the self-similar nature of network traffic, identifying its characteristics, and developing appropriate traffic models are crucial to teletraffic engineering and the performance evaluation of networks with self-similar teletraffic [89], [111], [118].

Many analytical studies have shown that self-similar network traffic can have a detrimental impact on network performance, including amplified queueing delays and packet loss rates in broadband wide area networks [117], [139]. While Poisson and simple Markov-based models have been acceptable, for example, for remote-login and file-transfer [54], real packet sequences do not obey Poisson models in most cases [89], [125], [147]. Paxson and Floyd [125] found that wide-area network traffic consists of more bursts than Poisson models predict over many time scales. This difference has implications for congestion control mechanisms and performance. Park et al. [118] observed the effect of congestion control on network performance when networks were subject to highly self-similar traffic conditions. Crovella and Bestavros [21] found evidence and possible causes of self-similarity in World Wide Web (WWW) traffic, such as WWW document file size data. Heyman et al. [59] discovered that long-range dependence can significantly affect data cell loss rates when both the Hurst parameter and traffic intensity are high.

The self-similar nature of teletraffic in modern communication networks is unquestionable. The fact that many network performance evaluation studies were conducted using Markov-based traffic models has produced growing concerns about the potential impact of their inaccurate performance predictions, due to the essential difference between results obtained for self-similar and Markovian models.

Motivated by such concerns, we scrutinised the following areas to investigate the consequences of stochastic self-similarity on network performance:

- *Traffic models:* New, accurate self-similar traffic models are necessary to reveal the dynamics of individual streams of data and how they give rise to the self-similar characteristics observed in real-world situations. Such models must be analytically tractable or algorithmic to enable simulation studies. In the latter case, one should be able to generate fast synthetic streams of data with desired self-similar properties. For self-similar models to be practical, they must be able to capture actual properties of traffic in an Ethernet LAN, or in a WAN carrying VBR video traffic, for example.

- *Exhaustive performance evaluation studies of networks with self-similar teletraffic:* The performance of communication networks with the self-similar characteristics of teletraffic must be studied to determine the consequences of self-similarity of data streams on queueing performance.

1.2 Structure of this Thesis

Mathematical properties of stochastic self-similar processes are summarised in Chapter 2. A critical review of Hurst parameter estimation techniques is given in Chapter 3. Generators of pseudo-random self-similar data streams are surveyed in Chapter 4. A new generator of self-similar teletraffic is also proposed and assessed. Chapter 4 describes a comparative study of selected self-similar pseudo-random teletraffic generators. In Chapter 5, we investigate the generation of self-similar processes with arbitrary marginal distributions. The generation of a specific self-similar data stream (self-similar VBR video) is discussed in Chapter 6, and steady-state simulation studies of self-similar ef-

fects in queueing systems are reported in Chapter 7. Conclusions are presented in Chapter 8. A detailed summary of each chapter follows.

Chapter 2: Properties of Self-Similar Teletraffic explains why self-similarity is so important in computer systems and telecommunication networks, and shows where self-similar behaviour can be observed. The definitions of deterministic and stochastic self-similarity are given, and their relevant statistics are described. Then, important properties of stochastic self-similar processes are presented, including three definitions of long-range dependence, the slowly decaying variance of the sample mean, the Hurst effect, the power-law behaviour of the spectral density at the origin, and the role played by heavy-tailed distributions. The relationships among self-similarity, time scales and power-law behaviour are explained. We show that the relevant statistics of self-similar network traffic exhibit power-law behaviour over a wide range of time scales.

Chapter 3: Critical Review of Hurst Parameter Estimation Techniques exhaustively evaluates the most commonly used methods for estimating the self-similarity parameter H. The Durbin-Levinson algorithm is used to generate exact self-similar FGN (Fractional Gaussian Noise) sequences. Mean values of estimated H and other statistical tests on estimated H values were used to statistically prove which of the Hurst parameter estimators is more accurate than the others. The results of our analysis have pointed to the wavelet-based H estimator and Whittle's MLE as the most efficient estimators of H.

Chapter 4: Algorithmic Generators of Self-Similar Teletraffic presents a detailed survey of self-similar generators proposed for generating sequential and fixed-length self-similar pseudo-random sequences. Both sequential and fixed-length sequence generators can be used for sequential and non-sequential simulations. If a fixed-length sequence generator is used for sequential simulation, however, then a sufficiently long self-similar sequence of numbers must be generated before the simulation begins. We considered the following sequential and fixed-length self-similar generators:

A. Six sequential generators based on: fractal-binomial-noise-driven Poisson processes (FBNDP), superposition of fractal renewal processes (SFRP), output processes of $M/G/\infty$ queueing systems (MGIP), Pareto-modulated

Poisson processes (PMPP), spatial renewal processes and fractional Gaussian noise (SRP-FGN), and superposition of autoregressive processes (SAP).

B. Five fixed-length generators based on: fast Fourier transform (FFT), fractional-autoregressive integrated moving average (F-ARIMA), random midpoint displacement (RMD), successive random addition (SRA), and fractional Gaussian noise and Daubechies wavelets (FGN-DW).

We evaluate and compare the operational properties of the fixed-length and sequential generators of self-similar pseudo-random sequences considered in the same chapter. The statistical accuracy and time required to produce long sequences are discussed theoretically and studied experimentally. The evaluation of the generators concentrated on two aspects: (i) how accurately self-similar processes can be generated (assuming a given mean, variance and self-similarity parameter H), and (ii) how quickly the generators can generate long self-similar sequences.

Six sequential and five fixed-length generators of self-similar sequences are compared and analysed using the most efficient estimators of H found in Chapter 3: the wavelet-based H estimator and Whittle's MLE.

Chapter 5: Generation of a Self-Similar Process with an Arbitrary Marginal Distribution investigates the transformation of self-similar processes into suitable self-similar processes with arbitrary marginal distributions, based on the method of inverse transformation of cumulative probability functions. We show the degree to which the self-similarity and autocorrelation function of the original self-similar process are preserved when the self-similar sequences are converted into self-similar processes with arbitrary marginal distributions.

Chapter 6: Modelling and Generation of Self-Similar VBR Video Traffic shows that pseudo-random self-similar sequences representing VBR video streams, compressed under JPEG/ MPEG standards, might be required to simulate ATM networks. We show how such pseudo-random VBR video streams can be obtained by appropriate transformation of data generated by the FGN-DW generator and the combined gamma/Pareto model of teletraffic. JPEG, MPEG-1 and MPEG-2 compression standards are considered. Results of a steady-state simulation of a single buffer fed by these

synthetic video streams are used to show that simulations based on synthetic streams of teletraffic can give the same qualitative and quantitative results as simulations based on real traces of VBR video.

Chapter 7: Steady-State Simulation of Self-Similar and Queueing Processes considers basic features of steady-state simulation studies of self-similar queueing processes. We investigate run lengths of sequential steady-state simulations of queueing models with self-similar arrivals to estimate mean response times, under two different methods of output data analysis. The results are theoretically and experimentally compared with simulation run lengths of the same queueing models fed by Poisson processes. We also demonstrate how input self-similarity can affect the overflow probability of finite buffers.

Chapter 8: Conclusions summarise the main contributions and suggest the direction of future research work.

1.3 Contributions of this Thesis

The following is a list of the main contributions of this thesis to the field of modelling of self-similar teletraffic for simulation. Contributions are noted in order of appearance in this thesis, with corresponding Chapter numbers. We show relevant publications in parentheses, and list them in Section 1.4.

1. Comparative analysis and evaluation of estimators of the self-similarity parameter H. The estimators considered include the wavelet-based H estimator and Whittle's Maximum Likelihood Estimator (MLE), and estimators obtained from periodogram analysis, R/S-statistic analysis, variance-time analysis and IDC(t) analysis. Their properties were assessed on the basis of mean estimates and other statistical tests to statistically prove which of the estimators should be recommended in practice (Chapter 3, [JMP01a]).

2. Proposal of two new fixed-length pseudo-random generators of self-similar teletraffic (SRA and FGN-DW generators) (Chapter 4, [JMP99a], [JMP99b], [JMP99d]).

3. Comparative analysis of practical pseudo-random generators of self-similar teletraffic. Both sequential and fixed-length generators have been considered (Chapter 4, [JMP99a], [JMP99b], [JMP99d]).

4. Transformation of marginal distributions of self-similar processes: a study of necessary conditions for preservation of the Hurst parameter and ACFs in resulted processes. Our studies allow us to formulate the hypothesis that the transformation preserves both the Hurst parameter and ACF of the input self-similar process if the variance of the output marginal distribution of the transform process is finite (Chapter 5, [JMP99c], [JMP01b]).

5. Influence of compression algorithms on correlation structure of compressed teletraffic: generalisation of findings of Garrett and Willinger, showing that video compression algorithms (MPEG-1 and MPEG-2) lead to self-similar processes. We considered outcomes of MPEG-1 and MPEG-2 in addition to previously studied outcomes of JPEG (Garrett and Willinger), to show that the results (of Garrett and Willinger) do not depend on the compression algorithms (Chapter 6).

6. We have analysed run-lengths of sequential steady-state simulations of queueing models with self-similar arrivals, under two different methods of simulation output data analysis (spectral analysis and batch means). Steady-state simulations of queueing processes with self-similar input processes were conducted for studying the influence of the degree of self-similarity on run-lengths of sequential simulation and on the overflow probability of finite buffers (Chapter 7, [JMP00], [PJL02]).

1.4 Publications by the Author Related to this Thesis

The following papers were prepared during the PhD study:

[JMP99a] Jeong, H.-D.J., McNickle, D., and Pawlikowski, K. A Search for Computationally Efficient Generators of Synthetic Self-Similar Teletraffic. In

Proceedings of the Twenty Second Australasian Computer Science Conference (Auckland, New Zealand, 1999), vol. 21, pp. 75-86.

[JMP99b] Jeong, H.-D.J., McNickle, D., and Pawlikowski, K. A Comparative Study of Three Self-Similar Teletraffic Generators. In *Proceedings of 13th European Simulation Multiconference (ESM'99)* (Warsaw, Poland, 1999), pp. 356-362.

[JMP99c] Jeong, H.-D.J., McNickle, D., and Pawlikowski, K. Generation of Self-Similar Time Series for Simulation Studies of Telecommunication Networks. In *Proceedings of the First Western Pacific and Third Australia-Japan Workshop on Stochastic Models in Engineering, Technology and Management* (Christchurch, New Zealand, 1999), pp. 221-230.

[JMP99d] Jeong, H.-D.J., McNickle, D., and Pawlikowski, K. Fast Self-Similar Teletraffic Generation Based on FGN and Wavelets. In *Proceedings of IEEE International Conference on Networks (ICON'99)* (Brisbane, Australia, 1999), pp. 75-82.

[JMP00] Jeong, H.-D.J., McNickle, D., and Pawlikowski, K. Some Problems in Sequential Simulation with Self-Similar Processes. In *Proceedings of the 2000 Summer Computer Simulation Conference* (Vancouver, British Columbia, Canada, 2000), pp. 175-180.

[JMP01a] Jeong, H.-D.J., McNickle, D., and Pawlikowski, K. Hurst Parameter Estimation Techniques: A Critical Review. In *Proceedings of Operational Research Society of New Zealand (ORSNZ) Conference Twenty Naught One* (Christchurch, New Zealand, 2001), pp. 165-172.

[PJL02] Pawlikowski, K., Jeong, H.-D.J., and Lee, J.R. On Credibility of Simulation Studies of Telecommunication Networks. *IEEE Communications Magazine* 40, 1 (2002), pp. 132-139.

[JMP01b] Jeong, H.-D.J., McNickle, D., and Pawlikowski, K. Generation of Self-Similar Processes for Simulation Studies of Telecommunication Networks. *Mathematical and Computer Modelling*, (In press, 2003).

Chapter 2

PROPERTIES OF SELF-SIMILAR TELETRAFFIC: AN OVERVIEW

2.1 Mathematical Definition of Self-Similarity

Self-similarity can be classified into two types: deterministic and stochastic. In the first type, deterministic self-similarity, a mathematical object is said to be self-similar (or fractal) if it can be decomposed into smaller copies of itself. That is, self-similarity is a property in which the structure of the whole is contained in its parts. An example of a deterministic approximately self-similar object is given in Figure 2.1. The term *self-similar* (or *fractal*) was coined by Mandelbrot [104]. He and his co-workers brought self-similar processes to the attention of statisticians, mainly through applications in such areas as hydrology and geophysics [104], [105].

In this thesis we focus on the latter, i.e., stochastic self-similarity. In this case, probabilistic properties of self-similar processes remain unchanged or invariant when the process is viewed at varying time scales. This is in contrast to Poisson processes which lose their burstiness and flatten out when time scales are changed. However, such a time series exhibits burstiness over a wide range

Figure 2.1: *Dryopteris Erythrosora, the Autumn Fern* [47]: a simple example of a deterministic self-similar object.

of time scales. Self-similarity can statistically describe teletraffic that is bursty on many time scales.

One can distinguish two types of stochastic self-similarity. A continuous-time stochastic process $\mathbf{Y}_t$ is strictly *self-similar* with a self-similarity parameter $H(\frac{1}{2} < H < 1)$, if $\mathbf{Y}_{ct}$ and $c^H\mathbf{Y}_t$ (the rescaled process with time scale ct) have identical finite-dimensional probability for any positive time stretching factor c [9], [120], [155]. This means that, for any sequence of time points $t_1, t_2, \ldots, t_n$, and for any $c > 0$,

$$\{Y_{ct_1}, Y_{ct_2}, \ldots, Y_{ct_n}\} \stackrel{d}{=} \{c^H Y_{t_1}, c^H Y_{t_2}, \ldots, c^H Y_{t_n}\},$$

where $\stackrel{d}{=}$ denotes equivalence in distribution. This definition of the strictly self-similarity is in a sense of probability distribution (or narrow sense), quite different from that of the second-order self-similar process (or self-similar process in a broad sense). Self-similarity in the broad sense is observed at the mean, variance and autocorrelation level, whereas self-similarity in the narrow sense is observed at the probability distribution level, see Figure 2.2.

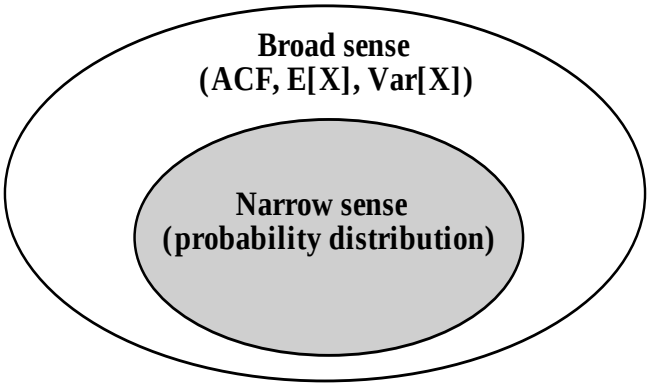


Figure 2.2: Block diagram of the definitions of the strictly (in a narrow sense) and second-order (or in a broad sense) self-similar stochastic processes.

When the weakly continuous-time self-similar process $\mathbf{Y}_t$ has stationary increments, i.e., the finite-dimensional probability distributions of $\mathbf{Y}_{t_0+t} - \mathbf{Y}_{t_0}$ do not depend on t_0, we can construct a stationary incremental process $\mathbf{X} = \{X_i = Y_{i+1} - Y_i : i = 0, 1, 2, \cdots\}$. Namely, in the discrete-time case, let $\mathbf{X}$ be a (discrete-time) stationary incremental process with mean $\mu = E[\mathbf{X}]$, variance $\sigma^2 = E[(\mathbf{X} - \mu)^2]$, and (normalised) autocorrelation function (ACF) $\{\rho_k\}, k = 0, 1, 2, \cdots$, where

$$\rho_k = \frac{E[(X_i - \mu)(X_{i+k} - \mu)]}{\sigma^2}. \tag{2.1}$$

$\mathbf{X}$ is strictly stationary if $\{X_{i_1}, X_{i_2}, \ldots, X_{i_n}\}$ and $\{X_{i_1+k}, X_{i_2+k}, \ldots, X_{i_n+k}\}$ possess the same joint distribution. However, we limit our attention to processes with a weaker form of stationarity, i.e., second-order stationarity (or weak, broad, or wide sense stationarity). Let $\mathbf{X}^{(m)} = \{X_1^{(m)}, X_2^{(m)}, \cdots\}$, $m = 1, 2, 3, \cdots$, be a sequence of batch means, that is,

$$X_i^{(m)} = \frac{1}{m}(X_{im-m+1} + \cdots + X_{im}), i \geq 1, \tag{2.2}$$

and let $\{\rho_k^{(m)}\}$ denote the ACF of $\mathbf{X}^{(m)}$. The process $\mathbf{X}$ is called *exactly second-order self-similar* with $0.5 < H < 1$, if for all $m \geq 1$,

$$\rho_k^{(m)} = \rho_k, k \geq 0. \tag{2.3}$$

In other words, the process $\mathbf{X}$ and the aggregated processes $\mathbf{X}^{(m)}$, $m \geq 1$, have an identical correlation structure. The process $\mathbf{X}$ is *asymptotically second-*

order self-similar with $0.5 < H < 1$, if for all k large enough,

$$\rho_k^{(m)} \to \rho_k, \text{ as } m \to \infty. \tag{2.4}$$

In general, two features of stochastic processes can be considered: (i) *long-range dependence* (LRD) or *short-range dependence* (SRD), and (ii) *self-similarity* or *non-self-similarity* [173]. LRD involves the tail behaviour of the ACF of a stationary sequence, while self-similarity typically refers to the scaling behaviour of the finite dimensional distributions of continuous time or discrete time processes. However, as it will be discussed later, a relationship between the ACFs and self-similarity does exist. Namely, in the case of asymptotic second-order self-similarity, by the restriction $\frac{1}{2} < H < 1$ in the definition, self-similarity implies LRD, and vice versa. Thus, we use the terms long-range dependence and (exact or asymptotic) self-similar processes in an interchangeable fashion, because both refer to the tail behaviour of the autocorrelations and are essentially equivalent.

2.2 Properties of Long-Range Dependent Self-Similar Processes

The most striking feature of some second-order self-similar processes is that the ACFs of the aggregated processes do not degenerate as the non-overlapping batch size m increases to infinity. Such processes are known as long-range dependent. This is in contrast to traditional processes used in models of teletraffic, all of which have the property that the ACFs of their aggregated processes degenerate as the non-overlapping batch size m increases to infinity, i.e., $\rho_k^{(m)} \to 0$ or $\rho_k^{(m)} = 0(|k| > 0)$, for $m > 1$. The main properties of second-order LRD self-similar processes are described below; see [9], [17], and [89], for more detail.

Two equivalent definitions of long-range dependence (LRD) are as follows:

- The first definition of LRD is given in (2.5). The stationary process $\mathbf{X}$ is said to be a long-range dependent process if its ACF is non-summable

[17], i.e.,

$$\sum_{k=-\infty}^{\infty} \rho_k = \infty. \tag{2.5}$$

The details of how ACF decays with k are of interest because the behaviour of the tail of ACF completely determines its summability.

- Another definition of LRD is given by

$$\rho_k \sim L(t)k^{-(2-2H)}, \text{ as } k \to \infty, \tag{2.6}$$

where $\frac{1}{2} < H < 1$ and $L(\cdot)$ slowly varies at infinity, i.e.,

$$\lim_{t\to\infty} \frac{L(xt)}{L(t)} = 1,$$

for all $x > 0$; see [89]. The Hurst parameter H characterises the relation in (2.6), which specifies the form of the tail of the ACF.

One can show that, for $\frac{1}{2} < H < 1$,

$$\rho_k = \frac{1}{2}[(k+1)^{2H} - 2k^{2H} + (k-1)^{2H}], \tag{2.7}$$

see Figure 2.3 [9]. For $0 < H < \frac{1}{2}$, the process has SRD and the ACFs sum up to zero, namely:

$$\sum_{k=-\infty}^{\infty} \rho_k = 0. \tag{2.8}$$

For $H = \frac{1}{2}$, all ACFs at non-zero lags are zero and we deal with processes with IID Gaussian increments. For $\frac{1}{2} < H < 1$, a process has LRD and the ACFs decay to zero very slowly (i.e., Equation (2.5)). For $H = 1$, all autocorrelation coefficients are equal to one, no matter how far apart in time the sequences are. This case has no practical importance in the case of real teletraffic modelling. If we had $H > 1$, then ACFs would diverge to infinity. Namely, we would have

$$\rho_k = \begin{cases} 1 & \text{for } k = 0, \\ \frac{1}{2}k^{2H}g(k^{-1}) & \text{for } k > 0, \end{cases} \tag{2.9}$$

where

$$g(x) = (1+x)^{2H} - 2 + (1-x)^{2H}. \tag{2.10}$$

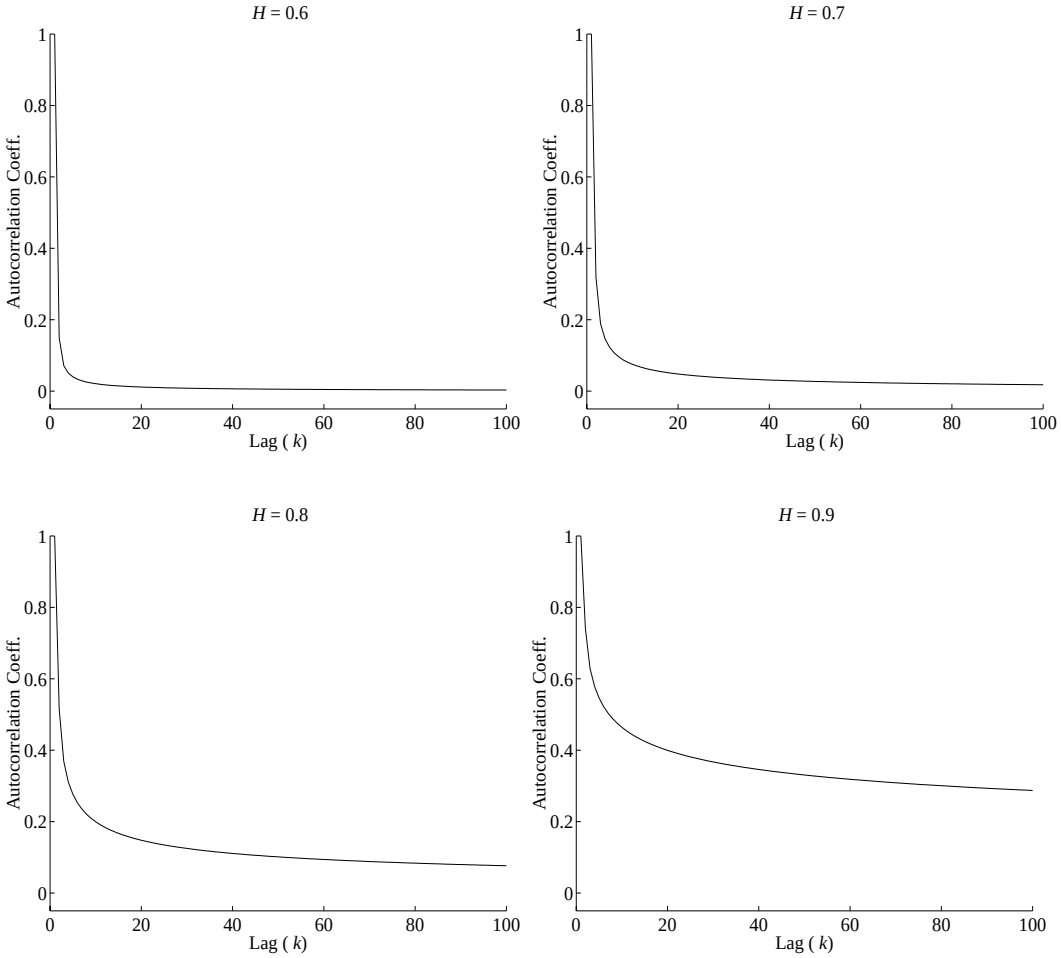


Figure 2.3: Autocorrelation function plots for $\rho_k = \frac{1}{2}[(k+1)^{2H} - 2k^{2H} + (k-1)^{2H}]$ in Equation (2.7) ($H = 0.6, 0.7, 0.8, 0.9$).

One can see that $g(x) \to \infty$ as $H > 1$; see [9] (p.52), [146]. If $0 < H < 1$ and $H \neq \frac{1}{2}$, then the first non-zero term in the Taylor expansion of $g(x)$ is equal to $2H(2H-1)x^2$. Therefore,

$$\rho_k/(H(2H-1)k^{2H-2}) \to 1, \quad \text{as} \quad k \to \infty. \tag{2.11}$$

This contradicts the fact that ρ_k must be between -1 and +1. Therefore, it can be concluded that if covariances exist and $\lim_{k\to\infty} \rho_k = 0$, then $0 < H < 1$. In some cases of teletraffic models, it was shown by statistical measurements that $\frac{1}{2} < H < 1$ [89].

We call $\mathbf{X}$ an *asymptotically* second-order process with *long memory*, *long-*

range dependence, or *strong dependence*, when it satisfies Equation (2.4), and we call it an *exactly* second-order process with LRD when it satisfies Equation (2.3). The asymptotically and exactly second-order processes are characterised by ACFs which decay hyperbolically. The process with LRD contrasts with processes with *short memory*, *short-range dependence*, or *weak dependence*. These processes have ACFs that decay exponentially.

2.2.1 Slowly Decaying Variance

The variances of the aggregated self-similar processes $\mathbf{X}^{(m)}, m \geq 1$, decrease more slowly than the reciprocal of the non-overlapping batch size m. From Equation (2.2), this property is given by

$$Var[\mathbf{X}^{(m)}] \rightarrow c_1 m^{-\beta_1},$$

as $m \rightarrow \infty$, where c_1 is a constant and $0 < \beta_1 < 1$.

For short-range dependent processes such as Poisson processes and related processes, $\beta_1 = 1$. Here X_i might represent the number of packets, cells, or bytes that have arrived at a telecommunication buffer during the i-th time interval of T_s seconds. Note that X_i is obtained by

$$X_i = N_i - N_{i-1} \tag{2.12}$$

when it is constructed from an underlying counting process $\mathbf{N}$. To illustrate this, let the process $\mathbf{X}$ defined in Equation (2.12) be constructed from a Poisson process. Then, for any independent random variables,

$$\begin{aligned} \mathrm{V}ar[\mathbf{X}^{(m)}] &= \mathrm{V}ar[\frac{1}{m}(X_1 + X_2 + \ldots + X_m)] \\ &= \frac{1}{m^2} \cdot m \cdot \mathrm{V}ar[X_1] \\ &= \frac{1}{m} \cdot \mathrm{V}ar[X_1]. \end{aligned}$$

In this case, $Var[\mathbf{X}^{(m)}]$ decays as m^{-1} for all $m = 1, 2, \ldots$. This property can be illustrated by the variance-time plot $log(Var[\mathbf{X}^{(m)}])$ against $log(m)$; see Section 3.4.5. If this plot forms a straight line with an absolute slope of less than one over a wide range of m, then we say the process $\mathbf{X}$ possesses the slowly decaying variance property.

2.2.2 Hurst Effect

Historically, the importance of self-similar processes lies in the fact that they provide an elegant explanation and interpretation of strong correlations in some empirical data. Namely, for a given sequence of random variables $\mathbf{X} = \{X_t\}_{t=1}^n = \{X_1, X_2, \cdots, X_n\}$, one can consider the so-called *rescaled adjusted range* $\frac{R(t,m)}{S(t,m)}$ (or R/S-statistic), with

$$\begin{aligned} R(t,m) &= \max_i[N_{t+i} - N_t - \frac{i}{m}(N_{t+m} - N_t), 0 \le i \le m] - \\ &\quad \min_i[N_{t+i} - N_t - \frac{i}{m}(N_{t+m} - N_t), 0 \le i \le m], \end{aligned} \tag{2.13}$$

where $1 \le t \le n$, m is the batch size and $N_t = \sum_{i=1}^t X_i$; and

$$S(t,m) = \sqrt{m^{-1} \sum_{i=t+1}^{t+m} (X_i - \bar{X}_{t,m})^2}, \tag{2.14}$$

where $\bar{X}_{t,m} = m^{-1} \sum_{i=t+1}^{t+m} X_i$.

Hurst found empirically that for many time series observed in nature, the expected value of $\frac{R(t,m)}{S(t,m)}$ asymptotically satisfies the power-law relation:

$$E[\frac{R(t,m)}{S(t,m)}] \to c_2 m^H, \text{ as } m \to \infty, \text{ with } \frac{1}{2} < H < 1,$$

where c_2 is a finite positive constant [9]. This empirical finding was in contradiction to previously known results for Markovian and related processes. For a stationary process with SRD, $E[\frac{R(t,m)}{S(t,m)}]$ behaves asymptotically like a constant times $m^{\frac{1}{2}}$. Therefore, for large values of m, the R/S-statistic plot is randomly scattered around a straight line with slope $\frac{1}{2}$. Hurst's finding that for the Nile River data, and for many other hydrological, geophysical, and climatological data, $\frac{R(t,m)}{S(t,m)}$ is randomly scattered around a straight line with slope $H > \frac{1}{2}$, is known as the *Hurst effect*, and H is known as the Hurst parameter (or self-similarity parameter). Mandelbrot and Wallis [105] showed that the Hurst effect can be modelled by FGN with the self-similarity parameter $\frac{1}{2} < H < 1$.

2.2.3 Spectral Density Obeys a Power-Law

In the frequency domain, an essentially equivalent definition of LRD for a process $\mathbf{X}$ with given spectral density

$$f(\lambda) = \frac{\sigma^2}{2\pi} \sum_{k=-\infty}^{\infty} \rho_k e^{ik\lambda},$$

is that in the case of LRD processes, this function is required to satisfy the following property:

$$f(\lambda) \sim c_{f_1} \lambda^{-\gamma}, \text{ as } \lambda \to 0, \tag{2.15}$$

where c_{f_1} is a positive constant and $0 < \gamma < 1, \gamma = 2H - 1 < 1$ [9]. Thus, LRD manifests itself in the spectral density that obeys a power-law in the vicinity of the origin. This implies that $f(0) = \sum_k \rho_k = \infty$. Thus, it requires a spectral density which tends to $+\infty$ as the frequency λ approaches 0. In contrast to LRD, SRD is characterised by a spectral density function $f(\lambda)$ which is positive and finite for $\lambda = 0$.

For a fractional Gaussian noise (FGN) process, the spectral density $f(\lambda, H)$ is given by

$$f(\lambda, H) = 2c_f(1 - cos(\lambda))\mathcal{B}(\lambda, H) \tag{2.16}$$

with $0 < H < 1$ and $-\pi \leq \lambda \leq \pi$, where

$$\begin{aligned} c_f &= \sigma^2(2\pi)^{-1} sin(\pi H)\Gamma(2H+1), \\ \mathcal{B}(\lambda, H) &= \sum_{k=-\infty}^{\infty} |2\pi k + \lambda|^{-2H-1}, \end{aligned} \tag{2.17}$$

and $\sigma^2 = \text{Var}[X_k]$ and $\Gamma(\cdot)$ is the gamma function; see [9].

The spectral density $f(\lambda, H)$ in Equation (2.16) obeys a power-law at the origin, i.e.,

$$f(\lambda, H) \to c_f \lambda^{1-2H}, \text{ as } \lambda \to 0, \tag{2.18}$$

where $\frac{1}{2} < H < 1$. We will use these formulae in Chapter 4.4.

2.3 Heavy-Tailed Distributions

There is an intimate relationship between heavy-tailed distributions and LRD processes. Let X be a random variable with a cumulative distribution function (CDF) $F(x) = Pr[X \leq x]$ and complementary CDF

$$\bar{F}(x) = 1 - F(x) = Pr[X > x].$$

A distribution of X is *heavy-tailed* if

$$Pr[X > x] \sim L(x)x^{-\alpha}, \quad x \to \infty, \tag{2.19}$$

where $0 < \alpha < 2$ is called the *tail index* (or *shape parameter* or coefficient of heavy-tailedness), and $L(\cdot)$ slowly varies at infinity. That is, the tail of the distribution asymptotically decays hyperbolically. This is in contrast to *light-tailed* distributions such as exponential and normal distributions which possess an exponentially decreasing tail [49], [50].

The simplest heavy-tailed distribution is the *Pareto* distribution. This exhibits power-law behaviour over its entire range and its probability density function (PDF) is

$$f(x) = \alpha b^{\alpha} x^{-(\alpha+1)}, \tag{2.20}$$

where α is the shape parameter, $\alpha > 0$, and b is the minimum allowed value of $x, 0 < b \leq x$. Its CDF is given by

$$F(x) = Pr[X \leq x] = 1 - \left(\frac{b}{x}\right)^{\alpha}. \tag{2.21}$$

If $\alpha \leq 2$, then the distribution has an infinite variance; if $\alpha \leq 1$, then the distribution has an infinite mean. Thus, as α decreases, an arbitrarily large portion of the PDF may be present in the tail of the distribution. In practical terms, a random variable that follows a heavy-tailed distribution can take on extremely large values with non-negligible probability. Figure 2.4 shows 10,000 observations synthetically generated from a Pareto distribution with $\alpha = 1.6$ and $b = 1$. This figure shows the characteristic, visually striking behaviour of heavy-tailed random variables. In such observations, most of the values are small ($97.48\% < 10$), but the greatest contribution to the sample mean or variance comes from the few large values ($2.52\% \geq 10$). Heavy-tailed

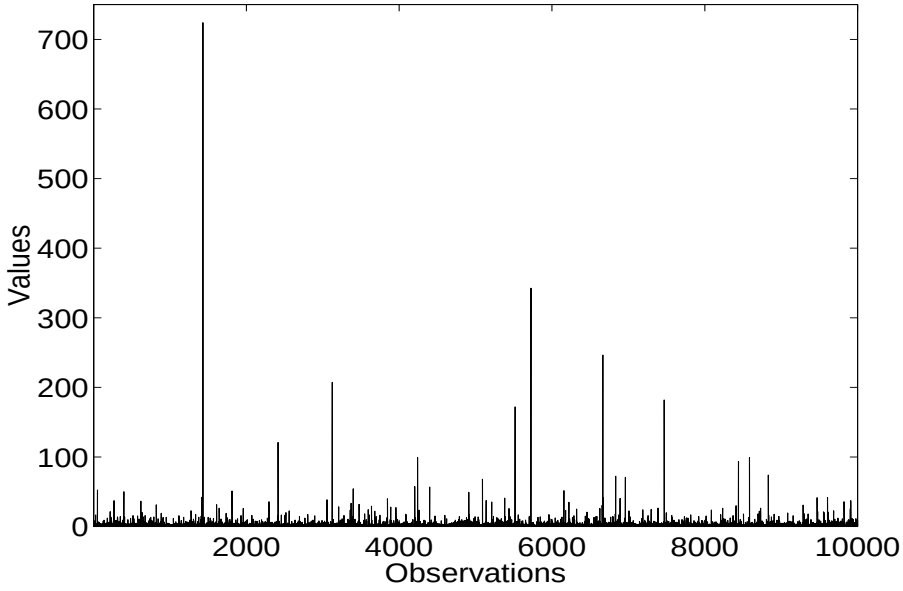


Figure 2.4: A sequence plot generated from a Pareto distribution with $\alpha = 1.6$ and $b = 1$.

distributions can be used to characterise probability distributions that describe teletraffic processes such as inter-arrival times and burst length.

Other examples of heavy-tailed distributions are:

- Lognormal distribution
  $F(x) = \Phi\left(\frac{\log x - \mu}{\sigma}\right)$, for $\sigma > 0$, where Φ is the standard normal distribution.
- Weibull distribution
  $F(x) = 1 - e^{-x^{\beta}}$, for $0 < \beta < 1$, see [80] for a more detailed discussion.

2.3.1 Slow Convergence to Steady-State in Simulations with Heavy-Tailed Input

Computer network designers and researchers are increasingly interested in employing heavy-tailed distributions in simulation studies of computer systems and telecommunication networks since recent evidence shows that some characteristics of these networks may be described well using heavy-tailed distribu-

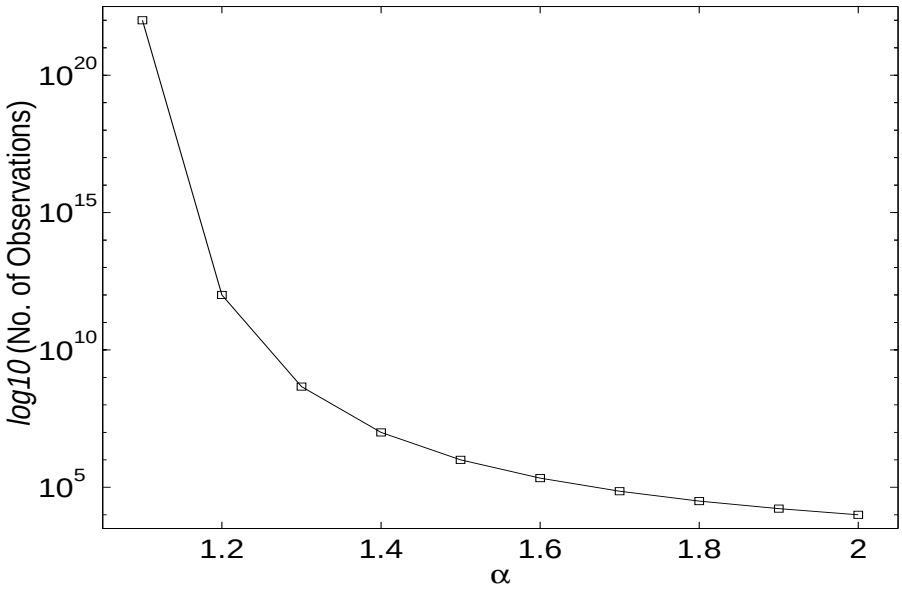


Figure 2.5: Number of theoretically required observations to achieve two digit accuracy in simulations with heavy-tailed input for $c_2 = 1$ and $\alpha = 1.1$ to 2.0 obtained from Equation (2.28).

tions [23]. However, properties of heavy-tailed distributions are not only very different from the conventional distributions described above, but also make simulation stability hard to achieve.

Erramilli et al. [28] analyse the impact of heavy-tailed service times in packet loss systems. Packet losses occur at much higher rates than the long term rates (e.g., ATM cell loss rates of the order of 10^{-10}), and correspondingly the impact on applications will be considerably more than that indicated by the long term rate. They also show that Pareto service time densities converge so slowly to the Erlang B result that one can significantly underestimate the true blocking probability in typical engineering intervals. Crovella and Lipsky [23] show that if α is close to 1, a huge number of observations are required to stop simulations (see Figure 2.5). They also show that such simulations can take a very long time to reach steady-state; and be much less stable at steady-state than typical for traditional systems.

Figure 2.5 shows that the number of required observations to obtain convergence in the sample mean significantly increases when α is close to 1. There

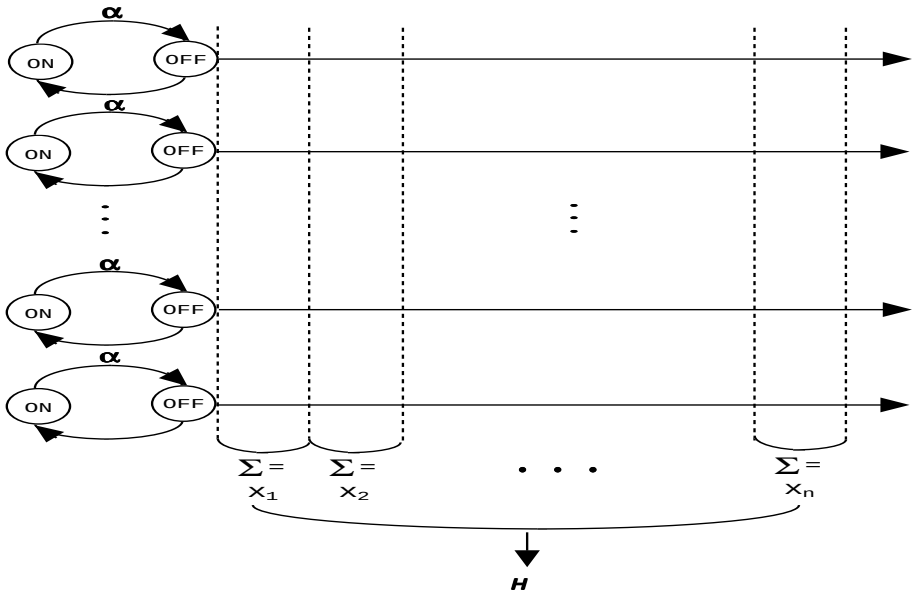


Figure 2.6: The superposition of ON/OFF processes with a heavy-tailed distribution converges to a self-similar count process with $H = (3 - \alpha)/2$.

is a relationship between the shape parameter α of heavy-tailed distributions and the self-similarity parameter H: $H = (3 - \alpha)/2$. The relationship with $H = (3 - \alpha)/2$ arises in the ON/OFF model whose ON- or OFF-periods have heavy-tailed distributions [159]. The range of the Hurst parameter (i.e., $\frac{1}{2} < H < 1$) is equivalent to the range of the thickness of the heavy-tailed distribution (i.e., $1 < \alpha < 2$)[89], [159]. The convergence of a superposition of sources converges to a self-similar process depending on the normalisation used [3]; see also Figure 2.6.

For $\alpha = 1.6$, 1.2 and 1.1 (i.e., for $H = 0.7$, 0.9 and 0.95, as $H = (3 - \alpha)/2$ characterises the *thickness* of the heavy-tailed distribution), the number of theoretically required observations to achieve two digit accuracy in Figure 2.5 is 215,443, 10^{12} and 10^{22}, respectively. Thus, it may be infeasible in any reasonable amount of time to observe a steady-state in a simulation with such a heavy-tailed distribution. Over any reasonable time scale, such a simulation would be always in a transient state.

The behaviour of the sample means of common statistics can be statistically analysed on the basis of observations obtained from simulations. We

consider the convergence properties of sums of random variables. If observations $x_1, x_2, \ldots, x_n$ are sequences of independent and identically distributed (IID) random variables $X_1, X_2, \ldots, X_n$, then the *central limit theorem* (CLT) states that the distribution of the variable $\bar{X}$ becomes closer to the normal distribution as the number of collected observations n increases [122]. Unfortunately, the CLT applies only to sums of random variables with a finite variance. Otherwise we need to use the limit theorems for heavy-tailed random variables first formulated by Lévy [23], [151].

When X_i are IID and drawn from some distribution $F(x)$ with mean μ and variance $\sigma^2 < \infty$, the usual CLT is defined by

$$Z \xrightarrow{d} N(0, \sigma^2), \tag{2.22}$$

where

$$\begin{aligned} Z &= n^{-1/2}(\bar{X} - \mu), \\ \bar{X} &= \frac{1}{n}\sum_{i=1}^{n} X_i \end{aligned} \tag{2.23}$$

and $N(0, \sigma^2)$ is a normal distribution [23].

However, when X_i are IID and drawn from some distribution $F(x)$ that is heavy-tailed with the shape parameter $1 < \alpha < 2$, then the CLT applied to

$$Z = n^{1-1/\alpha}(\bar{X} - \mu). \tag{2.24}$$

says that the limit distribution of Z, as $n \to \infty$, converges to a heavy-tailed distribution; see Crovella and Lipsky [23]. For large n, (2.24) can be written as

$$|\bar{X} - \mu| \sim n^{1/\alpha - 1}. \tag{2.25}$$

Equation (2.25) shows how slowly $|\bar{X} - \mu|$ converges to $n^{1/\alpha-1}$. Alternatively, if α is close to 1, then the speed of convergence is very slow, but for $\alpha = 1$, the average does not converge at all; the mean is infinite.

We assume that the sample mean $\bar{X}$ can be used to form an estimate of the mean μ that is accurate until the relative error drops to $10^{-i} \cdot 100\%$. In other words, one might describe a simulation that has reached steady-state

when the observed mean of the input $\bar{X}$ agrees with μ until the relative error drops to $10^{-i} \cdot 100\%$ given by

$$\frac{|\bar{X} - \mu|}{\mu} \leq 10^{-i}. \tag{2.26}$$

Assuming

$$|\bar{X} - \mu| = c_1 n^{1/\alpha - 1}, \tag{2.27}$$

where c_1 is a positive constant, then, from Equations (2.26) and (2.27), the number of required observations is given by

$$c_2 10^{\frac{i}{1-1/\alpha}} \leq n, \tag{2.28}$$

where c_2 is a positive constant and i is a digit accuracy. For more detailed discussion, see Crovella and Lipsky [23].

Thus, we conclude that heavy-tailed distributions have convergence properties very different from those of conventional distributions. This means that the convergence of a simulation with heavy-tailed random variables to the steady-state can be very slow.

2.4 Self-Similar Processes in Computer Systems and Telecommunication Networks

Many recent studies of real teletraffic data have shown that teletraffic exhibits self-similar behaviour over a wide range of time scales in computer systems and telecommunication networks (see [8] and [172]). The self-similar nature of teletraffic (in the sense that long-range dependent behaviour is exhibited over a range of time scales: milliseconds, seconds, minutes and hours) can have a significant impact on computer network performance [29], [114]. When Park et al. [117], [118] explored the relationship between file sizes, transport protocols and self-similarity, they found that self-similar network traffic can have a detrimental impact on network performance (including amplified queueing delay, retransmission rate and packet loss rate).

The properties of teletraffic in self-similar scenarios are very different from the traditional models, and can lead to incorrect conclusions about the performance of analysed networks. For example, Paxson and Floyd [125] analysed

twenty one wide area network (WAN) traces, investigating a number of wide-area transmission control protocol (TCP) arrival processes to determine the error introduced by modelling them using Poisson processes. They showed that in cases of connection arrivals and TELNET packet inter-arrivals [109], Poisson models resulted in serious overestimations of the burstiness of TCP traffic over a wide range of time scales (i.e., time scales of 0.1 seconds and longer). On the other hand, if the strongly correlated character of teletraffic is explicitly taken into account, this can lead to more efficient traffic control mechanisms [70], [113].

Various studies of real teletraffic include local area networks (LAN) and WAN, networks operating under asynchronous transfer mode (ATM), Internet and World Wide Web (WWW), transmission control protocol/ Internet protocol (TCP/IP), video, queueing performance, congestion control of self-similar traffic, and others. We will first consider cases in which apparent self-similar behaviour of teletraffic has been found. Then we will discuss theoretical models proposed.

2.4.1 Self-Similar Nature of Real Teletraffic

- *LAN, MAN and WAN traffic:*

  In the early-1990s, Leland et al. [89] and Willinger et al. [90], [172] demonstrated that Bellcore *Ethernet LAN traffic is statistically self-similar in nature.* While they looked at the count processes (the number of packets submitted within a time unit in the second column), we took the same traces of teletraffic as in [89] and having tested inter-arrival times for self-similarity, we have found that they have self-similar properties too. On the basis of our statistical analysis we can conclude that traces of teletraffic investigated by Willinger et al. [90] have self-similar properties not only when one investigates their count processes, but also when one characterises them by processes assembled from inter-event times, see in Appendix B.

  As Addie et al. [5] shown, teletraffic in a commercial public broadband data network FASTPAC, an Australian high speed data network providing services of 2 Mb/sec. and 10 Mb/sec., is also self-similar. The traffic measurements were taken by a bus monitor which counted the number

of segments within every consecutive time interval. Each time interval is 1 millisecond long. The measurements were taken for a period of one busy hour on six different days: March 29 and April 11-15, 1994.

Paxson and Floyd [126] studied how Poisson processes fail as accurate models for WAN packet arrival processes. They found that only user-initiated TCP session arrivals such as remote-login and file-transfer are well-modelled as Poisson processes. Other TCP and FTP connection arrivals, over time scales of 0.1 seconds and longer, are not captured by Poisson models. Furthermore, the Poisson arrivals commonly used to model packet arrivals generated by the user side of a TELNET connection result in serious overestimations of the burstiness of the traffic. They also found that the distribution of the number of bytes in each burst has a very heavy upper tail, and a small part of the largest bursts has a majority of the traffic in almost all of the FTP traffic. FTP connection arrivals completely dominate FTP traffic. For example, for FTP traffic (PKT-1 trace and PKT-3 trace), the upper 2% tail holds about 50% of all the traffic; and for the PKT-2 trace and PKT-5 trace, the upper 2% tail holds 85% of all the traffic.

Erramilli et al. [29] showed that LRD has considerable impact on queueing performance and is a dominant characteristic for determining several issues of packet traffic engineering problems such as dimensioning of buffers and determining usable capacity.

- *Internet and WWW traffic:*

  Crovella and Bestavros [21] found evidence and possible causes of self-similarity in World Wide Web (WWW) traffic. They presented the LRD structure of WWW traffic using a set of traces of actual user executions of the National Centre for Super-computing Applications (NCSA) Mosaic (the first popular graphical World Wide Web browser in early 1993). Then, using WWW traffic, user preference, and WWW document file size data, they showed that the transmission times and quiet times for any particular WWW session are heavy-tailed, which is an essential characteristic of the proposed mechanism for the self-similarity of traffic.

- *Video traffic:*

  Analysis of variable-bit-rate (VBR) video traffic measurements produced similar results to that of the Internet and WWW traffic. Beran et al. [10] analysed a few million observed frames from VBR video services and found that LRD is an inherent feature of VBR video traffic. They also showed that the LRD property allows us to distinguish between the measured data and synthetic traffic generated VBR source models currently used in the literature, see also [77], [101], [102].

- *Others:*

  Gribble et al. [51] found evidence of self-similarity in file systems. They demonstrated the existence of self-similar behaviour in high-level file system events, such as file opens, block reads or writes, file closes, and file deletes, over short-term time scales of less than a day in length, but showed that this behaviour does not persist across time scales of days, weeks, and months.

2.4.2 Theoretical Models of Self-Similar Teletraffic

- *LAN and WAN traffic:*

  Later, Willinger et al. [173] and Taqqu et al. [159] developed and proved a result for self-similar traffic modelling . They found that the superposition of strictly independent alternating ON/OFF traffic models whose ON- or OFF-periods have heavy-tailed distributions with infinite variance can be used to model aggregate network traffic that exhibits self-similar (or long-range dependent) behaviour typical for measured Ethernet LAN traffic over a wide range of time scales.

- *ATM network traffic:*

  Georganas [45] and Likhanov et al. [95] proposed self-similar traffic models constituting of a superposition of infinite number of individual Pareto source models (i.e., an M/G/1 queueing system where the service time is Pareto distributed with infinite variance and the transmission discipline is a first-in, first-out (FIFO) manner for sources). They showed that self-similar traffic arriving at an ATM buffer results in a heavy-tailed

buffer occupancy distribution. They also found that the buffer cell loss probability decreases with the buffer size, not exponentially as in traditional Markovian models, but hyperbolically. Tsybakov and Georganas [162], [163], [165] constructed mathematical models for self-similar cell traffic and analysed the overflow behaviour of a finite-sized ATM buffer fed by self-similar traffic. They demonstrated that both asymptotic upper and lower bounds to the overflow probability decrease hyperbolically with increasing buffer size, and this decay is much slower than exponential decay. They also pointed out that previous calculations and buffer dimensioning of ATM switches will have to be reconsidered in view of the new analytical results obtained from self-similar processes. Tsybakov and Georganas [164] also gave a model for ATM cell traffic in communication networks and found general conditions of self-similarity for this model. A consequence is that the merging of traffic, as in a statistical multiplexer or an ATM switch, does not result in smoothing of traffic [35]. Bursty traffic that is multiplexed tends to produce bursty aggregate traffic. Thus, they showed that ATM network traffic is also self-similar.

Ryu [146] showed that self-similar models based on fractal point processes are able to capture accurately the queueing behaviour of various traffic streams and are tractable for queueing analysis under diverse scenarios of ATM networks.

- *Internet and WWW traffic:*

  Crovella and Bestavros [21] found that many characteristics of WWW use can be modelled using heavy-tailed distributions, including the distribution of transfer times, the distribution of user requests for documents, and the underlying distribution of WWW document sizes.

  Twenty one researchers [22] met to take stock of the field of Internet performance modelling in 1999, as this has played an increasingly significant role for computer network users. They reported three dimensions to the problems: (i) measurement techniques for the Internet have begun to develop, but significant unresolved problems remain. One of the main reasons for this is that traffic streams in the Internet exhibit high variability across a wide range of time scales (i.e., self-similar behaviour); (ii) models and solution techniques are not yet well developed because all related network conditions simultaneously interact; (iii) methods to

improve resource management and control in the Internet are being developed, but change rapidly and so are often not subjected to rigorous performance evaluation, which is also difficult on the Internet.

- *TCP/IP network traffic:*

  Park et al. [117], [118] examined a mechanism that creates self-similar network traffic, and presented some of its performance implications. This mechanism is the transfer of files or messages whose size is drawn from a heavy-tailed distribution. They showed that properties of the transport layer play an important role in preserving and modulating the relationship between file sizes, transport protocols and self-similar network traffic. In particular, the reliable transmission and flow control mechanisms of TCP serve to maintain the LRD structure induced by heavy-tailed file size distributions. They also presented data on the relationship between self-similar traffic and network performance as captured by performance measures such as packet loss rate, retransmission rate and queueing delay. They showed that increased self-similarity results in degradation of performance, and queueing delay exhibits a dramatic increase as self-similarity increases.

  Veres et al. [169] demonstrated how induced self-similarity is propagated and spread in modern computer networks by TCP congestion control, which represents the dominant transport protocol of the Internet. This is due to its congestion control algorithm, which adapts to self-similar fluctuations on several time scales. They also demonstrated that if a TCP connection shares a bottleneck link with a self-similar background traffic flow, it propagates the correlation structure of the background flow to a characteristic time scale. They showed that if congestion periods are LRD, then the end-user perceived end-to-end traffic is also LRD, and found that the self-similarity of a TCP stream can be passed on to other TCP streams.

- *Video traffic:*

  Garrett and Willinger [42], and Rose [141] showed that models for VBR video traffic using heavy-tailed distributions give more accurate results than the commonly used stochastic processes (see Appendix A.1). The main finding of their analysis was that the autocorrelation of the VBR

video sequence decays hyperbolically and can be modelled using F-ARIMA and FGN self-similar processes. Trace-driven simulations showed that statistical multiplexing results in significant bandwidth efficiency even when LRD is present. Simulations of their gamma/Pareto model fed by Hosking's algorithm (one of the F-ARIMA processes) showed LRD and heavy-tailed marginals to be important features which are not taken into account in currently used VBR video traffic models. Sahinoglu and Tekinay [150] presented a survey of the self-similarity phenomenon observed in multimedia traffic such as video and voice, and its important implications for network performance.

- *Queueing performance:*

  The impact of self-similar models on queueing performance has been investigated in many papers. Beginning with the work by Norros [114], there has been mounting evidence that clearly shows that the performance of queueing models with self-similar inputs can be dramatically different from the performance predicted by traditional models of teletraffic based on Markovian processes [27].

  Erramilli et al. [29] gave conditions under which the use of accurate and simple traffic models that capture LRD in a parsimonious manner is justified, and obtained practically relevant solutions to performance problems of high-speed communication networks that carry self-similar-like traffic. Willinger et al. [173] complemented this evidence by illustrating the practical relevance of such findings for (i) parsimonious traffic modelling for high-speed networks, (ii) efficient simulation of actual network traffic, and (iii) analysing queueing models and protocols under realistic traffic scenarios. The traditional models of teletraffic that assume independent arrivals, based on Poisson processes, Markov-modulated Poisson processes and other related processes, with an ACF that drops off exponentially rather than hyperbolically, are not able to capture the self-similar nature of teletraffic [126]. Addie et al. [4], [6] developed a heuristic method to evaluate the performance of a queue fed by a self-similar traffic stream. They showed the accuracy of their evaluation on six different real traffic networks and found that the inaccuracy was usually low.

  Jelenković [67] developed techniques for approximating buffer overflow probabilities in a network multiplexer. He also found that MPEG video

traffic in multimedia networks exhibits multiple time scales and sub-exponential characteristics that are heavy-tailed. Resnick and Samorodnitsky [134] and Dahl and Willemain [24] investigated how self-similar processes can have a large impact on queue performance, and demonstrate that the distributions of the waiting time and the queue length have heavy tails, using a fixed sample size of 10,000 observations.

Greiner et al. [49], [50] investigated how power-tailed distributions (a subclass of heavy-tailed distributions) can occur in computer systems and developed a truncated analytical model that in the limit is power-tail. They demonstrated power-tailed distributions for modelling a steady-state $GI/M/1$ queueing system in simulation studies of telecommunication networks and showed that steady-state behaviour varies smoothly with $\alpha > 1$.

Neame et al. [112] showed that the queueing performance of the M/Pareto process as a model for an arbitrary broadband traffic stream depends on the level of aggregation in the process. The M/Pareto process is composed of a number of overlapping bursts as a sub-group of the more general $M/G/\infty$ processes. Bursts arrive according to a Poisson process with rate λ. The duration of each burst is random, and chosen from a Pareto distribution. Then, the M/Pareto process converges to a long-range dependent Gaussian process when the aggregation level is high. They also showed that the M/Pareto process is capable of modelling the queueing performance of real broadband traffic sequences such as Ethernet LAN and VBR MPEG video sequences when the aggregation level is low. However, their M/Pareto process still has some limitations because the choice of the arrival rate is complicated by the fact that the correct value of the arrival rate differs depending on the service rate.

Neidhardt and Wang [113] pointed out that larger values of H are not always associated with longer queues, but Dahl and Willemain [24] did not observe this phenomenon with their choice of parameter values. Likewise Grossglauer and Bolot [53] showed that some buffer overflow problems are relatively insensitive to certain long-range dependent structures, whereas Dahl and Willemain [24] found strong sensitivities for the means and maxima of queue statistics. Clearly, more research is needed to fully understand the effect of self-similar processes on the queueing behaviour

in different queueing systems.

Jeong et al. [75] found that when self-similar input with $H \geq 0.6$ is assumed in a steady-state simulation of queueing models using the spectral analysis method, one needs many more observations to obtain the final simulation results with a relative precision $\leq 10\%$, at 95% confidence level, than when assuming Poisson processes.

Boxma and Cohen [12] presented an approximation for the waiting time distribution for an M/G/1 queueing system with a heavy-tailed service time distribution by using the heavy traffic limit theorem. This resulting approximation is sharp even when the traffic intensity is not heavy. For different service disciplines, such as first-come-first-served (FCFS), processor sharing (PS) and last-come-first-served preemptive resume (LCFS-PR), we need to investigate further the behaviour of other queueing systems with heavy-tailed inter-arrival and/or service time distributions. It is important to perform additional empirical experiments to fully understand the effect of heavy-tailed traffic characteristics on performance measures, and develop useful approximations.

- *Congestion control system:*

  Park et al. [118], and Park and Tuan [119] investigated the effect of multiple time scale congestion control on network performance when the network is subject to highly self-similar traffic conditions. Decreasing the arrival rate results in a decline in packet loss rate, whereas link utilisation increases. They also found that increasing sophistication of control leads to improved performance that is preserved even under highly self-similar traffic conditions.

  Östring et al. [116] found that when existing self-similarity of teletraffic is employed in designing available-bit-rate (ABR) congestion control algorithms, then queue lengths at finite buffers can be lower and cell losses can be significantly smaller. In the case of finite buffer sizes, self-similarity of teletraffic causes higher cell losses within the system and these cell losses decrease hyperbolically as the buffer size increases, rather than decreasing exponentially as in case of Poisson processes.

- *Others:*

  An ON/OFF source model was used for synthesising file system traffic that exhibits the same self-similar behaviour as observed in the file system sequences [51]. A comprehensive bibliographical guide to self-similar traffic and performance modelling for modern high-speed computer networks is presented in [171].

2.5 Conclusions

Given the strong empirical evidence that self-similar processes are better than Poisson processes in capturing crucial network teletraffic characteristics such as burstiness, it has become vital to develop and apply tools for understanding self-similar processes, and for generating synthetic network traffic for use in simulations.

The methods of verification and modelling of self-similar behaviour of teletraffic are further discussed in this thesis. The scope of the thesis is practically limited to issues related with important, but one characteristic of self-similarity: the Hurst parameter.

In Chapters 3 – 7 some definitions and statistical properties of self-similarity described above are used to compare and analyse sequences generated by self-similar pseudo-random teletraffic generators.

Chapter 3

CRITICAL REVIEW OF HURST PARAMETER ESTIMATION TECHNIQUES

3.1 Introduction

Many studies have investigated possible causes of LRD phenomena in real networks. Two main sources of them have been identified. Firstly, LRD can be the result of users' behaviour. For instance, network users usually do not care about sizes of files stored in their servers. Typically there are collections of very short and very long files which are transmitted between computers whenever it is needed, regardless of their sizes. Despite the existence of very powerful compression algorithms, data compression is generally rarely used. Lack of rationality when dealing with very long files is one of likely causes of the probability of distributions of file sizes being heavy-tailed and, as explained in Chapter 2, heavy-tailed inter-events times cause LRD of count processes.

Secondly, the other source of LRD can exist within networks themselves, since they have become complex adaptive systems. In the case of retransmission protocols, for example, such protocols as the exponential back-off algorithm, LRD can occur when network becomes congested. Hara and Taketsugu [55] found that DCA-TDMA (dynamic channel allocation-based time division multiple access) cellular networks have a hidden cause of LRD: a strong inter-

dependencies between cells cause inter-cell interference.

While researchers largely acknowledge the significance of LRD phenomena, they still disagree on two issues: (i) selection of the best LRD models for buffer dimensioning and bandwidth allocation, and (ii) selection of the best measures of LRD, including statistical estimators; see e.g., Krunz and Matta [82]. In this thesis, we focus on the second issue.

The LRD phenomenon is well characterised by the Hurst parameter. As mentioned in Chapter 2, the Hurst parameter shapes ACFs of LRD processes, and it is needed for determining variance of such a process. In this chapter we limit our attention to estimators of the Hurst parameter.

For practical reasons, we consider here the methods of estimation of the Hurst parameter H from a finite time series. Some popular Hurst parameter estimation techniques are based on the idea of estimating the slope of a linear fit in a log-log plot; for detailed discussion, see [9]. For example, the R/S-statistic estimation technique is a well-known example of this approach, although, as we will see, it has poor statistical performance; notably it has a high bias when the value of the Hurst parameter ($\frac{1}{2} < H < 1$) is small or large. Another example is the periodogram plot based on a linear fit in a $log_{10}(P(\lambda))$ against $log_{10}(\lambda)$ plot, where $P(\cdot)$ is the periodogram and λ is frequency.

We report results of our comparative analysis of H estimators. Estimators considered include the wavelet-based H estimator and Whittle's Maximum Likelihood Estimator (MLE), as well as estimators based on periodogram analysis, R/S-statistic analysis, variance-time analysis and IDC(t) analysis. We have focused on the wavelet-based H estimator and Whittle's MLE of H, as they are asymptotically unbiased and efficient in theory, at least in the FGN case [2], [167].

This chapter is organised as follows. In Section 3.2 we describe the generation of exact pseudo-random self-similar sequences, which are used as reference sequences for assessing estimation techniques of the Hurst parameter. We use an exact self-similar FGN process based on the *Durbin-Levinson* algorithm. Section 3.4 describes the most frequently used estimation techniques of the Hurst parameter. To enable estimation techniques to be analysed, the appropriate sample size must first be determined. This is discussed in Section 3.3. Quality of the considered estimators of H is discussed on the basis of numeri-

Table 3.1: Mean generation times of the exact self-similar sequences (FGN process) when using the Durbin-Levinson algorithm. The results obtained using the SunOS 5.7 `time` command on a Pentium II (233 MHz, 512 MB); each mean time is an average over 30 iterations.

Sequence of					
32,768 Numbers	65,536 Numbers	131,072 Numbers	262,144 Numbers	524,288 Numbers	1,048,576 Numbers
Mean running time (*hour:minute:second*)					
00:01:44	00:07:28	00:39:08	02:41:09	10:55:04	44:13:37

cal results showing mean values of the estimated H parameters, as well as the results of appropriate statistical tests, see Section 3.5.

3.2 Generation of Exact Self-Similar Sequences

To assess the quality of various estimators of H experimentally, one needs to apply them to self-similar stochastic processes of exactly known properties. In the study presented in this chapter, the role of such a reference process will be played by an FGN process.

Following the recommendation given in [1], [13], [158], we generated exact self-similar FGN sequences by using the Durbin-Levinson algorithm. This algorithm, with complexity $O(n^2)$, is the fastest currently known algorithm for generation of exactly self-similar processes; see [136]. Therefore, we use this algorithm to generate exact self-similar sequences for our comparative analysis of H estimation techniques. This algorithm written in C is given in Appendix C, see also Taqqu et al. [158]. Table 3.1 shows the mean running times of the FGN generator based on the Durbin-Levinson algorithm. The generation of a sequence of one million numbers took approximately 44 hours. Such a rate of generation of pseudo-random self-similar sequences would be too slow in simulation practice, for example, when simulating performance of telecommunication networks. So, generators of approximately self-similar sequences are of great practical importance and are discussed in Chapter 4.

3.3 Determining Minimal Sample Sizes

We determine the minimal length of a sequence used as the sample for estimating H parameter, to ensure that our estimates can be satisfactorily accurate, despite using finite sequences. Such an approach makes our analysis different from majority of published results on H estimation which did not consider this practical aspect. The exact estimation of the Hurst parameter from finite sequences is still an open problem. Thus, when undertaking a comparative study of different estimators, we assume that we deal with satisfactorily long sequences for producing H estimates close to the values that would be obtained if sequences were infinite long.

Different reported studies have used sequences of different length (different sample sizes). For example, Mandelbrot and Wallis [105] used a sample of 9,000 numbers using the R/S-statistic. Leland et al. [89], [172] analysed sequences of 360,000 observations, where each observation represented the number of bytes sent over the Ethernet per 10 milliseconds. Garrett and Willinger [42] presented a statistical analysis of a two-hour long empirical sample of VBR video with 171,000 frames. Paxson [124], for each input H value, used ten samples of 32,768 numbers to obtain estimates of the Hurst parameter. He confirmed that the stochastic dependence present in the generator was consistent with having the required value of H using Whittle's MLE. Rose [141] studied traffic modelling of VBR MPEG video and its impacts on ATM networks with different sequences of 40,000 frames of MPEG traffic data (it takes 30 minutes to obtain each sequence). An R/S-statistic estimation technique and Whittle's MLE were used to estimate the Hurst parameter for MPEG traffic in [141]. Abry and Veitch [2], [167] compared the wavelet-based H estimator and Whittle's MLE using small synthetic sample sequences of 4,096 numbers and the real Ethernet LAN data set. A minimisation procedure is involved in Whittle's MLE estimator which requires many repetitive calculations, leading to a significantly higher overall cost than in the case of the wavelet-based H estimator, which requires simple calculations of discrete wavelet transforms in $O(n)$ operations.

We evaluate the most commonly used methods for estimating the self-similarity parameter H to find the best estimator in the sense of bias. We determine the minimal length of sequences for which bias becomes negligible, and

then analyse Hurst parameter estimation techniques using synthetic sequences generated by the exact pseudo-random self-similar FGN process discussed in Section 3.2. To determine the minimal length of sequences, we investigate accuracy of H estimates as a function of the length of batch sizes and sample sizes.

Firstly, for each $H = 0.6$, 0.7, 0.8 and 0.9, each sequence with one million numbers is divided into sub-sequences (i.e., batch size) m of $2^{10}, 2^{11}, \ldots, 2^{19}, 2^{20}$ numbers. For example, let a sequence $(x_1, \ldots, x_n)$ be divided into i sub-sequences $(x_1, \ldots, x_l)$, $(x_{l+1}, \ldots, x_{2l})$, ..., $(x_{(i-1)l+1}, \ldots, x_{il})$, $l > 0$, $i = [n/l]$. Each estimate $\hat{H}_j, j = 1, \ldots, i$ is obtained using the considered estimation techniques of H. The mean and variance of these estimates are given by

$$\hat{H} = \sum_{j=1}^{i} \frac{\hat{H}_j}{i}, \tag{3.1}$$

and

$$\hat{\sigma}^2[\hat{H}] = \sum_{j=1}^{i} \frac{(\hat{H}_j - \hat{H})^2}{i}, \tag{3.2}$$

respectively [11].

In Figures 3.1 and 3.2, log_2 (Batch Size) is plotted against $\hat{H}$ obtained from the exact self-similar FGN process using the wavelet-based H estimator and Whittle's MLE, for $H = 0.6$, 0.7, 0.8 and 0.9. These estimators have also been widely used in the analysis of computer network traffic [2], [89], [167]. Figure 3.1 shows that for all H values, all curves for $\hat{H}$, using the wavelet-based H estimator, quickly converge toward the true values. The 95% confidence intervals of H, $\hat{H} \pm 1.96\hat{\sigma}[\hat{H}]$, are computed using the mean from Equation (3.1) and the variance from Equation (3.2).

As a measure of bias of the estimators we use their relative inaccuracy ΔH, as defined as:

$$\Delta H = \frac{\hat{H} - H}{H} * 100\%, \tag{3.3}$$

where H is the exact value assumed and $\hat{H}$ is its empirical mean value. Further, we assume that a sample size is acceptable when the relative inaccuracy drops below one percent.

Table 3.2 shows that the relative inaccuracies for all curves of estimates of H are less than one percent from $m = 2^{12}$. The relative inaccuracy ΔH in Table 3.2 is calculated using the formula: These curves converge to the required values. The range of scale $(i_1, i_2) = (4, 10)$ is used. Figure 3.2 also shows that all curves of $\hat{H}$ using Whittle's MLE converge to true values. The bottom-most curve matches the true values for all m, except for $H = 0.6$ at $m = 2^{10}$. However, for $H = 0.7$, 0.8 and 0.9, the curves of $\hat{H}$ converge more slowly toward the true values than for the wavelet-based H estimator as m increases. Table 3.3 shows that its relative inaccuracies are less than one percent from $m = 2^{15}$.

Figures 3.3 and 3.4 show estimates of H obtained from the exact self-similar FGN process using the wavelet-based H estimator and Whittle's MLE as the length of the sample size n increases from 2^{10} to 2^{17}, for $H = 0.6$, 0.7, 0.8 and 0.9. (We chose the length 2^{17} for the largest sample, because mean values of estimates of H using the wavelet-based H estimator and Whittle's MLE reached the required values when $n \geq 2^{13}$ and $n \geq 2^{10}$, respectively, for all H values.) Figure 3.3 shows that for all H values, all curves of H estimates from the wavelet-based H estimator converge to the required values. They have a positive bias if $n \leq 2^{12}$, and then the curves converge to the required values for $n \geq 2^{13}$. Table 3.4 shows that the relative inaccuracies for all curves of H estimates are less than one percent for $n \geq 2^{13}$. In contrast, Figure 3.4 shows that all curves of estimates of H from Whittle's MLE converge to the exact values faster than estimates from the wavelet-based H estimator, and reach the required values already at $n = 2^{10}$. Table 3.5 shows that its relative inaccuracies are less than one percent for $n \geq 2^{10}$.

Thus, minimal lengths of sequences for estimating H parameter are recommended to be between 2^{15} and 2^{17}. We have chosen the minimum sequence length to be 32,768 (2^{15}) in our study of these estimators.

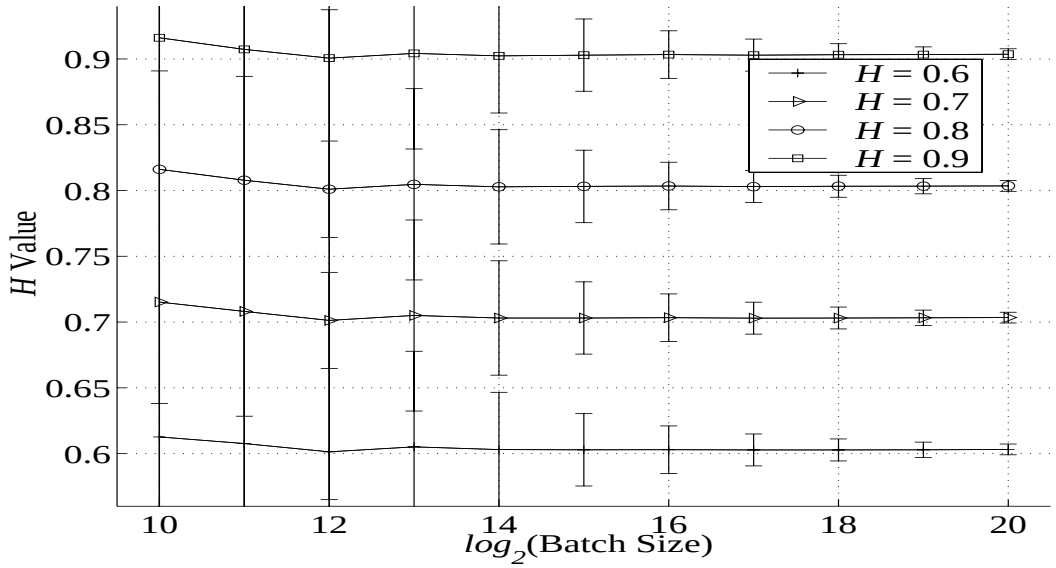


Figure 3.1: For $H = 0.6$, 0.7, 0.8 and 0.9, estimates of H obtained from the exact self-similar FGN process using the wavelet-based H estimator as the length of the batch size increases from 2^{10} to 2^{20}. The vertical bars at each batch size give 95% confidence intervals for the H values.

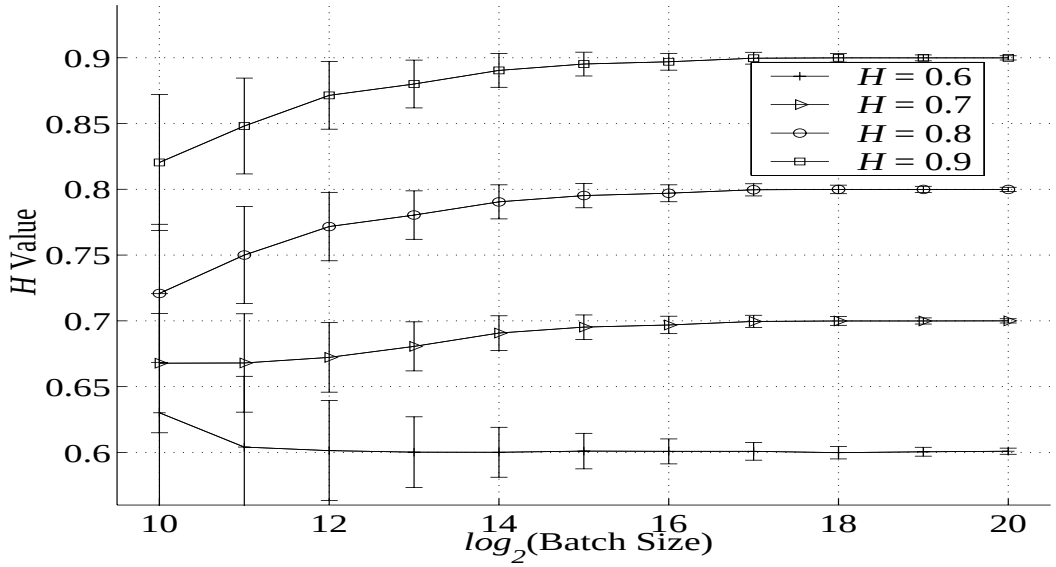


Figure 3.2: For $H = 0.6$, 0.7, 0.8 and 0.9, estimates of H obtained from the exact self-similar FGN process using Whittle's MLE as the length of the batch size increases from 2^{10} to 2^{20}.

Table 3.2: Relative inaccuracy, ΔH, of mean values of estimates of H using the wavelet-based H estimator for the exact self-similar FGN process as the length of the batch size increases, for $H = 0.6$, 0.7, 0.8 and 0.9.

Batch Size	ΔH (%)			
m	.6	.7	.8	.9
2^{10}	+2.1130	+2.1740	+2.0170	+1.7920
2^{11}	+1.2910	+1.1530	+0.9807	+0.8061
2^{12}	+0.2280	+0.1856	+0.1207	+0.0677
2^{13}	+0.8436	+0.7212	+0.5857	+0.4626
2^{14}	+0.5144	+0.4488	+0.3544	+0.2589
2^{15}	+0.4824	+0.4440	+0.3826	+0.3192
2^{16}	+0.5050	+0.4694	+0.4197	+0.3704
2^{17}	+0.4717	+0.4271	+0.3725	+0.3225
2^{18}	+0.4679	+0.4375	+0.3941	+0.3506
2^{19}	+0.4867	+0.4543	+0.4119	+0.3700
2^{20}	+0.5317	+0.4914	+0.4425	+0.3967

Table 3.3: Relative inaccuracy, ΔH, of mean values of estimates of H using Whittle's MLE for the exact self-similar FGN process as the length of the batch size increases, for $H = 0.6$, 0.7, 0.8 and 0.9.

Batch Size	ΔH (%)			
m	.6	.7	.8	.9
2^{10}	+5.0180	−4.6000	−9.8830	−8.8450
2^{11}	+0.6811	−4.5720	−6.2520	−5.7690
2^{12}	+0.2291	−3.9760	−3.5360	−3.1770
2^{13}	+0.0254	−2.7640	−2.4500	−2.2190
2^{14}	+0.0131	−1.3220	−1.1830	−1.0720
2^{15}	+0.1747	−0.6896	−0.5995	−0.5320
2^{16}	+0.1264	−0.4406	−0.3788	−0.3462
2^{17}	+0.1415	−0.0636	−0.0516	−0.0446
2^{18}	−0.0304	−0.0143	−0.0163	−0.0119
2^{19}	+0.0833	−0.0086	−0.0113	−0.0089
2^{20}	+0.1583	−0.0057	−0.0075	−0.0067

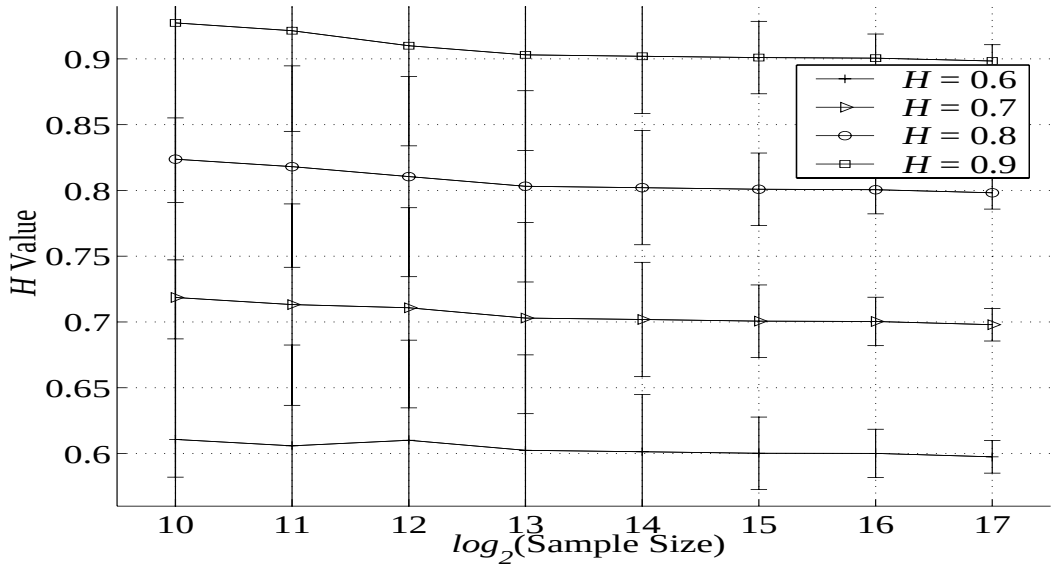


Figure 3.3: The wavelet-based H estimator for sample sizes from 2^{10} to 2^{17}.

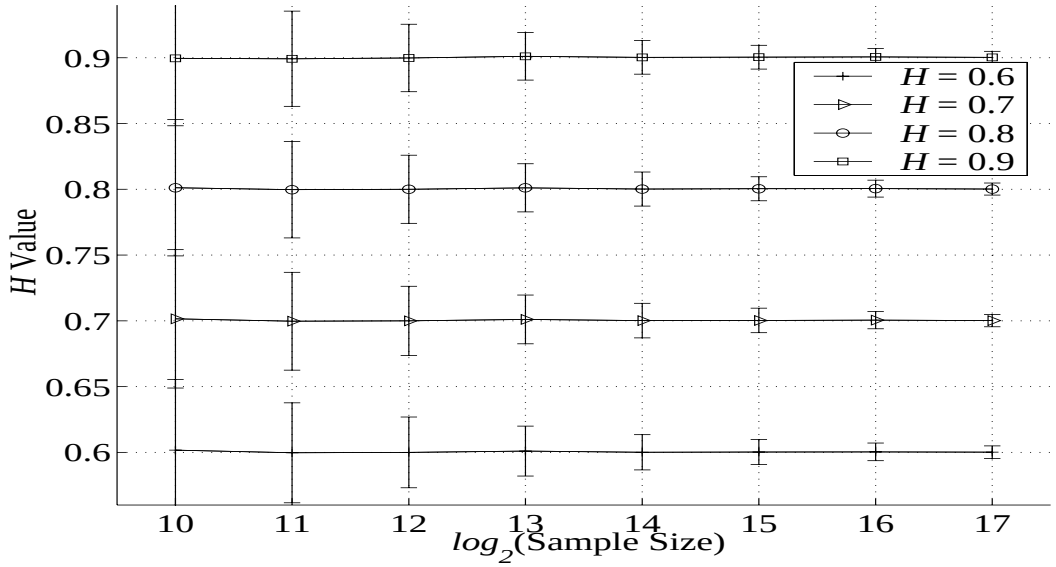


Figure 3.4: Whittle's MLE for sample sizes from 2^{10} to 2^{17}.

3.4 Hurst Parameter Estimators

3.4.1 Wavelet-Based H Estimator

The original wavelet-based H estimator, proposed by Abry and Veitch in 1998 [2], is a fast estimator based on a wavelet transform, with a fast pyramidal

algorithm for the wavelet transform, with its complexity of order $O(n)$. However, as argued in [167] this estimator suffers from a bias associated with its log-log regression component. Later, Veitch and Abry [167] proposed a new improved method of estimation of the H parameter within a so-called wavelet-based joint estimator, which allows us to estimate both H and so-called power parameter, an independent quantitative parameter with the unit of variance;

Table 3.4: Relative inaccuracy, ΔH, of mean values of estimates of H using the wavelet-based H estimator for the exact self-similar FGN process as the length of the sample size increases, for $H = 0.6$, 0.7, 0.8 and 0.9.

Sample Size	ΔH (%)			
n	.6	.7	.8	.9
2^{10}	+1.7870	+2.6530	+2.9670	+3.0340
2^{11}	+0.9794	+1.8880	+2.2650	+2.3760
2^{12}	+1.6810	+1.5360	+1.3170	+1.1110
2^{13}	+0.4068	+0.4232	+0.3838	+0.3292
2^{14}	+0.2352	+0.2762	+0.2615	+0.2278
2^{15}	+0.0258	+0.0920	+0.1096	+0.1075
2^{16}	+0.0095	+0.0593	+0.0672	+0.0603
2^{17}	- 0.4140	- 0.3001	- 0.2304	- 0.1759

Table 3.5: Relative inaccuracy of Whittle's MLE.

Sample Size	ΔH (%)			
n	.6	.7	.8	.9
2^{10}	+0.2873	+0.2193	+0.1459	- 0.0601
2^{11}	- 0.0512	- 0.0381	- 0.0314	- 0.0891
2^{12}	+0.0032	+0.0050	+0.0046	- 0.0235
2^{13}	+0.1594	+0.1501	+0.1419	+0.1252
2^{14}	+0.0155	+0.0238	+0.0310	+0.0332
2^{15}	+0.0435	+0.0443	+0.0441	+0.0435
2^{16}	+0.0656	+0.0647	+0.0663	+0.0661
2^{17}	+0.0303	+0.0303	+0.0275	+0.0289

see [142] and [167] for details. The resulted wavelet-based H estimator, that we further simply call the wavelet-based H estimator, is asymptotically unbiased and (almost) the most efficient [167].

The wavelet transform generates the wavelet coefficients $d_x(i,j)$ from a sequence of given numbers. For an LRD process, the variance of the wavelet coefficients at each level i is defined by

$$E[d_x(i,\cdot)^2] = \mathcal{C}\mathcal{A}_f 2^{i(2H-1)}, \tag{3.4}$$

where $\mathcal{C} > 0$ and

$$\mathcal{A}_f = 2(2\pi)^{1-2H} c_\gamma \mathcal{E}(2H-1)\sin((1-H)\pi),$$

where $\mathcal{E}(\cdot)$ is the Euler function and c_γ is a positive constant. The power parameter $\mathcal{A}_f$ that plays a major role in fixing the absolute size of LRD generated effects in applications is an independent quantitative parameter with the dimensions of variance.

Abry and Veitch ([2], [167]) have suggested that a spectral estimator can be obtained by calculating a time average μ_i of the $d_x(i,\cdot)$ at a given scale, that is,

$$\mu_i = \frac{1}{n_i}\sum_{j=1}^{n_i} d_x^2(i,j), \tag{3.5}$$

where n_i is the number of wavelet coefficients at scale i, i.e., $n_i = 2^{-i}n$, where n is the number of the data points. The estimator uses a weighted linear regression as the variances of $\log_2(\mu_i)$ vary with $\log_2(2^i)$. $\hat{H}$, an estimated Hurst parameter H is obtained from a linear regression [167]:

$$y_i = \log_2(\mu_i) = (2\hat{H}-1)i + c + 1/(n_i \ln 2), \tag{3.6}$$

where c is a constant; see [167] for details. Mean values of experimental values of the $d_x(i,\cdot)$ for $1 \le i \le 10$, and regression lines defined by points y_i [see Equation (3.6)], for scale $(i_1, i_2) = (4,10)$, are shown in Figure 3.5, for the exact self-similar FGN process with $H = 0.6$, 0.7, 0.8 and 0.9.

In practice, we need to select the scales $i = i_1, \ldots, i_2$ over which the power-law behaviour in Equation (3.4) holds, since the definition of LRD in $f(\lambda, H) \to c\lambda^{1-2H}$, as $\lambda \to 0$, is asymptotic. For a more detailed discussion,

see also Veitch and Abry [167]. Table 3.6 shows that the mean values of $\hat{H}$ obtained from 100 exact self-similar sequences are biased when the scale i_1 is small (the minimum scale is one).

Table 3.7 shows variances of $\hat{H}$ values $\mathrm{Var}[\hat{H}]$ for the exact FGN generator for different scales and H values. The variances gradually increase as the value of i_1 increases. Our results showed that the wavelet-based H estimator is the least biased at scales $i_1 = 4$ and 5. Therefore, scale $i_1 = 4$ is chosen for all our experiments.

Furthermore, Figure 3.5 shows that the estimates of H at higher scales are not only highly variable, but also biased, as Roughan et al. [143], [144]

Table 3.6: ΔH of mean values of estimated H using the wavelet-based H estimator for the exact self-similar FGN process for different scales and H values. n = 32,768 (2^{15}).

Scale	ΔH (%)			
(i_1, i_2)	.6	.7	.8	.9
(1,10)	+5.8590	+6.1380	+6.0870	+5.8800
(2,10)	+1.3000	+1.7920	+1.9410	+1.9290
(3,10)	+0.2200	+0.4090	+0.4639	+0.4628
(4,10)	+0.0258	+0.0920	+0.1096	+0.1075
(5,10)	-0.0783	-0.0788	-0.0904	-0.1033
(6,10)	-0.4554	-0.3627	-0.3088	-0.2806

Table 3.7: $\mathrm{Var}[\hat{H}]$ of the wavelet-based H estimator for the exact self-similar FGN process for different scales and H values. n = 32,768 (2^{15}).

Scale	Estimated Variances			
(i_1, i_2)	.6	.7	.8	.9
(1,10)	1.713e-05	1.700e-05	1.697e-05	1.703e-05
(2,10)	4.356e-05	4.362e-05	4.383e-05	4.420e-05
(3,10)	7.630e-05	7.716e-05	7.851e-05	8.058e-05
(4,10)	1.892e-04	1.922e-04	1.951e-04	1.985e-04
(5,10)	4.863e-04	4.912e-04	4.952e-04	4.955e-04
(6,10)	1.228e-03	1.277e-03	1.337e-03	1.393e-03

discussed. Slope of all curves in Figure 3.5 at $\log_2(2^i) = 9$ sharply goes down because of insufficient numbers of data points at the end of these values of scale. Thus, this results in the under-sampling of events at these scales, and, if these scales were used in the H estimation, they might result in noticeable inaccuracies.

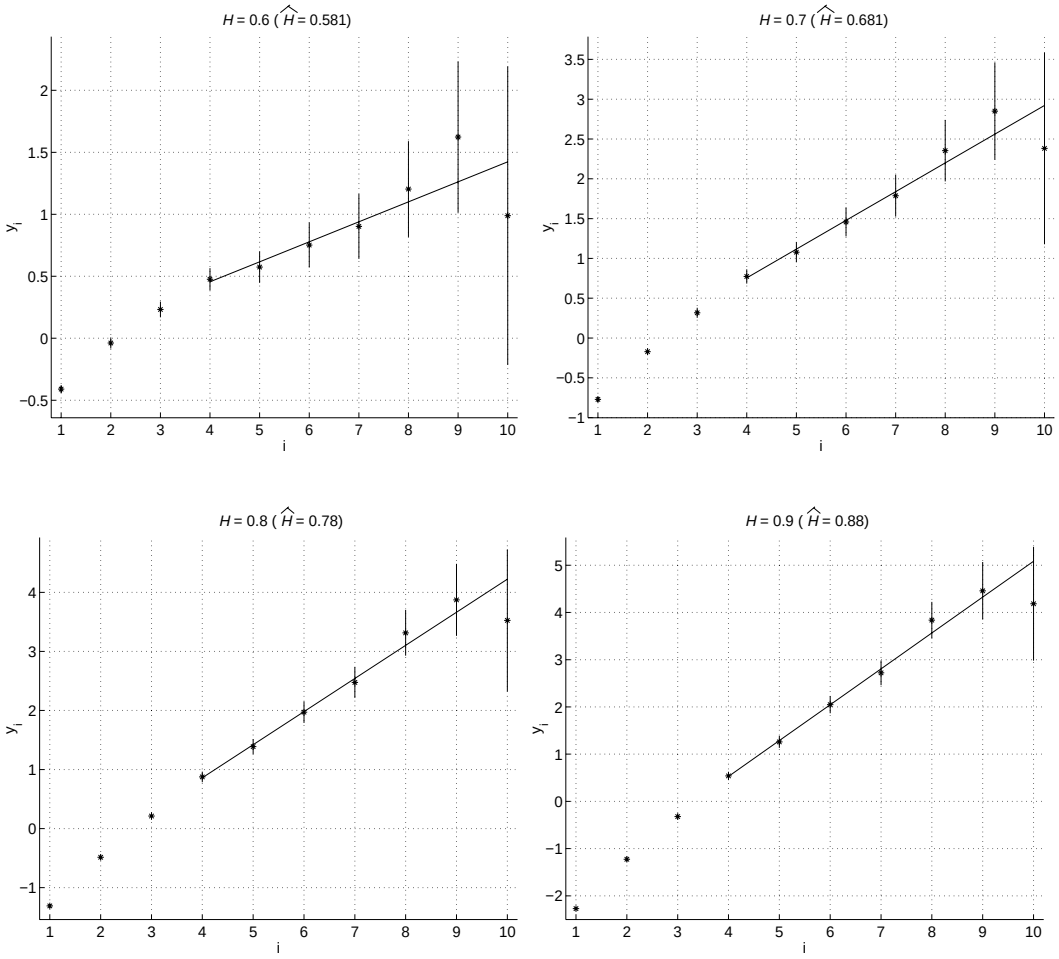


Figure 3.5: Mean values of $d_x(i, \cdot)$, and their confidence intervals, for $1 \leq i \leq 10$, and regression curves for wavelet-based H estimates in the case of the exact self-similar FGN process, with $H = 0.6$, 0.7, 0.8 and 0.9, for scale (i_1, i_2) = (4,10). n = 32,768 (2^{15}).

3.4.2 Whittle's Approximate MLE

It is possible to employ more refined data analysis to obtain confidence intervals (CIs) for the Hurst parameter H applying MLEs (Maximum Likelihood-type Estimators) and related methods based on the periodogram [9]. Several versions of the periodogram-based estimator can be found in the literature (see [8], [9], [141]). Whittle's approximate MLE, proposed by Whittle in 1953 [9], has been studied extensively and has shown to have desirable statistical properties for Gaussian processes, as presented in [8] and [172]. It is defined as follows.

Let $f(\lambda, \theta)$, for $-\pi \leq \lambda \leq \pi$, be the spectral density of the self-similar process with $\theta = (Var(\varepsilon_i), H, \theta_3, \ldots, \theta_k)$, where $Var(\varepsilon_i)$ is the variance of the innovation ε_i in the infinite autoregressive representation of the process $\mathbf{X} = \{X_i\}_{i=0}^{i=\infty}$, with $X_i = \sum_{j=1}^{\infty} \alpha_j X_{i-j} + \varepsilon_i$, H is the Hurst parameter, and the parameters $(\theta_3, \ldots, \theta_k)$ describe the SRD behaviour of the process. This implies $\int_{-\pi}^{\pi} \log(f(\lambda, (1, \eta)))d\lambda = 0$.

Let $P(\lambda), -\pi \leq \lambda \leq \pi$, denote the periodogram of the given sequence $\{X_1, X_2, \ldots, X_n\}$ defined by

$$P(\lambda) = \frac{1}{2\pi n} \left| \sum_{j=1}^{n} X_j e^{ij\lambda} \right|^2 . \tag{3.7}$$

$\hat{H}$, Whittle's Approximate MLE is defined as the value of H which minimises $Q(H)$, where

$$Q(H) = \int_{-\pi}^{\pi} \frac{P(\lambda)}{f(\lambda, H)} d\lambda. \tag{3.8}$$

Along with Whittle's MLE, the estimate of σ_H^2 is given by

$$\hat{\sigma}_H^2 = 4\pi \left[\int_{-\pi}^{\pi} \left(\frac{\partial \log f(\omega_i)}{\partial H} \right)^2 d\omega \right]^{-1}, \tag{3.9}$$

where $\omega_i = 2\pi i/n (i = 1, 2, \ldots, n^*), n^* = (n-1)/2$ if $n-1$ is even and $n^* = (n-1)/2 - 1/2$ if $n-1$ is odd.

Combining Whittle's approximate MLE approach with a technique of data aggregation leads to the so-called aggregated Whittle's MLE, proposed in [172].

Another semi-parametric, Whittle's type estimator, known as the local Whittle's MLE was proposed in [140]. However, comparative studies of these estimators showed that the original Whittle's MLE remains by far the most accurate [157], and because of that we use it in the research reported in this thesis.

3.4.3 Periodogram-Based Estimators

A few different techniques of H estimation on the basis of periodograms have been proposed. We discuss them briefly in this section.

Periodogram

The periodogram estimator, proposed by Daniell in 1948 [19], is a graphical method of assessing H. The periodogram of $X_i = \{X_1, X_2, \ldots, X_n\}$ is defined by Equation (3.7). This shows that if the autocorrelations are summable, i.e., SRD, then, near the origin, it should be scattered randomly around a constant level. If the autocorrelations are non-summable, i.e., LRD-type, the points are scattered around a negative slope. The periodogram plot is obtained by plotting $log_{10}(P(\lambda))$ against $log_{10}(\lambda)$. Periodograms for the exact self-similar FGN process are shown in Figure 3.6. An estimate of the Hurst parameter is given by $\hat{H} = \frac{(1-\hat{\beta}_3)}{2}$, where $\hat{\beta}_3$ is the slope of a regression line which is fitted to a number of low frequencies [9]. In some cases, this plot will lead to a quite inaccurate estimate of H since the periodogram estimation method is biased and inconsistent [9]. However, this plot can reveal the power spectrum near the origin [111]. In practice we use only the lowest 10% of the frequencies for the calculation, since this phenomenon holds only for frequencies close to zero. Properties of this periodogram-based estimator of H are analysed further in this chapter, since it has been found to be superior to other estimators of this class [158].

Modified Periodogram

A modification of the periodogram estimation technique compensates for the fact that on a log-log plot most of the points are scattered on the far right, thus exerting a very strong influence on the least-squares line fitted to the

periodogram. The frequency axis is divided into logarithmically equally spaced boxes, and the periodogram values corresponding to the frequencies inside each box are averaged. Several values at the very low frequencies are left untouched, since there are so few of them to begin with. Taqqu and et al. [158] showed that in general, results did not seem to show an improvement on the original periodogram technique.

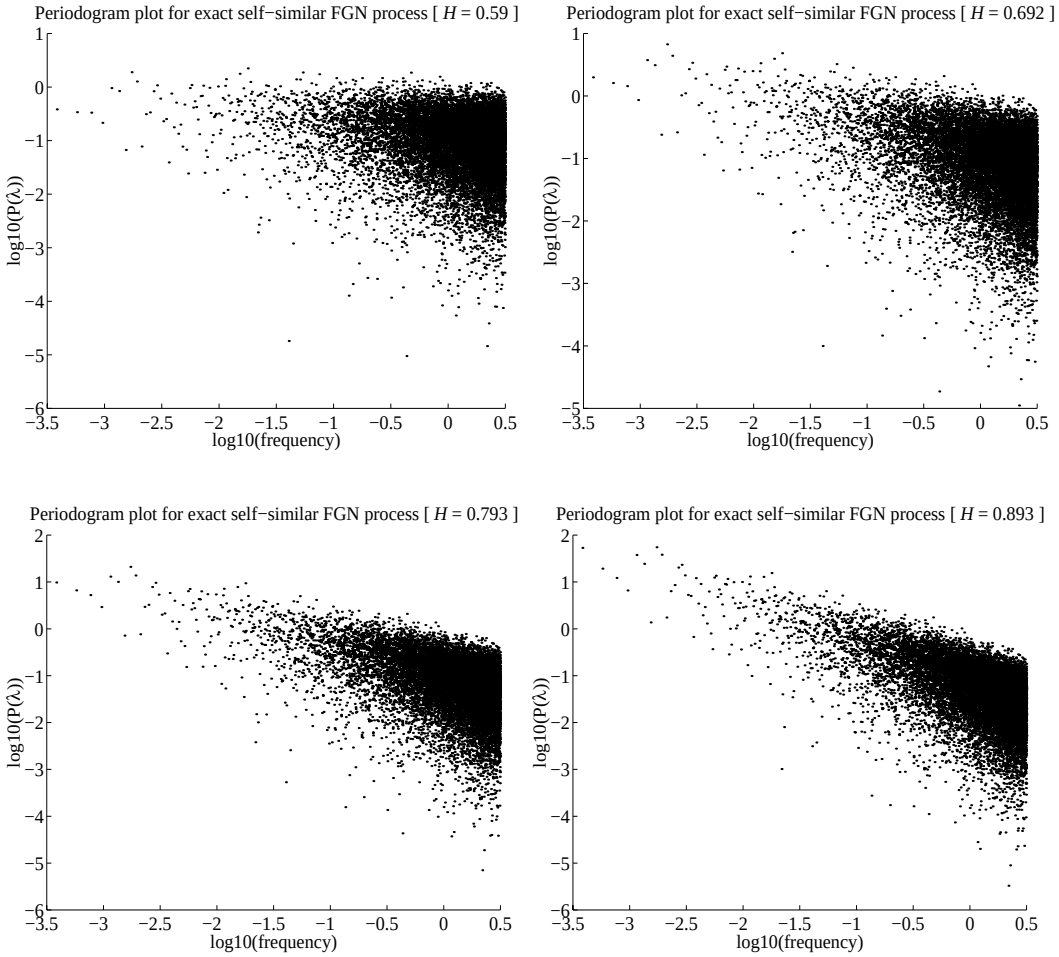


Figure 3.6: Periodogram plots for time series generated by the exact self-similar FGN process with $H = 0.6$, 0.7, 0.8 and 0.9. $n = 32{,}768$ (2^{15}).

Cumulative Periodogram

The cumulative periodogram can be calculated instead of using the periodogram itself, and it can be plotted in a log-log scale [158]. The slope of a regression line which is fitted to a number of low frequencies would then be proportional to $|\lambda|^{2-2\hat{H}}$. The advantage of the cumulative periodogram is that it is somewhat smoother than the original periodogram technique, although more approximations are used in the cumulative periodogram. However, both periodograms are biased because we treat them as discrete functions, and, thus, instead of integrating, which would result in the slope of a regression line proportional to $|\lambda|^{2-2\hat{H}}$, we approximate them by

$$C_{\kappa_1}(\lambda_i) = \frac{2\pi}{n} \sum_{j=\kappa_1+1}^{i} P(\lambda_j), \kappa_1 < i \leq \kappa_2,$$

where n is the length of the sequence, $\lambda_{\kappa_1} < \lambda_i \leq \lambda_{\kappa_2}$ (λ_{κ_1} and λ_{κ_2} are a lower and a higher cut-off frequency, respectively), and $P(\lambda_j)(\lambda_{\kappa_1+1} \leq \lambda_j \leq \lambda_i)$ is the periodogram given in Equation (3.7). For small enough i, $C_{\kappa_1}(\lambda_i) \sim c(\lambda_i^{2-2\hat{H}} - \lambda_{\kappa_1}^{2-2\hat{H}})$, where c is a constant. Thus, we select values of κ_1 and κ_2, and then calculate $C_{\kappa_1}(\lambda_i)$. Then, a least-squares fit of $C_{\kappa_1}(\lambda_i)$ against $c(\lambda_i^{2-2\hat{H}} - \lambda_{\kappa_1}^{2-2\hat{H}})$ is performed. However, as shown in [158], the original periodogram is consistently more accurate. Because of that, we have chosen the method based on the original periodogram in our further study.

3.4.4 R/S Statistic

An R/S statistic estimator, proposed by Hurst in 1951 [9], is used to estimate the Hurst parameter H based on the *rescaled adjusted range* (or *R/S statistic*) from empirical data. Given empirical data of length $n(X_i : i = 1, \ldots, n)$, the whole sequence is subdivided into i non-overlapping batches. Now, we compute the R/S statistic $\frac{R_{t,m}}{S_{t,m}}$ from Equations (2.13) and (2.14). The R/S statistic plot is obtained by plotting $log_{10}(\frac{R_{t,m}}{S_{t,m}})$ against $log_{10}(m)$ (also called the *pox diagram of R/S*); see also Figure 3.7. When H is well defined, a typical R/S statistic plot starts with a transient zone representing the nature of SRD in the sample, but eventually settles down and fluctuates around a straight line of a slope $\hat{\beta}_2$. The graphical R/S statistic is used to determine whether such asymptotic behaviour appears to be supported by the data. An estimate

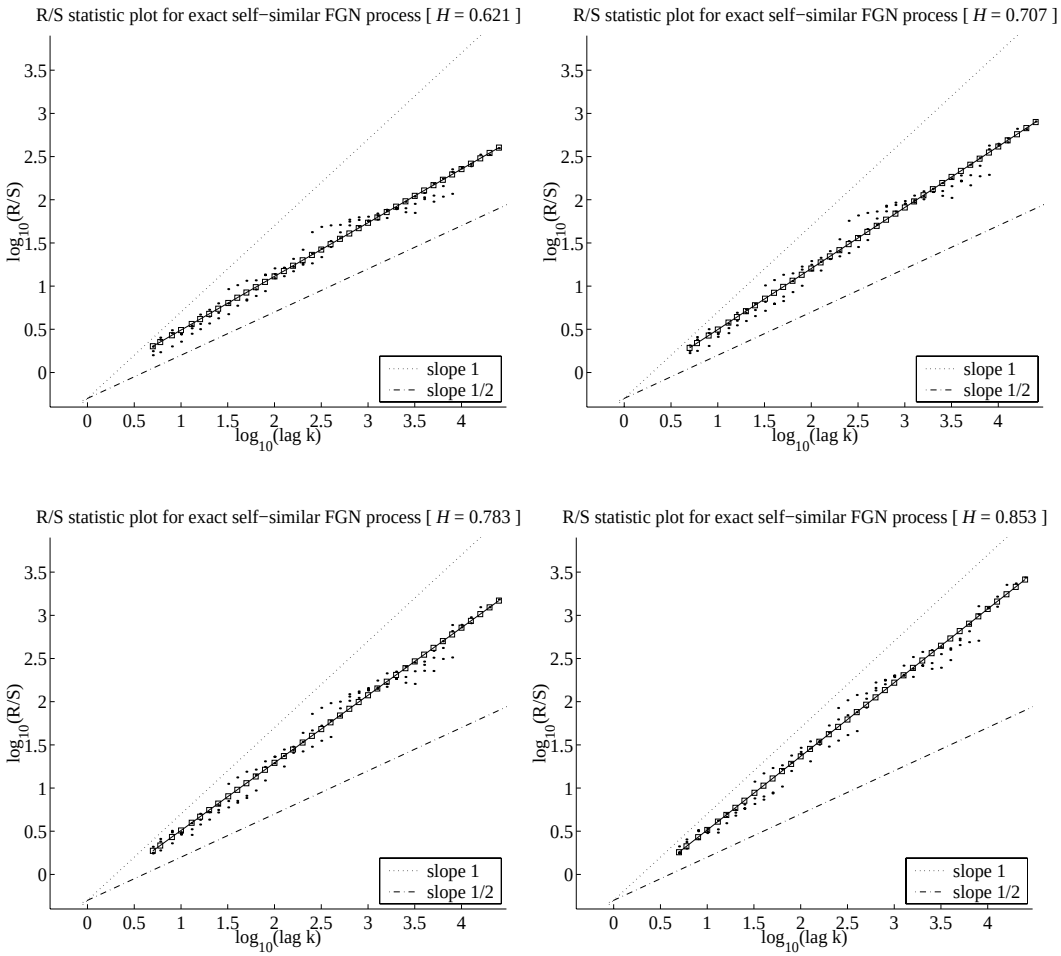


Figure 3.7: R/S statistic plots for time series generated by the exact self-similar FGN process with $H = 0.6$, 0.7, 0.8 and 0.9. $n = 32{,}768$ (2^{15}).

of H is given by the asymptotic slope $\hat{\beta}_2$ of the R/S statistic plot, which can take any value between 1/2 and 1, i.e., $\hat{H} = \hat{\beta}_2$ [9].

3.4.5 Moment-Based Estimators

A number of different estimators of H derived from formulas for moments have been proposed. The most important of them is briefly discussed in this section.

Variance-Time

The variance-time estimator, proposed by Cox and Smith in 1953 [19], is based on the property of self-similar processes that the variances of the sample means converge more slowly to zero than m^{-1} [89], [154], i.e.,

$$\mathrm{Var}(\mathbf{X}^{(m)}) \sim cm^{2H-2},$$

where $c > 0$, and $\mathbf{X}^{(m)}$ is the process obtained by aggregation of the original process $\mathbf{X}$, as defined on page 13. In contrast, the variances of the aggregated processes $\mathbf{X}^{(m)}, m \geq 1$, decrease linearly for large m in log-log plots against m with slopes arbitrarily flatter than -1. The variance-time plot is obtained by plotting $log_{10}(Var(\mathbf{X}^{(m)}))$ against $log_{10}(m)$ and by fitting a simple regression line through the resulting points; see also Figure 3.8. An estimate of the Hurst parameter is given by $\hat{H} = 1 - \frac{\hat{\beta}_1}{2}$, where $\hat{\beta}_1$ is the slope, $0 < \hat{\beta}_1 < 1$. The variance-time plot needs three steps for estimating H [9]:

(i) for different integers m in the range $2 \leq m \leq \frac{n}{2}$, and a sufficient number (say i) of a sub-series of length m, calculate the sample means $\bar{X}_1^{(m)}, \bar{X}_2^{(m)}, \ldots, \bar{X}_i^{(m)}$ and the overall mean $\bar{X}^{(m)}$

$$\bar{X}^{(m)} = \frac{1}{i} \sum_{j=1}^{i} \bar{X}_j^{(m)}; \tag{3.10}$$

(ii) for each m, calculate the sample variance of the sample means $\bar{X}_j^{(m)} (j = 1, \ldots, i)$:

$$Var(\mathbf{X}^{(m)}) = \frac{1}{i-1} \sum_{j=1}^{i} (\bar{X}_j^{(m)} - \bar{X}^{(m)})^2;$$

and (iii) plot $log_{10}(Var(\mathbf{X}^{(m)}))$ against $log_{10}(m)$.

In practice, it is assumed that both n and m are large. This ensures that both the length of each batch and the number of batches is large. If the sequence does not have long-range dependencies and finite variance, the estimate of $\hat{H}$ is 0.5 and the slope of the fitted line should be -1. Some points at the very low and high ends of the plots are not used practically to fit the least-squares line. In other words, short-range effects can distort the estimates of H if the very low end of the plot is used, and at the very high end of the plot there are hardly any blocks to obtain reliable estimates of the variance. This is probably the most popular estimator of H. Its properties are further investigated in this chapter.

Absolute Moments

This estimation technique is very similar to the variance-time plot in Section 3.4.5. The sequence is split in the same way, and the aggregated sequence Equation (3.10) computed. Instead of calculating the sample variance, one can find the k^{th} absolute moment $A_k^{(m)}$ of the aggregated sequence given by

$$A_k^{(m)} = \frac{1}{n/m} \sum_{j=1}^{n/m} \left| X_j^{(m)} - \bar{X} \right|^k,$$

where $\bar{X}$ is the overall mean.

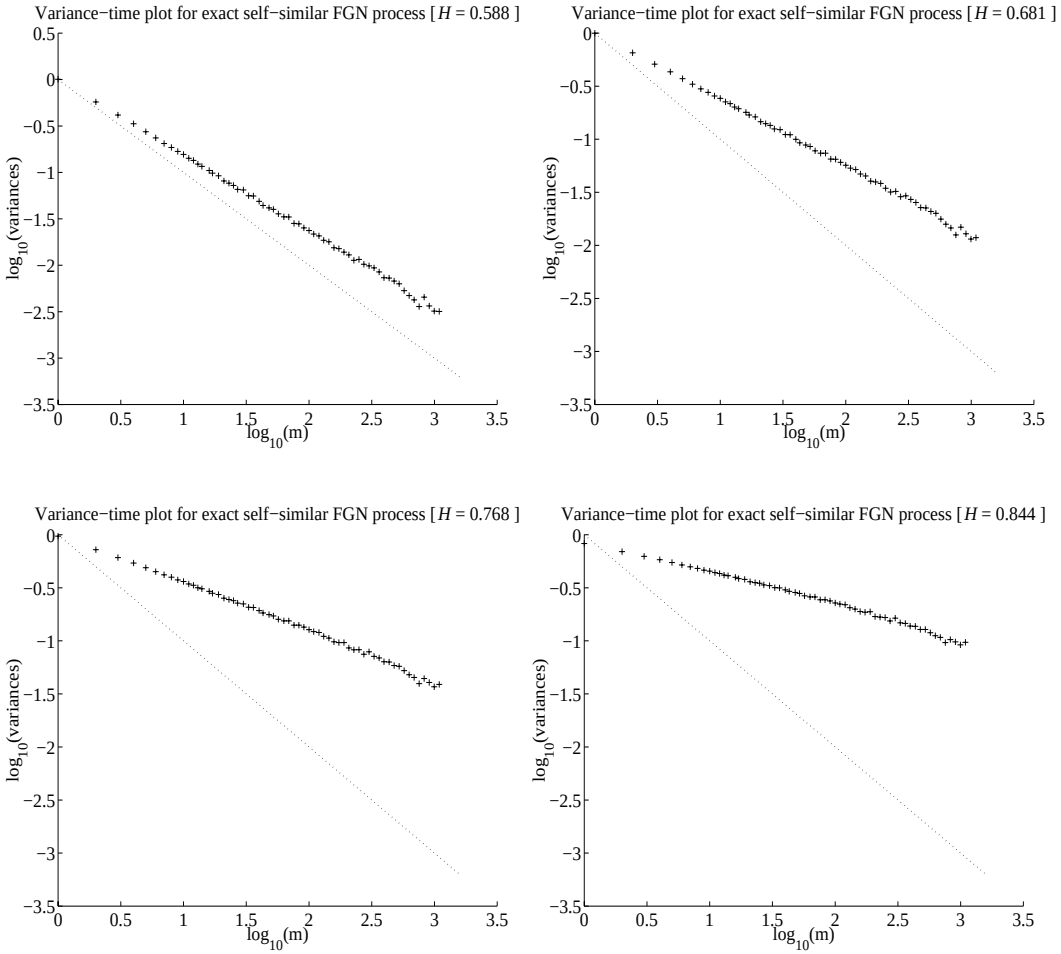


Figure 3.8: Variance-time plots for time series generated by the exact self-similar FGN process with $H = 0.6$, 0.7, 0.8 and 0.9. $n = 32{,}768$ (2^{15}).

To find an estimate of H, the logarithm of the statistic $log_{10}(A_k^{(m)})$ is plotted against $log_{10}(m)$. The aggregated sequence $X_i^{(m)}$ behaves asymptotically in a similar manner to $cm^{k(\hat{H}-1)}$ for large m, where c is a positive constant. Thus, $A_k^{(m)}$ is proportional to $m^{k(\hat{H}-1)}$. For $k = 2$, this technique reduces to the variance-time estimation technique. If the original sequences have long-range dependencies, it should result in a line with a slope $k(\hat{H} - 1)$ [158].

Fractal Dimension

The technique proposed by Higuchi [61] is very similar to the absolute moments estimation technique with $n = 1$. This technique calculates the length of a sequence and finds its fractal dimension. The fractal dimension (FD) is defined by

$$FD(m) = \frac{n-1}{m^3} \sum_{i=1}^{m} \left[\frac{n-i}{m}\right]^{-1} \sum_{l=1}^{[(n-i)/m]} \left| \sum_{j=i+(l-1)m+1}^{i+lm} X_j \right|,$$

where $[x]$ denotes the largest integer smaller or equal x (or the so-called Gauss' notation). If a sequence is LRD, then $E[FD(m)] \sim cm^{\hat{H}-2}$, where $E[\cdot]$ means the mean value, and c is a positive constant. This technique requires much more computational time since the difference from the absolute moments estimation technique lies in using a sliding window to compute the aggregated sequence, instead of using non-intersecting batches. While this modification may result in increased accuracy in shorter time sequences, there seems to be no advantage in using it for a sequence length of over 10,000 numbers [158].

Variance of Residuals

The technique introduced by Peng et al. [129] consists of several steps. The sequence is divided into batches of size m. Then, within each of the batches, the partial sums of the sequence are calculated, $Y_i = \sum_{j=1}^{i} X_j, i = 1, 2, \ldots, m$. A least-squares line, $a + bi$, is fitted to the partial sums within each batch, and the sample variance of the residuals is computed by

$$\frac{1}{m} \sum_{i=1}^{m} (Y_i - a - bi)^2.$$

This step is repeated for each of the batches, and the resulting sample variances are averaged. This is equivalent to calculating the sample variance of the entire sequence since the batches are all of the same size. Taqqu et al. [158] proved that for large m, the variance of residuals is proportional to $m^{2\hat{H}}$ for FGN, and a similar result is obtained for F-ARIMA. A straight line with a slope $2\hat{H}$ is obtained from a log-log plot against m [158].

3.4.6 Index of Dispersion for Counts/Intervals

The index of dispersion for counts, IDC(t), and the index of dispersion for intervals, IDI(t), proposed by Rao and Chakravati in 1956 [19], are often used to describe burstiness because they are relatively straightforward to estimate and convey much more information than simpler indices such as the coefficient of variation. The IDC(t) captures the variability of traffic over different time scales for a count process. Let $\{X_1, X_2, \ldots, X_n\}$ be the number of events recorded in consecutive time intervals of length T_1. Then, the index of dispersion for counts over t consecutive time intervals is defined as

$$\text{IDC}(t) = \frac{Var\left[\sum_{i=1}^{t} X_i\right]}{E\left[\sum_{i=1}^{t} X_i\right]}. \tag{3.11}$$

Self-similar processes produce a monotonically increasing IDC(t) of the form ct^{2H-1}, where c is a finite positive constant that does not depend on t. In other words, this behaviour contrasts entirely with traditional processes such as Poisson, Poisson-related processes, and Markov-modulated Poisson processes. Their IDC(t) curves are either constant or converge to a fixed value quite rapidly.

The IDC(t) needs three steps for estimating H [19], [54], [89]: (i) for a finite sequence of numbers, the variance of $\{\sum_{i=1}^{t} X_i\}$ can be calculated by dividing the whole series into non-overlapping blocks of length t and treating them as different instances of $\{\sum_{i=1}^{t} X_i\}$; (ii) calculate IDC(t); and (iii) plotting $log_{10}(\text{IDC}(t))$ against $log_{10}(t)$ results in an asymptotic straight line with slope $2H$ - 1; see Figure 3.9.

To characterise inter-event processes by the index of dispersion for intervals, one needs to know consecutive inter-event intervals $\Delta t_1 = t_{e_2} - t_{e_1}, \Delta t_2 = t_{e_3} - t_{e_2}, \ldots, \Delta t_n = t_{e_{n+1}} - t_{e_n}$, where t_{e_i} is the time instance when the event

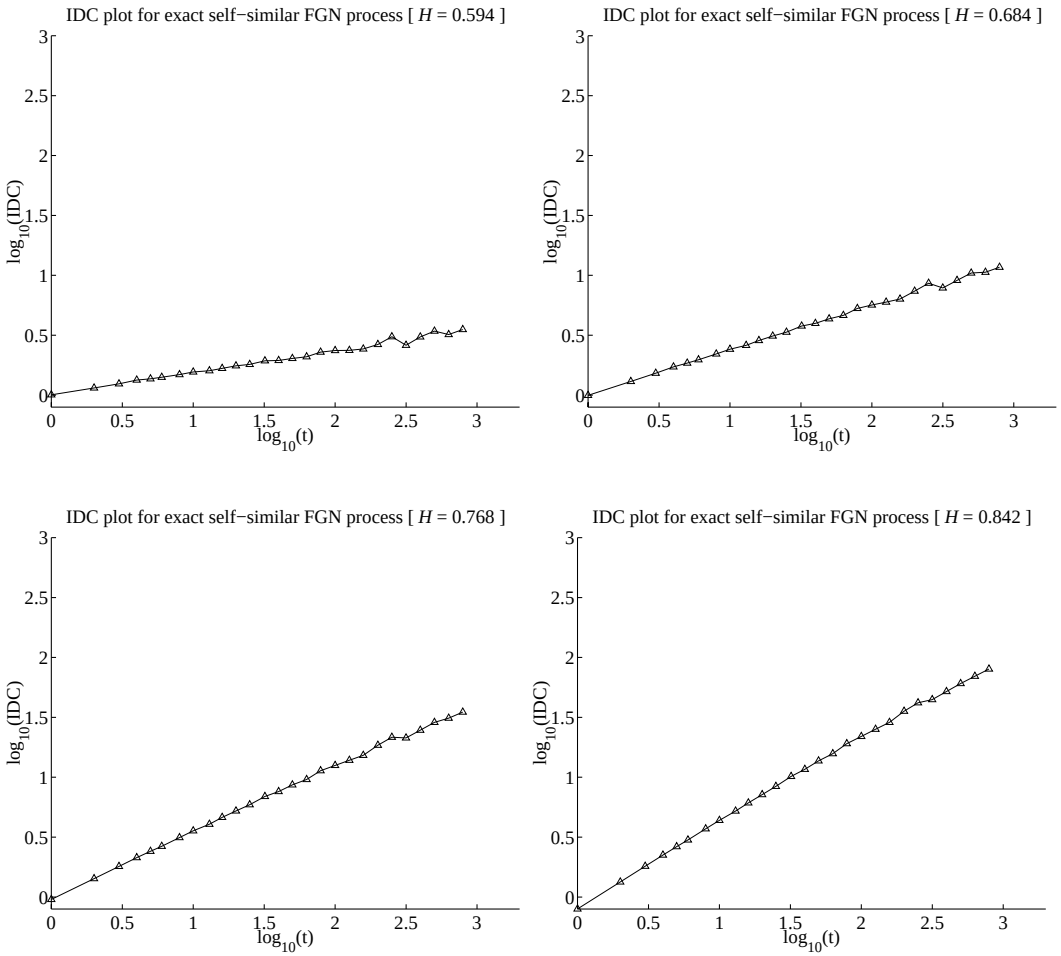


Figure 3.9: Index of dispersion for counts plots for time series generated by the exact self-similar FGN process with $H = 0.6$, 0.7, 0.8 and 0.9. $n = 32{,}768$ (2^{15}).

i happens. Then, for the time interval of length Δt, where $\Delta t = t_{e_n} - t_{e_1}$, the IDI($t$) is defined as

$$\mathrm{IDI}(\Delta t) = \frac{Var\left[\sum_{i=1}^{n} \Delta t_i\right]}{\Delta t \left\{E\left[\sum_{i=1}^{n} \Delta t_i\right]\right\}^2}. \tag{3.12}$$

The IDI(Δt) curves are also obtained by plotting $log_{10}(\mathrm{IDI}(\Delta t))$ against $log_{10}(\Delta t)$ using Equation (3.12). This method was used to obtain the results presented in Appendix B, Figures B.5 and B.6, for studying self-similarity of Bellcore Ethernet LAN teletraffic.

3.5 Numerical Comparison

3.5.1 Mean Values of Estimated H and Relative Inaccuracies, ΔH

Tables 3.8 and 3.9 show the relative inaccuracies, ΔH, of mean values of estimated H using these techniques. We give 95% confidence intervals for the means of the Hurst parameter estimators in parentheses.

- The results for the wavelet-based H estimator of H with the corresponding 95% confidence intervals $\hat{H} \pm 1.96\hat{\sigma}_{\hat{H}}$, (see Tables 3.8 and 3.9), show that for all input H values, confidence intervals for the exact self-similar FGN process are within the assumed theoretical values. For $H = 0.6$, the wavelet-based H estimator is the most accurate. For $0.6 \leq H \leq 0.9$, the relative inaccuracies are less than 0.11%.

Table 3.8: Relative inaccuracy, ΔH, of mean values of estimated H using the Hurst parameter estimation techniques for the exact self-similar FGN process for $H = 0.6$ and 0.7. We give 95% confidence intervals for the means of the Hurst parameter estimation techniques in parentheses. $n = 32{,}768$ (2^{15}).

Methods	Mean Values of Estimated H and ΔH			
	.6		.7	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
Wavelet-based	.6002 (.573,.628)	+0.026	.7006 (.673,.728)	+0.092
Whittle's MLE	.6003 (.591,.610)	+0.043	.7003 (.691,.710)	+0.044
Periodogram	.6008 (.599,.603)	+0.128	.7025 (.700,.705)	+0.358
R/S-statistic	.6277 (.624,.632)	+4.623	.7118 (.708,.716)	+1.689
Variance-time	.5964 (.594,.599)	-0.608	.6917 (.689,.695)	-1.192
IDC(t)	.5968 (.594,.600)	-0.528	.6919 (.689,.695)	-1.153

Table 3.9: Relative inaccuracy, ΔH, of mean values of estimated H using the Hurst parameter estimation techniques for the exact self-similar FGN process for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means of the Hurst parameter estimation techniques in parentheses. $n = 32{,}768$ (2^{15}).

Methods	Mean Values of Estimated H and ΔH			
	.8		.9	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
Wavelet-based	.8009 (.773,.828)	+0.110	.9010 (.874,.929)	+0.108
Whittle's MLE	.8004 (.791,.810)	+0.044	.9004 (.891,.909)	+0.043
Periodogram	.8040 (.802,.806)	+0.496	.9054 (.903,.908)	+0.598
R/S-statistic	.7916 (.787,.796)	-1.053	.8621 (.857,.867)	-4.210
Variance-time	.7814 (.778,.785)	-2.327	.8600 (.856,.864)	-4.439
IDC(t)	.7812 (.778,.785)	-2.349	.8593 (.855,.863)	-4.520

- Tables 3.8 and 3.9 show that Whittle's MLE is the most accurate for the exact self-similar FGN process, as shown in Figure 3.10. For $0.6 \leq H \leq 0.9$, the estimates of Whittle's MLE match the true values very closely (i.e., $|\Delta H| < 0.05\%$).

- The periodogram plots have slopes decreasing as H increases. The negative slopes of all plots for $H = 0.6$, 0.7, 0.8 and 0.9 are evidence of self-similarity. (Periodograms for FGN are shown in Figure 3.10.) The estimated Hurst parameters, assessed using periodogram plots, show positively biased $\hat{H}$ values. For $H = 0.6$ and 0.7, the confidence intervals of the estimated Hurst parameter contain the exact values, while for $H = 0.8$ and 0.9, they do not.

- For $H < 0.76$, the R/S statistic H estimates are positively biased, and for $0.76 < H$, it is negatively biased, with $|\Delta H| < 4.63\%$.

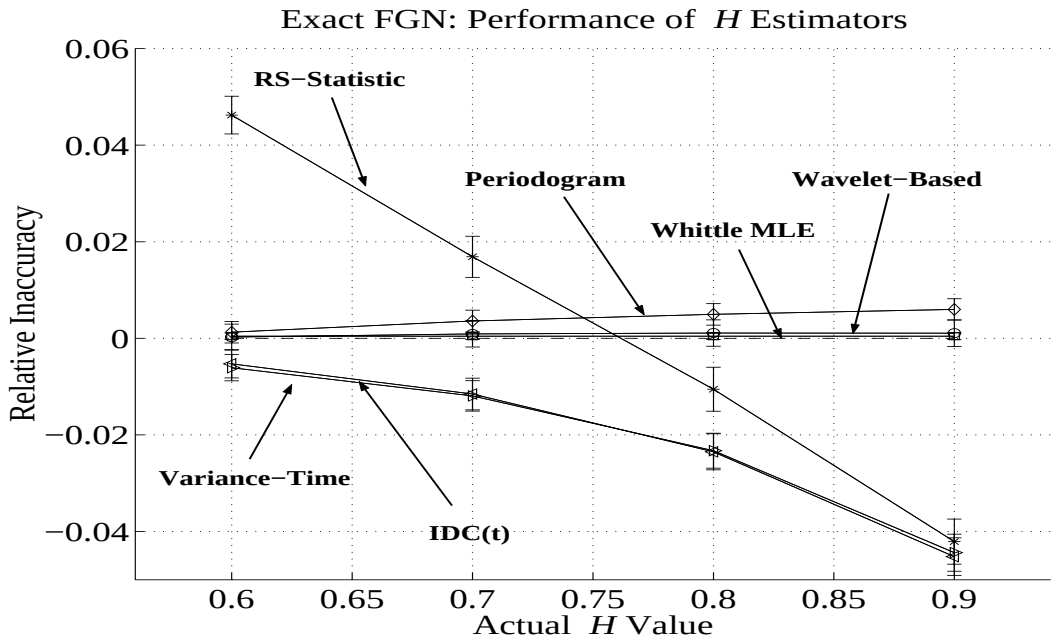


Figure 3.10: Bias performance of H estimators for the exact self-similar FGN process. $n = 32{,}768$ (2^{15}).

- The variance-time estimator produces negatively biased $\hat{H}$ values as H increases, with $|\Delta H| < 4.44\%$ and $\Delta H < 0$.

- The IDC(t) H estimator gives a negatively biased $\hat{H}$ as H increases, with $|\Delta H| < 4.53\%$ and $\Delta H < 0$.

So far we have concentrated on bias of the Hurst parameter estimators. A "good" estimator is not only one which produces an estimate whose expected value is close to the true parameter (low bias), but also one which has small variance. Table 3.10 shows that the variances of Whittle's MLE are much smaller than in the case of other estimators.

As the wavelet-based estimator and Whittle's MLE are substantially more accurate than the other methods, these have been chosen for further analysis. Figure 3.11 shows that for $H = 0.7$, 0.8 and 0.9, the estimates using Whittle's MLE are less biased, in all cases except $H = 0.6$. Figures 3.12 – 3.15 show that as indicated in Table 3.10, the H estimates from Whittle's MLE have a much lower variance than those from the wavelet-based H estimator.

Table 3.10: Variances of estimated H obtained using the Hurst parameter estimators for the exact self-similar FGN process for $H = 0.6$, 0.7, 0.8 and 0.9. $n = 32{,}768$ (2^{15}).

Methods	Variances of Estimated H			
	.6	.7	.8	.9
Wavelet-based	1.892e-04	1.922e-04	1.951e-04	1.985e-04
Whittle's MLE	1.093e-05	1.145e-05	1.179e-05	1.217e-05
Periodogram	1.288e-04	1.306e-04	1.291e-04	1.279e-04
R/S-statistic	3.971e-04	4.724e-04	5.400e-04	5.669e-04
Variance-time	1.928e-04	2.571e-04	3.405e-04	3.826e-04
IDC(t)	2.227e-04	2.812e-04	3.613e-04	3.947e-04

The same 100 sequences were used for Whittle's MLE and the wavelet-based H estimator. Thus we can further quantify their relative accuracies by counting how often (out of 100 Bernoulli trials) each estimator produced the more accurate estimate. The results are given in Table 3.11. The P values given in Table 3.11 give the chance of getting these descripancies or worse by chance alone if the two estimators were equally accurate. For more than 85% of the series Whittle's MLE is the most accurate for all the values of H.

3.6 Conclusions

Many studies of H parameter estimators have used sequences that are quite short. The exact estimation of the Hurst parameter from finite sequences is still an open problem. Thus, when undertaking a comparative study of different estimators, we assume that we deal with a satisfactorily long sequence for producing H estimates close to the values that would be obtained if sequences were infinite long.

We have determined the minimal length of a sequence for estimating H parameter to ensure that our estimates can be satisfactorily accurate, despite using finite sequences. Such a approach makes our analysis different from

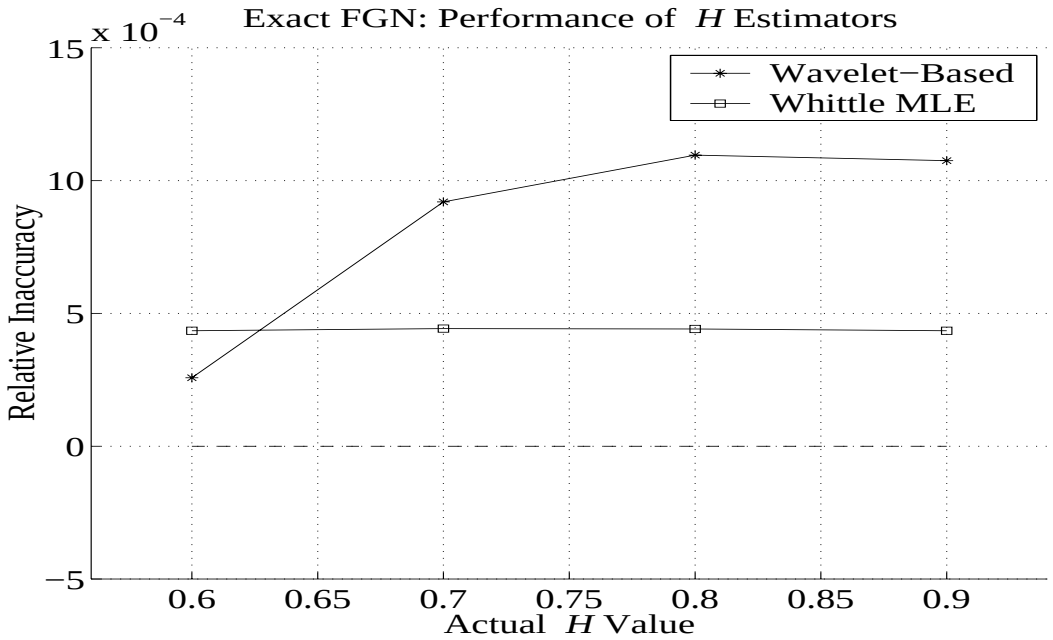


Figure 3.11: Bias performance of the wavelet-based H estimator and Whittle's MLE for the exact self-similar FGN process. $n = 32{,}768$ (2^{15}).

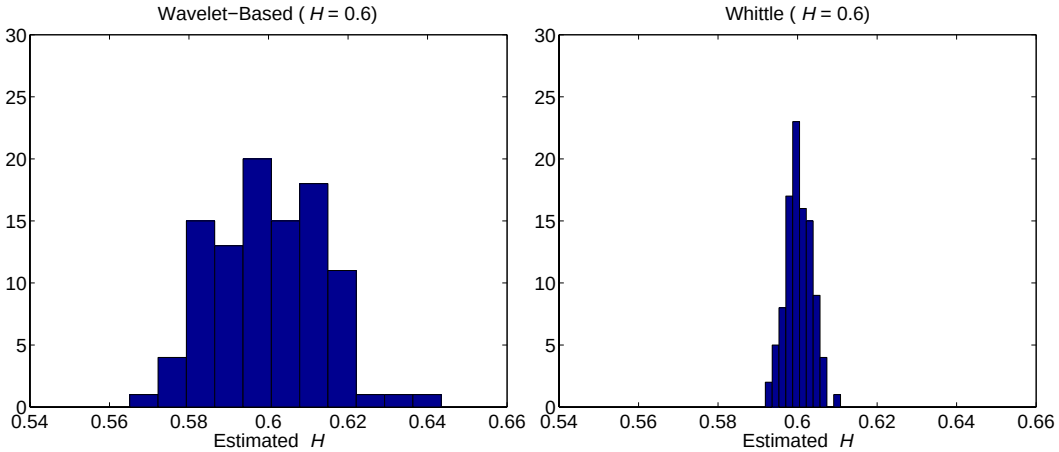


Figure 3.12: Histograms of estimated H values for the exact self-similar FGN process obtained from the wavelet-based H estimator and Whittle's MLE for $H = 0.6$. These are based on 100 replications. $n = 32{,}768$ (2^{15}).

majority of publications on H estimation which neglect this practical aspect. Minimal numbers of a sequence for analysing H parameter estimators are recommended to be between 2^{15} and 2^{17}. We have chosen the minimum sequence length to be 32,768 (2^{15}) for our review of these estimators.

A comparative analysis of the most frequently used H estimation techniques, the wavelet-based H estimator and Whittle's MLE estimator, periodogram, R/S-statistic, variance-time and IDC(t) estimators, has been done.

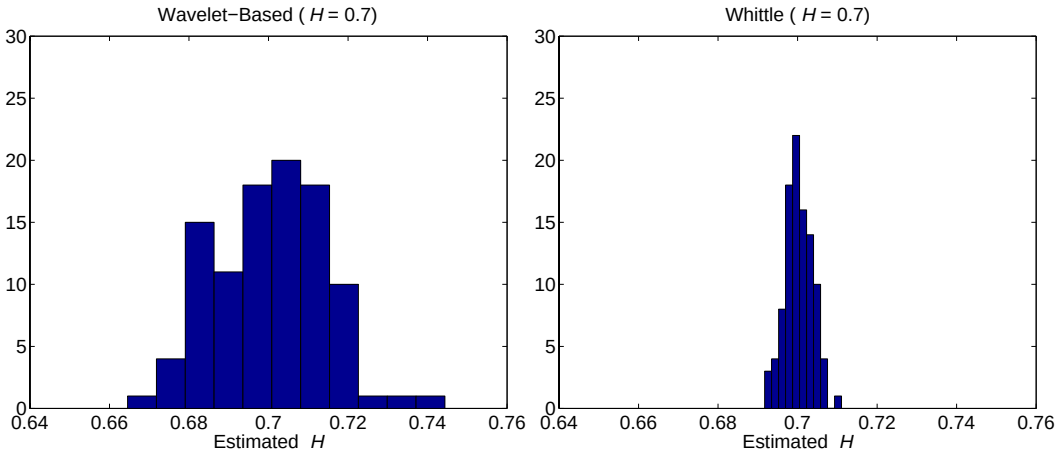


Figure 3.13: Histograms of estimated H values for the exact self-similar FGN process obtained from the wavelet-based H estimator and Whittle's MLE for $H = 0.7$. These are based on 100 replications. $n = 32{,}768$ (2^{15}).

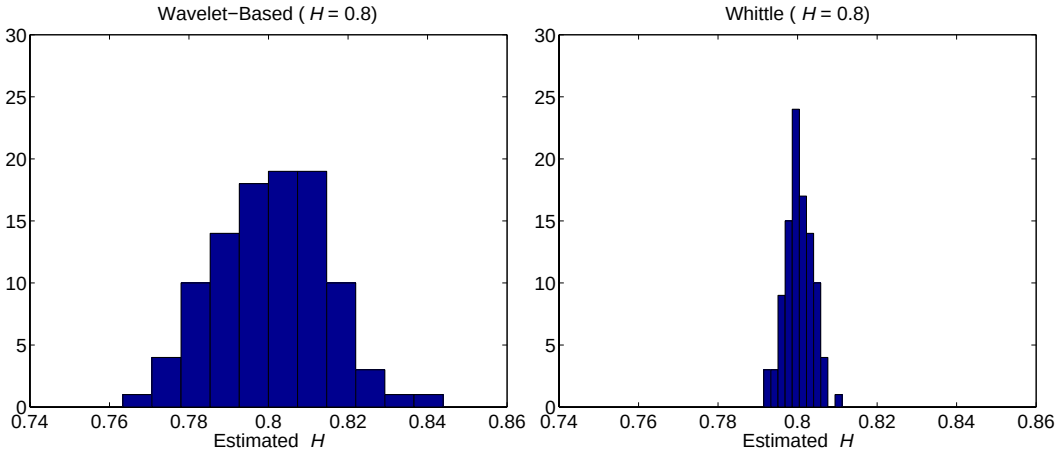


Figure 3.14: Histograms of estimated H values for the exact self-similar FGN process obtained from the wavelet-based H estimator and Whittle's MLE for $H = 0.8$. These are based on 100 replications. $n = 32{,}768$ (2^{15}).

Our results have shown that the wavelet-based H estimator and Whittle's MLE are the least biased of the H estimation techniques. While the wavelet-based H estimator has a lower bias when $H = 0.6$, and is computationally simpler and faster than Whittle's MLE, a superiority of Whittle's MLE is statistically closer to the required H value when samples are reasonably large. Thus, the wavelet-based H estimator may still be preferred in practical applications despite its relatively poorer accuracy than Whittle's MLE.

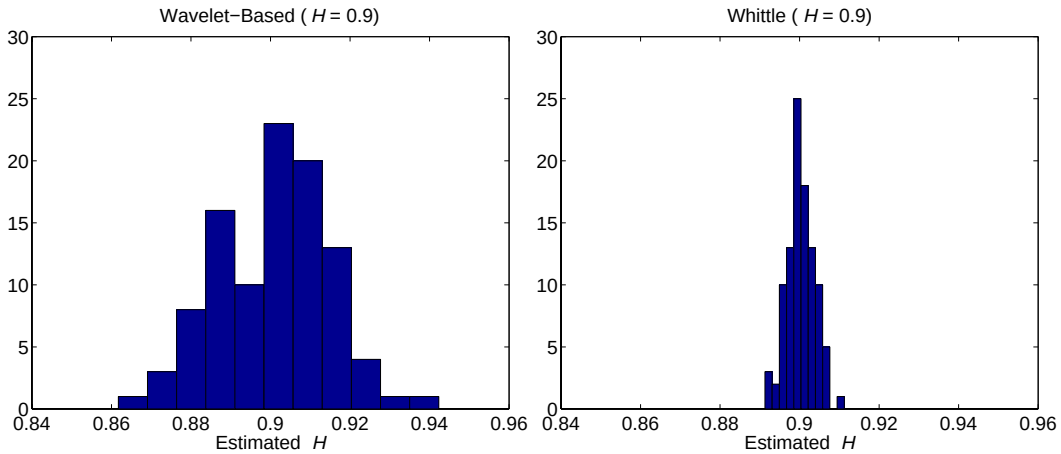


Figure 3.15: Histograms of estimated H values for the exact self-similar FGN process obtained from the wavelet-based H estimator and Whittle's MLE for $H = 0.9$. These are based on 100 replications. $n = 32{,}768$ (2^{15}).

Table 3.11: Sum of the closest estimate of H to the required value obtained from the exact self-similar FGN process using the wavelet-based H estimator and Whittle's MLE for $H = 0.6$, 0.7, 0.8 and 0.9. P-values are given at $p = \frac{1}{2}$ in parentheses. $n = 32{,}768$ (2^{15}).

Estimators	Sum of the closest estimate of H			
	0.6	0.7	0.8	0.9
Wavelet-Based H	14 (1.0)	9 (1.0)	14 (1.0)	11 (1.0)
Whittle's MLE	86 (4.1422e-14)	91 (1.6610e-18)	86 (4.1422e-14)	89 (1.2704e-16)
Total	100	100	100	100

Therefore, we use both the wavelet-based H estimator and Whittle's MLE in a comparative analysis of sequences generated by self-similar generators.

Chapter 4

ALGORITHMIC GENERATORS OF SELF-SIMILAR TELETRAFFIC

4.1 Introduction

Generators of synthetic self-similar sequences can be divided into two practical classes: sequential generators and fixed-length sequence generators.

4.1.1 Sequential Generators: An Overview

It is possible to construct a sequential Markovian model that mimics a self-similar sequence [137], [138]. However, a disadvantage of this method is that the connection between the model's parameters and its self-similar properties is difficult to understand. Markovian models for self-similar traffic are forced to include several control parameters with a wide range of input values, and it is more complicated to control these values in sequential generators than in generators of fixed-length sequences of self-similar processes with a given Hurst parameter. For example, a method based on superposition of two state Markovian processes was proposed by Andersen and Nielsen [7]. They showed

that it can be used to imitate a self-similar process with a certain Hurst parameter, over three to five time scales. However, five control parameters are needed, and the resulted self-similarity can gradually disappear as the time scale increases. Thus, in this respect, their method is unable to adequately model the self-similar counting processes.

Lowen and Teich [99], and Ryu and Lowen [148], [149] proposed four generators (i.e., fractal-binomial-noise-driven Poisson process [FBNDP], fractal renewal process [FRP], superposition of fractal renewal processes [SFRP], and fractal-short-noise-driven Poisson process [FSNDP]). We considered only SFRP and FBNDP because they are more flexible, more accurate and faster at generating self-similar processes than FRP and FSNDP, see [146], [149].

A sequential generator based on the *renewal reward processes*, has been proposed by Mandelbrot [103] and Taqqu and Levy [156], [173]. However, we excluded that generator because it requires $O(nM)$ computations to generate n numbers, where M is an aggregation level. Further, the generator behaves as a fractional Brownian motion when $n \ll M$, for $M \geq 16,000$; see [156] and [173], for detailed discussions.

We considered and investigated the following efficient candidate sequential generators:

- A generator based on the *fractal-binomial-noise-driven Poisson processes* (FBNDP), proposed by Lowen and Teich [99], and Ryu and Lowen [148], [149];
- A generator based on the *superposition of fractal renewal processes* (SFRP), proposed by Lowen and Teich [99], and Ryu and Lowen [148], [149];
- A generator based on the $M/G/\infty$ *processes* (MGIP), proposed by Cox and Isham [18], and Cox [17];
- A generator based on the *Pareto-modulated Poisson processes* (PMPP), proposed by Le-Ngoc and Subramanian [86];
- A generator based on the *spatial renewal processes and fractional Gaussian noise* (SRP-FGN), proposed by Taralp et al. [160]; and
- A generator based on the *superposition of autoregressive processes* (SAP), proposed by Granger [48].

4.1.2 Fixed-Length Sequence Generators: An Overview

The most frequently studied models of self-similar traffic in the discrete-time case belong to the class of *fractional autoregressive integrated moving-average* (F-ARIMA) processes and the class of *fractional Gaussian noise* (FGN) processes because they require the Hurst parameter and variance; see [89], [98], [124]. F-ARIMA (p, d, q) processes were introduced by Hosking [63], where p is the order of autoregression in the ARIMA process, d is the degree of differencing, and q is the order of the moving average. Hosking showed that the F-ARIMA processes are asymptotically self-similar with the Hurst parameter $H = d + \frac{1}{2}$, as long as $0 < d < \frac{1}{2}$.

To describe the FGN process, we first introduce the fractional Brownian motion (FBM) process $\mathbf{B}_H(t), t \geq 0$, which has a Hurst parameter H, $0 < H < 1$. The FBM process is a Gaussian process with zero-mean, stationary increments and the autocovariance function

$$Cov(B_H(t_1), B_H(t_2)) = \frac{1}{2}[t_1^{2H} + t_2^{2H} - (t_1 - t_2)^{2H}]Var[B_H(1)],$$

where t_1 and t_2 are time. This is statistically self-similar in the sense that $\mathbf{B}_H(at), t \geq 0$, has the same finite dimensional distributions as $a^H\mathbf{B}_H(t), t \geq 0$, for all $a > 0$. The FGN process $\mathbf{Y}$ is the incremental process of the FBM process. It is defined by $Y_i = B_H(i+1) - B_H(i), i \geq 1$, and its properties of stationarity, zero mean and variance $E[Y_i^2] = E[B_H^2(1)] = \sigma_0^2$ are derived from the FBM process. The ACF of the FGN process is given by Equation (2.7).

A different approach to generating synthetic self-similar sequences for packet traffic was proposed by Erramilli et al. [30], [31], [90], based on deterministic chaotic maps [16]. Chaos is present in a dynamic system if simple, low order, nonlinear deterministic equations can produce behaviour that mimics random processes. In particular, Erramilli and Singh have shown that a simple, two parameter nonlinear chaotic map, referred to as an intermittent map, can capture many of the fractal properties in actual packet traffic measurements. Clearly, the generation of synthetic traffic via nonlinear chaotic maps makes the dynamic system's approach to packet traffic modelling particularly appealing. After an appropriate chaotic map has been derived from a set of traffic measurements, generating a packet stream for an individual source is generally quick and easy. Deriving an appropriate nonlinear chaotic map based on

a set of actual traffic measurements, however, currently requires considerable guessing and experimenting.

We considered the following fully synthesised fixed-length sequence generators:

- A generator based on the *fractional-autoregressive integrated moving average* (F-ARIMA) process, proposed by Hosking [63];
- A generator based on the *fast Fourier transform* (FFT) algorithm, proposed by Paxson [124];
- A generator based on *fractional Gaussian noise and Daubechies wavelets* (FGN-DW), proposed by Jeong, McNickle and Pawlikowski [73];
- A generator based on the *random midpoint displacement* (RMD) algorithm and implemented by Lau, Erramilli, Wang and Willinger [84]; and
- A generator based on the *successive random addition* (SRA) algorithm, proposed by Saupe [20], in the version implemented by Jeong, McNickle and Pawlikowski [70].

Our comparative evaluation of self-similar pseudo-random teletraffic generators concentrates on two aspects:

(i) how accurately self-similar processes can be generated, and

(ii) how quickly the methods generate long self-similar sequences.

We describe six sequential generators, based on FBNDP, SFRP, MGIP, PMPP, SRP-FGN and SAP in Section 4.2, and five fixed-length generators of self-similar sequences, based on the F-ARIMA, FFT, FGN-DW, RMD and SRA methods, in Section 4.4. Then, in Section 4.3 and 4.5 we concentrate on the least biased estimators, the wavelet-based H estimator and Whittle's MLE, as discussed in Chapter 3, when presenting the numerical results of a comparative analysis of the generated sequences. In Section 4.6, the fastest and most accurate sequential generator is compared with the most accurate fixed-length sequence generator; finally, conclusions are presented.

4.2 Sequential Generators

4.2.1 Method Based on Fractal-Binomial-Noise-Driven Poisson Process

For the standard fractal renewal process (FRP), inter-event times are independent random variables. The marginal probability density function (PDF) of such a fractal renewal process assumes the form

$$f(t) = \begin{cases} 0, & t \leq A, \\ \delta A^{\delta} t^{-(\delta+1)}, & t > A, \end{cases} \tag{4.1}$$

where $0 < \delta < 2$ [146].

However, the resulting IDC(t) (see Section 3.4.6) has a dip near $t = t_0$, caused by the abrupt cutoff in the inter-event time PDF. The time instant t_0, which marks the lower limit for significant scaling behaviour in the IDC(t) and ACF, is also known as the fractal onset time. Furthermore, the power spectral density exhibits excessive oscillations for the same reason.

Selecting δ in the range $1 < \delta < 2$ proves far superior to $0 < \delta < 1$ for the same required value of α, but the form of the inter-event time PDF in Equation (4.1) can be further improved. The improved PDF of the FRP decays as a power law given by

$$f(t) = \begin{cases} \delta A^{-1} e^{-\delta t/A}, & 0 < t \leq A, \\ \delta e^{-\delta} A t^{-(\delta+1)}, & t > A, \end{cases} \tag{4.2}$$

which is continuous for all t and it produces smoother spectral density function than Equation (4.1).

The FRP is recast as a process with real-values that alternates between two values, zero and $R, R > 0$ [149]. This alternating FRP starts at a value of zero ("OFF"), and then switches to a value of R ("ON") at a time corresponding to the first event in the FRP. At the second such event, the alternating FRP switches back to zero, and proceeds to switch back and forth at every successive event in the FRP. Thus, all ON/OFF periods are IID with the same heavy-tailed distribution as in the FRP.

A method based on the fractal-binomial-noise-driven Poisson process (FBNDP) adds M IID alternating FRPs to generate a fractal binomial noise pro-

cess that serves as the rate function for a Poisson process. The FBNDP requires five input parameters to generate self-similar sequences: $A, \delta, R,\ \Delta t$ and M. The resulted Hurst parameter H assumes the value $(\alpha + 1)/2$. The algorithm advances by the intervals Δt.

If S is a simulation clock, which advances in time and $S^{(j)}$ is the elapsed time of the j-th FRP sequence, then $S^{(j)} = \tau_0^{(j)} + \tau_1^{(j)} + \cdots + \tau_k^{(j)}$ for some k and $j = 1, 2, \ldots, M$, where $\tau_k^{(j)}$ is the inter-arrival time. The sequence of self-similar pseudo-random numbers $X_0, X_1, \ldots$ is generated by the following steps:

Step 1. For each $j = 1, 2, \ldots, M$, generate $\tau_0^{(j)}$ from

$$\tau_0^{(j)} = \begin{cases} -\delta^{-1} A \log[U(\delta^{(V-1)})(\delta^{(V-U)})^{-1}], & V \geq 1, \\ A V^{1/(1-\delta)}, & V < 1, \end{cases} \tag{4.3}$$

where

$$V \equiv \frac{1 + (\delta - 1)e^{\delta}}{\delta} U, \tag{4.4}$$

and U is an IID uniformly distributed random variable over the unit interval [0,1); set $S^{(j)} = \tau_0^{(j)}$.

Step 2. Find j^* and $S^{(j^*)}$ such that $j^* = \text{argmin}_j\{S^{(j)}\}$.

Step 3. Calculate

$$x = \begin{cases} 0, & \text{if } S^{(j^*)} < A, \\ 1, & \text{if } S^{(j^*)} \geq A. \end{cases} \tag{4.5}$$

Step 4. If $x = 1$, then X_0 should be drawn from a Poisson probability distribution with $\lambda = 1$. If $x = 0$, then $X_0 = 0$.

Step 5. Set $i = 1$, and $y = 0$. Advance the simulation clock, i.e., $S \longleftarrow S^{(j^*)}$.

Step 6. Construct a new inter-event time $\tau_i^{(j)}$ from

$$\tau_i^{(j)} = \begin{cases} -\delta^{-1} A \log[U], & U \geq e^{-\delta}, \\ e^{-1} A U^{-1/\delta}, & U < e^{-\delta}, \end{cases} \tag{4.6}$$

and set $S^{(j^*)} \longleftarrow S^{(j^*)} + \tau_i^{(j)}$.

Step 7. Find a new j^* such that $j^* = \text{argmin}_j\{S^{(j)}\}$, and compute $S^{(j^*)} - S$.

Step 8. Repeat Step 6 through Step 8 to obtain x as in Step 3.

Step 9. Advance the simulation clock, i.e., $S \longleftarrow S^{(j^*)}$, and set $y = y + x$.

Step 10. Repeat Step 6 through Step 10 within time slot of length Δt.

Step 11. Compute $X_i = POISS(y)$, set $y = 0$, and $i = i + 1$.

Step 12. Repeat Step 6 through Step 11 until $i = n$, where n is the number of sample points.

An approximate self-similar sequence $\{X_0, X_1, X_2, \ldots\}$ is obtained from these steps. Tables 4.1 and 4.2 show mean values of the estimated H obtained using the wavelet-based H estimator and Whittle's MLE for the FBNDP method with four input values: $A = 9.92$, $R = 200$, $M = 4$ to 14, and $H = 0.6$ to 0.9. Our results show that the appropriate aggregate level M is between 4 and 10. These results show that no aggregation level in this range of M is consistently better than others. Without studying whether the marginal probability distributions of these mixtures of Poisson processes are close enough to normal distributions, we chose the aggregate level of $M = 10$, when comparing this generator with others in Section 4.3. Thus, the problem of selection of M for securing normality of marginal distributions of the output processes from such a generator remains an open problem.

Note that, for small input values of λ of Poisson processes, only the Poisson approximation can be used, but for large input values of λ we can use either the normal or the Poisson approximation. This implies that for large values of λ it must be possible to approximate the Poisson distribution by the normal distribution; see [38] (p.190) for details.

Generation of a sample sequence of 1,048,576 numbers (so, about $524 * 10^6$ inter-event times) took 9 minutes 38 seconds on a Pentium II (233 MHz, 512 MB). The FBNDP method requires $O(n)$ computations to generate n numbers. For a more detailed discussion, see [146].

Table 4.1: Mean values of estimated H obtained using the wavelet-based H estimator for the FBNDP method. We give 95% confidence intervals for the means in parentheses.

Aggregation Level M	Mean Values of Estimated H			
	.6	.7	.8	.9
4	.6023 (.575, .630)	.6864 (.659, .714)	.7817 (.754, .809)	.8469 (.819, .874)
6	.6085 (.581, .636)	.6871 (.660, .715)	.7804 (.753, .808)	.8496 (.822, .877)
8	.6080 (.581, .636)	.6870 (.660, .715)	.7803 (.753, .808)	.8489 (.821, .876)
10	.6086 (.581, .636)	.6875 (.660, .715)	.7827 (.755, .810)	.8502 (.823, .878)
12	.6053 (.578, .633)	.6903 (.663, .718)	.7832 (.756, .811)	.8501 (.823, .878)
14	.6049 (.577, .632)	.6895 (.662, .717)	.7842 (.757, .812)	.8497 (.822, .877)

Table 4.2: Mean values of estimated H obtained using Whittle's MLE for the FBNDP method. We give 95% confidence intervals for the means in parentheses.

Aggregation Level M	Mean Values of Estimated H			
	.6	.7	.8	.9
4	.6122 (.603, .622)	.6828 (.674, .692)	.7557 (.747, .765)	.8164 (.807, .826)
6	.6131 (.604, .623)	.6823 (.673, .692)	.7547 (.746, .764)	.8163 (.807, .825)
8	.6140 (.605, .623)	.6823 (.673, .692)	.7547 (.745, .764)	.8145 (.805, .824)
10	.6145 (.605, .624)	.6829 (.674, .692)	.7553 (.746, .764)	.8142 (.805, .823)
12	.6137 (.604, .623)	.6830 (.674, .692)	.7554 (.746, .765)	.8132 (.804, .822)
14	.6145 (.605, .624)	.6836 (.674, .693)	.7555 (.746, .765)	.8123 (.803, .821)

4.2.2 Method Based on Superposition of Fractal Renewal Processes

The fractal renewal process (FRP) was described in Section 4.2.1. This self-similar process results from the superposition of a number of independent and identical FRPs. We now consider a method based on the *superposition of fractal renewal processes* (SFRP), proposed by Lowen and Teich [99] and Ryu and Lowen [148], [149]. This method is defined as the superposition of M independent and identical FRPs. The method is characterised by M and the common inter-event PDF in Equation (4.2). This method requires three parameters, i.e., α and A from the individual FRPs, and M, the number of FRPs superposed. The resulted Hurst parameter H, and mean μ and variance σ^2 of the marginal output distribution of a related count process in the unit time interval, are given by

$$\begin{aligned} H &= (\alpha+1)/2, \\ \mu &= E[X_n] = \lambda, \\ \sigma^2 &= Var[X_n] = (1+(1/t_0))^{\alpha}\lambda, \end{aligned} \tag{4.7}$$

where

$$\lambda = M\delta[1+(\delta-1)^{-1}e^{-\delta}]^{-1}A^{-1}$$

is the aggregated arrival rate of events in the unit time interval, and

$$t_0 = (2^{-1}\delta^{-2}(\delta-1)^{-1}(2-\delta)(3-\delta)e^{-\delta}[1+(\delta-1)e^{\delta}]^2A^{\alpha})^{1/\alpha},$$

that is the value of time at which the resulting IDC(t) has a dip; see also page 71.

If S is a simulation clock, which advances in time and $S^{(j)}$ is the elapsed time of the j-th FRP sequence, then $S^{(j)} = \tau_0^{(j)} + \tau_1^{(j)} + \cdots + \tau_k^{(j)}$ for some k and $j = 1, 2, \ldots, M$. The inter-event times X_i are generated by the following steps:

Step 1. For each $j = 1, 2, \ldots, M$, and $i = 0$, generate $\tau_0^{(j)}$ from Equation (4.3) and Equation (4.4) in the FBNDP; set $S^{(j)} = \tau_0^{(j)}$.

Step 2. Find j^* such that $j^* = \text{argmin}_j\{S^{(j)}\}$, and set $X_0 = S^{(j^*)}$.

Step 3. Advance the simulation clock, i.e., $S \longleftarrow S^{(j^*)}$.

Step 4. Set $i = i + 1$. Construct a new inter-event time $\tau_i^{(j)}$ from Equation (4.6) in the FBNDP and set $S^{(j^*)} \longleftarrow S^{(j^*)} + \tau_i^{(j)}$.

Step 5. Find a new j^* such that $j^* = \text{argmin}_j\{S^{(j)}\}$, and compute $X_i = S^{(j^*)} - S$.

Step 6. Advance the simulation clock, i.e., $S \longleftarrow S^{(j^*)}$.

Step 7. Repeat Step 4 through Step 6 until a given $i = n$ is reached, where n is the number of sample points.

Using the previous steps, this method generates an approximate self-similar sequence $\{X_1, X_2, \ldots\}$. Tables 4.3 and 4.4 show that this method produces the most accurate result when the aggregation level M is between 4 and 10. It took 22 minutes, 44 seconds to generate a sequence of 1,048,576 numbers (so, about $1,362 * 10^6$ inter-event times) on a Pentium II (233 MHz, 512 MB). The results were obtained assuming $M = 10$ and $A = 3.8$. The SFRP method requires $O(n)$ computations to generate n numbers. For a more detailed discussion, see [146].

4.2.3 Method Based on $M/G/\infty$ Processes

An $M/G/\infty$ is a queueing system in which a server is available immediately for every arriving customer, regardless of how many customers are already being served. Applying this process to generate LRD count sequences, we assume that new arrivals can enter $M/G/\infty$ only at the beginning of time slot of length Δt. Let us call this method as MGIP. The method is based on simulation of customers that arrive at an infinite-server queueing system according to a Poisson process with an arrival rate λ. This method generates asymptotically self-similar sequences obtained from counting the number of customers from unlimited servers in the system, where the service time distribution G satisfies the heavy-tailed condition [17], [30], [121], [126]. Cox [17] showed that an infinite variance service time distribution results in an asymptotically self-similar process. Likhanov et al. [95] proposed a model for aggregate packet streams, based on combining sequences generated by several ON/OFF sources with a Pareto-distributed ON period. They showed that increasing the number

Table 4.3: Mean values of estimated H obtained using the wavelet-based H estimator for the SFRP method. We give 95% confidence intervals for the means in parentheses.

Aggregation Level M	Mean Values of Estimated H			
	.6	.7	.8	.9
4	.6076 (.580, .635)	.6986 (.671, .726)	.7929 (.765, .820)	.8603 (.833, .888)
6	.6076 (.580, .635)	.6986 (.671, .726)	.7929 (.765, .820)	.8603 (.833, .888)
8	.6192 (.592, .647)	.7084 (.681, .736)	.7997 (.772, .827)	.8655 (.838, .893)
10	.6166 (.589, .644)	.7091 (.682, .737)	.7986 (.771, .826)	.8686 (.841, .896)
12	.6134 (.586, .641)	.7085 (.681, .736)	.7978 (.770, .825)	.8718 (.844, .899)
14	.6143 (.587, .642)	.7063 (.679, .734)	.7983 (.771, .826)	.8708 (.843, .898)

Table 4.4: Mean values of estimated H obtained using Whittle's MLE for the SFRP method. We give 95% confidence intervals for the means in parentheses.

Aggregation Level M	Mean Values of Estimated H			
	.6	.7	.8	.9
4	.6372 (.628, .647)	.7124 (.703, .722)	.7886 (.779, .798)	.8517 (.843, .861)
6	.6377 (.628, .647)	.7132 (.704, .722)	.7895 (.780, .799)	.8501 (.841, .859)
8	.6382 (.629, .648)	.7136 (.704, .723)	.7889 (.780, .798)	.8505 (.841, .860)
10	.6383 (.629, .648)	.7132 (.704, .722)	.7886 (.779, .798)	.8495 (.840, .859)
12	.6380 (.629, .647)	.7128 (.704, .722)	.7885 (.779, .798)	.8497 (.841, .859)
14	.6372 (.628, .647)	.7127 (.703, .722)	.7891 (.780, .798)	.8485 (.839, .858)

of sources yields a limiting behaviour identical to the $M/G/\infty$ input sequence with a Pareto distribution. To implement their findings we need to assume a given coefficient utilisation of the queueing system $\rho, 0 < \rho < 1$, and a Pareto distribution of service times with finite mean service times and infinite variance of service times, i.e., with the shape parameter $\alpha, 1 < \alpha < 2$. The simulation will be advanced each time by Δt seconds. Then, the MGIP method consists of the following steps:

Step 1. Given ρ, α, Δt, $i = 1$.

Step 2. Simulate performance of an $M/G/\infty$ queueing system over Δt seconds. This means, generate pseudo-random numbers representing the number of Poisson arrivals to the $M/G/\infty$ queueing system within Δt seconds, and pseudo-random numbers from the Pareto distribution representing service times of these customers. Assume arrival rate $\lambda = \rho(\alpha - 1)/\alpha$, where ρ is traffic intensity and α is the shape parameter of the Pareto distribution, and service rate $(\alpha - 1)/\alpha$.

Step 3. Count the number of customers in the simulated $M/G/\infty$ queueing system at the end of this time slot of length Δt. This is X_i, the ith number of the output LRD self-similar sequence in the scale Δt. LRD sequences in larger time scales, say s, can be obtained by counting number of customers in the system at the end of each s time slots, i.e., by assuming a time lag equal to Δt.

Step 4. Set $i = i + 1$. Repeat Step 2 to Step 4, advancing the simulated time by the next Δt seconds, until $i \leq n$, where n is the number of sample points. Otherwise, stop.

A self-similar sequence $\{X_1, X_2, \ldots\}$ is obtained from these steps. Tables 4.5 and 4.6 show mean values of estimated H obtained using the wavelet-based H estimator and Whittle's MLE for the MGIP method with input traffic intensity $\rho = 0.9$ and service rate $\mu = (\alpha - 1)/\alpha$, where $\alpha = 3 - 2H$, for $H = 0.6$, 0.7, 0.8 and 0.9, and time lag $s = 4$ to 14. Our results show that this method is most efficient at time lag s between 4 and 8. Generation of an asymptotic self-similar sequence with 1,048,576 numbers with these time lags took 27 seconds on a Pentium II (233 MHz, 512 MB). $O(n)$ computations are required to generate a self-similar sequence.

Table 4.5: Mean values of estimated H obtained using the wavelet-based H estimator for the MGIP method. We give 95% confidence intervals for the means in parentheses.

Time Lag s	Mean Values of Estimated H			
	.6	.7	.8	.9
4	.5754 (.548, .603)	.6717 (.644, .699)	.8453 (.818, .873)	1.1180 (1.091, 1.146)
6	.5538 (.526, .581)	.6451 (.618, .673)	.8267 (.799, .854)	1.1150 (1.087, 1.142)
8	.5445 (.517, .572)	.6241 (.597, .652)	.8194 (.792, .847)	1.1120 (1.084, 1.139)
10	.5351 (.508, .563)	.6155 (.588, .643)	.8168 (.789, .844)	1.1130 (1.086, 1.141)
12	.5307 (.503, .558)	.6073 (.580, .635)	.8098 (.782, .837)	1.1100 (1.083, 1.138)
14	.5323 (.505, .560)	.6024 (.575, .630)	.8045 (.777, .832)	1.1080 (1.081, 1.136)

Table 4.6: Mean values of estimated H obtained using Whittle's MLE for the MGIP method. We give 95% confidence intervals for the means in parentheses.

Time Lag s	Mean Values of Estimated H			
	.6	.7	.8	.9
4	.5520 (.542, .562)	.6325 (.623, .642)	.7742 (.765, .783)	.9499 (.941, .959)
6	.5426 (.533, .552)	.6102 (.601, .620)	.7463 (.737, .756)	.9499 (.941, .959)
8	.5373 (.528, .547)	.5948 (.585, .604)	.7300 (.721, .739)	.9499 (.941, .959)
10	.5318 (.522, .541)	.5862 (.577, .596)	.7191 (.710, .728)	.9500 (.941, .959)
12	.5282 (.519, .538)	.5780 (.568, .588)	.7110 (.702, .720)	.9499 (.941, .959)
14	.5255 (.516, .535)	.5720 (.562, .582)	.7033 (.694, .713)	.9499 (.941, .959)

4.2.4 Method Based on Pareto-Modulated Poisson Processes

This method is based on the fact that a Pareto-modulated Poisson process (PMPP), based on a switched Poisson process with two states, with sojourn times governed by an independent and identical Pareto distribution, asymptotically generates a self-similar sequence [86]. Figure 4.1 shows a state diagram of the PMPP. The two states of the switched Poisson process can be viewed as intervals with the long and short burst rates of events. This process goes through consecutive cycles of being in State 1 and State 2. The time spent in each cycle is governed by a Pareto distribution characterised by $\alpha, 1 < \alpha < 2$. These cycles have the mean length (ML) equal

$$\text{ML} = \frac{\alpha}{\alpha - 1} + \frac{\alpha}{\alpha - 1} = \frac{2\alpha}{\alpha - 1} \text{ time units.} \tag{4.8}$$

Mean numbers of Poisson events (MNPE) generated in state S_1 and state S_2 are:

$$\begin{aligned} \text{MNPE in state} S_1 &= \lambda_1 \frac{\alpha}{\alpha - 1}, \text{ and} \\ \text{MNPE in state} S_2 &= \lambda_2 \frac{\alpha}{\alpha - 1}. \end{aligned} \tag{4.9}$$

Thus, mean number of Poisson events per cycle is given by

$$\lambda_1 \frac{\alpha}{\alpha - 1} + \lambda_2 \frac{\alpha}{\alpha - 1} = (\lambda_1 + \lambda_2) \frac{\alpha}{\alpha - 1}, \tag{4.10}$$

and using Equations (4.8) and (4.10), we get the mean number of Poisson events per time unit as

$$\bar{E} = \frac{(\lambda_1 + \lambda_2)}{2}. \tag{4.11}$$

As mentioned the PMPP can be used to generate asymptotically self-similar sequences. The quality generated sequences, in the sense of the closeness to exactly self-similar processes, depends on the size of frames within which one counts numbers of Poisson events occurring in underlining PMPP, see Figure 4.2. If frames have length T seconds, then the mean number of Poisson events occurring in a frame can be called the aggregation level of that method, given as

$$\bar{N} = \bar{E} * T, \tag{4.12}$$

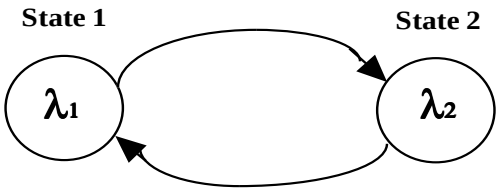


Figure 4.1: State diagram of the Pareto-modulated Poisson process [86]. It is a two-state switched Poisson process with the sojourn time in each state following an independent and identical Pareto distribution.

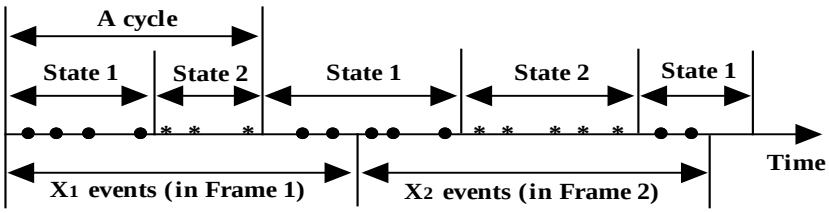


Figure 4.2: Graphical explanation of the concept of data aggregation in the generator based on PMPP. Note that $\bullet$ is an event in Poisson process with rate λ_1, and $*$ is an event in Poisson process with rate λ_2. X_i is the number of Poisson events occurring in Frame i.

where $\bar{E}$ is the number of events per second. There should exist the minimum acceptable aggregation level $\bar{N}_{min}$ below which this method would not generate satisfactory self-similar sequences, but we leave this issue for further research. In our investigations we assume $\lambda_1 = 100$, $\lambda_2 = 120$, and $T \geq 100$. Thus, we assume $\bar{N}_{min} \geq 11{,}000$.

The PMPP generator follows of the following steps: For a given $\lambda_1, \lambda_2, \alpha, T$ and n, let S be the time advance.

Step 1. Set $i = 1, k = 1$, $S = 0$, $X_i^{(0)} = 0$.

Step 2. Generate the sojourn times τ_k and τ_{k+1} for state $S_{k \bmod 2}$ and $S_{(k+1) \bmod 2}$ of PMPP using the Pareto distribution with shape parameter $\alpha, 1 < \alpha < 2$, see Equation (2.21).

Produce a sequence of Poisson arrivals within each state S_j with rate λ_j, $j = 1$, or 2.

Calculate $S = S + \tau_k + \tau_{k+1}$ if $S \geq iT$ then go to Step 3, otherwise assume $k = k + 1$ and repeat Step 2.

Step 3. Count $X_i^{(1)}$, the number of events that occur in the last frame of the length of T seconds.

$$X_i = X_i^{(0)} + X_i^{(1)}. \tag{4.13}$$

Count the number of events that occur in the remaining time interval (iT, S). This is $X_{i+1}^{(0)}$, the initial component of X_{i+1}. If $i < n$ then assume $i = i+1$, go to Step 2. Otherwise if $i = n$, then stop (the required number of pseudo-random self-similar numbers has been generated).

Tables 4.7 and 4.8 show mean values of estimated H obtained using the wavelet-based H estimator and Whittle's MLE in sequences generated by using the PMPP method. Our results were obtained for frames between 100 and 600. We compare this generator with others in Section 4.3, assuming $T = 300$. Generating a sample sequence of 1,048,576 numbers took 7 minutes, 11 seconds on a Pentium II (233 MHz, 512 MB). The PMPP method requires $O(n)$ computations to generate n numbers. For a more detailed discussion, see [86].

4.2.5 Method Based on Spatial Renewal Processes and Fractional Gaussian Noise

The SRP-FGN generator is a hybrid method that uses a fractional Gaussian noise (FGN) generator based on the spatial renewal process (SRP) developed by Taralp et al. [160]. Before discussing SRP, we first introduce the concept of sub-exponentiality. It means that the ACF of a stationary process decays not exponentially, but hyperbolically, for large lags [41], [80]. For example, Jelenković [67] observed distinctive sub-exponential characteristics of MPEG video traffic in the functional behaviour of its scene length distributions.

The SRP belongs to a class of sub-exponentially time-dependent stochastic processes. The SRP $\mathbf{Z}$ is composed of a chain of mutually independent renewal periods. For practical reasons, it is assumed that the SRP is a discrete time process and its ith period T_i has length k_i, where k_i is an integer, and the

Table 4.7: Mean values of estimated H obtained using the wavelet-based H estimator for the PMPP method. We give 95% confidence intervals for the means in parentheses.

Frame Length T	Mean Values of Estimated H			
	.6	.7	.8	.9
100	.6180 (.590, .645)	.7007 (.673, .728)	.7864 (.759, .814)	.8486 (.821, .876)
200	.6138 (.586, .641)	.6960 (.669, .724)	.7834 (.756, .811)	.8603 (.833, .888)
300	.6126 (.585, .640)	.6931 (.666, .721)	.7865 (.759, .814)	.8617 (.834, .889)
400	.5972 (.570, .625)	.6970 (.669, .724)	.7867 (.759, .814)	.8633 (.836, .891)
500	.5924 (.565, .620)	.6845 (.657, .712)	.7902 (.763, .818)	.8661 (.839, .894)
600	.5947 (.567, .622)	.6868 (.659, .714)	.7817 (.754, .809)	.8638 (.836, .891)

Table 4.8: Mean values of estimated H obtained using Whittle's MLE for the PMPP method. We give 95% confidence intervals for the means in parentheses.

Frame Length T	Mean Values of Estimated H			
	.6	.7	.8	.9
100	.6702 (.661, .680)	.7310 (.722, .740)	.7901 (.781, .799)	.8381 (.829, .847)
200	.6587 (.649, .668)	.7244 (.715, .734)	.7891 (.780, .798)	.8421 (.833, .851)
300	.6541 (.645, .664)	.7195 (.710, .729)	.7904 (.781, .800)	.8450 (.836, .854)
400	.6462 (.637, .656)	.7185 (.709, .728)	.7886 (.779, .798)	.8470 (.838, .856)
500	.6431 (.634, .652)	.7157 (.706, .725)	.7898 (.781, .799)	.8497 (.841, .859)
600	.6437 (.634, .653)	.7146 (.705, .724)	.7896 (.780, .799)	.8505 (.841, .860)

sample of $\mathbf{Z}$ during the period is represented by a sequence of k_i numbers $Y_1, Y_2, \ldots, Y_{k_i}$, governed by the normal distribution. The consecutive number of the output self-similar time series $X_i \sum_{j=1}^{k_i} Y_j$.

To improve statistical properties of the output sequence (it fit to normal distribution and the required correlation function), it has been proposed to aggregate a number of such sequences [160]. We investigated this suggestion experimentally by considering various levels of aggregation. Tables 4.9 and 4.10 show the mean values of estimated H obtained using the wavelet-based H estimator and Whittle's MLE for the SRP-FGN method. Our results also show that the SRP-FGN method is most accurate if the level of aggregation $M = 10$, supporting the advice of Taralp et al. [160], who wrote that the aggregation level needs not be high (i.e., ≈ 10) to obtain accurate results.

The aggregate output sequence $\{X_1, X_2, \ldots\}$ is computed by after having summed the sequences and normalisation of the sample variance to one. The SRP-FGN model has a normal marginal distribution, and is characterised by a sub-exponential ACF. Figure 4.3 shows a block diagram of the SRP-FGN generator, which consists of the following steps:

Step 1. Given H, n, M, $i = 0$, $m = 0$, $s = 0$, $l = 1$, $X_0 = 0$.

Step 2. Generate a random length k_i of the renewal cycle T_i governed by the following a cumulative probability distribution function $F_T(k)$.

$$F_T(k) = \begin{cases} 0, & 0 \le k < 1, \\ 1 - \frac{H\{(k+1)^{2H-1} - 2k^{2H-1} + (k-1)^{2H-1}\}}{(2^{2H-1} - 2)}, & 1 \le k. \end{cases} \tag{4.14}$$

Step 3. The SRP $\mathbf{Z}$ is composed of a chain of renewal periods where the ith period T_i is k_i in length. Generate k_i random numbers $Y_1, Y_2, \ldots, Y_{k_i}$ governed by the normal distribution. Set $m = m + k_i$.
If $m < M$, then $X_i = X_i + \sum_{j=1}^{k_i} Y_j$, set $s = s + m$, and go to Step 2.
If $m \ge M$, then set $m = m - M$, and go to Step 4.

Step 4. The output value X_i is computed as follows:
If $s = 0$, then $X_i = \sum_{j=lM-M+1}^{lM} Y_j$, and set $s = 0$.
If $s > 0$, then $X_i = X_i + \sum_{j=1}^{M-s} Y_j$, and set $s = 0$.

Step 5. Set $i = i + 1$, and $X_i = 0$. If $i < n$ and if $m = 0$, then go to Step 2. Otherwise,

Table 4.9: Mean values of estimated H obtained using the wavelet-based H estimator for the SRP-FGN method. We give 95% confidence intervals for the means in parentheses.

Aggregation Level M	Mean Values of Estimated H			
	.6	.7	.8	.9
4	.6024 (.575, .630)	.7106 (.683, .738)	.8111 (.784, .839)	.9162 (.889, .944)
6	.5881 (.561, .616)	.7074 (.680, .735)	.8103 (.783, .838)	.9062 (.879, .934)
8	.5935 (.566, .621)	.6972 (.700, .725)	.8148 (.787, .842)	.9036 (.876, .931)
10	.5942 (.567, .622)	.6960 (.669, .724)	.8056 (.778, .833)	.9031 (.876, .931)
12	.5925 (.565, .620)	.6976 (.670, .725)	.8037 (.776, .831)	.9051 (.878, .933)
14	.5947 (.567, .622)	.6966 (.669, .724)	.7956 (.768, .823)	.9079 (.880, .935)

Table 4.10: Mean values of estimated H obtained using Whittle's MLE for the SRP-FGN method. We give 95% confidence intervals for the means in parentheses.

Aggregation Level M	Mean Values of Estimated H			
	.6	.7	.8	.9
4	.5949 (.585, .604)	.7058 (.697, .715)	.8165 (.807, .826)	.9228 (.914, .932)
6	.5967 (.587, .606)	.7105 (.701, .720)	.8223 (.813, .831)	.9249 (.916, .934)
8	.5990 (.589, .608)	.7129 (.704, .722)	.8215 (.812, .831)	.9240 (.915, .933)
10	.6007 (.591, .610)	.7133 (.704, .723)	.8203 (.811, .829)	.9227 (.914, .932)
12	.6016 (.592, .611)	.7159 (.707, .725)	.8197 (.811, .829)	.9225 (.913, .932)
14	.6031 (.594, .613)	.7152 (.706, .724)	.8203 (.811, .829)	.9199 (.911, .929)

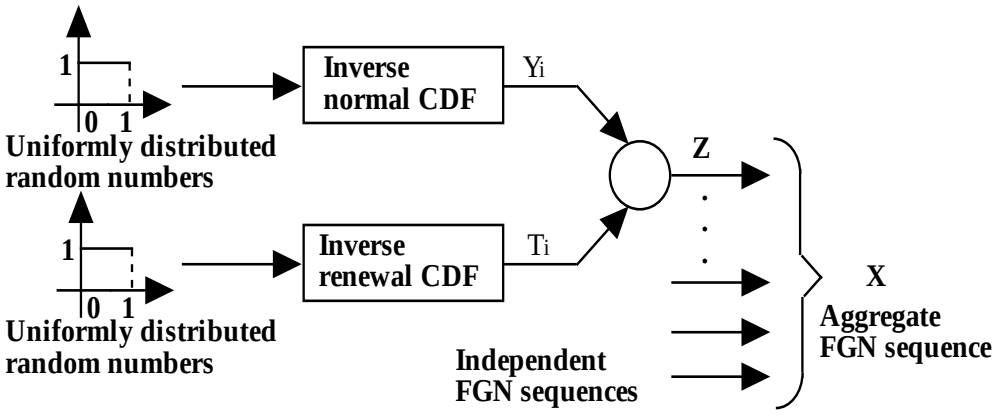


Figure 4.3: Block diagram of the SRP-FGN method [160].

- if $m < M$, then $X_i = \sum_{j=k_i-m+1}^{k_i} Y_j$, set $s = m$, $l = 1$, and go to Step 2.
- if $m \geq M$, then set $m = m - M$, $s = 0$, $l = l + 1$, and go to Step 4.

If $i = n$, where n is the number of sample points, then stop.

This generator produces approximately self-similar sequences $\{X_0, X_1, X_2, \ldots\}$. We obtained the points of the inverse renewal CDF using Equation(4.14). In order to obtain more accurate results of the tail behaviour, we chose a number of intervals, $T = 10,000$, for the renewal CDF. The SRP-FGN renewal CDF $F_T(i)$ and complementary CDF are plotted in Figures 4.4 and 4.5. $F_T(i)$ gradually has longer tails as the H value increases. The SRP-FGN method generates sample sequences $\{X_1, X_2, \ldots, X_n\}$ with $O(n)$. It took 26 seconds to generate a sequence of 1,048,576 numbers on a Pentium II (233 MHz, 512 MB).

4.2.6 Method Based on Superposition of Autoregressive Processes

The method based on the superposition of autoregressive processes (SAP) proposed by Granger [48] generates asymptotically self-similar sequences when aggregating several independent autoregressive processes. In the simplest case

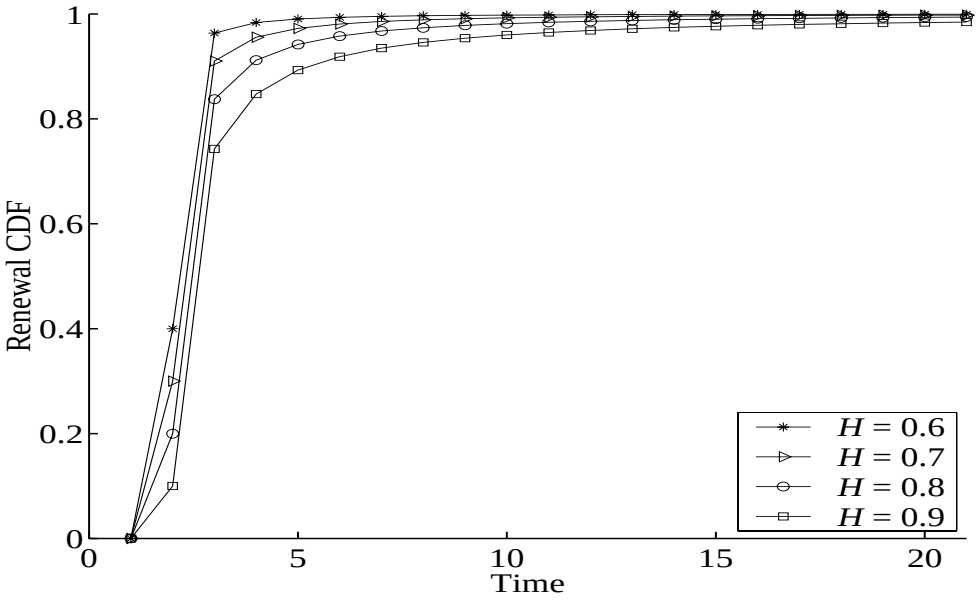


Figure 4.4: Cumulative distribution function $F_T(i)$ of the SRP-FGN method for $H = 0.6$, 0.7, 0.8 and 0.9.

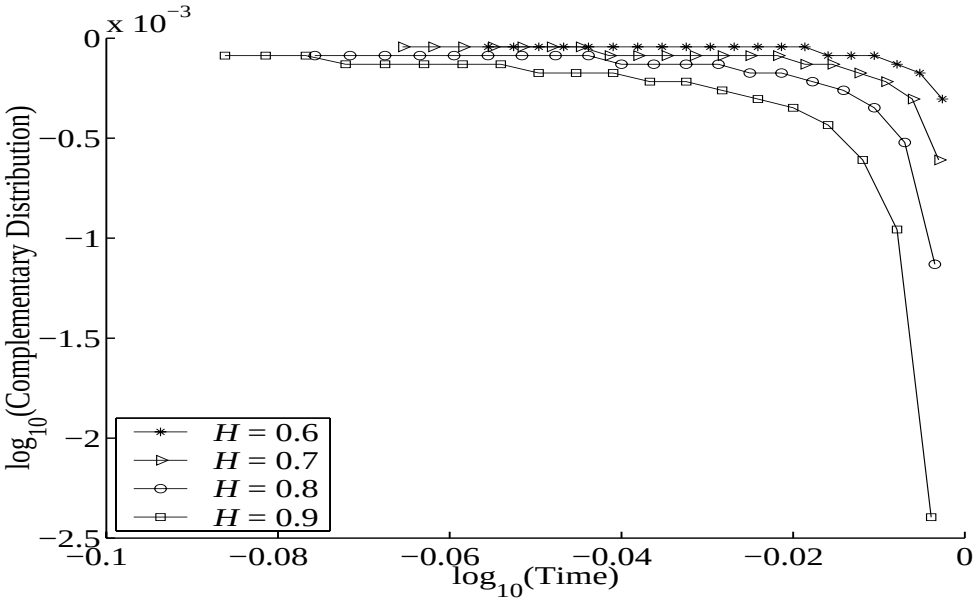


Figure 4.5: Complementary cumulative distribution function of the SRP-FGN method for $H = 0.6$, 0.7, 0.8 and 0.9.

this can be the sum of two autoregressive processes of the first order:

$$\begin{aligned} z_{1i} &= A_{1i} z_{1,i-1} + y_{1i}, \\ z_{2i} &= A_{2i} z_{2,i-1} + y_{2i}, \quad i = 1, 2, \ldots \end{aligned} \tag{4.15}$$

where A_{1i} and A_{2i} are randomly chosen from a beta distribution $B(\alpha_1, \alpha_2)$ on $[0, 1]$ with shape parameters α_1 and α_2, where $\alpha_1 > 0, \alpha_2 > 0$. y_{1i} and y_{2i} are a pair of IID sequences of random variables with a mean of zero and variance $\sigma^2 = 1$.

As shown in [48], using the least-square fitting it can be found that $\alpha_2 = 7.7929 * log(H) + 4.9513$. Thus, the Hurst parameter H is linearly dependent on the shape parameter α_2 of the beta-distribution, while α_1 can be selected arbitrary, for example, $\alpha_1 = 1$ in all cases that we investigated.

The PDF $f(x)$ of the beta distribution is given by

$$f(x) = \begin{cases} \frac{x^{\alpha_1 - 1}(1-x)^{\alpha_2 - 1}}{\beta(\alpha_1, \alpha_2)}, & 0 < x < 1, \\ 0, & \text{otherwise}, \end{cases} \tag{4.16}$$

where $\beta(\alpha_1, \alpha_2)$ is defined by

$$\beta(\alpha_1, \alpha_2) = \int_0^1 x^{\alpha_1 - 1}(1-x)^{\alpha_2 - 1} dx = \frac{\Gamma(\alpha_1)\Gamma(\alpha_2)}{\Gamma(\alpha_1 + \alpha_2)}.$$

This method, as based on the superposition of the autoregressive processes, consists of the following steps:

Given: α_1, α_2, $i = 0$.

Step 1. Set $i = i + 1$. Determine z_{1i} and z_{2i} using Equation (4.15).

Step 2. Calculate the sum,

$$X_i = z_{1i} + z_{2i}, \quad i = 1, 2, \ldots.$$

Step 3. Repeat Step 1 and Step 2 until $i = n$, where n is the number of sample points.

Using the previous steps, the method based on the superposition of the autoregressive process generates an asymptotically self-similar sequence $\{X_1, X_2, \ldots\}$. The CPU time required to generate 1,048,576 numbers was 35 seconds

on a Pentium II (233 MHz, 512 MB). Unlike the other sequential generators, the SAP generator does not require an aggregation level to be assumed as an input parameter, but such input parameters as the shape parameter α_1, instead.

4.3 Comparison of Sequential Generators

All six sequential generators based on FBNDP, SFRP, MGIP, PMPP, SRP-FGN and SAP, generate approximately self-similar sequences. We investigate their properties in greater detail in this section. All have been implemented in C on a Pentium II (233 MHz, 512 MB) computer. The mean times required for generating sequences of a given length were obtained using the SunOS 5.7 `time` command and were averaged over 30 replications, each with sequences of 32,768 (2^{15}), 65,536 (2^{16}), 131,072 (2^{17}), 262,144 (2^{18}), 524,288 (2^{19}) and 1,048,576 (2^{20}) numbers.

We have analysed the accuracy of the six methods. For each of $H = 0.5$, 0.6, 0.7, 0.8 and 0.9, sample sequences generated were analysed as follows. The FBNDP process was analysed with input $M = 10$, $R = 200$, and cutoff parameter $A = 9.92$; the SFRP method required the following three parameters: $M = 10$, $H = 0.6$, 0.7, 0.8 and 0.9, and cutoff parameter $A = 3.8$; the $M/G/\infty$ process (MGIP) with input traffic intensity $\rho = 0.9$, service rate $\mu = \alpha/(\alpha-1)$, where $\alpha = 3 - 2H$, and time lag $s = 8$; the Pareto-modulated Poisson process (PMPP) with input $T = 300$, $\lambda_1 = 100$ and $\lambda_2 = 120$; the SRP-FGN methods with input $M = 10$, and the interval number of the renewal CDF $(T) = 10{,}000$. The input of the superposition of the autoregressive process (SAP) with a beta-distribution ($B(\alpha_1, \alpha_2)$) on $[0, 1]$ was $B(1, 2.9)$, $B(1, 8.1)$, $B(1, 21.3)$, and $B(1, 71.5)$.

4.3.1 Accuracy of Generated Sequences

For each of $H = 0.6$, 0.7, 0.8 and 0.9, and for each of $\alpha_2 = 2.9$, 8.1, 21.3 and 71.5, all results are averaged over 30 sequences.

(a) Tables 4.11 and 4.12 show the results of the six sequential methods using the wavelet-based H estimator with the corresponding 95% confidence

Table 4.11: Mean values of estimated H using the wavelet-based H estimator for the six sequential generators for $H = 0.6$ and 0.7. We give 95% confidence intervals for the means in parentheses.

Methods	Mean Values of Estimated H and ΔH			
	.6		.7	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
FBNDP	.6086 (.581, .636)	+1.440	.6875 (.660, .715)	-1.789
SFRP	.6166 (.589, .644)	+2.759	.7091 (.682, .737)	+1.296
MGIP	.5445 (.517, .572)	-9.252	.6241 (.597, .652)	-10.84
PMPP	.5924 (.565, .620)	-1.266	.6845 (.657, .712)	-2.221
SRP-FGN	.5942 (.567, .622)	-0.968	.6960 (.669, .724)	-0.569
SAP	.5989 (.595, .603)	-0.182	.6852 (.680, .690)	-2.112

intervals $\hat{H} \pm 1.96\hat{\sigma}_{\hat{H}}$. For all input H and α_2 values, the SRP-FGN method produced sequences with the least biased H values compared with the other six methods.

For $H = 0.6$, 0.7 and 0.8, the absolute relative error for the FBNDP method was less than 5%, but for $H = 0.9$, it was greater than 5% (i.e., -5.5%). For $H = 0.6$, the estimated H value for the method was positively biased, but for $H = 0.7$, 0.8 and 0.9, were gradually more negatively biased as the H value increased.

For $H = 0.6$, 0.7, 0.8 and 0.9, each relative error for the SFRP method was +2.76%, +1.29%, -0.17% and -3.49%, respectively. As in the FBNDP method, estimated H values for the method ranged from positively biased to negatively biased as the H value increased.

A shortcoming of the MGIP method was that it generated approximately self-similar sequences with strongly biased H values. For $H = 0.6, 0.7, 0.8$

and 0.9, each relative error was -9.25%, -10.84%, +2.42% and +23.5%, respectively. Although these inter-event processes can be used to produce synthetic teletraffic with bursts appearing over a wider range of time scales than Poisson processes, the associated arrival processes do not appear to be self-similar when the aggregation level is low [93], [126].

For $H = 0.6$, 0.7, 0.8 and 0.9, the values of the Hurst parameter from the sample sequences of the PMPP method were lower than the desired values. Each relative error was -1.27%, -2.22%, -1.23% and -3.77%, respectively. Furthermore, this method required four control parameters (i.e., two Poisson arrival rates λ_1 and λ_2, an aggregation number M, and the shape parameter α), and generated a self-similar sequence that was positively biased to negatively biased as the shape parameter α approached one.

For $H = 0.6$, 0.7, 0.8 and 0.9, the values of the Hurst parameter from

Table 4.12: Mean values of estimated H using the wavelet-based H estimator for the six sequential generators for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means in parentheses.

Methods	Mean Values of Estimated H and ΔH			
	.8		.9	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
FBNDP	.7827 (.755, .810)	-2.157	.8502 (.823, .878)	-5.538
SFRP	.7986 (.771, .826)	-0.174	.8686 (.841, .896)	-3.491
MGIP	.8194 (.792, .847)	+2.424	1.1120 (1.084, 1.139)	+23.50
PMPP	.7902 (.763, .818)	-1.229	.8661 (.839, .894)	-3.766
SRP-FGN	.8056 (.778, .833)	+0.700	.9031 (.876, .931)	+0.344
SAP	.7845 (.781, .788)	-1.937	.8971 (.894, .900)	-0.325

the sample sequences of the SRP-FGN method match the required values well. For $H = 0.6$, 0.7, 0.8 and 0.9, each relative error was -0.97%, -0.57%, +0.70% and +0.34%, respectively.

For $\alpha_2 = 2.9$, 8.1, 21.3 and 71.5, all values of the Hurst parameter from the sample sequences of the SAP method were lower than the required values. Each relative error was -0.18%, -2.11%, -1.94% and -0.33%, respectively.

(b) Tables 4.13 and 4.14 show the results of the six sequential methods using Whittle's MLE with the corresponding 95% confidence intervals $\hat{H} \pm 1.96\hat{\sigma}_{\hat{H}}$. As for the results obtained from the wavelet-based H estimator, the SRP-FGN method produced sequences with the least biased H values compared with the other six methods.

For $H = 0.6$ and 0.7, the absolute relative error for the FBNDP method was less than 3%, while for $H = 0.8$ and 0.9, it was greater than 5% (i.e., -5.53% and -9.29%). For $H = 0.6$, the estimated H value for the method was positively biased; and for $H = 0.7$, 0.8 and 0.9, they gradually became more negatively biased as the H value increased.

For $H = 0.6$, 0.7, 0.8 and 0.9, relative error for the SFRP method was +6.21%, +1.78%, -1.43% and -5.37%, respectively. As in the FBNDP method, estimated H values ranged from positively biased to negatively biased as the H value increased.

A shortcoming of the MGIP method was that it generated approximately self-similar sequences with biased H values for $H = 0.7$, similar to results obtained from the wavelet-based H estimator. For $H = 0.6$, 0.7, 0.8 and 0.9, relative error was -8.0%, -9.6%, -3.2% and +5.5%, respectively.

For $H = 0.8$ and 0.9, the values of the Hurst parameter from the sample sequences of the PMPP method were lower than the required values, but for $H = 0.6$ and 0.7, they were higher. Relative error for $H = 0.6$, 0.7, 0.8 and 0.9 was +7.18%, +2.24%, -1.28% and -5.59%, respectively. This method generated a self-similar sequence that ranged from negatively biased to positively biased as the shape parameter α approached one.

For $H = 0.6$, 0.7, 0.8 and 0.9, the Hurst parameter values from the sample sequences of the SRP-FGN method match the required values well. For

H = 0.6, 0.7, 0.8 and 0.9, relative error was +0.11%, +1.91%, +2.54% and +2.53%, respectively. The results were consistently overestimated.

For $\alpha_2 = 2.9$ and 8.1, all values of the Hurst parameter from the sample sequences of the SAP method were higher than the required values. Relative error was +24.14% and +10.45%, respectively; thus, these results were overestimated. For $\alpha_2 = 21.3$ and 71.5, relative errors was -0.01% and -8.45%.

(c) Sequence plots show higher levels of data correlation of data, and thus evidence of LRD properties, as the H value increased; see Figure 4.6 for results of the SRP-FGN method.

Our results show that all six sequential generators produced approximately self-similar sequences, but that relative inaccuracy ($|\Delta H|$) increased as H increased.

Table 4.13: Mean values of estimated H using Whittle's MLE for the six sequential generators for $H = 0.6$ and 0.7. We give 95% confidence intervals for the means in parentheses.

Methods	Mean Values of Estimated H and ΔH			
	.6		.7	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
FBNDP	.6122 (.603, .622)	+2.028	.6828 (.674, .692)	-2.452
SFRP	.6372 (.628, .647)	+6.205	.7124 (.703, .722)	+1.777
MGIP	.5520 (.542, .562)	-7.995	.6325 (.623, .642)	-9.641
PMPP	.6431 (.634, .652)	+7.176	.7157 (.706, .725)	+2.236
SRP-FGN	.6007 (.591, .610)	+0.110	.7133 (.704, .723)	+1.905
SAP	.7448 (.736, .754)	+24.14	.7731 (.764, .782)	+10.45

4.3.2 Complexity and Speed of Generation

The computational complexities of all six sequential generators of pseudo-random self-similar sequences of a given length n are $O(n)$. However, the number of arithmetic operations per number required by each of them are different, as shown in Table 4.15. The MGIP, SAP and SRP-FGN methods are the fastest of the six sequential generators requiring 4,976, 4,481 and 4,173 operations per number, respectively. While the SFRP method was the slowest (85,199 operations per number), the FBNDP and PMPP methods require similar amounts of time to generate the same number of self-similar sequences, requiring 11,028 and 19,507 arithmetic operations per number, respectively. Figure 4.7 shows the experimental mean running times of the six sequential generators. All require $O(n)$ computations to generate n numbers.

In summary, our results show that the generator based on the SRP-FGN al-

Table 4.14: Mean values of estimated H using Whittle's MLE for the six sequential generators for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means in parentheses.

Methods	Mean Values of Estimated H and ΔH			
	.8		.9	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
FBNDP	.7557 (.747, .765)	-5.531	.8164 (.807, .826)	-9.291
SFRP	.7886 (.779, .798)	-1.426	.8517 (.843, .861)	-5.366
MGIP	.7742 (.765, .783)	-3.225	.9499 (.941, .959)	+5.549
PMPP	.7898 (.781, .799)	-1.281	.8497 (.841, .859)	-5.586
SRP-FGN	.8203 (.811, .829)	+2.539	.9227 (.914, .932)	+2.526
SAP	.7999 (.791, .809)	-0.009	.8239 (.815, .833)	-8.451

gorithm is the fastest if long sequences of self-similar pseudo-random numbers are required. The MGIP and SAP methods are nearly as fast. The SRP-FGN generator is also the most efficient when evaluated by Whittle's MLE. Therefore, the SRP-FGN generator was compared with the most efficient fixed-length sequence generator, which we discuss in Section 4.6. However, as pointed out in [144] there are general pitfalls in using generators of LRD processes based on FRPs, since the generated processes fail to capture fully the required autocorrelation structure. For a more detailed discussion, see [144].

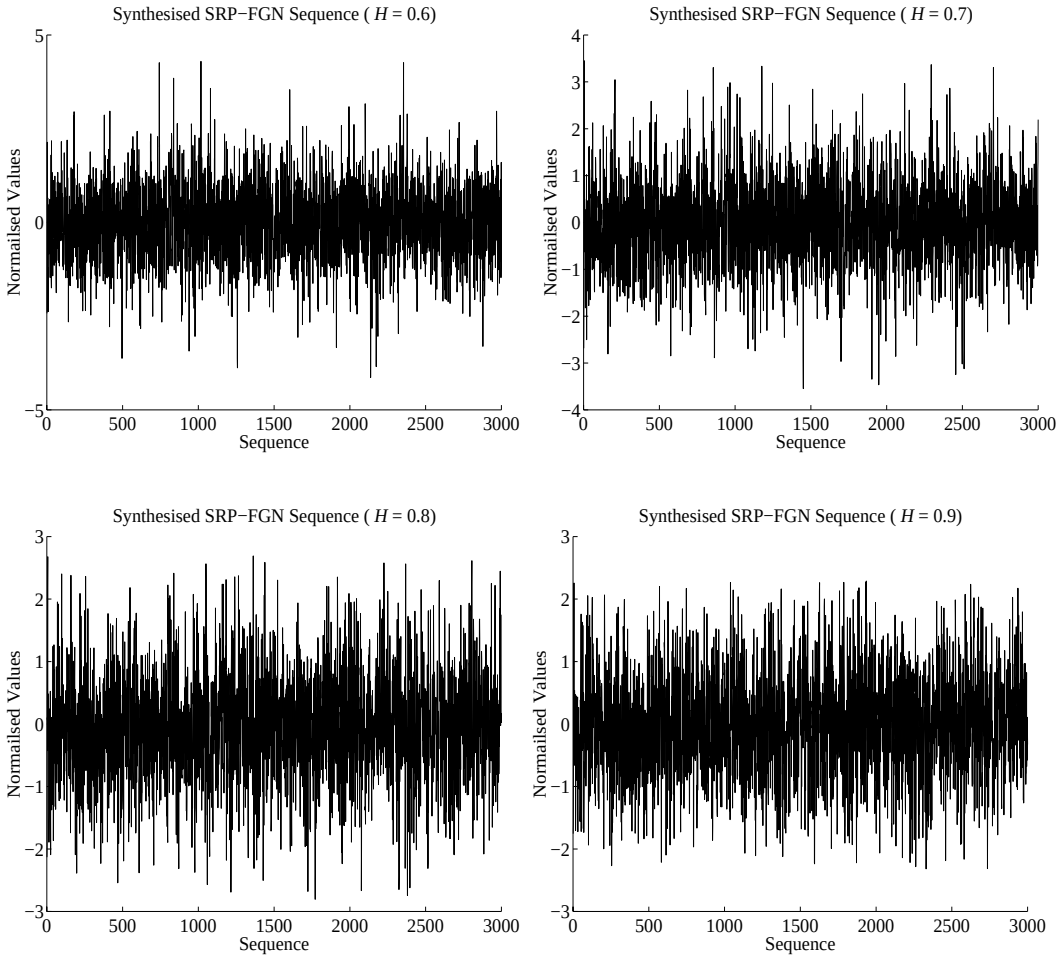


Figure 4.6: Sequence plots for the SRP-FGN method ($H = 0.6, 0.7, 0.8, 0.9$).

Table 4.15: Computational complexities and arithmetic mean operations required for each of the six sequential generators (s: the time lag in the $M/G/\infty$ queueing system, M: aggregation level, T: the number of intervals in the renewal CDF).

Method	Complexity	Operations per number	Mean operations per number when assuming optimum values
FBNDP	$O(n)$	$161M$ + 9,418	11,028
SFRP	$O(n)$	$1378M$ + 71,419	85,199
MGIP	$O(n)$	$130s$ + 3,936	4,976
PMPP	$O(n)$	$130M$ + 4,037	19,507
SRP-FGN	$O(n)$	$14T$ + 4,159	4,173
SAP	$O(n)$	4,481	4,481

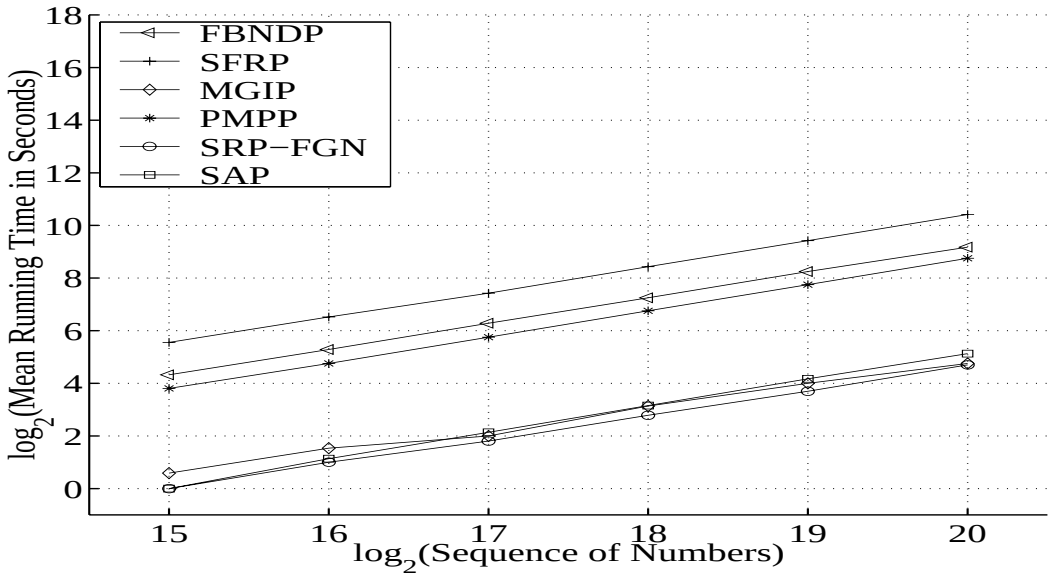


Figure 4.7: Mean running times of the six sequential generators. Running times were obtained using the SunOS 5.7 `time` command on a Pentium II (233 MHz, 512 MB); each mean is averaged over 30 iterations.

4.4 Fixed-Length Sequence Generators

The numerical results for the following fixed-length sequence generators are given in Section 4.5.

4.4.1 Fast Fourier Transform Method

The fast Fourier transform method generates approximately self-similar sequences based on the fast Fourier transform (FFT) and the fractional Gaussian noise (FGN) process. Its main weakness is in the accuracy of the power spectrum calculation, which involves an infinite summation. Paxson [124] has proposed a solution to this problem by applying a finite approximation. Another possible method to generate self-similar sequences is to run the FFT of white noise through the power spectrum, and then apply the inverse FFT. An overview of the FFT method is given as follows. For a more detailed discussion, see [70], [87], [124].

Step 1. Given: H. Start for $i = 1$ and continue until $i = n/2$. Calculate a sequence of values $\{f_1, \ldots, f_{\frac{n}{2}}\}$, where $f_i = \hat{f}(\frac{2\pi i}{n}, H)$, corresponding to the power spectrum of an FGN process for frequencies from $\frac{2\pi}{n}$ to π, $1/2 < H < 1$, and the total length of the sequence generated, n. For an FGN process, the power spectrum $f(\lambda, H)$ is defined by Equations (2.16) and (2.17). As mentioned, the infinite summation in Equation (2.16) for $\mathcal{B}(\lambda, H)$ poses the main difficulty in computing the power spectrum exactly. Paxson [124] proposed to use the approximation given by Equation (4.17):

$$\mathcal{B}(\lambda, H) \approx a_1^d + b_1^d + a_2^d + b_2^d + a_3^d + b_3^d + \frac{a_3^{d'} + b_3^{d'} + a_4^{d'} + b_4^{d'}}{8H\pi}, \quad (4.17)$$

where $d = -2H - 1, d' = -2H, a_i = 2i\pi + \lambda, b_i = 2i\pi - \lambda$.

Step 2. Multiply the sequence of values $\{f_1, \ldots, f_{\frac{n}{2}}\}$ by an exponential random variable with a mean of one. Paxson [124] used this step since, when estimating the power spectrum of a process using the periodogram, the power spectrum estimated for a given frequency is distributed asymptotically as an exponential random variable with a mean equal to the actual power (see also Beran [8] (p.409)).

Step 3. Generate $\{Z_1, \ldots, Z_{\frac{n}{2}}\}$, a sequence of complex values such that $|Z_i| = \sqrt{\hat{f}_i}$ and the phase of Z_i is uniformly distributed between 0 and 2π. This random phase technique, taken from Schiff [152], preserves the spectral density corresponding to $\{\hat{f}_i\}$. It also makes the marginal distribution of the final sequence normal, as proved by Lindeberg ([37], p. 256), and defines the requirements for FGN.

Step 4. Start for $i = 0$ and continue until $i < n$. Construct $\{Z_0^{'}, \ldots, Z_{n-1}^{'}\}$, an *expanded* version of $\{Z_1, \ldots, Z_{\frac{n}{2}}\}$:

$$Z_i^{'} = \begin{cases} 0, & \text{if} \quad i = 0, \\ Z_i, & \text{if} \quad 0 < i \leq \frac{n}{2}, \text{and} \\ \overline{Z_{n-i}}, & \text{if} \quad \frac{n}{2} < i < n. \end{cases} \tag{4.18}$$

where $\overline{Z_{n-i}}$ denotes the complex conjugate of Z_{n-i}. $\{Z_i^{'}\}$ retains properties of the power spectrum used to construct $\{Z_i\}$, but because $\{Z_i^{'}\}$ is symmetric about $Z_{\frac{n}{2}}^{'}$, it now corresponds to the FFT of a real-valued signal.

Step 5. Calculate the inverse FFT $\{Z_i^{'}\}$ to obtain the approximate FGN sequence $\{X_i\}$ with a mean of zero and variance of one. Then form the final sequence $X_1, X_2, \ldots, X_n$ by assigning $X_i \longleftarrow Z_i^{'}$, $i = 1, 2, \ldots, n$.

This method generates an approximately self-similar sequence $\{X_1, X_2, \ldots, X_n\}$ with the exact values. Generating a sample sequence of 1,048,576 numbers took 33 seconds on a Pentium II (233 MHz, 512 MB). The FFT method requires $O(n\log n)$ computations to generate n numbers because of the Fourier transform algorithm [124], [130]. The Danielson-Lanczos' FFT[1] algorithm was used [131]. For a more detailed discussion, see [124].

[1] Available at `http://ita.ee.lbl.gov/`.

4.4.2 Fractional-Autoregressive Integrated Moving Average Method

Hosking ([62], [63]) states that the F-ARIMA method[2] is used to generate an approximately self-similar process with a Hurst parameter of $H = d + \frac{1}{2}$. We used the F-ARIMA(0,d,0) method for generating self-similar sequences, where d is the fractional differencing parameter, $0 < d < \frac{1}{2}$. Hosking's algorithm is used to generate the process $\mathbf{X} = \{X_i : i = 0, 1, 2, \ldots, n\}$ with a normal marginal distribution, a mean of zero and variance σ_0^2, and an autocorrelation function (ACF), $\{\rho_k\}(k = 0, \pm 1, \ldots)$ defined as

$$\rho_k = \gamma_k/\gamma_0 = \frac{\Gamma(1-d)\Gamma(k+d)}{\Gamma(d)\Gamma(k+1-d)}, \tag{4.19}$$

where

$$\gamma_k = \sigma_0^2 \frac{(-1)^k \Gamma(1-2d)}{\Gamma(k-d+1)\Gamma(1-k-d)};$$

see [62] and page 63 on [9].

Step 0. Set $N_0 = 0$ and $D_0 = 1$. X_0, the first pseudo-random element in the output self-similar sequence, is generated from the normal distribution $N(0, \sigma_0^2)$, where σ_0^2 is the required variance of the X_i.

Step i ($i = 1, \ldots, n-1$.) Compute $mean_i$ and var_i of X_i recursively, using the following equations:

$$\begin{aligned} N_i &= \rho_i - \sum_{j=1}^{i-1} \phi_{i-1,j}\rho_{i-j}, \\ D_i &= D_{i-1} - N_{i-1}^2/D_{i-1}, \\ \phi_{ii} &= N_i/D_i, \\ \phi_{ij} &= \phi_{i-1,j} - \phi_{ii}\phi_{i-1,i-j}, j = 1, \ldots, i-1, \end{aligned}$$

where $\phi_{ij}, i = 0, j = 0, \ldots, n-1$, is given by

$$\phi_{ij} = -\binom{i}{j} \frac{(j-d-1)!(i-d-j)!}{(-d-1)!(i-d)!},$$

[2]The autocorrelation functions of two other simple processes, F-ARIMA(1,d,0) and F-ARIMA(0,d,1), behave similarly at high lags, but the F-ARIMA(0,d,1) autocorrelation function drops more sharply between lags 1 and 2. For a more detailed discussion, see [62].

$$mean_i = \sum_{j=1}^{i} \phi_{ij} X_{i-j}, \tag{4.20}$$

$$var_i = (1 - \phi_{ii}^2) var_{i-1}. \tag{4.21}$$

Generate X_i from $N(mean_i, var_i)$. Increase i by 1. If $i = n$, then stop.

A self-similar sequence $\{X_1, X_2, \ldots, X_n\}$ is obtained in n steps. The F-ARIMA method is too computationally intensive to generate long sample sequences. Generation of an F-ARIMA traffic sample sequence with 1,048,576 numbers took 41 hours, 0 minutes and 22 seconds on a Pentium II (233 MHz, 512 MB). This method requires $O(n^2)$ computation time.

4.4.3 Random Midpoint Displacement Method

The *random midpoint displacement* (RMD) method generates an approximately self-similar sequence in the time interval $[0, T]$. The RMD algorithm is an approximate fractional Brownian motion (FBM) generation method. The basic concept of the RMD algorithm is to interpolate the interval $[0, T]$ recursively and calculate the values of the process at the midpoints from the values at the endpoints.

Figure 4.8 illustrates the first three steps of the process. This method leads to the generation of the sequence $(d_{3,1}, d_{3,2}, d_{3,3}, d_{3,4})$. The interval between 0 and 1 is subdivided to construct the increments governed by a normal distribution. Adding offsets to the midpoints makes the marginal distribution of the final result normal. For more detailed discussions of the RMD method, see [70], [84], [127].

Step 1. If the process $\mathbf{Y}$ is to be computed for any time instance t between 0 and 1, then begin by setting $Y_0 = 0$ and selecting Y_1 as a pseudo-random number from a normal distribution with mean 0 and variance $Var[Y_1] = \sigma_0^2$. Then $Var[Y_1 - Y_0] = \sigma_0^2$.

Step 2. Next, $Y_{\frac{1}{2}}$ is constructed as the average of Y_0 and Y_1, that is,

$$Y_{\frac{1}{2}} = \frac{1}{2}(Y_0 + Y_1) + d_1.$$

The offset d_1 is a normal random number (NRN), which is multiplied by a scaling factor $\frac{1}{2}$, with mean 0 and variance S_1^2 of d_1. Compare the visualisation of this step and the next one in Figure 4.8. For $Var[Y_{t_2} - Y_{t_1}] = |t_2 - t_1|^{2H}\sigma_0^2$ to be true for $0 \leq t_1 \leq t_2 \leq 1$, then

$$\begin{aligned} Var[Y_{\frac{1}{2}} - Y_0] &= \frac{1}{4}Var[Y_1 - Y_0] + S_1^2 \\ (\frac{1}{2})^{2H}\sigma_0^2 &= \frac{1}{4}\sigma_0^2 + S_1^2. \end{aligned}$$

Thus $S_1^2 = (\frac{1}{2^1})^{2H}(1 - 2^{2H-2})\sigma_0^2$.

Step 3. Reduce the scaling factor by $\sqrt{2}$; that is, now assume $\frac{1}{\sqrt{8}}$, and divide the two intervals from 0 and $\frac{1}{2}$ and from $\frac{1}{2}$ to 1 again. $Y_{\frac{1}{4}}$ is set as the average $\frac{1}{2}(Y_0 + Y_{\frac{1}{2}})$ plus an offset $d_{2,1}$, which is an NRN multiplied by the current scaling factor $\frac{1}{\sqrt{8}}$. The corresponding formula holds for $Y_{\frac{3}{4}}$, that is,

$$Y_{\frac{3}{4}} = \frac{1}{2}(Y_{\frac{1}{2}} + Y_1) + d_{2,2},$$

where $d_{2,2}$ is a random offset computed as before. Therefore, the variance S_2^2 of $d_{2,*}$ must be chosen such that

$$\begin{aligned} Var[Y_{\frac{1}{4}} - Y_0] &= \frac{1}{4}Var[Y_{\frac{1}{2}} - Y_0] + S_2^2 \\ (\frac{1}{2^2})^{2H}\sigma_0^2 &= \frac{1}{4}(\frac{1}{2})^{2H}\sigma_0^2 + S_2^2. \end{aligned}$$

Thus $S_2^2 = (\frac{1}{2^2})^{2H}(1 - 2^{2H-2})\sigma_0^2$.

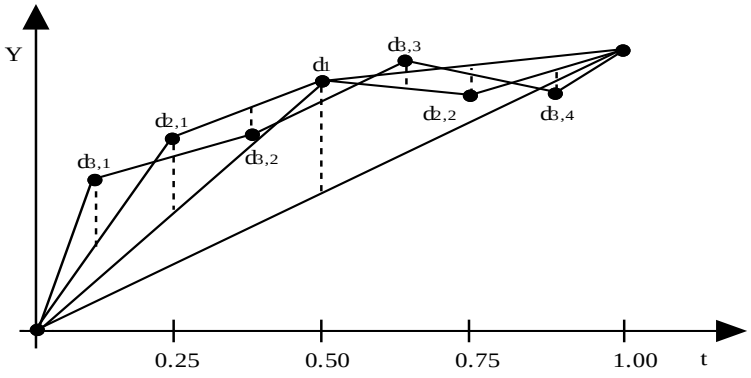


Figure 4.8: The first three stages in the RMD method.

Step 4. Proceed in the same manner: reduce the scaling factor by $\sqrt{2}$, that is, scale by $\frac{1}{\sqrt{16}}$. Then set

$$\begin{aligned} Y_{\frac{1}{8}} &= \frac{1}{2}(Y_0 + Y_{\frac{1}{4}}) + d_{3,1}, \\ Y_{\frac{3}{8}} &= \frac{1}{2}(Y_{\frac{1}{4}} + Y_{\frac{1}{2}}) + d_{3,2}, \\ Y_{\frac{5}{8}} &= \frac{1}{2}(Y_{\frac{1}{2}} + Y_{\frac{3}{4}}) + d_{3,3}, \\ Y_{\frac{7}{8}} &= \frac{1}{2}(Y_{\frac{3}{4}} + Y_1) + d_{3,4}. \end{aligned}$$

In each formula, $d_{3,*}$ is computed as a different NRN multiplied by the current scaling factor $\frac{1}{\sqrt{16}}$. The following step computes $\mathbf{Y}$ at $t = \frac{1}{16}, \frac{3}{16}, \ldots, \frac{15}{16}$ using a scaling factor again reduced by $\sqrt{2}$, and continues as indicated above. The variance S_3^2 of $d_{3,*}$ is chosen such that

$$\begin{aligned} Var[Y_{\frac{1}{8}} - Y_0] &= \frac{1}{4}Var[Y_{\frac{1}{4}} - Y_0] + S_3^2 \\ (\frac{1}{2^3})^{2H}\sigma_0^2 &= \frac{1}{4}(\frac{1}{2^2})^{2H}\sigma_0^2 + S_3^2, \end{aligned}$$

that is, $S_3^2 = (\frac{1}{2^3})^{2H}(1 - 2^{2H-2})\sigma_0^2$. The variance S_n^2 of $d_{n,*}$, therefore, yields $(\frac{1}{2^n})^{2H}(1 - 2^{2H-2})\sigma_0^2$.

Step 5. Calculate the values at the midpoints in the previous same manner until the given n is equal to $2^{NoOfSteps}$. Then form the final sequence $X_0, X_1, X_2, \ldots$ by assigning $X_i \longleftarrow Y_{i/2^{NoOfSteps}}$, $i = 0, 1, 2, \ldots$.

A self-similar sequence $\{X_0, X_1, \ldots, X_n\}$ is obtained from the previous steps. Generation of an approximately self-similar sequence with 1,048,576 numbers took 17 seconds on a Pentium II (233 MHz, 512 MB). The theoretical algorithmic complexity is $O(n)$ [128].

4.4.4 Successive Random Addition Method

An alternative method for the direct generation of an FBM process is based on the *successive random addition* (SRA) algorithm [20], [68], [69], [70]. The SRA method uses the midpoints as the RMD method does, but adds a displacement of a suitable variance to all the points [128]. Adding offsets to all points should make the resultant sequence self-similar and of normal distribution [128]. The SRA method consists of the following steps:

Step 1. If the process $\mathbf{Y}$ is to be computed for time instances t between 0 and 1, then begin by setting $Y_0 = 0$ and selecting Y_1 as a pseudo-random number from a normal distribution with mean 0 and variance $Var[Y_1] = \sigma_0^2$. Then $Var[Y_1 - Y_0] = \sigma_0^2$.

Step 2. Next, $Y_{\frac{1}{2}}$ is constructed by the interpolation of the midpoint, that is, $Y_{\frac{1}{2}} = \frac{1}{2}(Y_0 + Y_1)$.

Step 3. Add a displacement of a suitable variance to all points, i.e.,

$$\begin{aligned} Y_0 &= Y_0 + d_{1,1}, \\ Y_{\frac{1}{2}} &= Y_{\frac{1}{2}} + d_{1,2}, \\ Y_1 &= Y_1 + d_{1,3}. \end{aligned}$$

The offsets $d_{1,*}$ are governed by a normal random number. For $Var[Y_{t_2} - Y_{t_1}] = |t_2 - t_1|^{2H}\sigma_0^2$ to be true, for any $t_1, t_2, 0 \leq t_1 \leq t_2 \leq 1$, it is required that

$$\begin{aligned} Var[Y_{\frac{1}{2}} - Y_0] &= \frac{1}{4}Var[Y_1 - Y_0] + 2S_1^2 \\ (\frac{1}{2})^{2H}\sigma_0^2 &= \frac{1}{4}\sigma_0^2 + 2S_1^2, \end{aligned}$$

that is, $S_1^2 = \frac{1}{2}(\frac{1}{2^1})^{2H}(1 - 2^{2H-2})\sigma_0^2$.

Step 4. Step 2 and Step 3 are repeated. Therefore,

$$S_n^2 = \frac{1}{2}(\frac{1}{2^n})^{2H}(1 - 2^{2H-2})\sigma_0^2,$$

where σ_0^2 is the initial variance and $0 < H < 1$.

Step 5. Calculate the values at the midpoints as noted previously until the given n is equal to $2^{NoOfSteps}$. Then form the final sequence $X_0, X_1, X_2, \ldots$ by assigning $X_i \longleftarrow Y_{i/2^{NoOfSteps}}$, $i = 0, 1, 2, \ldots$.

Using these steps, the SRA method generates an approximately self-similar sequence $\{X_0, X_1, \ldots, X_n\}$. It took 15 seconds to generate a sequence of 1,048,576 numbers on a Pentium II (233 MHz, 512 MB). The theoretical algorithmic complexity is $O(n)$ [128].

4.4.5 Fractional Gaussian Noise and Daubechies Wavelets Method

We present a new generator of pseudo-random self-similar sequences based on fractional Gaussian noise (FGN) and Daubechies wavelets (DW), called the FGN-DW method [72], [73]. A pseudo-random generator of self-similar teletraffic based on Haar wavelet transforms has been proposed in [135], [136] and [168]. We used Daubechies wavelets because the generator based on Daubechies wavelets produces more accurate self-similar sequences than one based on Haar wavelets. In other words, not only estimates of H obtained from the Daubechies wavelets are closer to the true values than those from the Haar wavelets, but also variances obtained from the Daubechies wavelets are lower. The reason behind is that the Daubechies wavelets produce smoother coefficients of wavelets that are used in the discrete wavelet transform than the Haar wavelets [26], [143], [170]. Haar wavelets are discontinuous, and they do not have good time-frequency localisation properties, since their Fourier transforms decay as $|\lambda|^{-1}$, for $\lambda \to \infty$, meaning that the resulting decomposition has a poor scale. Therefore, Daubechies wavelets produce more accurate coefficients than Haar wavelets; for a more detailed discussion, see [26], [170].

Our method for generating synthetic self-similar FGN sequences in a time domain is based on a discrete wavelet transform (DWT). Wavelets can provide compact representations for a class of FGN processes [40], [78], [143], because the structure of wavelets naturally matches the self-similar structure of long-range dependent processes [2], [26], [170].

We claim that the FGN-DW method is sufficiently fast for the practical generation of synthetic self-similar sequences that can be used as simulation input data. The general strategy behind our method is similar to Paxson's, who used the Fourier transform [124].

Figure 4.9 graphically illustrates a discrete Fourier and a discrete wavelet transform (see also Appendix E). Wavelet analysis transforms a sequence onto a time-scale grid, where the term *scale* is used instead of *frequency*, because the mapping is not directly related to frequency as in the Fourier transform. The wavelet transform delivers good resolution in both time and scale, as compared to the Fourier transform, which provides only good frequency resolution. The algorithm consists of the following steps:

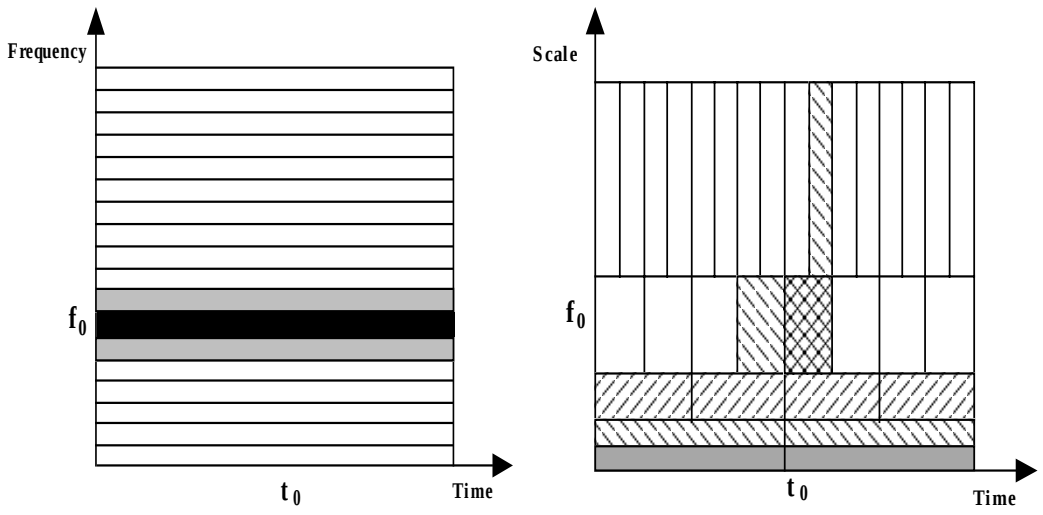
Step 1. Given: H. Start for $i = 1$ and continue until $i = n$. Calculate a sequence of values $\{f_1, f_2, \cdots, f_n\}$ using Equation (4.22) (following), where $f_i = \hat{f}(\frac{\pi i}{n}; H)$, corresponding to the spectral density of an FGN process for frequencies f_i ranging between $\frac{\pi}{n}$ and π.

The main difficulty with using Equation (2.16) when computing the spectral density is that it requires to execute the infinite summation. The approximation of $f(\lambda, H)$ is given in [9] as

$$f(\lambda, H) = c_f|\lambda|^{1-2H} + O(|\lambda|^{min(3-2H,2)}), \tag{4.22}$$

where c_f is Equation (2.17) and $O(\cdot)$ represents the residual error.

This formula was used in the generation of self-similar sequences proposed in this thesis. Another generator of self-similar sequences based on FGN was also proposed by Paxson [124], but his method was based on a more complicated approximation of $f(\lambda, H)$ as shown in Equation (4.17). Equation (4.22) can be used to determine $f(\lambda, H)$ for $\lambda \to \infty$, or for $n \to \infty$ at $\lambda = \frac{\pi}{n}$. For a large value of λ, $f(\lambda, H)$ can be calculated by Equation (2.16).



(a) A discrete Fourier transform. (b) A discrete wavelet transform.

Figure 4.9: A graphical representation of a discrete Fourier transform and a discrete wavelet transform.

Step 2. Multiply $\{f_i\}$ by realisations of an independent exponential random variable with a mean of one to obtain $\{\hat{f}_i\}$, because the spectral density estimated for a given frequency is distributed asymptotically as an independent exponential random variable with mean $f(\lambda, H)$ [8].

Step 3. Generate a sequence $\{Y_1, Y_2, \cdots, Y_p\}$ of complex numbers such that $|Y_i| = \sqrt{\hat{f}_i}$ and the phase of Y_i is uniformly distributed between 0 and 2π. This random phase technique, taken from Schiff [152], preserves the spectral density corresponding to $\{\hat{f}_i\}$. It also makes the marginal distribution of the final sequence normal and produces the requirements for FGN.

Step 4. Calculate the two synthetic coefficients of orthonormal Daubechies wavelets that are used in the inverse DWT (IDWT) [46]. The output sequence $\{X_1, X_2, \ldots, X_n\}$ representing approximately self-similar FGN process (in time domain) is obtained by applying the IDWT operation to the sequence $\{Y_1, Y_2, \cdots, Y_n\}$.

Using the previous steps, the proposed FGN-DW method generates a fast and sufficiently accurate self-similar FGN process $\{X_1, X_2, \ldots, X_n\}$. (Appendix D provides a program written in C and Matlab for implementing this method using the pyramidal algorithm of IDWT.) It took 16 seconds to generate a sequence of 1,048,576 numbers on a Pentium II (233 MHz, 512 MB). Its theoretical algorithmic complexity is $O(n)$. Moreover, the accuracy of Daubechies wavelets is slightly better than Haar wavelets, but there is no difference in the time taken to obtain the same number of coefficients. For more detailed discussion, see also Section 4.4.5 and Appendix E.

Autocorrelation Test for the FGN-DW Method

ACFs characterise the correlation structure and are used to investigate the behaviour of self-similar sequences. For H = 0.6, 0.7, 0.8 and 0.9, Figure 4.10 shows the ACF of sequences obtained by the FGN-DW method and the theoretical ACFs from Equation (2.6) and Equation (2.9). As the H value increased, the ACF curves decayed hyperbolically and LRD was observed. For H = 0.6, 0.7, 0.8 and 0.9, relative inaccuracy ΔACF estimated from the ACF was -0.0208%, -0.0061%, -0.0076% and -0.0218%, respectively.

Comparison of Daubechies Wavelets and Haar Wavelets for Generation of LRD Sequences

Comparison results of sequences produced by generators based on Haar and Daubechies wavelets with several coefficients are shown in Tables 4.16 – 4.19. The relative error associated with each wavelet were also compared. The results indicate that Daubechies wavelets with sixteen coefficients produce the most accurate results and are slightly more accurate than Haar wavelets. In addition, Tables 4.20 and 4.21 show that the variances obtained from Daubechies wavelets are smaller than those obtained from Haar wavelets. However, the

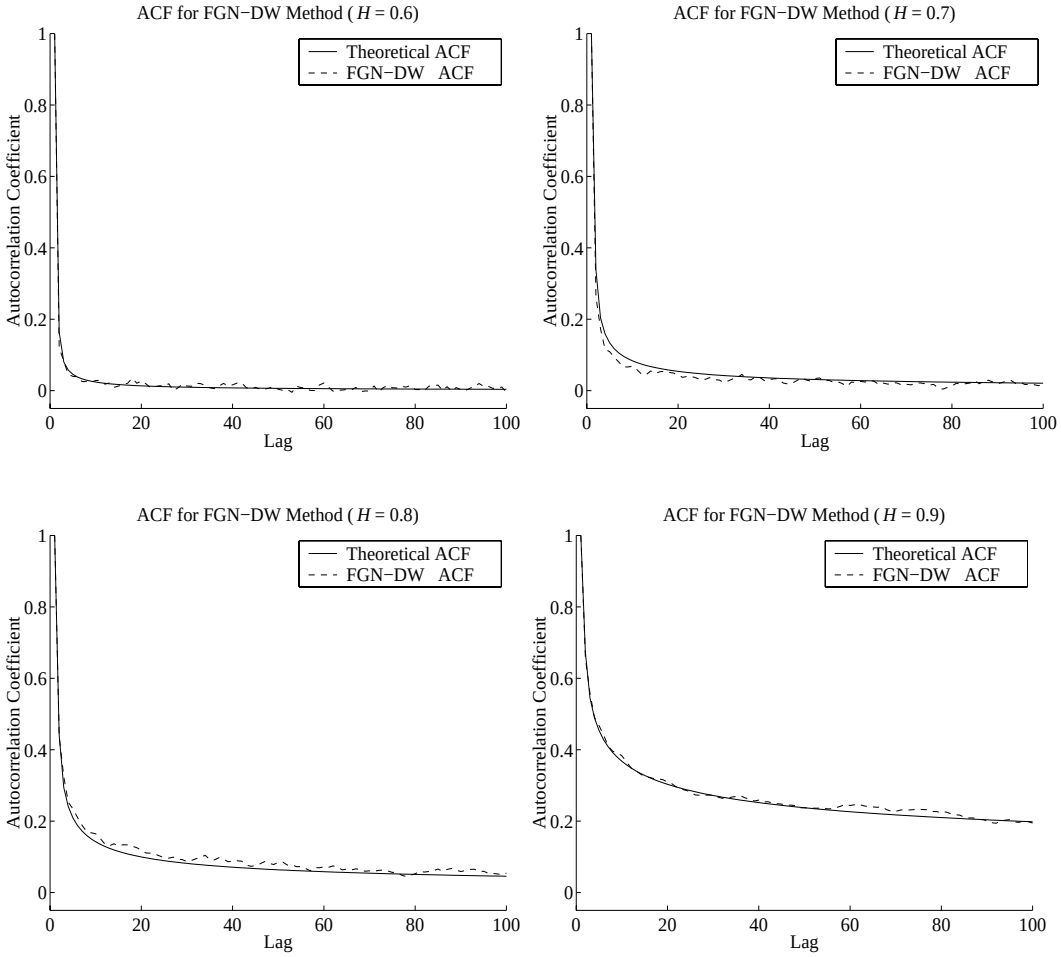


Figure 4.10: Autocorrelation function plots for the FGN-DW method (H = 0.6, 0.7, 0.8, 0.9).

two wavelets theoretically require the same $O(n)$ operations to transform n coefficients of their wavelets, and there is no difference in the time required to obtain a given number of their coefficients.

4.5 Comparison of Fixed-Length Generators

Paxson [124] and Lau et al. [84] suggest that the FFT- and RMD-based methods are sufficiently fast in the generation of simulation input data for practical applications. In this thesis, we report on the properties of these two methods and the F-ARIMA-based method, and compare them with SRA and FGN-DW, two recently proposed alternative methods for the generation of pseudo-random self-similar sequences [70], [73]. These five fixed-length sequence generators are comparable because all of them have the same statistical properties, such as normal marginal distributions, means and variances. They were implemented in C on a Pentium II (233 MHz, 512 MB) computer. The mean times required for generating sequences of a given length were obtained using the SunOS 5.7 `time` command and were averaged over 30 replications, each with sequences of 32,768 (2^{15}), 65,536 (2^{16}), 131,072 (2^{17}), 262,144 (2^{18}), 524,288 (2^{19}) and 1,048,576 (2^{20}) numbers.

We have analysed the accuracy with which five considered generators generate normal pseudo-random sequences with the required value of H. For $H =$ 0.6, 0.7, 0.8 and 0.9, each method was used to generate 30 sample sequences of 32,768 (2^{15}) numbers starting from different random seeds. Self-similarity and marginal distributions of the sequences generated were assessed by the same techniques as those used in Section 4.3.

4.5.1 Accuracy of Generated Sequences

A summary of the results of our analysis follows:

The estimates of the Hurst parameter for the wavelet-based H estimator and Whittle's MLE are shown in Tables 4.22 – 4.25. The relative inaccuracy

Table 4.16: Comparison of Daubechies wavelets and Haar wavelets: mean values of estimated H obtained using the wavelet-based H estimator for $H = 0.6$ and 0.7. Daub(#) stands for the Daubechies wavelets with # coefficients.

Methods	Mean Values of Estimated H and ΔH			
	.6		.7	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
Haar	.6073 (.580, .635)	+1.220	.7141 (.687, .742)	+2.018
Daub(2)	.6019 (.574, .630)	+0.323	.6984 (.671, .726)	-0.229
Daub(4)	.6026 (.575, .630)	+0.433	.7039 (.676, .731)	+0.554
Daub(8)	.6026 (.575, .630)	+0.430	.7031 (.676, .731)	+0.445
Daub(16)	.6013 (.574, .629)	+0.214	.6987 (.671, .726)	-0.185

Table 4.17: Comparison of Daubechies wavelets and Haar wavelets: mean values of estimated H obtained using the wavelet-based H estimator for $H = 0.8$ and 0.9. Daub(#) stands for the Daubechies wavelets with # coefficients.

Methods	Mean Values of Estimated H and ΔH			
	.8		.9	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
Haar	.8217 (.794, .849)	+2.712	.9299 (.902, .958)	+3.327
Daub(2)	.7943 (.767, .822)	-0.709	.8898 (.862, .917)	-1.137
Daub(4)	.8055 (.778, .833)	+0.684	.9074 (.880, .935)	+0.821
Daub(8)	.8039 (.776, .831)	+0.486	.9049 (.877, .932)	+0.545
Daub(16)	.7962 (.769, .824)	-0.474	.8938 (.866, .921)	-0.694

Table 4.18: Comparison of Daubechies wavelets and Haar wavelets: mean values of estimated H obtained using Whittle's MLE for $H = 0.6$ and 0.7. Daub(#) stands for the Daubechies wavelets with # coefficients.

Methods	Mean Values of Estimated H and ΔH			
	.6		.7	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
Haar	.5766 (.567, .586)	-3.905	.6567 (.647, .666)	-6.188
Daub(2)	.5814 (.572, .591)	-3.106	.6662 (.657, .676)	-4.829
Daub(4)	.5836 (.574, .593)	-2.736	.6702 (.661, .680)	-4.254
Daub(8)	.5845 (.575, .594)	-2.578	.6719 (.663, .681)	-4.017
Daub(16)	.5849 (.575, .594)	-2.521	.6725 (.663, .682)	-3.924

Table 4.19: Comparison of Daubechies wavelets and Haar wavelets: mean values of estimated H obtained using Whittle's MLE for $H = 0.8$ and 0.9. Daub(#) stands for the Daubechies wavelets with # coefficients.

Methods	Mean Values of Estimated H and ΔH			
	.8		.9	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
Haar	.7399 (.731, .749)	-7.508	.8256 (.817, .835)	-8.263
Daub(2)	.7535 (.744, .763)	-5.813	.8428 (.834, .852)	-6.357
Daub(4)	.759 (.750, .768)	-5.127	.8494 (.840, .859)	-5.620
Daub(8)	.7612 (.752, .770)	-4.854	.852 (.843, .861)	-5.336
Daub(16)	.762 (.753, .771)	-4.745	.853 (.844, .862)	-5.223

ΔH was calculated using Equation (3.3). All results were averaged over 30 sequences.

(a) The results for the wavelet-based H estimator with the corresponding 95% confidence intervals $\hat{H} \pm 1.96\hat{\sigma}_{\hat{H}}$, (see Tables 4.22 and 4.23), show that for all input H values, the F-ARIMA, the FFT and the FGN-DW methods produced sequences with less biased H values than other methods.

Using the FFT method, for $H = 0.6$, 0.7 and 0.8, the values of the Hurst parameter from the sample sequences match the required values well,

Table 4.20: Variances of estimated H obtained using the wavelet-based H estimator for Daubechies wavelets and Haar wavelets for $H = 0.6$, 0.7, 0.8 and 0.9. Daub(#) stands for the Daubechies wavelets with # coefficients.

Methods	Variances of Estimated H			
	.6	.7	.8	.9
Haar	2.070e-04	2.263e-04	2.534e-04	2.923e-04
Daub(2)	1.949e-04	2.154e-04	2.393e-04	2.670e-04
Daub(4)	2.508e-04	2.540e-04	2.586e-04	2.645e-04
Daub(8)	2.440e-04	2.432e-04	2.438e-04	2.456e-04
Daub(16)	2.116e-04	2.055e-04	2.007e-04	1.973e-04

Table 4.21: Variances of estimated H obtained using Whittle's MLE for Daubechies wavelets and Haar wavelets for $H = 0.6$, 0.7, 0.8 and 0.9. Daub(#) stands for the Daubechies wavelets with # coefficients.

Methods	Variances of Estimated H			
	.6	.7	.8	.9
Haar	1.3111e-05	1.4501e-05	1.7235e-05	2.1660e-05
Daub(2)	1.1914e-05	1.2353e-05	1.3133e-05	1.4408e-05
Daub(4)	1.1653e-05	1.2084e-05	1.2638e-05	1.3323e-05
Daub(8)	1.1201e-05	1.1720e-05	1.2344e-05	1.3030e-05
Daub(16)	1.0946e-05	1.1519e-05	1.2156e-05	1.2812e-05

Table 4.22: Mean values of estimated H using the wavelet-based H estimator for the five fixed-length sequence generators for $H = 0.6$ and 0.7. We give 95% confidence intervals for the means in parentheses.

Methods	Mean Values of Estimated H and ΔH			
	.6		.7	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
F-ARIMA	.5974 (.593, .601)	-0.427	.6990 (.693, .704)	-0.142
FFT	.6005 (.596, .604)	+0.083	.6967 (.692, .700)	-0.469
FGN-DW	.6013 (.574, .629)	+0.214	.6987 (.671, .726)	-0.185
RMD	.5963 (.591, .601)	-0.613	.6907 (.684, .696)	-1.332
SRA	.5848 (.579, .589)	-2.528	.6797 (.674, .685)	-2.899

Table 4.23: Mean values of estimated H using the wavelet-based H estimator for the five fixed-length sequence generators for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means in parentheses.

Methods	Mean Values of Estimated H and ΔH			
	.8		.9	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
F-ARIMA	.7947 (.787, .801)	-0.663	.8900 (.880, .899)	-1.115
FFT	.7862 (.782, .790)	-1.719	.8639 (.859, .867)	-4.012
FGN-DW	.7962 (.769, .824)	-0.474	.8938 (.866, .921)	-0.694
RMD	.7805 (.773, .787)	-2.443	.859 2 (.852, .866)	-4.536
SRA	.7700 (.763, .776)	-3.744	.8499 (.842, .856)	-5.568

but for $H = 0.9$, the accuracy of the match is lower. Relative error was +0.08%, -0.47%, -1.75% and -4.11%, respectively.

The estimated values of the F-ARIMA method were similar to the FFT method. For $H = 0.6$, 0.7, 0.8 and 0.9, all values of the Hurst parameter from the sample sequences were lower than the required values. Relative error was -0.43%, -0.14%, -0.66% and -1.12%, respectively.

The FGN-DW method demonstrated a high level of accuracy and was fast. For $H = 0.6$, 0.7, 0.8 and 0.9, the relative error was +0.21%, -0.19%, -0.47% and -0.69%, respectively.

The RMD method generated approximately self-similar sequences [84], [71]. For $H = 0.6$, 0.7, 0.8 and 0.9, the Hurst parameter tended to be lower than the required value. Relative error was -0.61%, -1.33%, -2.44% and -4.54%, respectively.

The SRA method results were similar to the RMD results. SRA generated self-similar sequences with the most biased H values. For a more detailed discussion, see [68], [71].

(b) The results for Whittle's MLE with the corresponding 95% confidence intervals $\hat{H} \pm 1.96\hat{\sigma}_{\hat{H}}$, (see Tables 4.24 and 4.25), show that for all input H values, the FFT and the FGN-DW methods produced sequences with less biased H values than other methods.

The FFT method demonstrated a high level of accuracy. For $H = 0.6$, 0.7, 0.8 and 0.9, the values of the Hurst parameter from the sample sequences match the required values very well. Relative error was +0.03%, +0.03%, +0.03% and +0.02%, respectively.

Using the F-ARIMA method, for $H = 0.6$, 0.7, 0.8 and 0.9, all values of the Hurst parameter from the sample sequences were lower than the required values. Relative error was -3.28%, -5.31%, -6.64% and -7.51%, respectively.

The FGN-DW method is more accurate than the F-ARIMA, RMD and SRA methods, but not the FFT method. For $H = 0.6$, 0.7, 0.8 and 0.9, the relative error was -2.52%, -3.92%, -4.75% and -5.22%, respectively.

The RMD method generated approximately self-similar sequences [84], [71]. For $H = 0.6$, 0.7, 0.8 and 0.9, the Hurst parameter tended to be

Table 4.24: Mean values of estimated H using Whittle's MLE for the five fixed-length sequence generators for $H = 0.6$ and 0.7. We give 95% confidence intervals for the means in parentheses.

Methods	Mean Values of Estimated H and ΔH			
	.6		.7	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
F-ARIMA	.5803 (.571, .590)	-3.281	.6628 (.654, .672)	-5.308
FFT	.6002 (.591, .610)	+0.027	.7002 (.691, .710)	+0.033
FGN-DW	.5849 (.575, .594)	-2.521	.6725 (.663, .682)	-3.924
RMD	.5765 (.567, .586)	-3.910	.6567 (.647, .666)	-6.180
SRA	.5762 (.567, .586)	-3.965	.6563 (.647, .666)	-6.249

Table 4.25: Mean values of estimated H using Whittle's MLE for the five fixed-length sequence generators for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means in parentheses.

Methods	Mean Values of Estimated H and ΔH			
	.8		.9	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
F-ARIMA	.7469 (.738, .756)	-6.642	.8324 (.823, .842)	-7.507
FFT	.8003 (.791, .809)	+0.033	.9002 (.891, .909)	+0.024
FGN-DW	.762 (.753, .771)	-4.745	.853 (.844, .862)	-5.223
RMD	.7401 (.731, .749)	-7.482	.8261 (.817, .835)	-8.214
SRA	.7395 (.730, .749)	-7.567	.8252 (.816, .834)	-8.311

lower than the required value. Relative error was -3.91%, -6.18%, -7.48% and -8.21%, respectively.

The SRA method results were similar to the RMD results. It generated self-similar sequences with the most biased H values. For a more detailed discussion, see [68], [71].

(c) Sequence plots show stronger data correlation as the H value increased; see Figure 4.11 for the FGN-DW method. In other words, generated sequences demonstrated evidence of LRD properties.

Our results show that all five generators produced approximately self-similar sequences, with the relative inaccuracy ΔH increasing with H, but

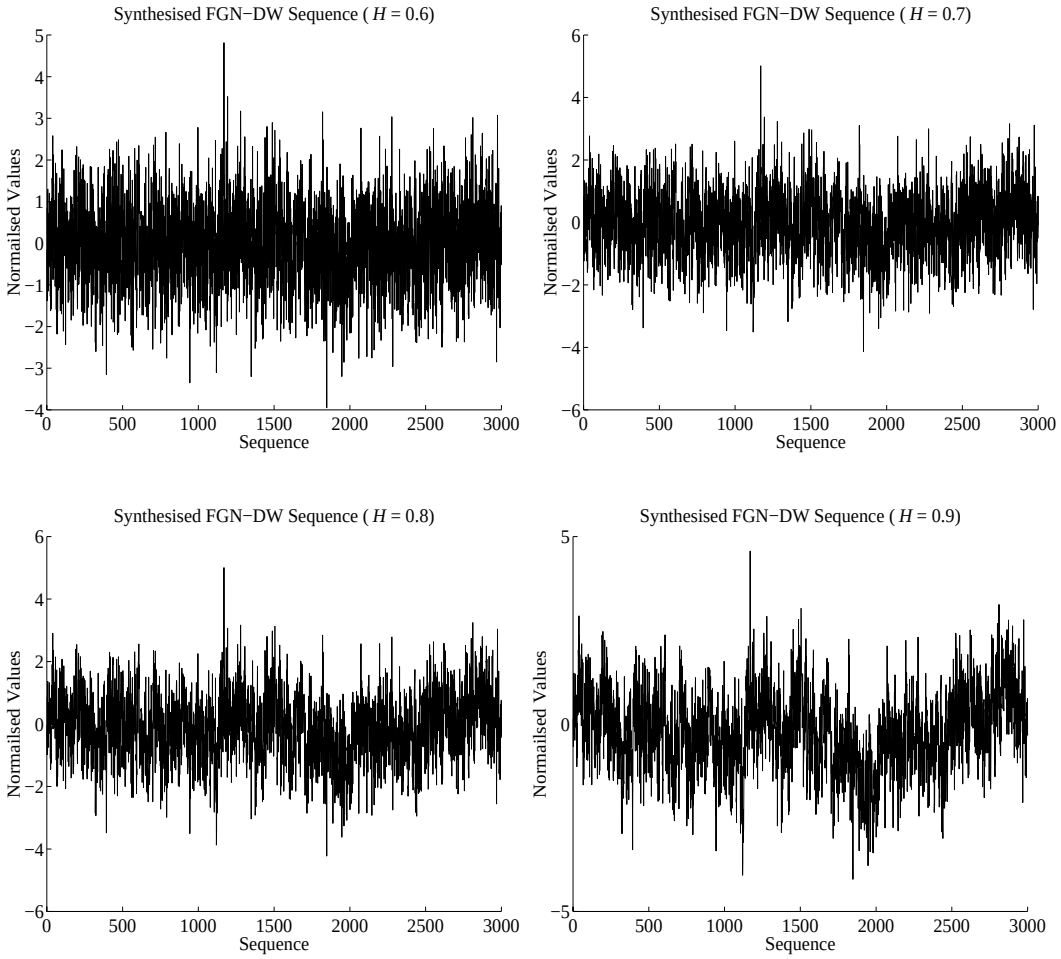


Figure 4.11: Sequence plots for the FGN-DW method ($H = 0.6, 0.7, 0.8, 0.9$).

always remaining below 9%.

4.5.2 Complexity and Speed of Generation

The computational complexities of the five fixed-length sequence generators for generating pseudo-random self-similar sequences of a given length n are shown in Table 4.26. The number of arithmetic operations per number required by each of them are also shown in Table 4.26. The F-ARIMA method was the slowest, requiring $178n^2$ + 3,936 time operations per number. FFT was also slower than the FGN-DW, RMD and SRA methods.

The results of our experimental analysis of the mean times required by each generator are shown in Figure 4.12. Our main conclusions are:

(a) The F-ARIMA method was the slowest of the five methods, as expected.

(b) On average, the FFT method was faster than F-ARIMA, but slower than the other three. This was caused by the relatively high complexity of the inverse FFT algorithm. FFT requires $O(n\log n)$ computations to generate n numbers [130] and $49\log n$ + 4,175 time operations per number.

(c) The FGN-DW, RMD and SRA methods were equally fast. The theoretical complexity of forming a spectral density, and constructing normally distributed complex numbers, is $O(1)$, while the inverse DWT is $O(n)$

Table 4.26: Computational complexities and arithmetic operations required by each of the five fixed-length sequence generators.

Method	Complexity	Operations per Number
F-ARIMA	$O(n^2)$	$178n^2$ + 3,936
FFT	$O(n\log n)$	$49\log n$ + 4,175
FGN-DW	$O(n)$	4,091
RMD	$O(n)$	4,103
SRA	$O(n)$	4,167

[124], [170]. Thus, the time complexity of FGN-DW is also $O(n)$ and this method requires 4,091 operations per number. The theoretical algorithmic complexity of the RMD and SRA methods is also $O(n)$ [128] and they require 4,103 and 4,167 operations per number, respectively.

Overall, our results showed that the generator based on FGN-DW is the fastest of the five generators if long sequences of self-similar pseudo-random numbers are required.

4.6 Sequential Generators versus Fixed-Length Sequence Generators

We have presented the results of a comparative analysis of six sequential generators of (long) pseudo-random self-similar sequences. All six sequential generators, based on the FBNDP, SFRP, MGIP, PMPP, SRP-FGN and SAP methods, generated approximately self-similar sequences; SRP-FGN was the most accurate. However, our results show that for most input H values, the

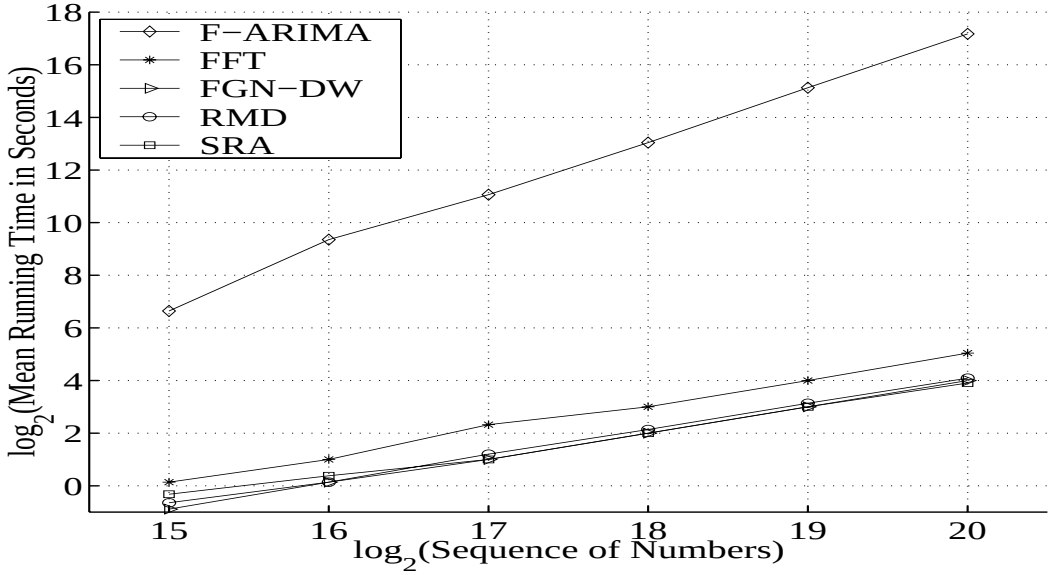


Figure 4.12: Mean running times of the five fixed-length sequence generators. Running times were obtained using the SunOS 5.7 `time` command on a Pentium II (233 MHz, 512 MB); each mean is averaged over 30 iterations.

MGIP and SAP-based generators were strongly biased. The FBNDP method was biased for $H = 0.9$.

The analysis of mean times required to generate sequences of a given length demonstrates that all six sequential generators are more attractive than the F-ARIMA-based generator for practical simulation studies of computer networks because they are much faster. However, these generators require more input parameters, and selecting appropriate values is a problem that remains. Furthermore, in the case of SAP, the question of how to define the relationship between the Hurst parameter and the two shape parameters (i.e., $\alpha_1 > 0$ and $\alpha_2 > 0$) of a beta-distribution remains.

We have also presented the results of a comparative analysis of five fixed-length generators of self-similar sequences. All five, based on the F-ARIMA, FFT, FGN-DW, RMD and SRA methods, generated approximately self-similar sequences, with the relative inaccuracy of the resultant $\hat{H}$ below 9% if $0.6 \leq H \leq 0.9$. However, the analysis of mean times required to generate sequences of a given length shows that the FFT, FGN-DW, RMD, and SRA generators are more attractive for practical simulation studies of computer networks because they generate sequences much faster. When the wavelet-based H estimator and Whittle's MLE (the least biased of the H estimation techniques), are applied (see Chapter 3), FFT produces the most accurate results, with the FGN-DW results almost as accurate. Thus, FFT and FGN-DW are the most practical in both accuracy and speed for simulation studies with self-similar input.

Table 4.27 and Figure 4.13 show a comparison of the three fastest and most accurate generators of the six sequential and five fixed-length sequence generators: the SRP-FGN, FFT and FGN-DW generators. While estimated H values for the FGN-DW method obtained using the wavelet-based H estimator were more accurate than those for the SRP-FGN and FFT methods, those for the FFT method obtained using Whittle's MLE were the most accurate.

Even though the SRP-FGN and FGN-DW methods have the same computational complexity, $O(n)$, the FGN-DW method was faster than the SRP-FGN and FFT methods, as shown in Figure 4.13. Furthermore, while the SRP-FGN method required three input parameters (i.e., H, M and T), the FFT and FGN-DW methods required only the Hurst parameter H. Thus, the

FFT method was more accurate than the SRP-FGN and FGN-DW methods, and the FGN-DW method was faster than the other two. Overall, all three methods are more attractive for practical simulation studies of telecommunication networks than the other nine generators.

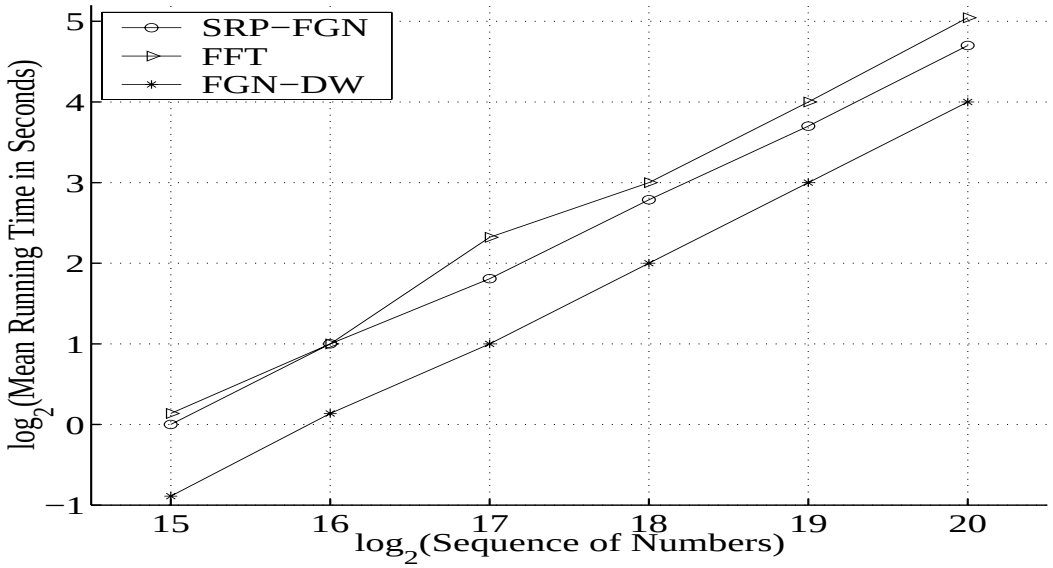


Figure 4.13: Mean running times of the SRP-FGN, FFT and FGN-DW generators. Running times were obtained using the SunOS 5.7 `time` command on a Pentium II (233 MHz, 512 MB); each mean is averaged over 30 iterations.

Table 4.27: Comparison of the most efficient SRP-FGN, FFT and FGN-DW methods. Relative inaccuracies of mean values of estimated H obtained using the wavelet-based H estimator and Whittle's MLE.

Estimator	Method	ΔH			
		.6	.7	.8	.9
Wavelet-Based	SRP-FGN	−0.968	−0.569	+0.700	+0.344
	FFT	+0.083	−0.469	−1.719	−4.012
	FGN-DW	+0.214	−0.185	−0.474	−0.694
Whittle's MLE	SRP-FGN	+0.110	+1.905	+2.539	+2.526
	FFT	+0.027	+0.033	+0.033	+0.024
	FGN-DW	−2.521	−3.924	−4.745	−5.223

4.7 Conclusions

One of the problems that computer network researchers face when conducting simulation studies is how to generate long synthetic sequential self-similar sequences. Three aspects must be considered: (i) how accurately self-similar processes can be generated, (ii) how quickly the methods generate long self-similar sequences, and (iii) how appropriately self-similar processes can be used in sequential simulations.

Most of the existing synthetic methods for generating self-similar sequences require large amounts of CPU time. Some current methods must store either part, or all, of the sequence in memory before generating numbers of a sequence. In addition, they are often inaccurate or inappropriate in simulation studies of computer networks. Sequential generators of self-similar sequences depend on the level of approximation, and need several input parameters to be assumed, while fixed-length sequence generators need only to assume the Hurst parameter to generate self-similar sequences.

Certainly, more efficient and accurate generators of self-similar sequences of pseudo-random numbers are needed. A comparative study of self-similar pseudo-random teletraffic generators was undertaken. Overall, our results, obtained using the wavelet-based H estimator and Whittle's MLE, which are the least biased of the H estimation techniques considered in Chapter 3, have revealed that the fastest and most accurate generators of the six sequential and five fixed-length sequence generators considered are the SRP-FGN, FFT and FGN-DW methods. However, these methods have both strengths and weaknesses. The FFT and FGN-DW methods are more attractive for non-sequential simulations, because they can generate the required number of sequences more accurately and quickly than the SRP-FGN method. If the FFT and FGN-DW methods are used for sequential simulations, sufficient numbers of sequences must be generated before the simulation begins. However, the required number of sequences is not easy to predict in practical simulations. On the other hand, the SRP-FGN method is more attractive for sequential simulations, because it does not need to "know" the required number of sequences beforehand. Unfortunately, this method is less accurate and requires more generating time than the FFT method.

The FGN-DW method was used to synthesise VBR video traffic, as dis-

cussed in Chapter 6, and the SRP-FGN method was used to investigate the queueing behaviour in steady-state simulation studies of self-similar and non-self-similar queueing systems, as outlined in Chapter 7.

For simulation studies of computer networks, self-similar processes with arbitrary marginal distributions are needed. In order to obtain these, it is needed to transform a given sequence from the exactly self-similar FGN process into a sequence that represents a realisation of a specific process. In the next chapter, we will investigate such a transformation for five different marginal distributions by studying their impact on the H values and ACFs of input processes.

Chapter 5

GENERATION OF A SELF-SIMILAR PROCESS WITH AN ARBITRARY MARGINAL DISTRIBUTION

5.1 Introduction

Simulation studies of telecommunication networks often require generation of random variables, or stochastic processes, characterised by different probability distributions. Thus far we have discussed generation of self-similar sequences with a normal marginal distribution. We can obtain sequences of numbers from normal distributions with different mean values and variances by applying such standard transformations as shifting and rescaling/normalisation. In practical simulation studies, however, generation of self-similar processes of several different non-normal marginal probability distributions might be required. The most common method of transforming realisations of one random variable into realisations of another random variable is based on the inverse of cumulative distribution functions. This method and its application in transformations of self-similar processes are discussed in Section 5.2.2 in detail.

The theory of transformations of strictly and second-order self-similar processes has not been fully developed. In this chapter, we look at applica-

tions of the inverse cumulative distribution function (ICDF) transformation[1] to the generation of long-range dependent sequences governed by non-normal marginal distributions from long-range dependent sequences of normal marginal distributions.

For studying the properties of the ICDF transformation in the context of self-similar process we investigate its properties when it is applied to the exact self-similar process, taking the self-similar FGN process as the reference [74], [76], [124]. This FGN process was generated by the *Durbin-Levinson* algorithm, described in Section 3.2. We consider output processes with different marginal probability distributions (exponential, gamma, Pareto, uniform and Weibull), with finite and infinite variances, and compare H parameters and ACFs of output processes with those characterising input self-similar FGN processes. Our findings are summarised in Section 5.4.

5.2 Generation of LRD Self-Similar Processes with an Arbitrary Marginal Distribution

Simulation studies of telecommunication networks require a mechanism to transform self-similar processes into processes with arbitrary marginal distributions [92], [94], [124]. In this chapter, we investigate preservation of the LRD self-similarity and ACFs in output processes with different marginal distributions when transforming exact self-similar FGN processes into processes with five different marginal distributions (exponential, gamma, Pareto, uniform and Weibull), with finite and infinite variances, using the ICDF transformation.

5.2.1 The Method of the Inverse Cumulative Distribution Function

The ICDF transformation is based on the observation that given any random variable X_i with a CDF $F(x)$, the random variable $u = F(x)$ is independent

[1]The TES (Transform-Expand-Sample) process [107], [108] and the ARTA (Autoregressive-to-Anything) process [14], [15] could be used the generation of correlated sequences.

and uniformly distributed between 0 and 1. Therefore, x can be obtained by generating uniform realisations and calculating $x = F^{-1}(u)$.

We assume that a process $\mathbf{X}$ is a Gaussian process with zero mean, variance of one and a given ACF $\{\rho_k\}$. Let $F_X(x)$ be its marginal CDF and $F_Y(y)$ be a marginal CDF of the process $\mathbf{Y}$. The process $\mathbf{Y}$ with the desired marginal CDF $F_Y(y)$ can be generated by the ICDF transformation from the process $\mathbf{X}$. Following the ICDF transformation, when transforming a random variable X_i into a random variable Y_i, we use the formula:

$$F_X(x) = F_Y(y), \tag{5.1}$$

Thus:

$$y = F_Y^{-1}\left(F_X(x)\right) \tag{5.2}$$

hence the method is called the ICDF transformation.

Here we consider five marginal distributions of output processes: exponential, gamma, Pareto, uniform and Weibull distributions that are frequently used in simulation practice.

Exponential Marginal Probability Distribution

The exponential distribution has the CDF

$$F_Y(y) = \begin{cases} 0, & \text{for } y \leq 0, \\ 1 - e^{-\lambda y}, & \text{for } y > 0, \end{cases} \tag{5.3}$$

where λ is the mean of a random variable Y.

To generate a random variable Y with an exponentially distribution from a random variable X of normal distribution, one applies the transformation:

$$y_i = -(\frac{1}{\lambda}) * \log(F_X(x_i)), \tag{5.4}$$

where $F_X(\cdot)$ is the CDF of the normal distribution.

Gamma Marginal Probability Distribution

The gamma distribution has the CDF

$$F_Y(y) = \begin{cases} 0, & \text{for } y \leq 0, \\ 1 - e^{-y/\beta_\Gamma} \sum_{j=0}^{\alpha_\Gamma - 1} \frac{(y/\beta_\Gamma)^j}{j!}, & \text{for } y > 0, \end{cases} \tag{5.5}$$

if α_Γ (the shape parameter) is a natural number, and β_Γ is the scale parameter, $\beta_\Gamma > 0$. If α_Γ is not integer, then there is no closed form of the CDF for the gamma distribution. A few methods for generating pseudo-random numbers governed by such a gamma probability distribution have been proposed [85] (pp. 487-490). We chose the Newton-Raphson technique, and used an implementation of this technique given in [131].

Pareto Marginal Probability Distribution

The Pareto distribution has the CDF

$$F_Y(y) = \begin{cases} 0, & \text{for } y < 1, \\ 1 - \left(\frac{b}{y}\right)^\alpha, & \text{for } 1 \leq y \leq \infty, \end{cases} \tag{5.6}$$

where α is a shape parameter and b is the minimum allowed value of y, $0 < b \leq y$. We assume $b = 1$ [also see Equation (2.21)]. To generate random variables with a Pareto distribution Y from random variables of normal distribution X, one applies the transformation:

$$y_i = 1/(F_X(x_i))^{1/\alpha}. \tag{5.7}$$

Uniform Marginal Probability Distribution

The uniform distribution has the CDF

$$F_Y(y) = \begin{cases} 0, & \text{for } y < a, \text{and } b < y \\ \frac{y-a}{b-a}, & \text{for } a \leq y \leq b, \end{cases} \tag{5.8}$$

where a is a lower limit and b is an upper limit, $a < b$. To generate pseudo-random numbers with a uniform distribution Y from random variables of normal distribution X, one applies the transformation:

$$y_i = a + (b - a)F_X(x_i). \tag{5.9}$$

Weibull Marginal Probability Distribution

The Weibull distribution has the CDF

$$F_Y(y) = \begin{cases} 0, & \text{for } y \leq 0, \\ 1 - e^{-(y/\beta)^\alpha}, & \text{for } 0 < y, \end{cases} \tag{5.10}$$

where α is a shape parameter and β is a scale parameter. To generate a random variable with a Weibull distribution Y from a random variable of normal distribution X, one applies the transformation:

$$y_i = \beta \left(-\log(F_X(x_i))\right)^{1/\alpha}. \tag{5.11}$$

5.2.2 Effects of Transformation

In simulation studies of such stochastic dynamic processes as those that occur in telecommunication networks one needs to decide both about their marginal probability distributions and autocorrelation structures. The problem of generating a strictly and/or second-order self-similar process of a given marginal distribution and an autocorrelation structure is difficult and has not been fully solved. No existing procedure is entirely satisfactory in terms of mathematical rigour, computational efficiency, accuracy of approximation, and precise and concise parameterisation [43].

Applications of the transformation in Equation (5.2) to transformations of correlated processes have been studied by several researchers [64], [76], [107]. In general, as proved by Beran (see [9], pp. 67-73), a transformation $y = G(x)$ applied to a strictly and/or second-order self-similar LRD sequence of numbers $\{x_1, x_2, \ldots\}$ does not preserve LRD properties in the output sequence $\{y_1, y_2, \ldots\}$. However, as proved in [64], if in Equation (5.2):

1. $F_X(\cdot)$ represents normal distribution,
2. $\{x_1, x_2, \ldots\}$ is a self-similar LRD sequence,
3. the transformation $G^2(x)$ is integrable, i.e.,

$$\int_{-\infty}^{+\infty} G^2(x) dF_X(x) \;<\; \infty, \text{ and} \tag{5.12}$$

4. $E(XY) \neq 0$,

then the output sequence $\{y_1, y_2, \ldots\}$ is asymptotically self-similar, with the same coefficient H as the sequence $\{x_1, x_2, \ldots\}$.

Related issues have been investigated. Wise et al. [174] and Liu and Munson [96] showed that, following the transformation of marginal distribution, the

transformation of ordinary ACF can be characterised when the input process is normal. They also indicated other processes for which this could be applied. Huang et al. [64] demonstrated that, if the process $\mathbf{X}$ is self-similar and has a normal marginal distribution, under general conditions, the output process $\mathbf{Y}$ is an asymptotically self-similar process with the same Hurst parameter ($\frac{1}{2} < H < 1$); for proof of the invariance of the Hurst parameter H, see [64]. Geist and Westall [44] demonstrated that arrival processes, obtained by the FFT method proposed by Paxson [124] (see Section 4.4.1), have ACFs that are consistent with LRD. However, it has not been fully developed to generate self-similar processes with arbitrary marginal distributions from self-similar processes with (normal) marginal distributions and autocorrelation structures [43], [44].

5.3 Numerical Results

The numerical results of this section are used to investigate how well the LRD self-similarity and ACFs of the original Gaussian processes are preserved when they are converted into processes with non-normal marginal distributions. For each of $H = 0.6$, 0.7, 0.8 and 0.9, 100 exact self-similar sample sequences of 32,768 (2^{15}) numbers starting from different random seeds are used.

The following five different marginal distributions are investigated: the exponential distribution with $\lambda = 9$; the uniform distribution with $a = 0$ and $b = 1$; the gamma distribution with $\alpha = 2$ and $\beta = 1$; the Pareto distributions with $\alpha = 1.2$, 1.4, 1.6, 1.8 and 20.0; and the Weibull distribution with $\alpha = 2$ and $\beta = 1$.

5.3.1 Analysis of H Parameters

Figure 5.1 shows an exact self-similar sequence taken from the exact self-similar FGN process, with mean = 0, variance = 1 and $H = 0.9$. Figures 5.2 – 5.10 show realisations of processes obtained by applying transformation (5.2) to the process depicted in Figure 5.1, for five different output marginal distributions described in Section 5.2.1: exponential, gamma, Pareto with $\alpha = 20.0$, uniform and Weibull, respectively. The estimates of H and the 95% confi-

dence intervals for these estimates (shown in parentheses), obtained using the wavelet-based H estimator, are 0.8940 (.8819, .9062), 0.9015 (.8894, .9137), 0.8929 (.8807, .9050), 0.9052 (.8930, .9173) and 0.9094 (.8972, .9215). Similarly, using Whittle's MLE of H, those are obtained: 0.8855 (.8810, .8901), 0.8938 (.8892, .8983), 0.8795 (.8750, .8841), 0.8922 (.8877, .8967) and 0.8997 (.8952, .9042). The LRD properties of the output process are preserved well.

However, the estimates of H depicted in Figures 5.7 – 5.10, for processes with Pareto marginal distributions with $\alpha = 1.2$, 1.4, 1.6 and 1.8, obtained using the wavelet-based H estimator, are 0.4285 (.4010, .4560), 0.5316 (.5041, .5591), 0.6098 (.5823, .6373) and 0.6667 (.63916, .6942), for $H = 0.9$, 0.8, 0.7 and 0.6 in the input processes, respectively. Using Whittle's MLE, those are also obtained: 0.6465 (.6371, .6559), 0.6868 (.6775, .6962), 0.7223 (.7130, .7316) and 0.7511 (.7419, .7603), for $H = 0.9$, 0.8, 0.7 and 0.6 in the input processes, respectively. As we see, the LRD properties in the output processes are not preserved. Note that all the output distributions considered have finite means and infinite variances.

For more rigorous proof, we analyse the self-similar sequences with five different marginal distributions generated by the exact self-similar FGN process using the wavelet-based H estimator and Whittle's MLE. Tables 5.1 – 5.4 show the estimated mean $\hat{H}$ values of the resulting process; see also Figure 5.11. For $H = 0.6$, 0.7, 0.8 and 0.9, each mean $\hat{H}$ value is obtained from 100 replications. We give 95% confidence intervals for the means in parentheses. All results in Tables 5.1 – 5.4 are presented together with their relative errors, defined in Equation (3.3). These results in Tables 5.1 – 5.2 show that all confidence intervals are within the required values, except for those with Pareto distribution is $\alpha = 1.2$, 1.4, 1.6 and 1.8. Values in Tables 5.3 – 5.4 show that, for gamma ($H = 0.6$), uniform ($H = 0.6$ and 0.7) and Weibull ($H = 0.6$, 0.7, 0.8 and 0.9), confidence intervals are within the required values, but others are slightly underestimated (i.e., $|\Delta H| < 4\%$).

If one considers output marginal distributions with infinite variances, then as it was proved in [64], the H values of the input process are not preserved. This fact is illustrated by results presented in Tables 5.1 – 5.4, where Pareto distributions with infinite variances ($\alpha = 1.2$, 1.4, 1.6 and 1.8) have been added to the previously considered five output distributions with finite variances for $H = 0.6$, 0.7, 0.8 and 0.9.

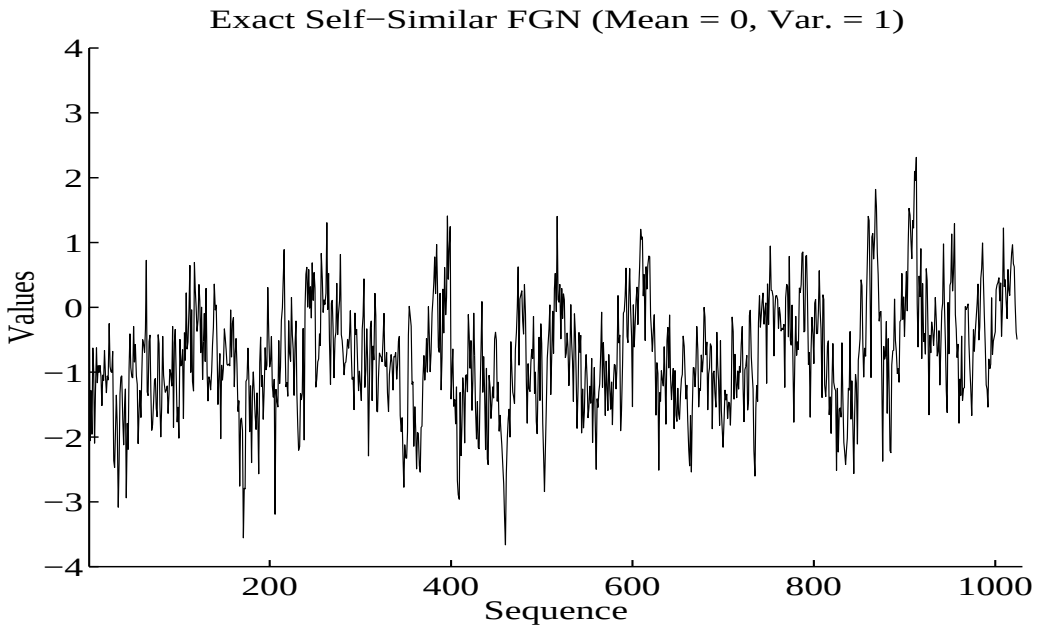


Figure 5.1: A realisation of the exact self-similar FGN process used as the input process for ICDF transformation, with $H = 0.9$.

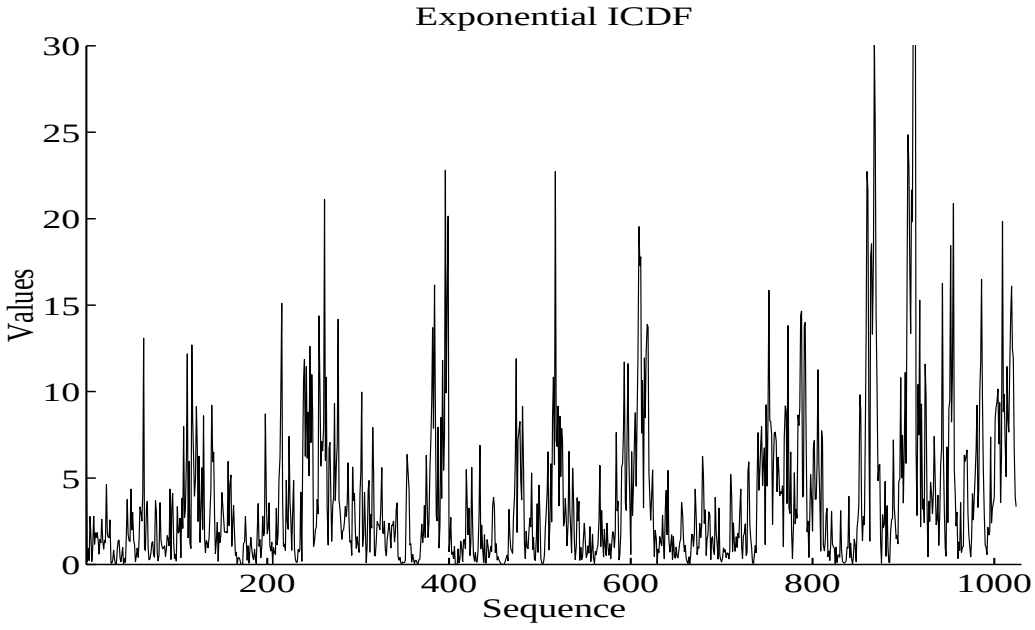


Figure 5.2: A realisation of the self-similar process with the exponential marginal distribution, with $H = 0.8855$ (on the basis of Whittle's MLE), using the exact self-similar FGN process with $H = 0.9$ as the input process.

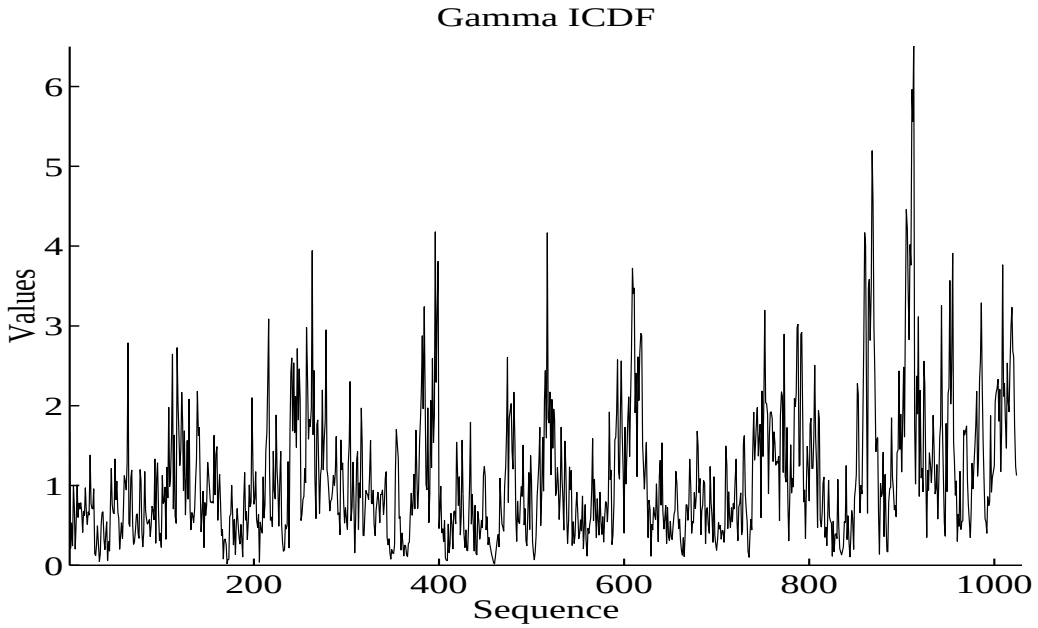


Figure 5.3: A realisation of the self-similar process with the gamma marginal distribution, with $H = 0.8938$ (on the basis of Whittle's MLE), using the exact self-similar FGN process with $H = 0.9$ as the input process.

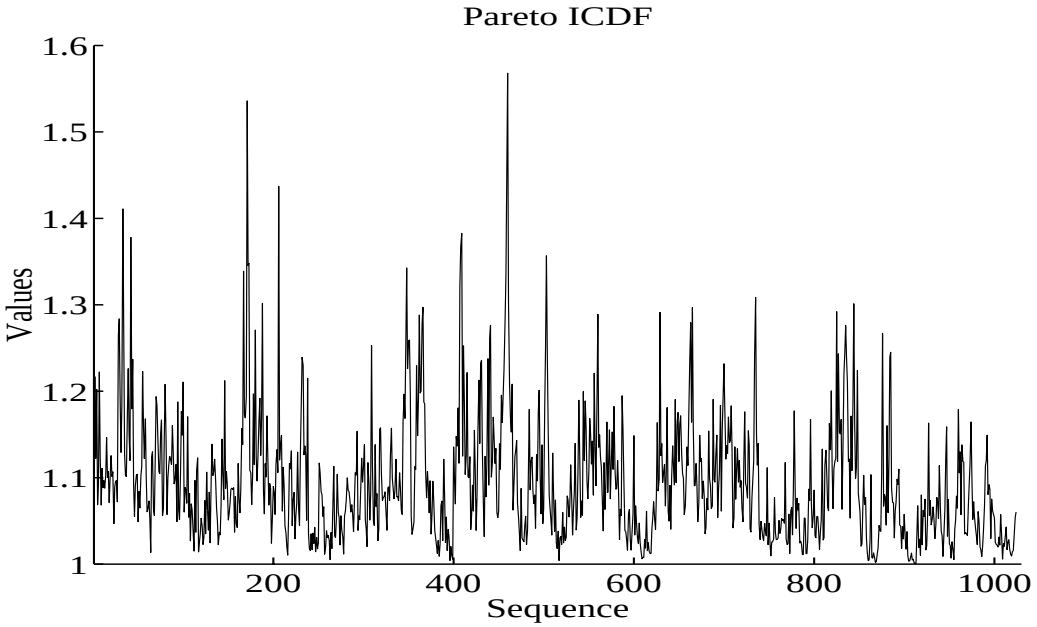


Figure 5.4: A realisation of the self-similar process with the Pareto marginal distribution, with $\alpha = 20.0$ and $H = 0.8795$ (on the basis of Whittle's MLE), using the exact self-similar FGN process with $H = 0.9$ as the input process.

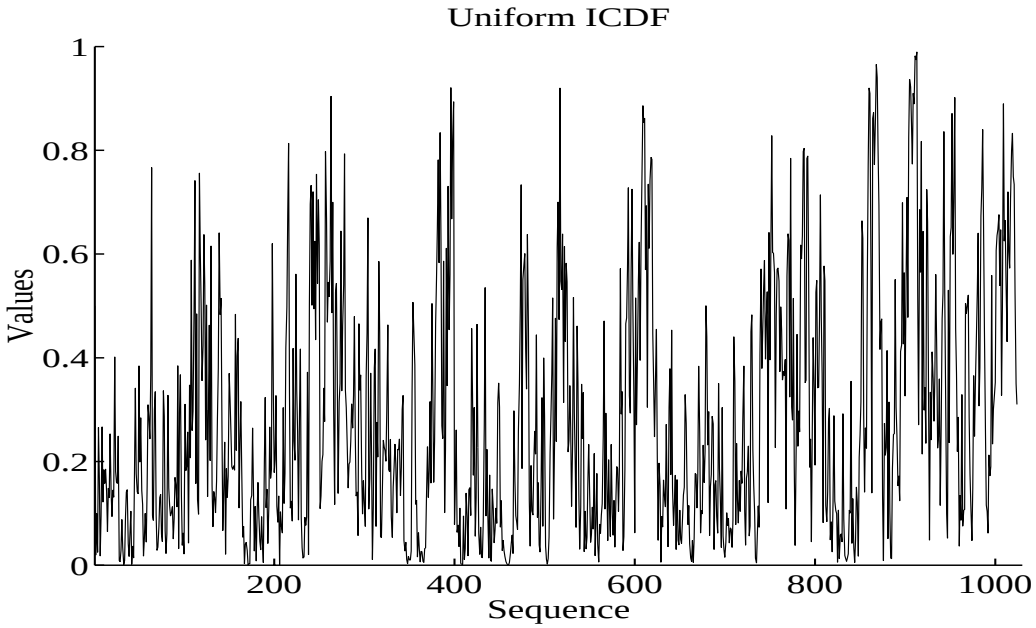


Figure 5.5: A realisation of the self-similar process with the uniform marginal distribution, with $H = 0.8922$ (on the basis of Whittle's MLE), using the exact self-similar FGN process with $H = 0.9$ as the input process.

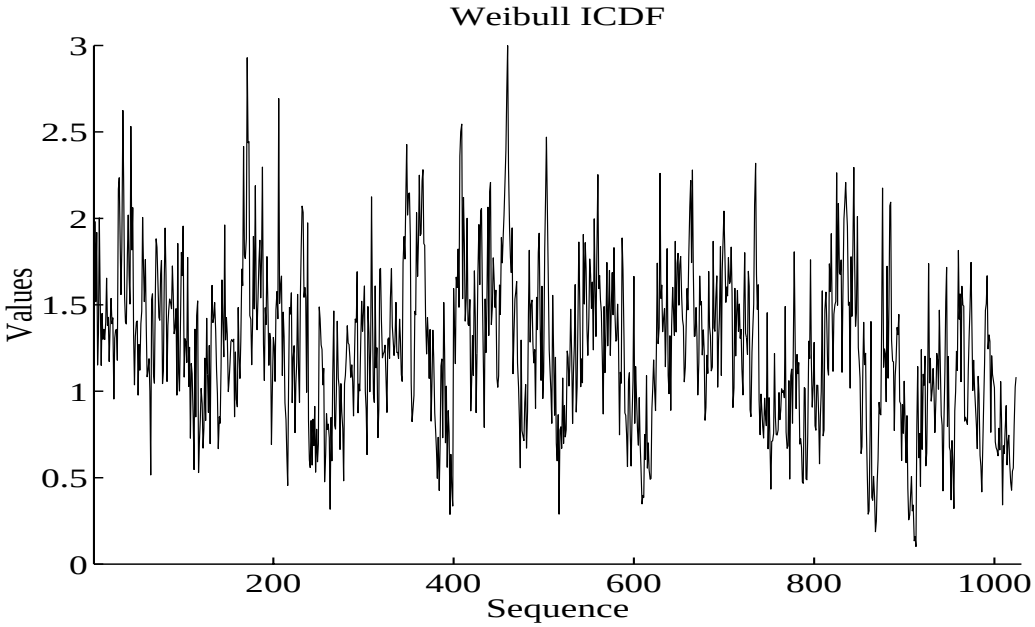


Figure 5.6: A realisation of the self-similar process with the Weibull marginal distribution, with $H = 0.8997$ (on the basis of Whittle's MLE), using the exact self-similar FGN process with $H = 0.9$ as the input process.

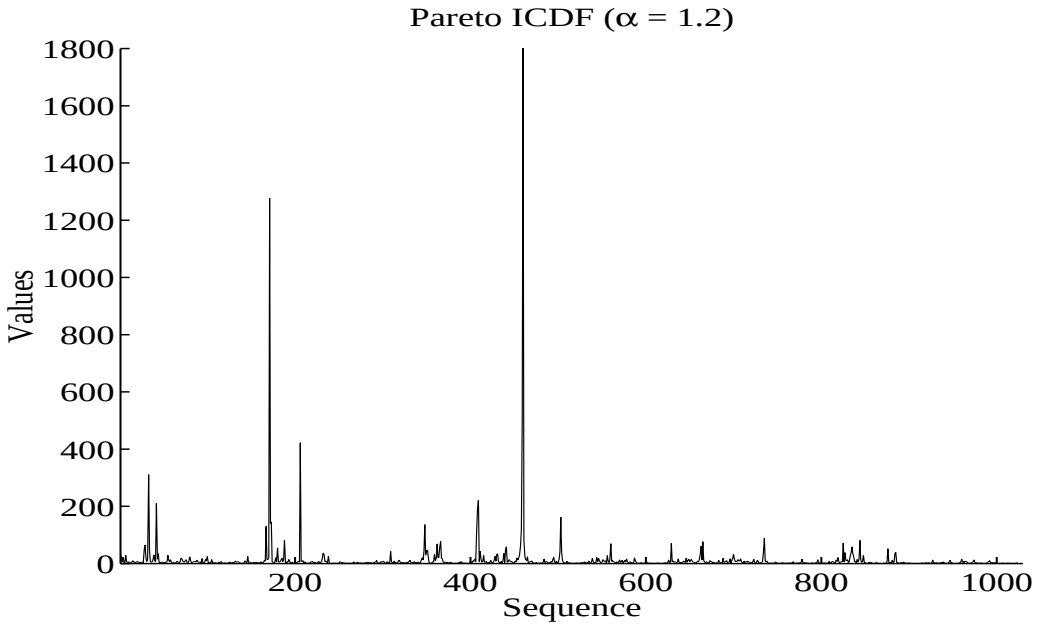


Figure 5.7: A realisation of the self-similar process with the Pareto marginal distribution, with $\alpha = 1.2$ and $H = 0.6465$ (on the basis of Whittle's MLE), using the exact self-similar FGN process with $H = 0.9$ as the input process.

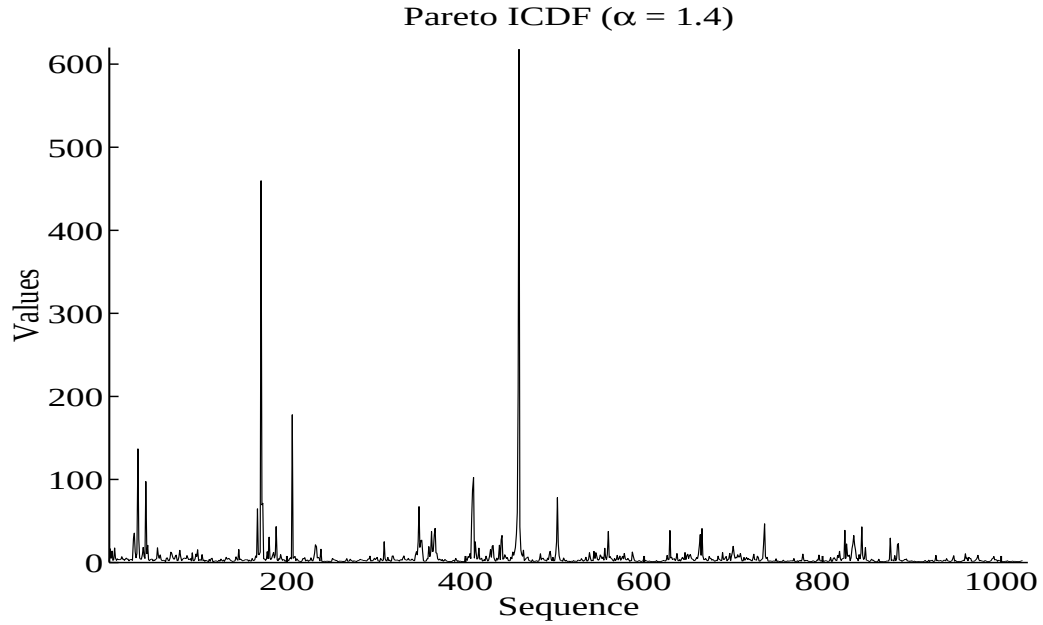


Figure 5.8: A realisation of the self-similar process with the Pareto marginal distribution, with $\alpha = 1.4$ and $H = 0.6868$ (on the basis of Whittle's MLE), using the exact self-similar FGN process with $H = 0.9$ as the input process.

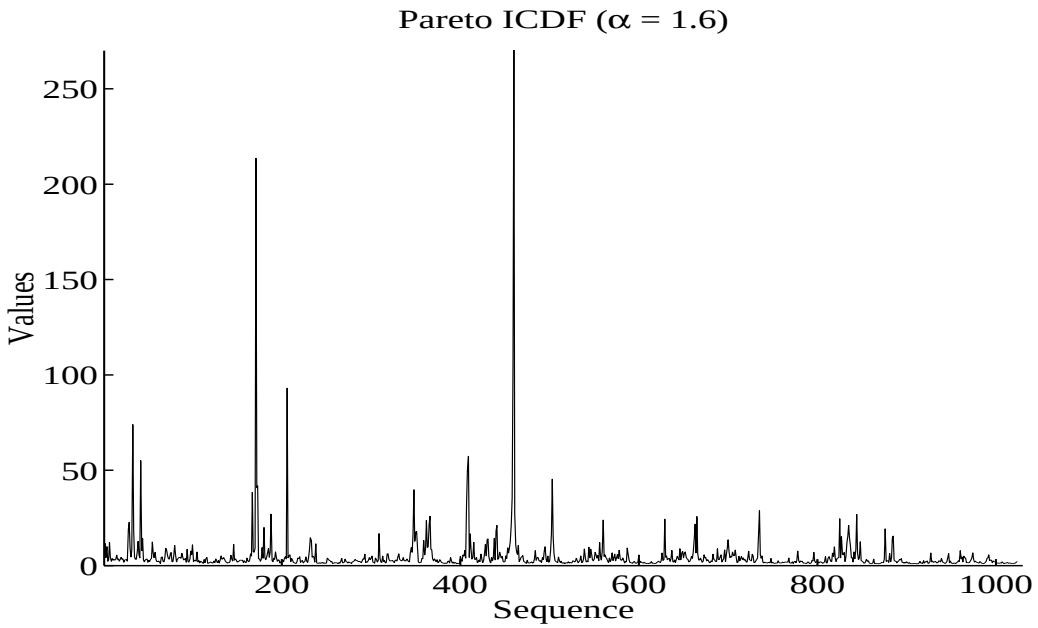


Figure 5.9: A realisation of the self-similar process with the Pareto marginal distribution, with $\alpha = 1.6$ and $H = 0.7223$ (on the basis of Whittle's MLE), using the exact self-similar FGN process with $H = 0.9$ as the input process.

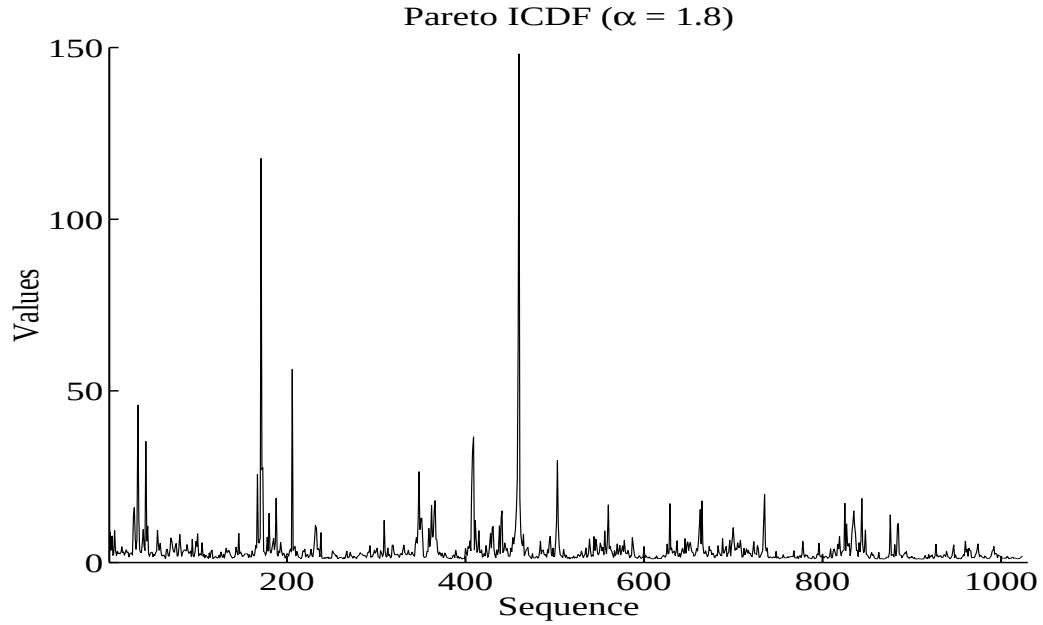


Figure 5.10: A realisation of the self-similar process with the Pareto marginal distribution, with $\alpha = 1.8$ and $H = 0.7511$ (on the basis of Whittle's MLE), using the exact self-similar FGN process with $H = 0.9$ as the input process.

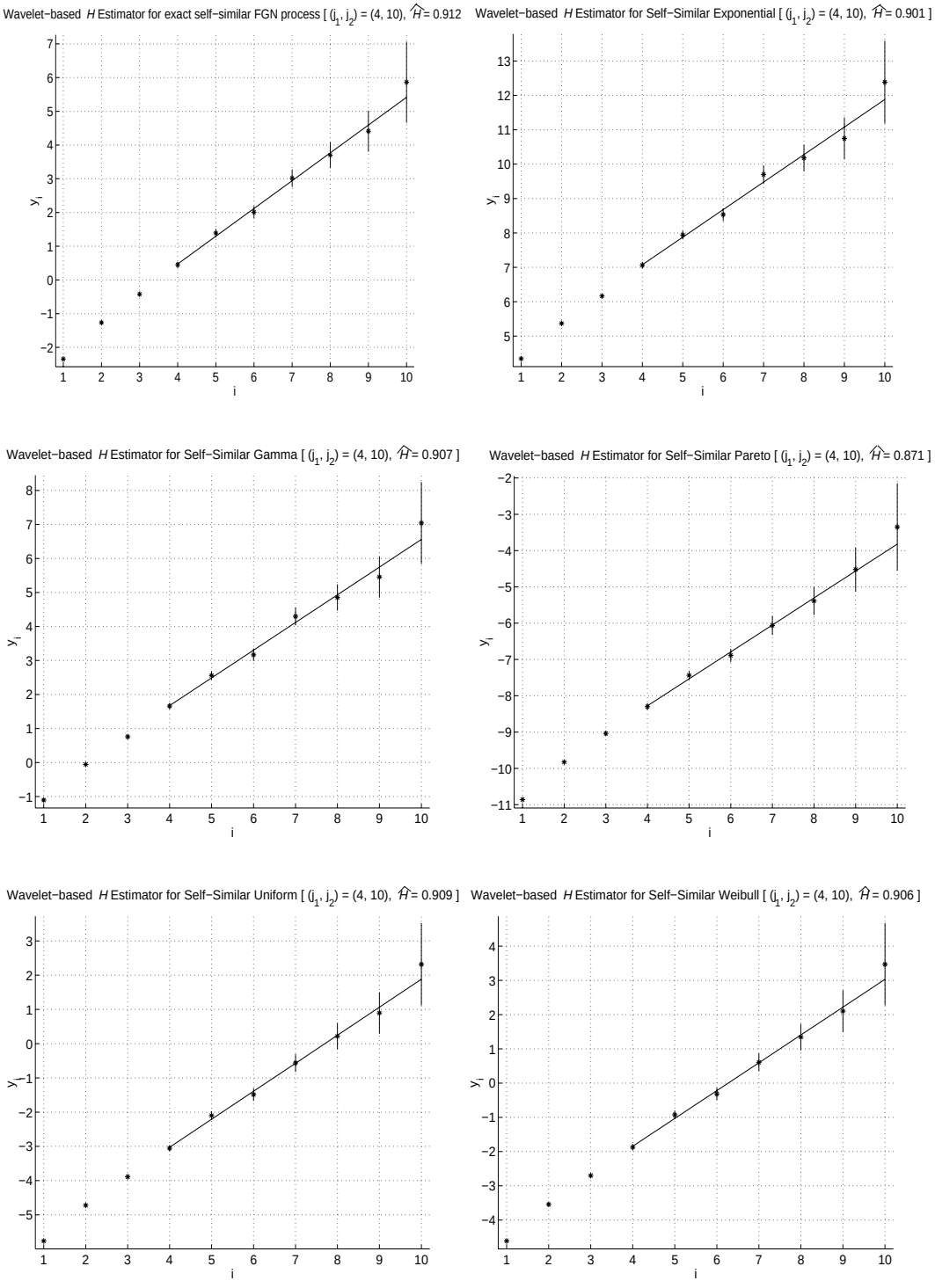


Figure 5.11: Wavelet-based H plots for the exact self-similar FGN process, five self-similar exponential, gamma, Pareto ($\alpha = 20.0$), uniform and Weibull marginal distributions for $H = 0.9$.

On the basis of our results we formulate the following hypothesis:

> Hypothesis: If *transformation (5.2) is applied to self-similar LRD processes with normal marginal distributions, then it preserves H parameter if the output marginal distribution has a finite variance.*

We think that this hypothesis could be analytically proved by showing that in the case of an infinite variance, the transformation in Equation (5.2) does

Table 5.1: Relative inaccuracy, ΔH, of mean values of estimated H obtained using the wavelet-based H estimator for the exact self-similar FGN process with different marginal distributions for $H = 0.6$ and 0.7. We give 95% confidence intervals for the mean values in parentheses.

Distribution	Mean Values of Estimated H and ΔH			
	.6		.7	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
Exponential	.5879 (.560,.615)	-2.039	.6830 (.656,.711)	-2.521
Gamma	.5945 (.567,.622)	-0.949	.6922 (.665,.720)	-1.198
Uniform	.5971 (.570,.625)	-0.514	.6964 (.669,.724)	-0.604
Weibull	.5981 (.571,.626)	-0.348	.6979 (.670,.725)	-0.394
Pareto ($\alpha = 20.0$)	.5857 (.558,.613)	-2.378	.6800 (.653,.708)	-2.862
Pareto ($\alpha = 1.2$)	.5014 (.474,.529)	-16.43	.5027 (.475,.530)	-28.19
Pareto ($\alpha = 1.4$)	.5098 (.482,.537)	-15.03	.5237 (.496,.551)	-25.18
Pareto ($\alpha = 1.6$)	.5189 (.491,.546)	-13.52	.5468 (.519,.574)	-21.89
Pareto ($\alpha = 1.8$)	.5281 (.501,.556)	-11.99	.5690 (.542,.597)	-18.71

not satisfy the assumption that its squared form in Equation (5.12) must be integrable [64].

Variances for Estimated H

Tables 5.5 – 5.6 show variances for estimated H obtained using the wavelet-based H estimator and Whittle's MLE for the exact self-similar FGN process

Table 5.2: Relative inaccuracy, ΔH, of mean values of estimated H obtained using the wavelet-based H estimator for the exact self-similar FGN process with different marginal distributions for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means in parentheses.

Distribution	Mean Values of Estimated H and ΔH			
	.8		.9	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
Exponential	.7800 (.753,.808)	-2.604	.8797 (.852,.907)	-2.356
Gamma	.7909 (.763,.818)	-1.243	.8912 (.864,.919)	-1.079
Uniform	.7953 (.768,.823)	-0.700	.8929 (.865,.920)	-0.892
Weibull	.7976 (.770,.825)	-0.410	.8975 (.870,.925)	-0.387
Pareto ($\alpha = 20.0$)	.7765 (.749,.804)	-2.940	.8764 (.849,.904)	-2.626
Pareto ($\alpha = 1.2$)	.5333 (.506,.561)	-33.34	.6300 (.603,.658)	-30.00
Pareto ($\alpha = 1.4$)	.5690 (.542,.597)	-28.87	.6778 (.650,.705)	-24.69
Pareto ($\alpha = 1.6$)	.6047 (.577,.632)	-24.42	.7177 (.690,.745)	-20.25
Pareto ($\alpha = 1.8$)	.6362 (.609,.664)	-20.47	.7495 (.722,.777)	-16.73

with different marginal distributions for $H = 0.6$, 0.7, 0.8 and 0.9. Estimated variances for the output processes with five different marginal distributions were slightly higher than the original, but those with the Pareto marginal distribution with $\alpha = 1.2$, 1.4, 1.6 and 1.8 had the highest variances. All variances gradually increased as the H value increased.

Table 5.3: Relative inaccuracy, ΔH, of mean values of estimated H obtained using Whittle's MLE for the exact self-similar FGN process with different marginal distributions for $H = 0.6$ and 0.7. We give 95% confidence intervals for the mean values in parentheses.

Distribution	Mean Values of Estimated H and ΔH			
	.6		.7	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
Exponential	.5856 (.576,.595)	-2.394	.6774 (.668,.687)	-3.230
Gamma	.5923 (.583,.602)	-1.290	.6878 (.678,.697)	-1.748
Uniform	.5962 (.587,.606)	-0.628	.6928 (.683,.702)	-1.034
Weibull	.5981 (.589,.608)	-0.313	.6969 (.688,.706)	-0.445
Pareto ($\alpha = 20.0$)	.5833 (.574,.593)	-2.780	.6737 (.664,.683)	-3.760
Pareto ($\alpha = 1.2$)	.5050 (.495,.515)	-15.84	.5155 (.506,.525)	-26.35
Pareto ($\alpha = 1.4$)	.5091 (.499,.519)	-15.15	.5277 (.518,.537)	-24.61
Pareto ($\alpha = 1.6$)	.5152 (.506,.525)	-14.13	.5426 (.533,.552)	-22.49
Pareto ($\alpha = 1.8$)	.5224 (.513,.532)	-12.93	.5584 (.549,.568)	-20.23

Consistency Test

A consistency test is used to check for any trends of estimated $\hat{H}$ values being greater, or less, than the required value of H more often than it should occur by chance. Having applied this test to 400 estimates of H obtained from the wavelet-based H estimator and Whittle's MLE, we obtained the results produced in Table 5.7.

Table 5.4: Relative inaccuracy, ΔH, of mean values of estimated H obtained using Whittle's MLE for the exact self-similar FGN process with different marginal distributions for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means in parentheses.

Distribution	Mean Values of Estimated H and ΔH			
	.8		.9	
	$\hat{H}$	$\Delta H(\%)$	$\hat{H}$	$\Delta H(\%)$
Exponential	.7749 (.766,.784)	-3.134	.8797 (.871,.889)	-2.258
Gamma	.7864 (.777,.796)	-1.701	.8892 (.880,.898)	-1.202
Uniform	.7897 (.781,.799)	-1.285	.8889 (.880,.898)	-1.235
Weibull	.7964 (.787,.806)	-0.451	.897 (.888,.906)	-0.335
Pareto ($\alpha = 20.0$)	.7706 (.761,.780)	-3.678	.8759 (.867,.885)	-2.674
Pareto ($\alpha = 1.2$)	.5438 (.534,.554)	-32.03	.6393 (.630,.649)	-28.96
Pareto ($\alpha = 1.4$)	.5666 (.557,.576)	-29.18	.6774 (.668,.687)	-24.73
Pareto ($\alpha = 1.6$)	.5919 (.582,.602)	-26.01	.7112 (.702,.721)	-20.97
Pareto ($\alpha = 1.8$)	.6169 (.607,.626)	-22.89	.7399 (.731,.749)	-17.79

Table 5.5: Variances for estimated H obtained using the wavelet-based H estimator for self-similar processes with different marginal distributions for H = 0.6, 0.7, 0.8 and 0.9.

Distribution	Variances of Estimated H			
	.6	.7	.8	.9
Original	1.8920e-04	1.9220e-04	1.9510e-04	1.9850e-04
Exponential	1.6620e-04	2.0330e-04	2.8780e-04	4.5280e-04
Gamma	1.9940e-04	2.0250e-04	2.1560e-04	2.6410e-04
Uniform	1.9930e-04	1.9920e-04	1.9620e-04	2.1290e-04
Weibull	1.8120e-04	1.9100e-04	2.0760e-04	2.3660e-04
Pareto ($\alpha = 20.0$)	1.6890e-04	2.0980e-04	3.0880e-04	4.9820e-04
Pareto ($\alpha = 1.2$)	5.0160e-03	1.0020e-02	9.7020e-03	9.1550e-03
Pareto ($\alpha = 1.4$)	3.5150e-03	6.6330e-03	7.1350e-03	7.3760e-03
Pareto ($\alpha = 1.6$)	2.4050e-03	4.4900e-03	5.5260e-03	5.7900e-03
Pareto ($\alpha = 1.8$)	1.6220e-03	3.0460e-03	4.2630e-03	4.3950e-03

Table 5.6: Variances for estimated H obtained using Whittle's MLE for self-similar processes with different marginal distributions for H = 0.6, 0.7, 0.8 and 0.9.

Distribution	Variances of Estimated H			
	.6	.7	.8	.9
Original	1.0930e-05	1.1447e-05	1.1789e-05	1.2172e-05
Exponential	1.2697e-05	1.5443e-05	2.0052e-05	3.0836e-05
Gamma	1.1583e-05	1.2920e-05	1.4497e-05	1.6641e-05
Uniform	1.1518e-05	1.2855e-05	1.4325e-05	1.7971e-05
Weibull	1.1581e-05	1.2447e-05	1.3394e-05	1.5497e-05
Pareto ($\alpha = 20.0$)	1.3430e-05	1.7630e-05	2.3820e-05	4.0410e-05
Pareto ($\alpha = 1.2$)	1.0190e-04	2.5820e-04	1.0370e-03	5.0670e-03
Pareto ($\alpha = 1.4$)	9.6520e-05	3.6780e-04	1.3100e-03	4.9850e-03
Pareto ($\alpha = 1.6$)	1.0100e-04	4.6260e-04	1.5050e-03	4.4040e-03
Pareto ($\alpha = 1.8$)	1.0280e-04	5.0840e-04	1.5720e-03	3.6650e-03

Table 5.7: A consistency test: number of sample sequences with $H > \hat{H}$, for estimated $\hat{H}$ obtained using the wavelet-based H estimator and Whittle's MLE, for self-similar process with different marginal distributions, for $H =$ 0.6, 0.7, 0.8 and 0.9.

Estimator	Distribution	Number of $H > \hat{H}$ of 100 Samples				
		.6	.7	.8	.9	Total out of 400
Wavelet-based	Exponential	97	100	98	93	388
	Gamma	88	97	94	85	364
	Pareto	96	100	100	100	396
	Uniform	86	94	98	95	373
	Weibull	79	88	85	78	330
Whittle's MLE	Exponential	100	100	100	100	400
	Gamma	99	100	100	99	398
	Pareto	100	100	100	99	399
	Uniform	89	97	99	99	384
	Weibull	71	79	83	77	310

Table 5.7 shows that for the wavelet-based H estimator, in the case of the exponential marginal distribution, 388 of 400 sample sequences had $H > \hat{H}$ and 12 had $H < \hat{H}$. In the gamma, Pareto and uniform marginal distributions, we obtained $H > \hat{H}$ in 364, 396 and 373 of 400 cases, respectively. 330 of 400 samples in the case of Weibull marginal distribution had $H > \hat{H}$, while 1 of 400 cases had $H = \hat{H}$, and 69 of 400 cases had $H < \hat{H}$. The results obtained using Whittle's MLE, see Table 5.7, are similar to the previous results. For the exponential, gamma, Pareto, uniform and Weibull marginal distributions, 400, 398, 399, 384 and 310 of 400 had $H > \hat{H}$, respectively. Therefore, most of the estimated $\hat{H}$ values obtained using the wavelet-based H estimator and Whittle's MLE for the output processes with the considered five marginal distributions were slightly lower than the required value.

On the basis of these results, one can see a clear trend that estimates of H obtained from the wavelet-based H estimator and Whittle's MLE for the output processes with five marginal distributions were smaller than the real value of H.

5.3.2 Analysis of Autocorrelation Functions

Preservation of H in output processes with marginal probability distributions and finite variances, which we showed in the previous section, is acompanied by preservation of ACFs in all these cases as well; for $H = 0.9$, see Figures 5.12 – 5.16. The output ACFs that significantly differ from the input ACFs of the exact FGN process are associated with Pareto distributions with infinite variances.

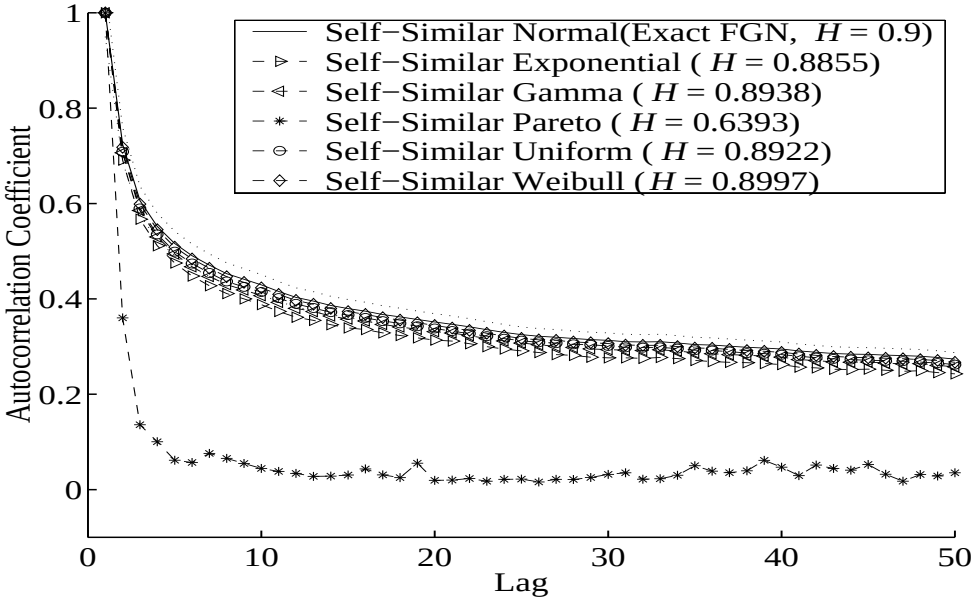
ACFs curves of self-similar LRD processes decay more slowly and hyperbolically rather than exponentially as H values increase. For example, Figure 5.8 shows ACFs for the exact self-similar FGN process, and five approximately self-similar processes with exponential, gamma, Pareto ($\alpha = 1.2$), uniform and Weibull marginal distributions for two different ranges of lags. The ACF curve obtained from the Pareto marginal distribution with $\alpha = 1.2$ and $H = 0.6393$ lies lower than other ACF curves with $H > 0.88$. In contrast, the ACF curve of a Poisson process assumes value one at lag equal 0, and zero otherwise. We considered here a Poisson process with $\lambda = 0.9$.

Note that all ACFs of marginal probability distributions with finite variances differ from the input ACFs by no more than 4% (Lower and upper dotted lines in Figures 5.12 – 5.16 are $\pm$ 4% apart from the input ACFs.). In all cases of output processes with Pareto distributions with infinite variances, the differences between their ACFs and the ACF of input FGN process are substantial. Thus, there is clear experimental evidence that the LRD self-similarity of the input process is not preserved in the output process generated by transformation (5.2), if the output process has an infinite variance. For $H = 0.6$, 0.7 and 0.8, Figures 5.17 – 5.19 show the effects of transforming ACFs from the original exact FGN process using the ICDF transformation.

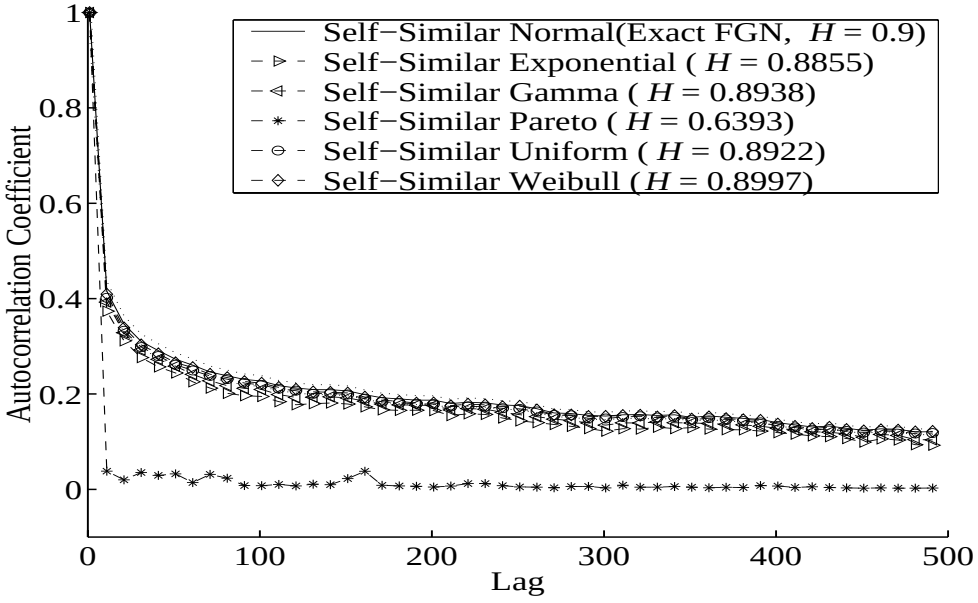
The results of the mean values of the difference, ΔACF, between the ACF of the exact self-similar FGN process and the ACFs characterising output processes with five different marginal distributions are summarised quantitatively in Tables 5.8 – 5.17. The difference, ΔACF, is calculated by

$$\Delta\text{ACF} = ACF - \widehat{ACF},$$

where ACF is the mean value of the ACF obtained from the original process and $\widehat{ACF}$ is the empirical mean value over a number of independently

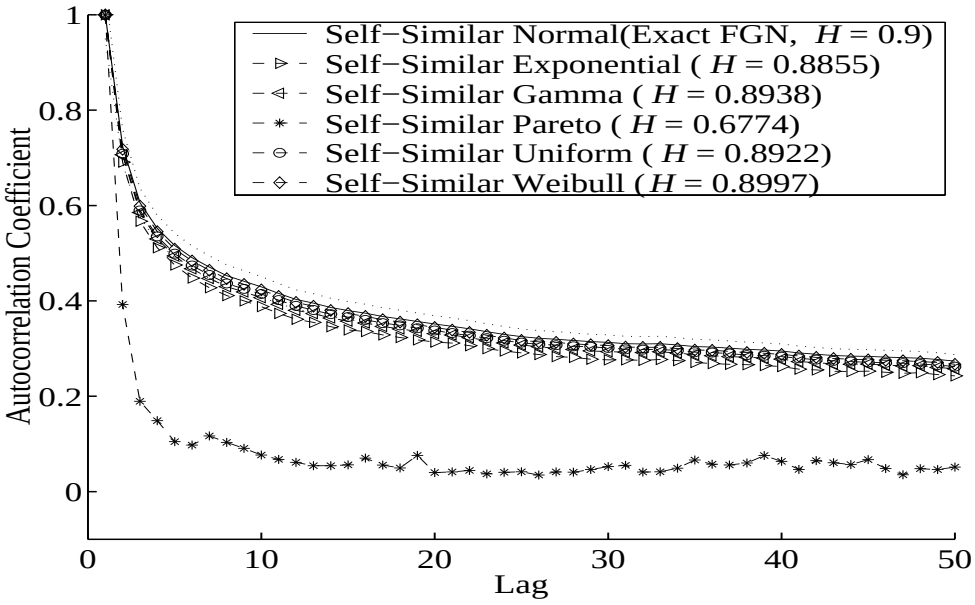


(a) Autocorrelation lags between 1 and 50.

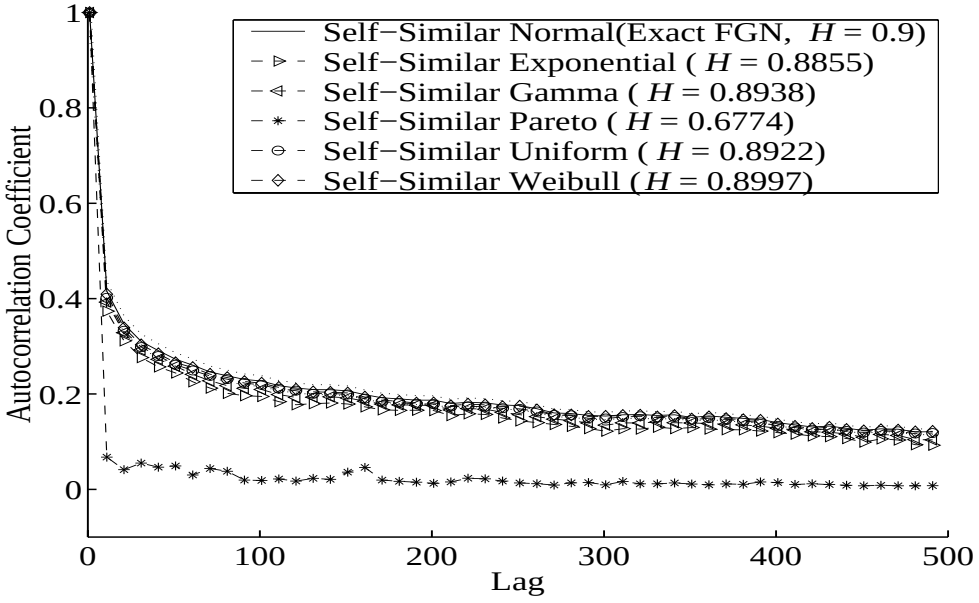


(b) Autocorrelation lags between 1 and 500.

Figure 5.12: Autocorrelation functions for the exact self-similar FGN process, five exponential, gamma, Pareto ($\alpha = 1.2$), uniform and Weibull marginal distributions in two different scales for $H = 0.9$. The output processes preserve LRD properties, except the Pareto marginal distribution with $\alpha = 1.2$.

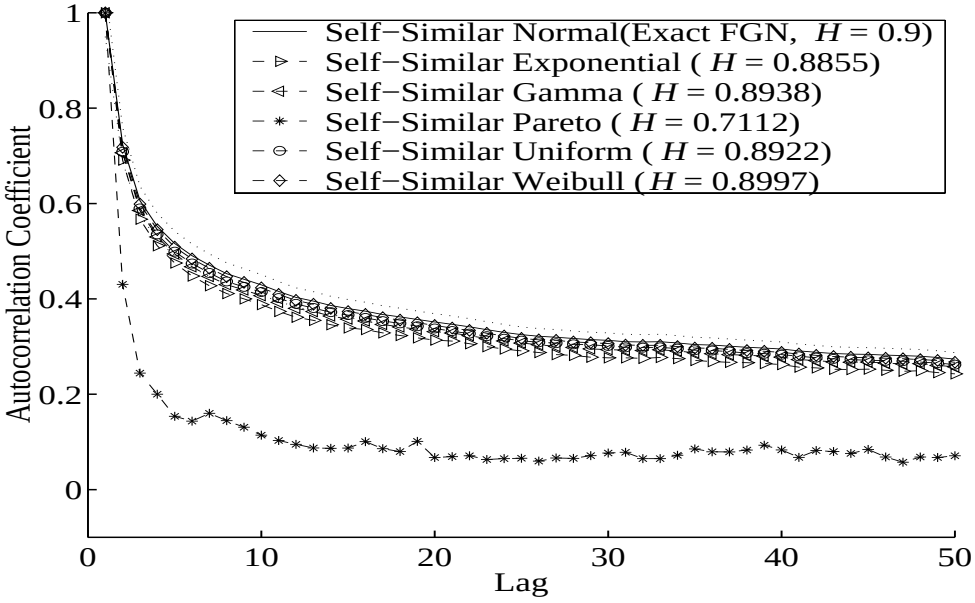


(a) Autocorrelation lags between 1 and 50.

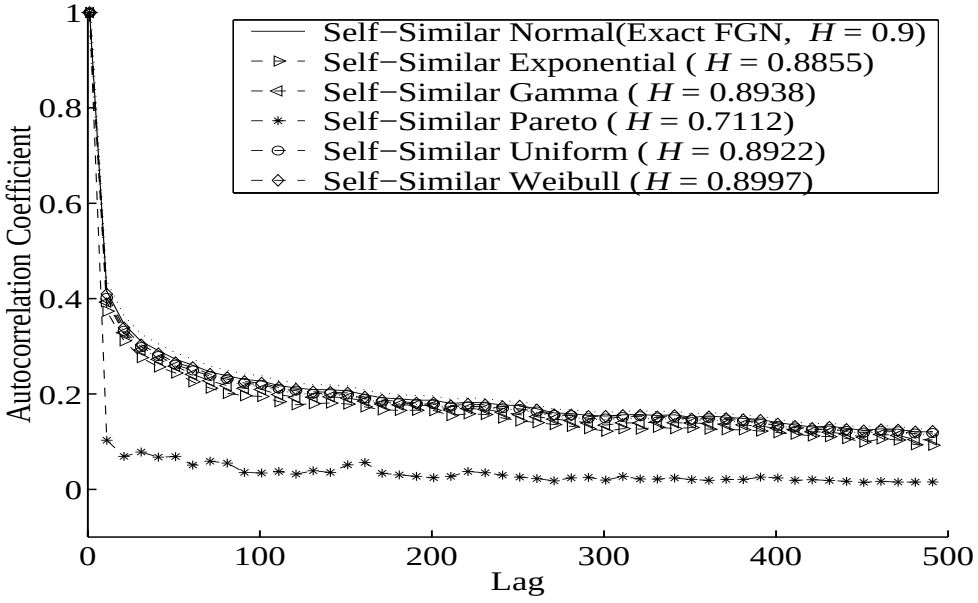


(b) Autocorrelation lags between 1 and 500.

Figure 5.13: Autocorrelation functions for the exact self-similar FGN process, five exponential, gamma, Pareto ($\alpha = 1.4$), uniform and Weibull marginal distributions in two different scales for $H = 0.9$. The output processes preserve LRD properties, except the Pareto marginal distribution with $\alpha = 1.4$.

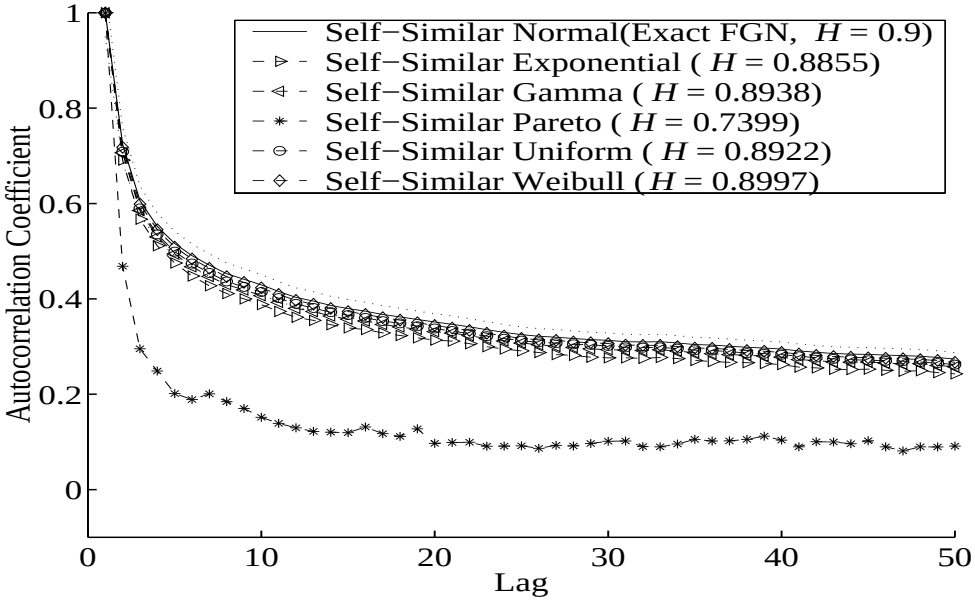


(a) Autocorrelation lags between 1 and 50.

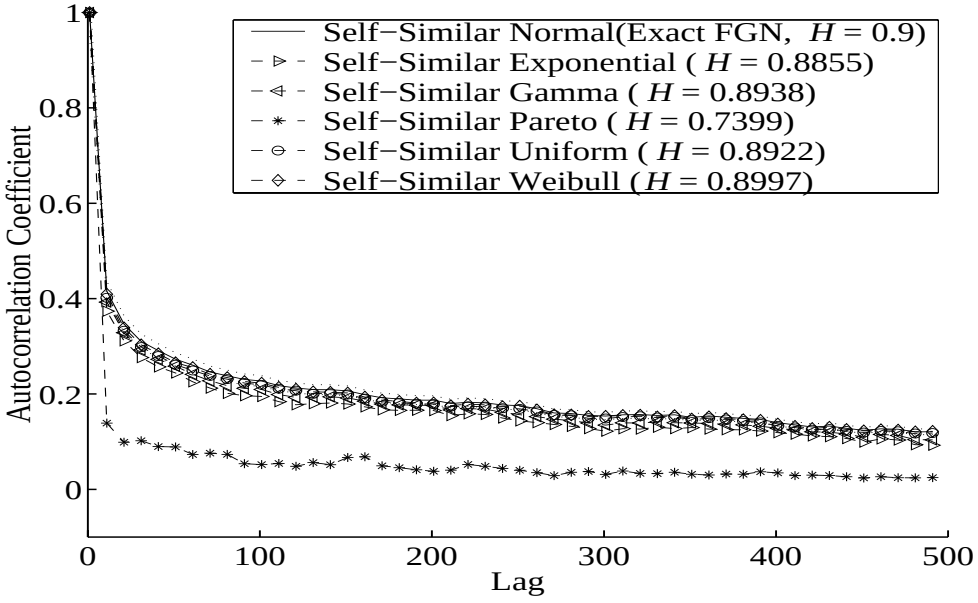


(b) Autocorrelation lags between 1 and 500.

Figure 5.14: Autocorrelation functions for the exact self-similar FGN process, five exponential, gamma, Pareto ($\alpha = 1.6$), uniform and Weibull marginal distributions in two different scales for $H = 0.9$. The output processes preserve LRD properties, except the Pareto marginal distribution with $\alpha = 1.6$.

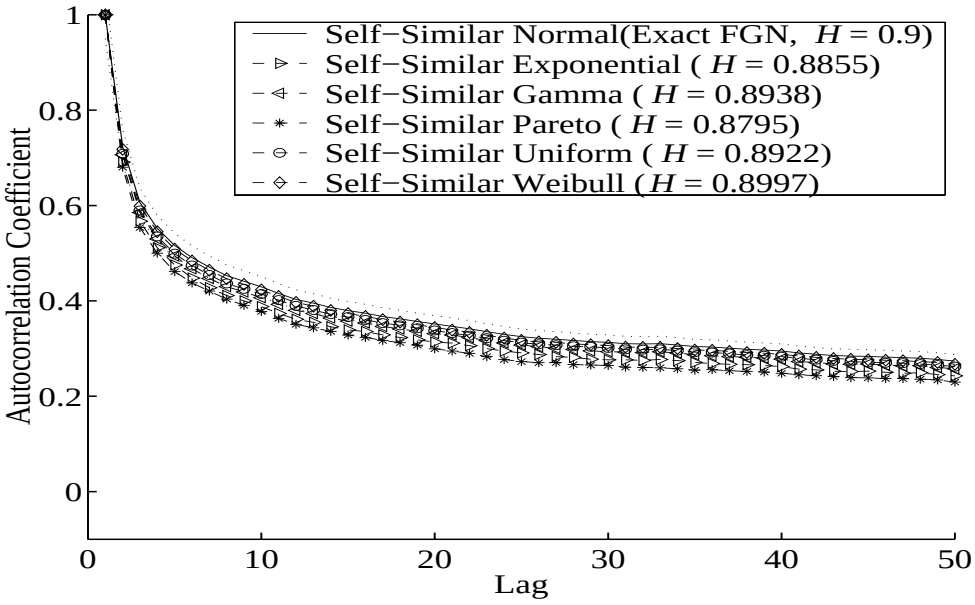


(a) Autocorrelation lags between 1 and 50.

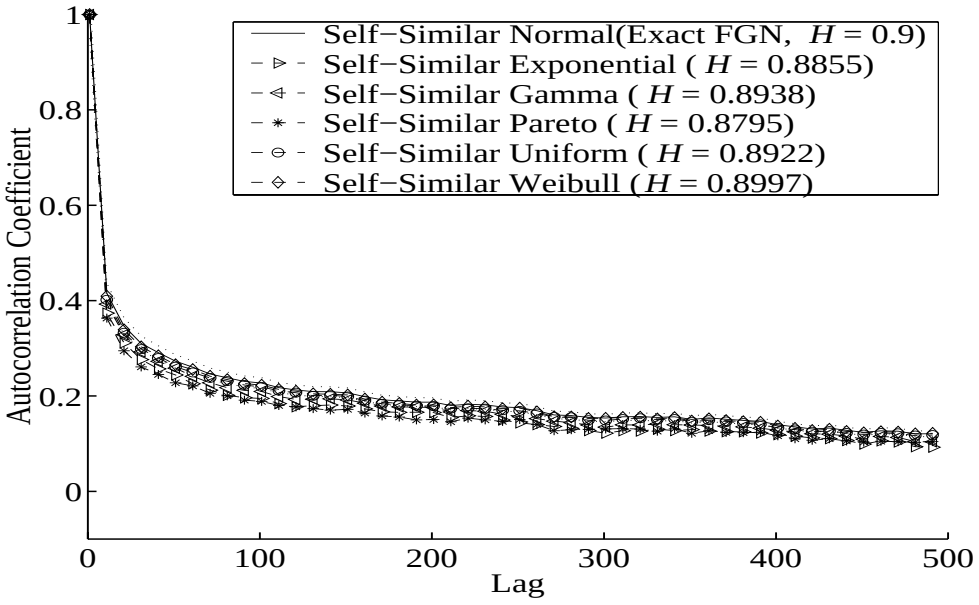


(b) Autocorrelation lags between 1 and 500.

Figure 5.15: Autocorrelation functions for the exact self-similar FGN process, five exponential, gamma, Pareto ($\alpha = 1.8$), uniform and Weibull marginal distributions in two different scales for $H = 0.9$. The output processes preserve LRD properties, except the Pareto marginal distribution with $\alpha = 1.8$.

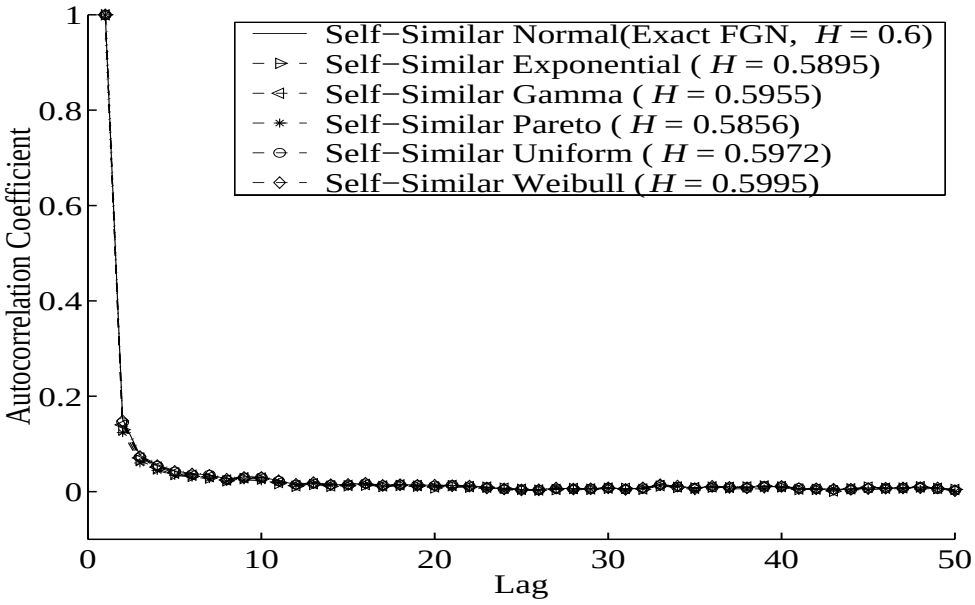


(a) Autocorrelation lags between 1 and 50.

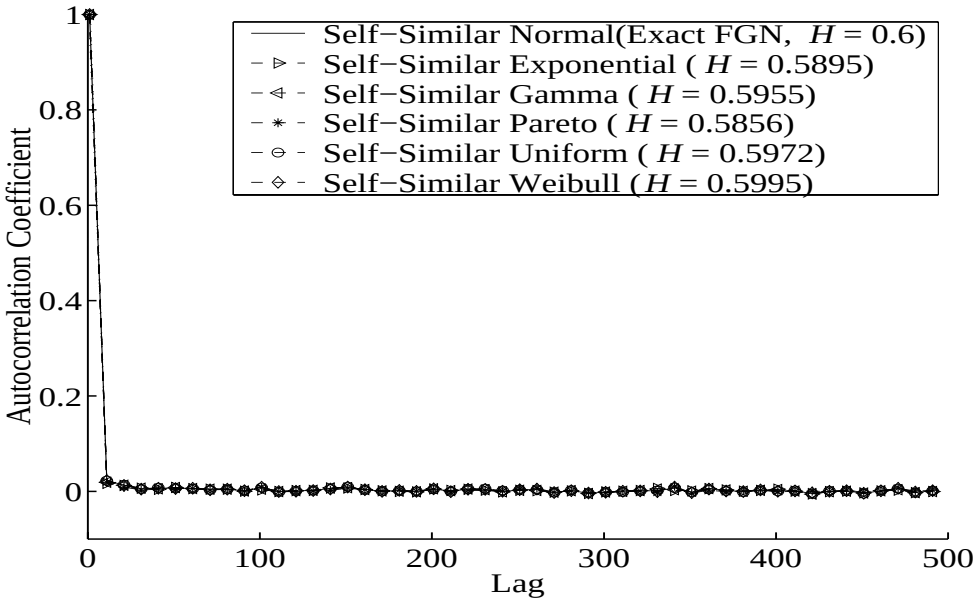


(b) Autocorrelation lags between 1 and 500.

Figure 5.16: Autocorrelation functions for the exact self-similar FGN process, five exponential, gamma, Pareto ($\alpha = 20.0$), uniform and Weibull marginal distributions in two different scales for $H = 0.9$. The output processes preserve LRD properties.

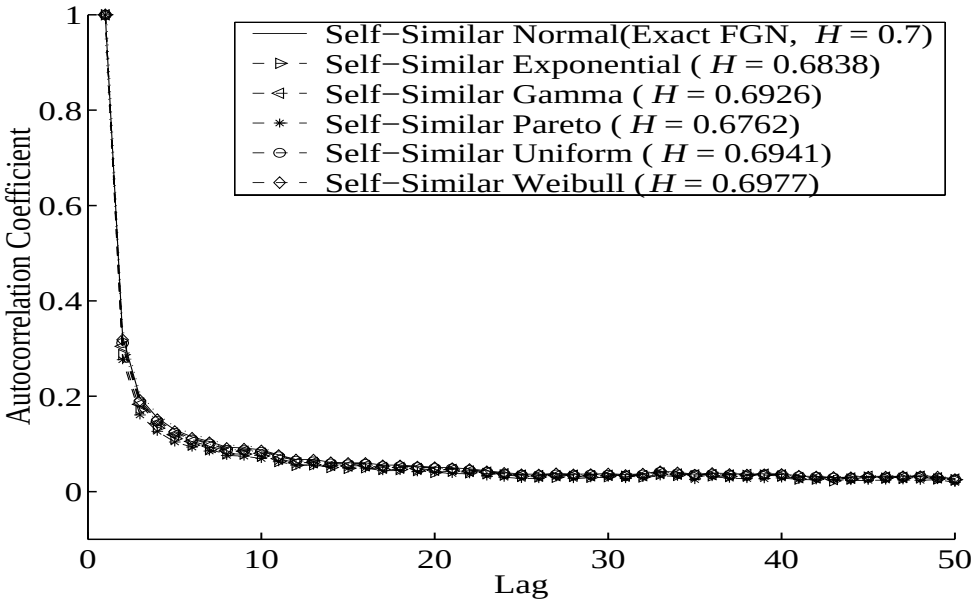


(a) Autocorrelation lags between 1 and 50.

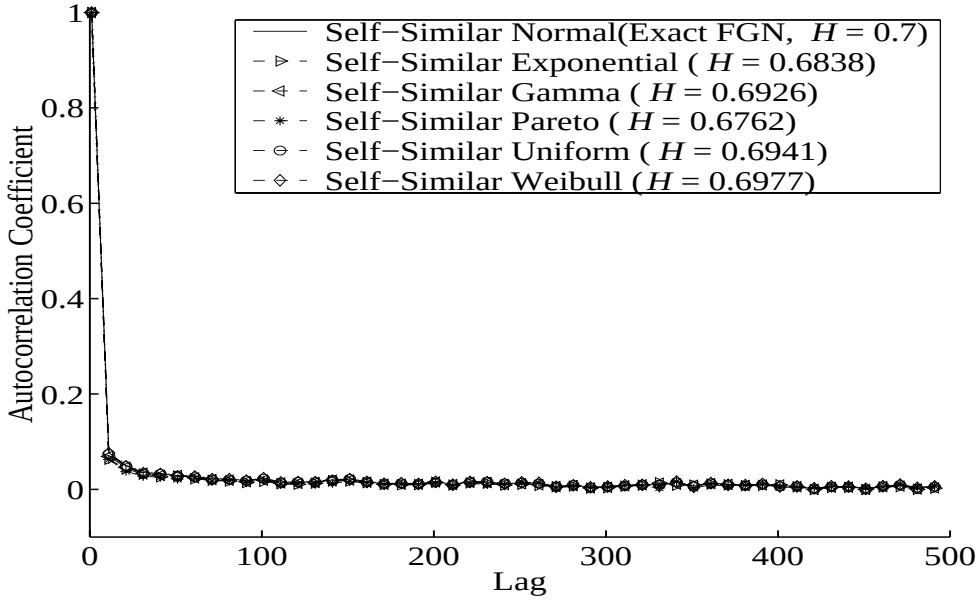


(b) Autocorrelation lags between 1 and 500.

Figure 5.17: Autocorrelation functions for the exact self-similar FGN process, five exponential, gamma, Pareto ($\alpha = 20.0$), uniform and Weibull marginal distributions in two different scales for $H = 0.6$. The output processes preserve LRD properties.

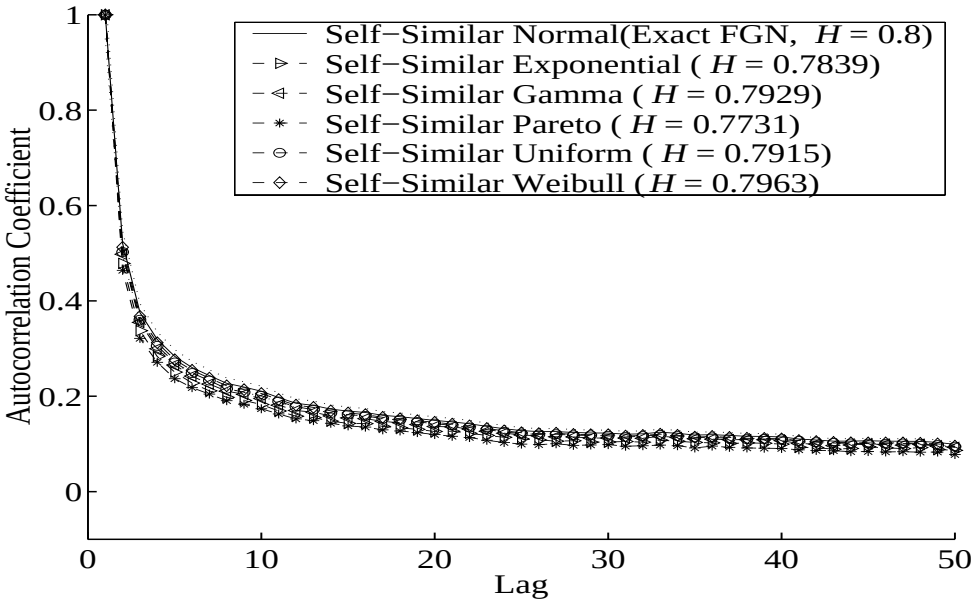


(a) Autocorrelation lags between 1 and 50.

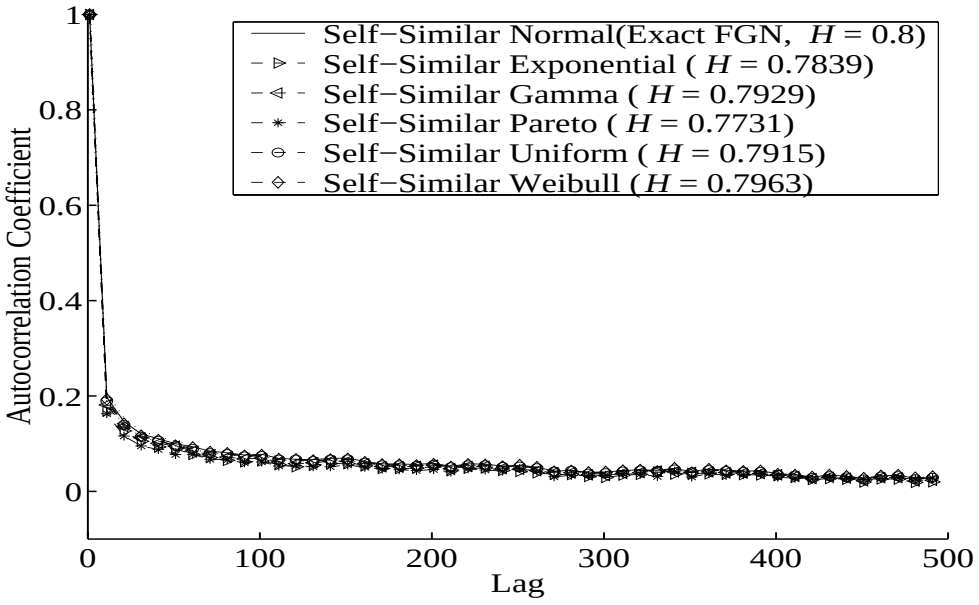


(b) Autocorrelation lags between 1 and 500.

Figure 5.18: Autocorrelation functions for the exact self-similar FGN process, five exponential, gamma, Pareto ($\alpha = 20.0$), uniform and Weibull marginal distributions in two different scales for $H = 0.7$. The output processes preserve LRD properties.



(a) Autocorrelation lags between 1 and 50.



(b) Autocorrelation lags between 1 and 500.

Figure 5.19: Autocorrelation functions for the exact self-similar FGN process, five exponential, gamma, Pareto ($\alpha = 20.0$), uniform and Weibull marginal distributions in two different scales for $H = 0.8$. The output processes preserve LRD properties.

generated sequences obtained using the ICDF transformation. We give 95% confidence intervals for the means in parentheses.

Exponential Marginal Distribution

For $H = 0.6$ and 0.7, and all lags, mean values of the difference, ΔACF, between the ACF of the original exact self-similar FGN process and the ACF transformed using the exponential marginal distribution are below 1%, as shown Tables 5.8 – 5.9. For $H = 0.8$ and low lags between 50 and 150, those are less than 2%, but for high lags, those are below 1%. For $H = 0.9$ and all lags, those are less than 4%.

Gamma Marginal Distribution

Tables 5.10 – 5.11 show that, for $H = 0.6$, 0.7 and 0.8, and all lags, all differences are below 1%, except for $H = 0.9$ and lags between 50 and 300, those are below 2%.

Pareto Marginal Distribution

These results obtained for the Pareto marginal distribution with $\alpha = 20.0$ in Tables 5.12 – 5.13 show the same trend as the exponential marginal distribution.

Uniform and Weibull Marginal Distributions

Tables 5.14 – 5.17 show that, for $H = 0.6$, 0.7, 0.8 and 0.9, and all lags, those obtained for the uniform and Weibull marginal distributions are below 1%.

Thus, for $H = 0.6$, 0.7, 0.8 and 0.9, and autocorrelation lags between 50 and 500, this is evidence of the preservation of the original ACF after the ICDF transformation is applied, except the Pareto marginal distribution with $\alpha =$ 1.2, 1.4, 1.6 and 1.8.

5.4 Conclusions

We investigated how well the LRD self-similarity and ACFs of the original processes were preserved when the self-similar processes were converted into suitable self-similar processes with five exponential, gamma, Pareto, uniform and Weibull marginal distributions. We used the ICDF transformation to

Table 5.8: Mean values of the difference, ΔACF, between the ACF of the original exact self-similar FGN process and the ACF transformed using the exponential marginal distribution for $H = 0.6$ and 0.7. We give 95% confidence intervals for the means in parentheses.

	ΔACF	
Lag	0.6	0.7
50	+1.302e-03 (+6.533e-04, +1.951e-03)	+4.958e-03 (+4.212e-03, +5.705e-03)
100	+1.088e-03 (+4.376e-04, +1.738e-03)	+3.611e-03 (+2.885e-03, +4.337e-03)
150	+4.605e-04 (-2.524e-04, +1.173e-03)	+2.193e-03 (+1.480e-03, +2.905e-03)
200	+6.947e-04 (+1.102e-05, +1.378e-03)	+2.334e-03 (+1.580e-03, +3.088e-03)
250	+3.128e-04 (-3.849e-04, +1.011e-03)	+1.651e-03 (+8.872e-04, +2.416e-03)
300	+2.245e-04 (-4.650e-04, +9.141e-04)	+1.358e-03 (+6.380e-04, +2.078e-03)
350	-2.937e-05 (-7.232e-04, +6.645e-04)	+9.952e-04 (+2.810e-04, +1.709e-03)
400	-8.934e-06 (-6.750e-04, +6.571e-04)	+9.604e-04 (+2.300e-04, +1.691e-03)
450	+4.430e-04 (-2.672e-04, +1.153e-03)	+1.116e-03 (+3.553e-04, +1.877e-03)
500	+2.085e-04 (-4.879e-04, +9.048e-04)	+8.998e-04 (+1.765e-04, +1.623e-03)

produce self-similar processes with five different marginal distributions for the stochastic simulation of telecommunication networks with self-similar teletraffic. Our results presented in this chapter provide clear experimental evidence that the LRD self-similarity of the input process is not preserved in the output process generated by transformation (5.2), if the output process has an infinite variance. On the basis of our results we formulate the following hypothesis:

Table 5.9: Mean values of the difference, ΔACF, between the ACF of the original exact self-similar FGN process and the ACF transformed using the exponential marginal distribution for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means in parentheses.

	ΔACF	
Lag	0.8	0.9
50	+1.516e-02 (+1.388e-02, +1.643e-02)	+3.226e-02 (+2.901e-02, +3.550e-02)
100	+1.153e-02 (+1.034e-02, +1.272e-02)	+2.769e-02 (+2.465e-02, +3.073e-02)
150	+8.489e-03 (+7.341e-03, +9.637e-03)	+2.329e-02 (+2.026e-02, +2.632e-02)
200	+7.910e-03 (+6.793e-03, +9.027e-03)	+2.194e-02 (+1.910e-02, +2.477e-02)
250	+6.955e-03 (+5.731e-03, +8.179e-03)	+2.097e-02 (+1.794e-02, +2.401e-02)
300	+6.158e-03 (+5.067e-03, +7.249e-03)	+1.945e-02 (+1.658e-02, +2.231e-02)
350	+5.396e-03 (+4.379e-03, +6.412e-03)	+1.801e-02 (+1.536e-02, +2.067e-02)
400	+5.182e-03 (+4.031e-03, +6.332e-03)	+1.737e-02 (+1.439e-02, +2.036e-02)
450	+5.030e-03 (+3.943e-03, +6.118e-03)	+1.653e-02 (+1.382e-02, +1.923e-02)
500	+4.376e-03 (+3.200e-03, +5.553e-03)	+1.541e-02 (+1.241e-02, +1.840e-02)

Hypothesis: If *transformation (5.2) is applied to self-similar LRD processes with normal marginal distributions, then it preserves H parameter and ACF of the input process if the output marginal distribution has a finite variance.*

Table 5.10: Mean values of the difference, ΔACF, between the ACF of the original exact self-similar FGN process and the ACF transformed using the gamma marginal distribution for $H = 0.6$ and 0.7. We give 95% confidence intervals for the means in parentheses.

	ΔACF	
Lag	0.6	0.7
50	+8.361e-04 (+3.519e-04, +1.320e-03)	+2.873e-03 (+2.329e-03, +3.418e-03)
100	+7.161e-04 (+2.411e-04, +1.191e-03)	+2.125e-03 (+1.607e-03, +2.643e-03)
150	+2.736e-04 (-2.570e-04, +8.042e-04)	+1.217e-03 (+6.898e-04, +1.744e-03)
200	+4.759e-04 (-2.985e-05, +9.817e-04)	+1.387e-03 (+8.331e-04, +1.940e-03)
250	+2.467e-04 (-2.710e-04, +7.644e-04)	+9.800e-04 (+4.116e-04, +1.548e-03)
300	+1.547e-04 (-3.606e-04, +6.701e-04)	+7.784e-04 (+2.446e-04, +1.312e-03)
350	-9.028e-06 (-5.193e-04, +5.013e-04)	+5.453e-04 (+2.730e-05, +1.063e-03)
400	-7.247e-05 (-5.678e-04, +4.229e-04)	+4.767e-04 (-6.574e-05, +1.019e-03)
450	+3.181e-04 (-2.034e-04, +8.397e-04)	+6.613e-04 (+1.140e-04, +1.209e-03)
500	+1.271e-04 (-4.023e-04, +6.566e-04)	+4.644e-04 (-7.901e-05, +1.008e-03)

Further research work is needed to investigate exact values of preservation of the second-order LRD self-similarity when transforming second-order self-similar processes into processes with arbitrary marginal distributions.

Table 5.11: Mean values of the difference, ΔACF, between the ACF of the original exact self-similar FGN process and the ACF transformed using the gamma marginal distribution for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means in parentheses.

	ΔACF	
Lag	0.8	0.9
50	+8.390e-03 (+7.471e-03, +9.309e-03)	+1.752e-02 (+1.527e-02, +1.976e-02)
100	+6.485e-03 (+5.655e-03, +7.315e-03)	+1.514e-02 (+1.301e-02, +1.727e-02)
150	+4.559e-03 (+3.742e-03, +5.377e-03)	+1.243e-02 (+1.028e-02, +1.458e-02)
200	+4.398e-03 (+3.605e-03, +5.191e-03)	+1.177e-02 (+9.762e-03, +1.377e-02)
250	+3.860e-03 (+2.977e-03, +4.743e-03)	+1.149e-02 (+9.350e-03, +1.362e-02)
300	+3.400e-03 (+2.629e-03, +4.171e-03)	+1.051e-02 (+8.511e-03, +1.251e-02)
350	+2.964e-03 (+2.267e-03, +3.662e-03)	+9.788e-03 (+7.939e-03, +1.164e-02)
400	+2.794e-03 (+1.972e-03, +3.615e-03)	+9.428e-03 (+7.341e-03, +1.152e-02)
450	+2.790e-03 (+2.030e-03, +3.550e-03)	+9.066e-03 (+7.183e-03, +1.095e-02)
500	+2.304e-03 (+1.479e-03, +3.129e-03)	+8.319e-03 (+6.238e-03, +1.040e-02)

Table 5.12: Mean values of the difference, ΔACF, between the ACF of the original exact self-similar FGN process and the ACF transformed using the Pareto marginal distribution with $\alpha = 20.0$ for $H = 0.6$ and 0.7. We give 95% confidence intervals for the means in parentheses.

	ΔACF	
Lag	0.6	0.7
50	+5.772e-04 (-1.494e-04, +1.304e-03)	+4.687e-03 (+3.912e-03, +5.461e-03)
100	+2.738e-04 (-3.514e-04, +8.990e-04)	+2.878e-03 (+2.202e-03, +3.554e-03)
150	+4.438e-04 (-2.304e-04, +1.118e-03)	+2.525e-03 (+1.790e-03, +3.260e-03)
200	+7.167e-05 (-6.693e-04, +8.126e-04)	+1.981e-03 (+1.129e-03, +2.834e-03)
250	-4.079e-04 (-1.146e-03, +3.298e-04)	+1.163e-03 (+3.492e-04, +1.976e-03)
300	+2.909e-05 (-7.067e-04, +7.649e-04)	+1.348e-03 (+5.725e-04, +2.124e-03)
350	-1.519e-04 (-8.799e-04, +5.761e-04)	+1.032e-03 (+2.501e-04, +1.815e-03)
400	+7.255e-04 (-5.164e-06, +1.456e-03)	+1.719e-03 (+8.941e-04, +2.544e-03)
450	-8.491e-05 (-7.577e-04, +5.879e-04)	+8.165e-04 (+8.781e-05, +1.545e-03)
500	+9.047e-05 (-6.734e-04, +8.543e-04)	+1.140e-03 (+2.972e-04, +1.982e-03)

Table 5.13: Mean values of the difference, ΔACF, between the ACF of the original exact self-similar FGN process and the ACF transformed using the Pareto marginal distribution with $\alpha = 20.0$ for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means in parentheses.

	ΔACF	
Lag	0.8	0.9
50	+1.622e-02 (+1.502e-02, +1.743e-02)	+3.648e-02 (+3.306e-02, +3.991e-02)
100	+1.169e-02 (+1.053e-02, +1.285e-02)	+3.063e-02 (+2.720e-02, +3.407e-02)
150	+1.040e-02 (+9.151e-03, +1.164e-02)	+2.912e-02 (+2.559e-02, +3.264e-02)
200	+9.090e-03 (+7.770e-03, +1.041e-02)	+2.683e-02 (+2.335e-02, +3.031e-02)
250	+7.337e-03 (+6.036e-03, +8.638e-03)	+2.380e-02 (+2.034e-02, +2.725e-02)
300	+6.968e-03 (+5.765e-03, +8.170e-03)	+2.274e-02 (+1.943e-02, +2.606e-02)
350	+6.168e-03 (+4.945e-03, +7.391e-03)	+2.118e-02 (+1.784e-02, +2.453e-02)
400	+6.404e-03 (+5.136e-03, +7.672e-03)	+2.070e-02 (+1.736e-02, +2.404e-02)
450	+5.105e-03 (+4.000e-03, +6.210e-03)	+1.862e-02 (+1.548e-02, +2.175e-02)
500	+5.446e-03 (+4.173e-03, +6.718e-03)	+1.867e-02 (+1.529e-02, +2.205e-02)

Table 5.14: Mean values of the difference, ΔACF, between the ACF of the original exact self-similar FGN process and the ACF transformed using the uniform marginal distribution for $H = 0.6$ and 0.7. We give 95% confidence intervals for the means in parentheses.

	ΔACF	
Lag	0.6	0.7
50	+8.645e-05 (-2.202e-04, +3.931e-04)	+9.570e-04 (+6.415e-04, +1.273e-03)
100	-1.389e-04 (-4.349e-04, +1.570e-04)	+3.895e-04 (+7.050e-05, +7.086e-04)
150	+2.516e-04 (-6.356e-05, +5.667e-04)	+8.735e-04 (+5.556e-04, +1.192e-03)
200	+2.339e-04 (-4.758e-05, +5.154e-04)	+6.848e-04 (+3.827e-04, +9.870e-04)
250	+3.628e-04 (+4.610e-06, +7.209e-04)	+6.785e-04 (+3.217e-04, +1.035e-03)
300	-7.109e-05 (-4.009e-04, +2.587e-04)	+2.690e-04 (-5.927e-05, +5.973e-04)
350	+3.580e-04 (+7.363e-05, +6.424e-04)	+5.128e-04 (+2.167e-04, +8.089e-04)
400	+3.474e-06 (-3.268e-04, +3.337e-04)	+3.751e-04 (+4.710e-05, +7.031e-04)
450	+1.056e-04 (-2.433e-04, +4.546e-04)	+3.138e-04 (-2.996e-05, +6.575e-04)
500	-6.386e-05 (-4.233e-04, +2.956e-04)	+6.514e-05 (-2.892e-04, +4.195e-04)

Table 5.15: Mean values of the difference, ΔACF, between the ACF of the original exact self-similar FGN process and the ACF transformed using the uniform marginal distribution for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means in parentheses.

	ΔACF	
Lag	0.8	0.9
50	+3.761e-03 (+3.378e-03, +4.145e-03)	+9.843e-03 (+8.740e-03, +1.095e-02)
100	+2.443e-03 (+2.033e-03, +2.852e-03)	+7.592e-03 (+6.453e-03, +8.730e-03)
150	+2.601e-03 (+2.242e-03, +2.961e-03)	+6.947e-03 (+5.852e-03, +8.042e-03)
200	+2.048e-03 (+1.641e-03, +2.456e-03)	+5.884e-03 (+4.763e-03, +7.005e-03)
250	+1.885e-03 (+1.477e-03, +2.294e-03)	+5.509e-03 (+4.393e-03, +6.625e-03)
300	+1.442e-03 (+1.041e-03, +1.844e-03)	+4.591e-03 (+3.470e-03, +5.712e-03)
350	+1.488e-03 (+1.113e-03, +1.863e-03)	+4.441e-03 (+3.346e-03, +5.536e-03)
400	+1.344e-03 (+9.432e-04, +1.744e-03)	+4.422e-03 (+3.239e-03, +5.606e-03)
450	+1.297e-03 (+9.223e-04, +1.672e-03)	+4.273e-03 (+3.172e-03, +5.373e-03)
500	+9.019e-04 (+4.688e-04, +1.335e-03)	+3.918e-03 (+2.643e-03, +5.193e-03)

Table 5.16: Mean values of the difference, ΔACF, between the ACF of the original exact self-similar FGN process and the ACF transformed using the Weibull marginal distribution for $H = 0.6$ and 0.7. We give 95% confidence intervals for the means in parentheses.

	ΔACF	
Lag	0.6	0.7
50	-1.073e-04 (-3.573e-04, +1.427e-04)	+3.811e-04 (+1.155e-04, +6.466e-04)
100	-2.009e-04 (-4.358e-04, +3.398e-05)	+1.149e-04 (-1.266e-04, +3.564e-04)
150	+4.847e-05 (-2.069e-04, +3.038e-04)	+3.632e-04 (+1.056e-04, +6.208e-04)
200	-8.680e-05 (-3.651e-04, +1.915e-04)	+1.614e-04 (-1.454e-04, +4.681e-04)
250	-8.478e-05 (-3.452e-04, +1.757e-04)	+1.366e-04 (-1.539e-04, +4.271e-04)
300	-6.725e-05 (-3.386e-04, +2.041e-04)	+1.189e-04 (-1.614e-04, +3.992e-04)
350	+4.888e-05 (-2.158e-04, +3.136e-04)	+1.839e-04 (-8.609e-05, +4.540e-04)
400	+1.576e-04 (-1.059e-04, +4.212e-04)	+3.006e-04 (+9.089e-06, +5.921e-04)
450	-7.474e-05 (-3.321e-04, +1.827e-04)	+5.913e-05 (-2.060e-04, +3.242e-04)
500	-1.222e-05 (-2.902e-04, +2.657e-04)	+1.543e-04 (-1.377e-04, +4.464e-04)

Table 5.17: Mean values of the difference, ΔACF, between the ACF of the original exact self-similar FGN process and the ACF transformed using the Weibull marginal distribution for $H = 0.8$ and 0.9. We give 95% confidence intervals for the means in parentheses.

	ΔACF	
Lag	0.8	0.9
50	+1.853e-03 (+1.441e-03, +2.265e-03)	+4.327e-03 (+3.234e-03, +5.421e-03)
100	+1.205e-03 (+8.247e-04, +1.585e-03)	+3.445e-03 (+2.370e-03, +4.520e-03)
150	+1.450e-03 (+1.057e-03, +1.843e-03)	+3.752e-03 (+2.643e-03, +4.862e-03)
200	+1.088e-03 (+6.629e-04, +1.512e-03)	+3.283e-03 (+2.178e-03, +4.388e-03)
250	+8.725e-04 (+4.367e-04, +1.308e-03)	+2.691e-03 (+1.602e-03, +3.780e-03)
300	+8.366e-04 (+4.505e-04, +1.223e-03)	+2.641e-03 (+1.626e-03, +3.656e-03)
350	+8.010e-04 (+4.554e-04, +1.147e-03)	+2.474e-03 (+1.530e-03, +3.418e-03)
400	+8.707e-04 (+4.568e-04, +1.285e-03)	+2.511e-03 (+1.510e-03, +3.512e-03)
450	+6.112e-04 (+2.588e-04, +9.637e-04)	+2.210e-03 (+1.309e-03, +3.111e-03)
500	+7.417e-04 (+3.488e-04, +1.135e-03)	+2.385e-03 (+1.446e-03, +3.324e-03)

Chapter 6

MODELLING AND GENERATION OF SELF-SIMILAR VBR VIDEO TRAFFIC

6.1 Introduction

Teletraffic in the Internet is rapidly growing and diversifying, and there is a strong need for QoS support in high-speed communication networks [34], [66]. The introduction of many new multimedia services requires a high bandwidth to transport data such as real-time digital video. Modern computer networks can no longer cope with uncompressed multimedia traffic, resulting in the development of several image and video compression standards such as the Joint Bilevel Imaging Group (JBIG), the Joint Photographic Experts Group (JPEG) and the Moving Picture Experts Group (MPEG). In this chapter we focus on VBR JPEG/MPEG video, i.e., on video streams compressed according to JPEG and MPEG standards and transmitted as VBR (Variable Bit Rate) components of an ATM network.

There are a number of research issues concerning the transmission of JPEG/MPEG video over modern high speed computer networks, such as the dimensioning of multiplexer buffers and monitoring of video cell streams. These

problems have been studied intensively over the last ten years in order to provide a consistent and desirable QoS for JPEG/MPEG video traffic, construct accurate models for JPEG/MPEG video traffic and utilise efficient resource allocation techniques. We look at the influence of compression algorithms on correlation structure of compressed teletraffic, see Section 6.4, where results of compression of *Star Wars* video under JPEG and MPEG-1 are discussed.

A number of researchers tried to fit a specific mathematical model to traces of real VBR video traffic. For example, several models (based on gamma [60], lognormal [83], and combined gamma/Pareto [42], [81]) have been suggested for VBR video traffic. Heyman et al. [60] used a 30-minute compressed video-teleconferencing sequence for simulation studies using the gamma model. Krunz et al. [83] used a 23-minute movie, *The Wizard of Oz*, to study statistical characteristics of VBR MPEG-coded video streams using the lognormal model. The gamma model for video traffic became inaccurate in the tail of distribution, and the lognormal model was too heavy-tailed at first and then fell off too rapidly. Garrett and Willinger [42] used a two-hour VBR video, *Star Wars*, and proposed a hybrid gamma/Pareto model based on the F-ARIMA process (see Section 6.3). They found that the tail behaviour of the marginal distribution can be accurately described using the heavy-tailed Pareto distributions. They also found that the autocorrelation of the VBR video sequence decays more hyperbolically than exponentially and can be modelled using self-similar processes.

Huang et al. [64] presented a unified approach to modelling VBR video traffic using both SRD and LRD empirical ACFs. The approach consists of the following four steps: (i) estimation of the Hurst parameter H from a video trace using variance-time and R/S-statistic estimators; (ii) modelling the ACF for a video trace; (iii) measurement of the "attenuation" factor using the ACF of Hosking's F-ARIMA processes in Equation (4.19); and (iv) generation a process with the desired ACF using the ICDF transformation. They applied this approach to 2 hours' trace of *Last Action Hero* video. Their approach is potentially accurate, but establishing an automatic search for the best background ACF remains an open problem.

Lombardo et al. [97] proposed the generation of pseudo-MPEG video traffic with a specific correlation structure based on FFT [124] and an ICDF transformation, assuming an arbitrary marginal distribution of the output process.

The proposed algorithm has been used to generate a sequence with the same statistical characteristics as those of the movie "The Simpsons", however, the robustness of this algorithm remains an issue open to further investigation. Applicability of gamma/Pareto model as marginal distributions of compressed video streams is discussed in Section 6.4, where we look at accuracy of this approximation in relation to data coming from different videos (*Star Wars* and *Titanic*) compressed under three different algorithms (JPEG, MPEG-1 and MPEG-2).

We also show that synthetically generated streams of VBR video, compressed under such standards as JPEG, MPEG-1 and MPEG-2, are statistically similar to real video traces. We also present the results of a steady-state simulation of a single buffer fed by these synthetic video streams, in order to show that simulations based on synthetic streams of teletraffic can give the same qualitative and quantitative results as simulations based on real traces of VBR video.

6.2 JPEG/MPEG Video Compression

Several algorithms have been developed to compress video data, in order to reduce the memory required for their storage, the time or bandwidth necessary for their transmission, and the effective data access or transfer rate.

A joint committee of the International Telegraph and Telephone Consultative Committee (CCITT) and the International Organisation for Standardisation (ISO) formed the JPEG still-image standard in 1986. The goal of JPEG was to develop a general-purpose compression algorithm for continuous-tone, still-frame, monochrome and colour images. The JPEG standard comprises four main components: sequential (baseline) DCT (discrete cosine transform), hierarchical, progressive DCT and lossless, as shown in Figure 6.1 [132], [161]. A baseline component provides a simple and efficient algorithm that is adequate for most image coding applications. A set of extended system features allows the baseline component to satisfy a broader range of applications. Among these optional features are 12-bit/pixel input, progressive sequential and hierarchical build-up, and arithmetic coding. An independent lossless method of compression is included for applications that require it.

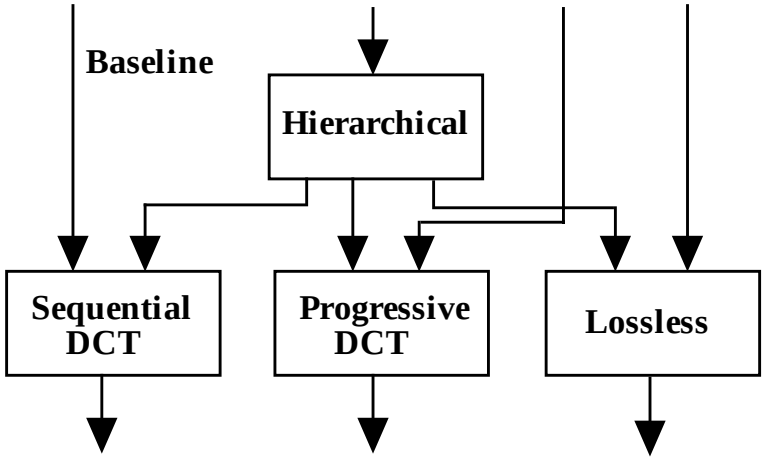


Figure 6.1: Four main components of the JPEG standard.

The MPEG coding scheme is widely used for video applications. MPEG is a group for audio and video compression formed under the auspices of ISO and the International Electrotechnical Commission (IEC) in 1988 [110]. The MPEG standard comprises different types, such as MPEG-1, MPEG-2, ..., MPEG-7, which have been designed to work in different situations. MPEG-1 addresses the compression of video and associated audio for the storage and retrieval of moving pictures and sound at a combined rate of about 1.5 Mbits/second (Mbps) using digital storage media (DSM).[1] MPEG-2 allows an interlaced format: video data streams consisting of two main layers. One is a base layer containing the most important video data. The other is used to improve the quality of video traffic. MPEG-2 is an extension of the MPEG-1 standard designed for broadcast television, including high-definition television (HDTV). MPEG-2 supports a higher bandwidth of up to 40 Mbits per second, five audio channels, a wider range of frame sizes, and interlaced video.

We focus on MPEG-1 and MPEG-2 of the MPEG standard family. MPEG-2 uses encoders from the MPEG-1 scheme, and in the case of multi-layer encoding, the statistical properties of its base layer are almost identical to MPEG-1. A video sequence is simply a series of pictures taken at closely spaced time intervals starting with a sequence header. The sequence header is

[1] The definition of DSM is broad enough to include CD-ROM, digital audio tape, writable optical disks, as well as computer and telecommunication networks such as an integrated services digital network and a local area network.

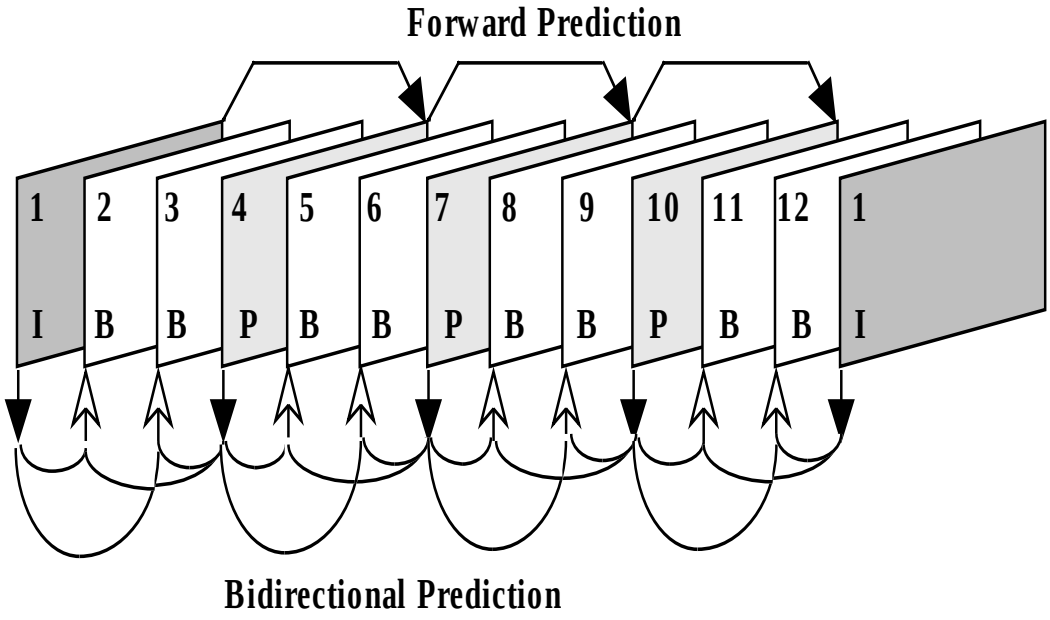


Figure 6.2: A typical MPEG group of pictures (GOP) in display order.

followed by one or more group(s) of pictures (GOP) and ends with a sequence end code. Additional sequence headers may appear between any GOP within the video sequence. This is achieved by using three types of frames [110]:

- Intra-coded picture (I-frame): coded independently from all other frames, based on DCT and entropy coding.
- Predictive-coded picture (P-frame): coded based on a prediction from a past I- or P-frame.
- Bi-directionally predictive-coded picture (B-frame): coded based on a prediction from a past and/or future I- or P-frame.

Only I- and P-frames can be used as a reference for past and/or future prediction. An ordered collection of I-, P- and B-frames is called a group of pictures. The proportion of I-, P- and B-frames is application-dependent and is left to the user. For example, for many scenes, spacing the reference frames at about one-twelfth of a second interval seems appropriate, i.e.,

Table 6.1: Parameters for generating the *Titanic* video sequence.

Parameters	Values
Coding algorithm	DCT
Duration	3 hours
Video frames	285890 (I-, P- and B-frames only)
Frame dimensions	720 x 576 pixels
Pixel resolution	24 bits/pixel (colour)
Frame rate	29.97/second
Average bandwidth	692150 bytes/second
Average compression rate	53.87
A group of pictures	15 frames (IBBPBBPBBPBBPBB)

IBBPBBPBBPBB.... Figure 6.2 shows the MPEG GOP pattern used by Garrett and Willinger [42] to encode the MPEG-1 version of *Star Wars*.

The GOP has a picture header followed by one or more *slices*. In turn, each slice comprises a slice header and one or more groups of DCT blocks called *macroblocks*. The first slice starts in the upper left corner of the picture and the last slice ends in the lower right corner. The macroblock is a group of six 8 x 8 DCT blocks: four blocks contain luminance samples and two contain chrominance samples. Each macroblock starts with a macroblock header containing information on which DCT blocks are actually coded.

We encoded three hours of *Titanic* video to obtain a realistic full-length trace of video traffic. This will be used as a control reference self-similar trace in our investigations. Parameters of the sequence are summarised in Table 6.1. We chose MPEG-2 to obtain encoded frame sequences of our trace. In this chapter we will use the following three self-similar sequences:

- Sequence A: two hours of *Star Wars* video encoded by JPEG [42]
- Sequence B: two hours of *Star Wars* video encoded by MPEG-1 [42]
- Sequence C: three hours of *Titanic* video encoded by MPEG-2.

The last trace was obtained by taking a sample that was approximately 50% longer than Sequence A and B.

6.3 Modelling for Self-Similar VBR Video Traffic

Following the recommendation of Garrett and Willinger [42], we chose to use the combined gamma/Pareto model for VBR video traffic. They along with Krunz and Makowski [81], showed that the gamma distribution can be used to capture the main part of the empirical distribution, but is inappropriate for the tail. Addition of a heavy-tailed Pareto distribution corrects this, as shown in Figure 6.3.

Let F_Γ and F_P be the CDF for the gamma and Pareto distributions, respectively. Note that F_Γ has no closed form of the CDF when α_Γ, the shape parameter of the gamma distribution assumes non-integer values. The gamma distribution has a PDF given by

$$f_\Gamma(x) = \begin{cases} 0, & \text{for } x \leq 0, \\ \frac{\beta_\Gamma^{-\alpha_\Gamma} e^{-x/\beta_\Gamma} x^{\alpha_\Gamma - 1}}{\Gamma(\alpha_\Gamma)}, & \text{for } x > 0, \end{cases} \tag{6.1}$$

where α_Γ is the shape parameter, $\alpha_\Gamma > 0$, β_Γ is the scale parameter, $\beta_\Gamma > 0$, and $\Gamma(\alpha_\Gamma)$ is the gamma function. If α_Γ is a positive integer, then the CDF for the gamma distribution is given by Equation (5.5).

The PDF $f_P(x)$ and CDF $F_P(x)$ of the Pareto distribution are given as:

$$f_P(x) = \alpha_P b_P^{\alpha_P} x^{-(\alpha_P + 1)}, \text{ for } 1 \leq x \leq \infty, \tag{6.2}$$

and

$$F_P(x) = \begin{cases} 0, & \text{for } x < 1, \\ 1 - \left(\frac{b_P}{x}\right)^{\alpha_P}, & \text{for } 1 \leq x \leq \infty, \end{cases} \tag{6.3}$$

where α_P is the shape parameter, $\alpha_P > 0$, and b_P is the minimum allowed value of $x, 0 < b_P \leq x$.

Thus, the combined gamma/Pareto distribution is determined by

$$f_{\Gamma/P}(x) = \begin{cases} f_\Gamma(x), & \text{for } x \le x^*, \\ f_P(x), & \text{for } x > x^*, \end{cases} \tag{6.4}$$

or by

$$F_{\Gamma/P}(x) = \begin{cases} 0, & \text{for } x \le 0, \\ F_\Gamma(x), & \text{for } 0 < x \le x^*, \\ F_P(x), & \text{for } x > x^*. \end{cases} \tag{6.5}$$

The complementary CDFs of $F_\Gamma(x)$ and $F_P(x)$ can be used to determine x^* in Equation (6.5). The parameters of the gamma distribution are obtained by matching the first and second moments of the empirical sequence to those of a gamma random variate. x^* can be obtained graphically by inspecting the tail behaviour of the empirical distribution, and determining where it starts to deviate from the tail of the gamma curve. The values of b_P and α_P for the estimated Pareto distribution can be obtained by finding $x = x^*$ for which the least-square fit of the Pareto tail gives $F_\Gamma(x) = F_P(x)$. Figures 6.3 – 6.5 show log-log plots of gamma and Pareto complementary CDF for real VBR video traffic. While the gamma curve fits the main part of the empirical video traffic well, the Pareto curve closely fits its tail part. Applying this method, we have determined values of x^* for all three samples; see Table 6.2.

Given a self-similar sequence of the FGN-DW process $\mathbf{X}$, we can transform the marginal distribution by mapping each point as

$$Z_i = F^{-1}_{\Gamma/P}(F_N(X_i)), \;\; i = 1, 2, \ldots, \tag{6.6}$$

where $F_N(\cdot)$ is the CDF of the normal distribution and $F^{-1}_{\Gamma/P}(\cdot)$ is the inverse CDF of the combined gamma/Pareto model given by

$$F^{-1}_{\Gamma/P}(y) = \begin{cases} F^{-1}_\Gamma(y), & \text{for } y \le 1 - (b_P/x^*)^{\alpha_P}, \\ F^{-1}_P(y) = b_P/(1-y)^{1/\alpha_P}, & \text{for } y > 1 - (b_P/x^*)^{\alpha_P}. \end{cases} \tag{6.7}$$

Note that for computing $F^{-1}_\Gamma(y)$, we used the Newton-Raphson technique mentioned in Section 5.2.1. Both the complete *Star Wars* video sequence in Figure 6.6 and the synthetic sequence generated by Equation (6.6), based on FGN-DW, and shown in Figure 6.7, look like self-similar. Figure 6.8 shows the procedure for the generation of synthetic self-similar JPEG/MPEG video

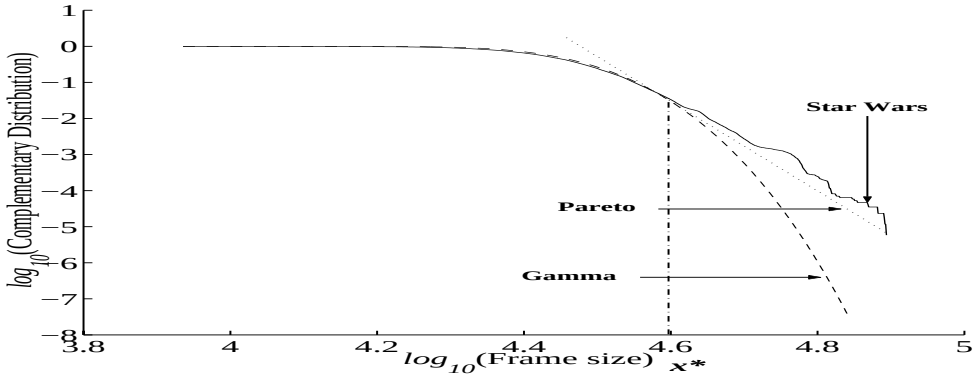


Figure 6.3: Complementary cumulative distributions of real *Star Wars* JPEG video traffic, gamma/Pareto model.

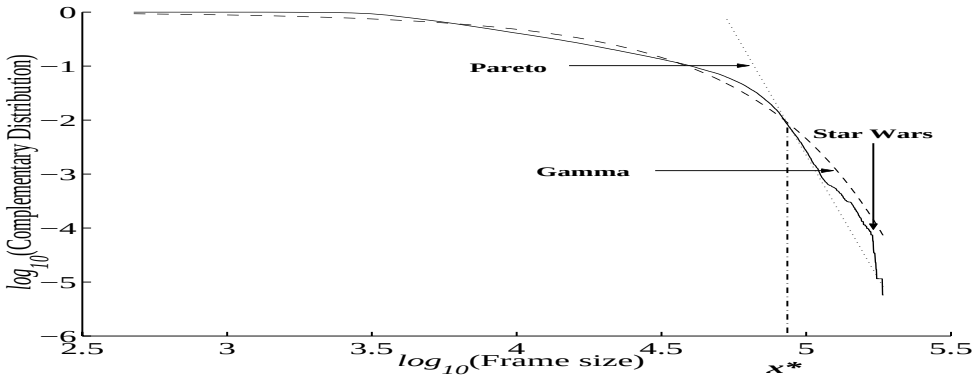


Figure 6.4: Complementary cumulative distributions of real *Star Wars* MPEG-1 video traffic, gamma/Pareto model.

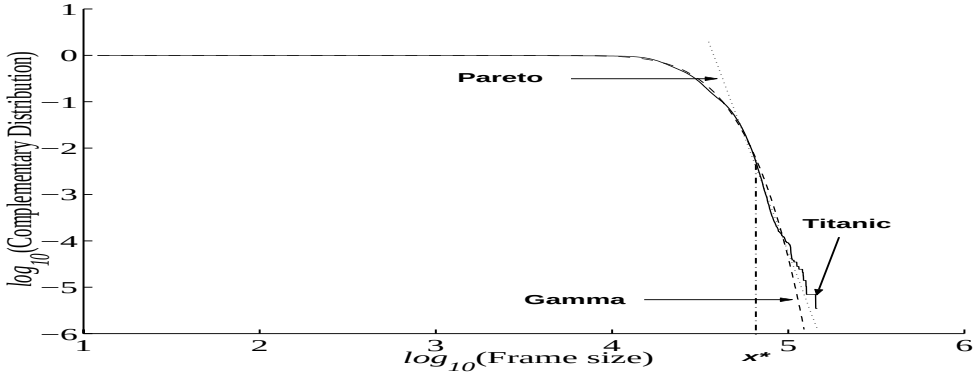


Figure 6.5: Complementary cumulative distributions of real *Titanic* MPEG-2 video traffic, gamma/Pareto model.

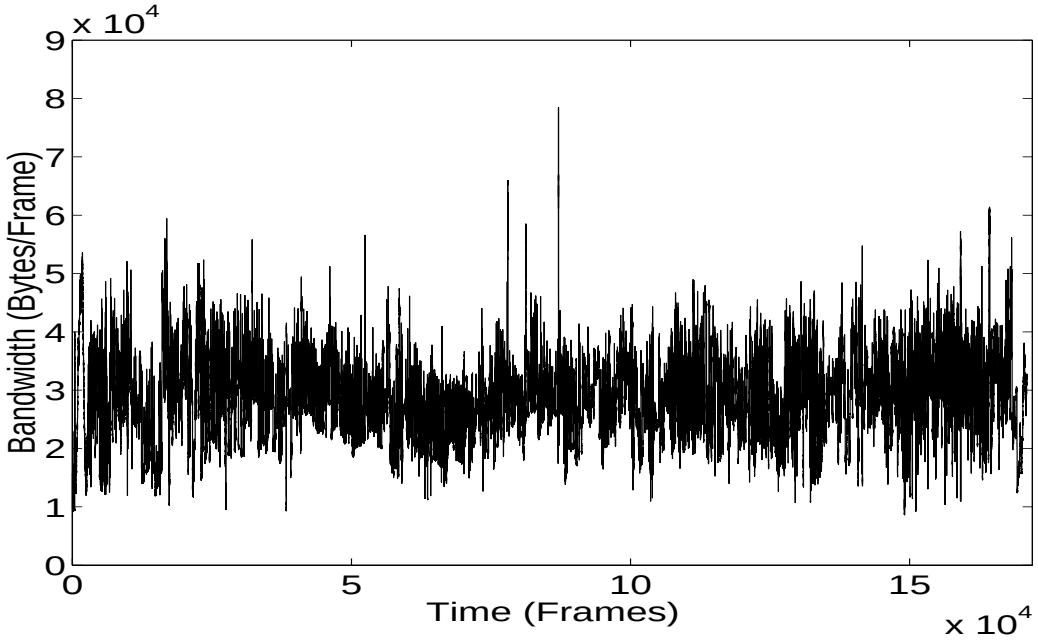


Figure 6.6: Sequence plot of two hours of real *Star Wars* JPEG video traffic.

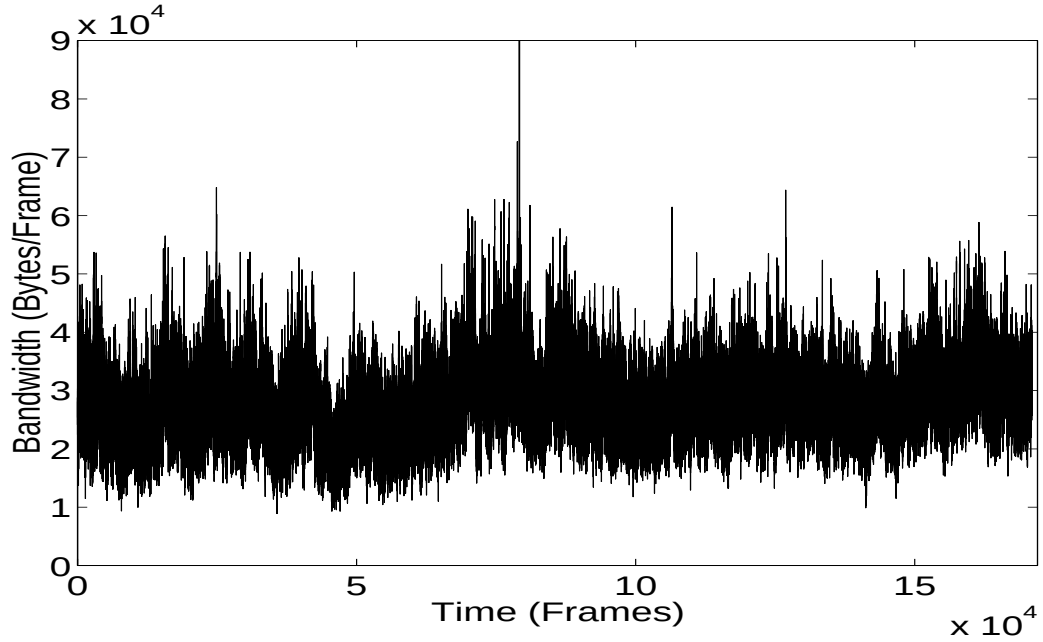


Figure 6.7: Sequence plot of two hours of synthetic FGN-DW traffic.

sequences. The procedure for the MPEG video consists of I-, P- and B-frames of sequences generated from FGN-DW, which are then combined in I-, B- and P-frame order (e.g., IBBPBBPBBPBBPBBI ...) before transforming the corresponding time series into time series with the gamma/Pareto marginal distributions defined in Equation (6.6). The procedure for the JPEG synthetic sequence generated from FGN-DW is simple, as shown in Figure 6.8. We used the sequences obtained from the previous procedure for simulation studies of VBR video traffic, which we describe in the next section.

Table 6.2: Estimated parameter values obtained from *Star Wars* and *Titanic* video traffic utilising the combined gamma/Pareto model.

Parameters	Estimated values		
	Star Wars JPEG	*Star Wars* MPEG-1	*Titanic* MPEG-2
Length (frames)	171,000	174,136	285,890
Duration	2 hours	2 hours	3 hours
Compression algorithm	Intra-frame	MPEG-1	MPEG-2
Sample mean	27,791	15,598	26,353
Standard dev.	6,254	18,165	11,600
Maximum	78,459	185,267	146,608
Minimum	8,622	476	12
Gamma α_Γ	25.8	0.737	5.16
Gamma β_Γ	1,100	21,154	5,106
Pareto α_P	12.42	9.19	10.06
Pareto b_P	30,000	51,500	37,800
x^*	39,810	86,003	57,280

6.4 Numerical Results

6.4.1 Analysis of Hurst Parameter Estimates for VBR Video Traffic

The Hurst parameter estimates obtained from the wavelet-based H estimator and Whittle's MLE, have been used to analyse *Star Wars* JPEG, *Star Wars*

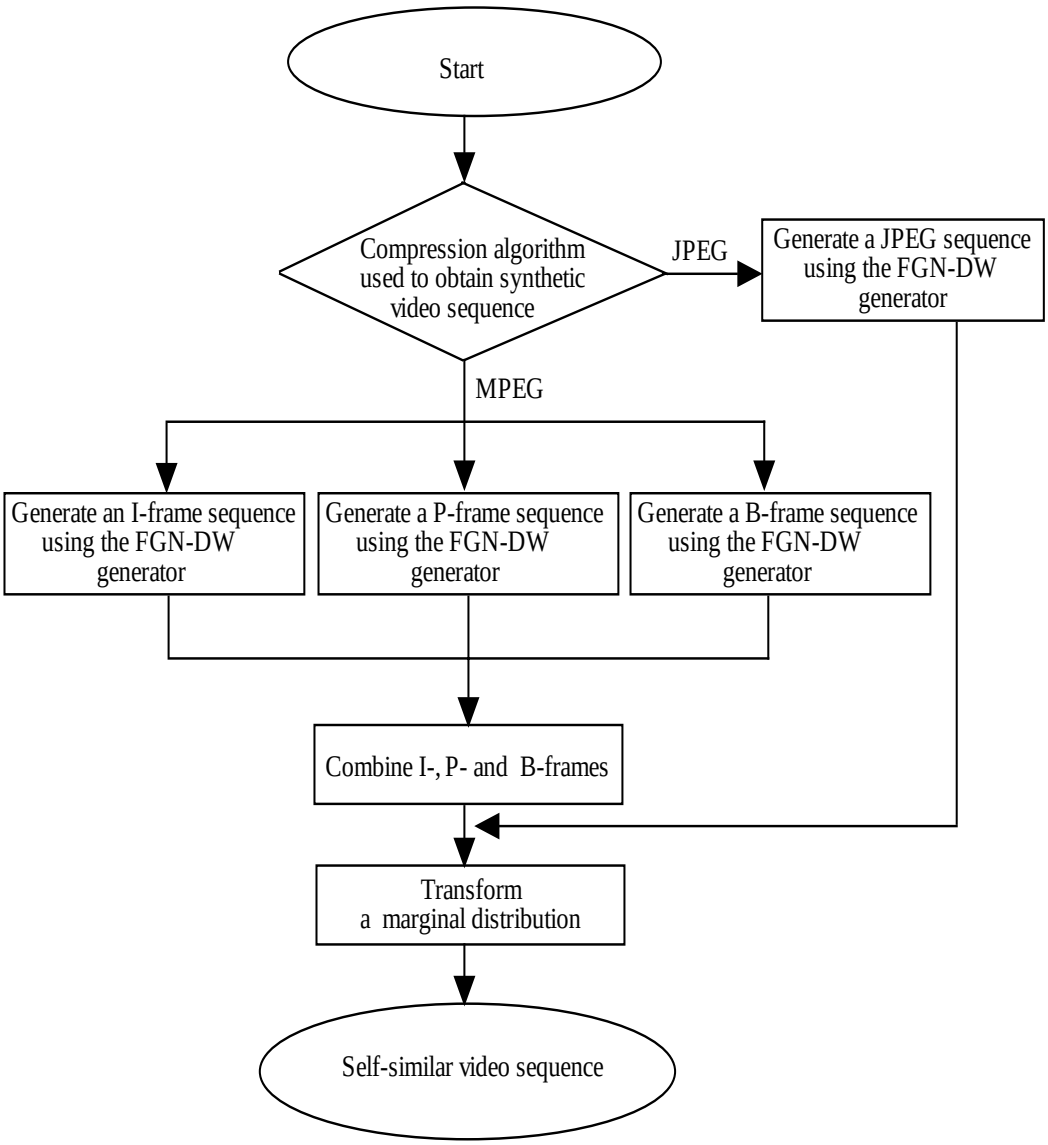


Figure 6.8: Flowchart for the generation of synthetic self-similar video sequences.

Table 6.3: Estimates of the Hurst parameter obtained from the wavelet-based H estimator and Whittle's MLE for *Star Wars* JPEG, *Star Wars* MPEG-1 and *Titanic* MPEG-2 video traffic. We give 95% confidence intervals for the means of two estimators in parentheses.

Estimators	*Star Wars* JPEG	*Star Wars* MPEG-1	*Titanic* MPEG-2
Wavelet-based	.8841(.879, .889)	.8634(.859, .868)	.9034(.895, .911)
Whittle's MLE	.8997(.887, .912)	.8680(.855, .880)	.8999(.886, .914)

MPEG-1 and *Titanic* MPEG-2 video sequences. Table 6.3 shows the estimates of the Hurst parameter for the three video sequences. Comparing *Star Wars* after JPEG and MPEG-1, we can formulate hypothesis that JPEG produces stronger dependent output video sequences. Our results show 2% difference in H parameter when using the wavelet-based H estimator, and 11% difference when using Whittle's MLE, see Table 6.3. On the other hand, two different videos (*Star Wars* and *Titanic*) show that regardless of compression algorithm resulted processes have the same marginal distribution well approximated by our gamma/Pareto model. This can be regarded as generalisation of a finding by Garrett and Willinger [42] who showed that gamma/Pareto model is a good approximation of marginal distributions for *Star Wars* compressed under JPEG.

Figure 6.9 depicts the wavelet-based H estimator curve of three hours of real *Titanic* video traffic. The wavelet-based H estimator calculates $\hat{H} = 0.9034$, shown in Table 6.3. Estimate of the Hurst parameter $\hat{H}$ obtained from Whittle's MLE is 0.8999. Figure 6.10 shows higher levels of data correlation for I-, P-, B- and all frames. The Hurst parameter estimates for the *Star Wars* JPEG and *Star Wars* MPEG-1 video sequences are also given in Table 6.3 [42].

6.4.2 Simulation Results of the VBR Video Traffic Model

We investigated frame loss probabilities of VBR video traffic obtained from the gamma/Pareto model using a queueing system with self-similar input. The frame loss probabilities are defined as the ratio of the number of lost

frames and total number of transmitted frames in a finite buffer system during a given period of the time. The queueing system consists of a single-server FIFO queue with a finite buffer size B and a constant service rate μ, using estimated parameter values in Table 6.2. We compared the performance of the original *Star Wars* JPEG sequence, the *Star Wars* MPEG-1 sequence and the *Titanic* MPEG-2 sequence using our queueing model.

Figures 6.11 – 6.13 show quantile-quantile plots for the distribution of frame sizes in bytes of real VBR video traffic, (i.e., *Star Wars* JPEG, *Star Wars* MPEG-1, and *Titanic* MPEG-2), and the combined gamma/Pareto model based on FGN-DW. We observed that although the compression algorithms used for encoding the various videos were different, the combined model fits the real traffic statistic well. Note especially that the distribution of the gamma/Pareto model matches the real *Star Wars* JPEG video traffic well.

Figures 6.14 – 6.17 show that the ACF of the combined gamma/Pareto model also fit the empirical video traffic statistic well. The ACF curve of the

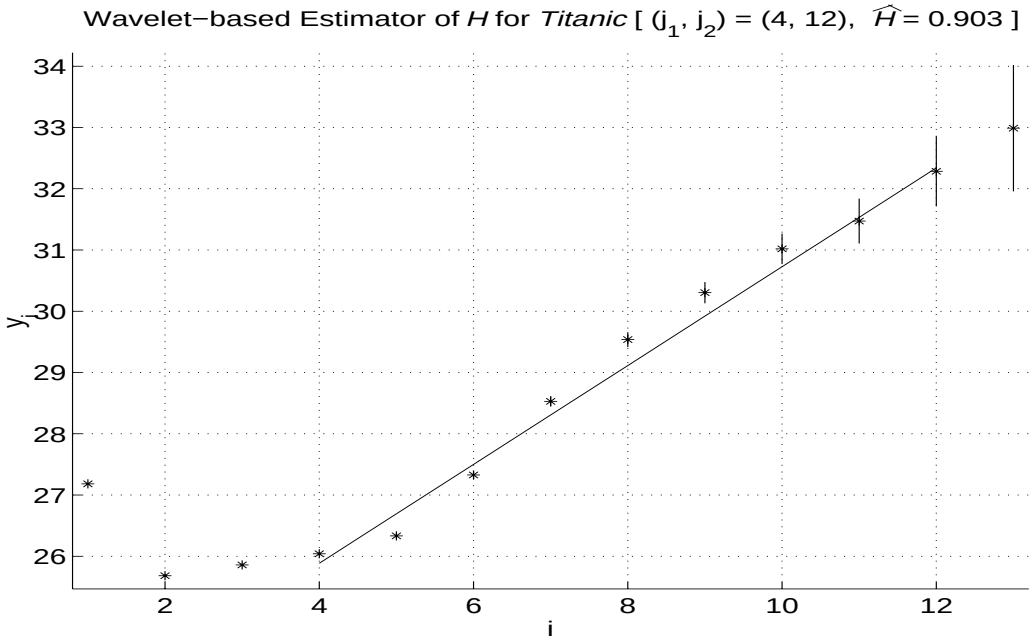


Figure 6.9: Wavelet-based estimator $\hat{H}$ of three hours of real *Titanic* video traffic.

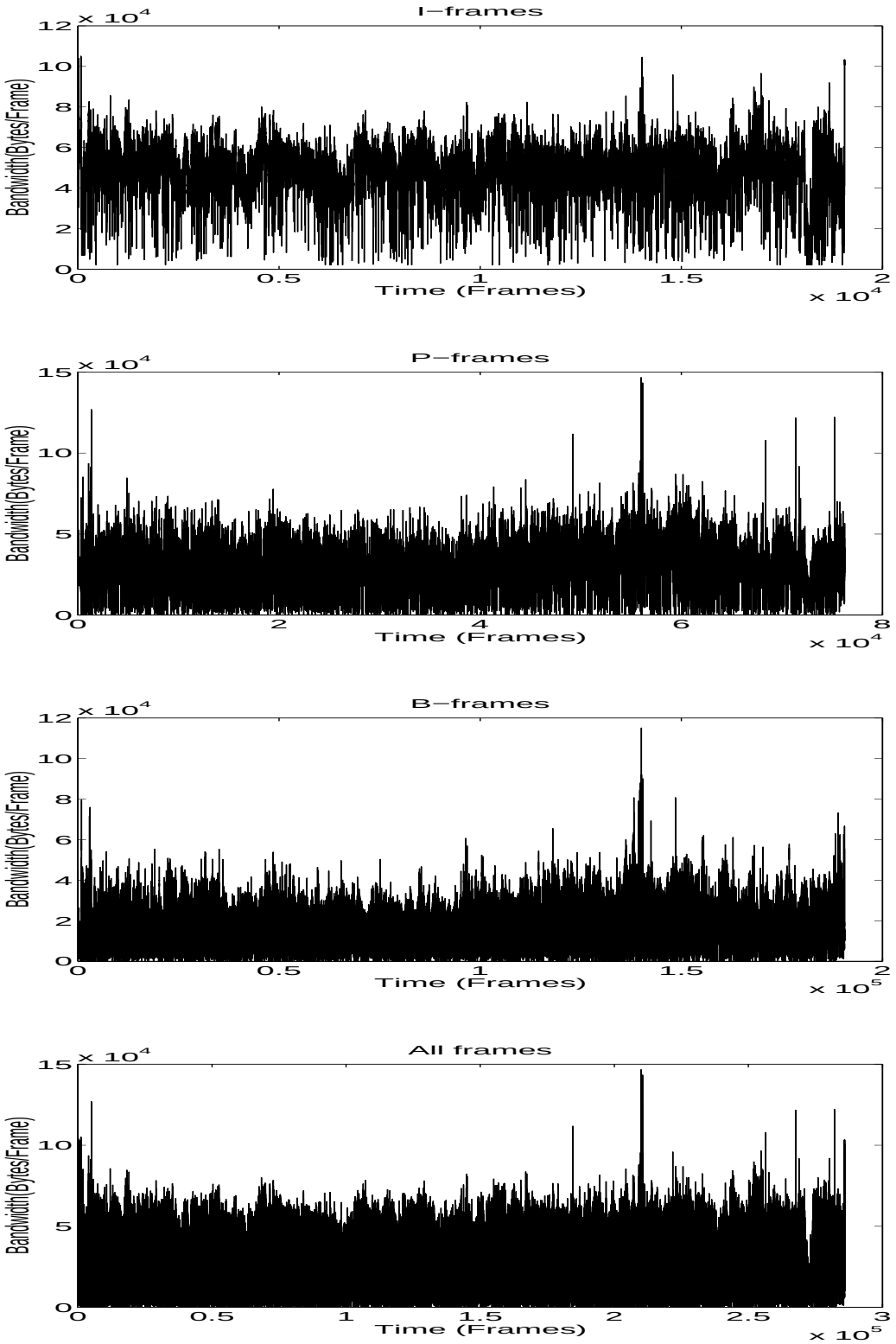


Figure 6.10: I-, P-, B- and all-frames plot of three-hours of real *Titanic* video traffic.

gamma/Pareto model at large lags (i.e., lags > 1,800) fit the real *Star Wars* JPEG video traffic well, but the model slightly underestimated at small lags. The ACF curves in Figures 6.15 – 6.16 also oscillated more than the one in Figure 6.14, due to the MPEG format. Furthermore, the autocorrelation structure in Figure 6.15 oscillated more than that in Figure 6.16 because they use different frame formats (i.e., while every 12th frame in Figure 6.15 is an I-frame, every 15th frame in Figure 6.16 is an I-frame). In addition, we found strong evidence of LRD, as all ACF curves obtained from the real video traffic and the gamma/Pareto model decayed slowly, while the SRD (i.e., Poisson model) in Figure 6.14 decayed quickly. The frame loss probability as a measurement of performance was examined at traffic intensities $\rho = 0.6$ and 0.8.

Star Wars JPEG Video Traffic

Figures 6.18 and 6.19 show the simulation results of real video traffic, the combined gamma/Pareto model and the SRD model at traffic intensities $\rho = 0.6$ and $\rho = 0.8$, respectively (for distances between the curves, see Tables 6.4 and 6.5). (Note that "Real Traffic - Model" in Tables 6.4 – 6.9 means the mean value of the difference between the frame loss probability for real video traffic ("Real Traffic") and the frame loss probability in the combined gamma/Pareto model ("Model"). "Real Traffic - SRD" means the mean value of the difference between the frame loss probability for real video traffic ("Real Traffic") and the frame loss probability in the SRD model ("SRD"). "Model - SRD" means the mean value of the difference between the frame loss probability in the combined gamma/Pareto model ("Model") and the frame loss probability in the SRD model ("SRD").)

The results of the gamma/Pareto model were averaged over 30 replications. The frame loss probabilities of the gamma/Pareto model in Figure 6.18 at $\rho = 0.6$ matched the real video traffic represented by the real *Star Wars* trace well for buffer sizes up to about 1,300 bytes. In the case of larger buffer sizes, discrepancies between the results for the trace of the real and artificial traffic has been most likely caused by the finite length of the real traffic trace we used. Figure 6.19 shows that the gamma/Pareto model based on FGN-DW matched the real video traffic well for buffer sizes up to about 2,300 bytes. Both the real video traffic and the gamma/Pareto model frame loss

probabilities tended to decrease much more slowly than those of the SRD model as buffer sizes increased. In other words, as in Tsybakov and Georganas [166], our analysis reveals that the frame loss probabilities shown in both figures decayed hyperbolically with increasing buffer size. Our results also show that the frame loss probabilities of the real video traffic and the gamma/Pareto model are much higher than those of the SRD model.

Star Wars MPEG-1 Video Traffic

Figures 6.20 and 6.21 show the simulation results of real *Star Wars* MPEG-1 video traffic, the gamma/Pareto model and the SRD model at traffic intensities $\rho = 0.6$ and 0.8; see also Tables 6.6 and 6.7. For $\rho = 0.6$, the frame loss probabilities of the gamma/Pareto model matched the real traffic well. However, for $\rho = 0.8$, the frame loss probabilities of the gamma/Pareto model were underestimated slightly more than those of the real *Star Wars* MPEG-1 video traffic when the buffer size was greater than 800. As we expected, curves of the real *Star Wars* MPEG-1 video traffic and the gamma/Pareto model decayed much more slowly than the SRD model and oscillated more with increasing buffer size than those obtained from the *Star Wars* JPEG because of the typical MPEG GOP pattern.

Titanic MPEG-2 Video Traffic

Figures 6.22 and 6.23 show the frame loss probabilities of real *Titanic* MPEG-2 video traffic, the combined gamma/Pareto model and the SRD model at $\rho = 0.6$ and 0.8 (see also Tables 6.8 and 6.9). Both figures show that the gamma/Pareto model based on FGN-DW matched the real video traffic well over all buffer sizes. These results also show that the two curves of real *Titanic* MPEG-2 video traffic and the gamma/Pareto model decreased more quickly than the *Star Wars* JPEG video traffic, but less quickly than the *Star Wars* MPEG-1 video traffic because of the frame format (every 15th frame is an I-frame). The sudden changes in frame loss probabilities in Figure 6.23 might have been caused by an insufficient length of data. As with the *Star Wars* MPEG-1 video, the decay demonstrated by the graphical curves of the real

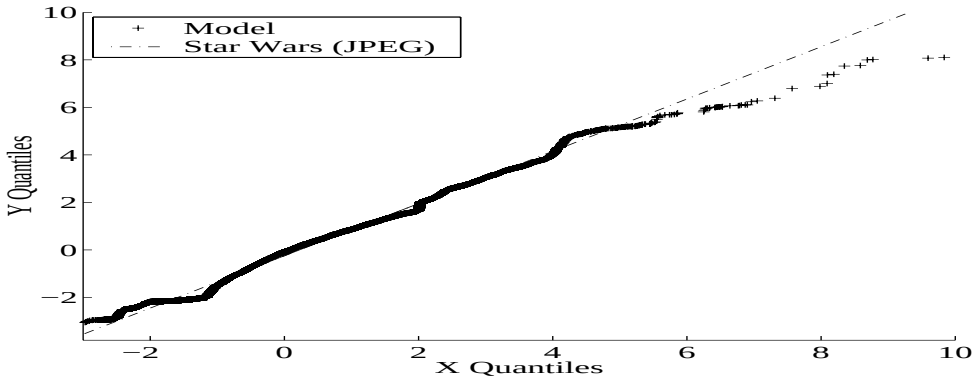


Figure 6.11: Distributions of real *Star Wars* JPEG video traffic and traffic from the gamma/Pareto models.

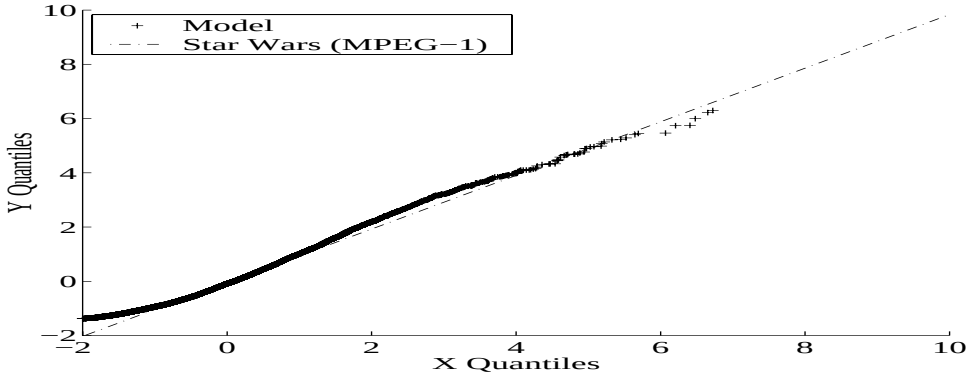


Figure 6.12: Distributions of real *Star Wars* MPEG-1 video traffic and traffic from the gamma/Pareto models.

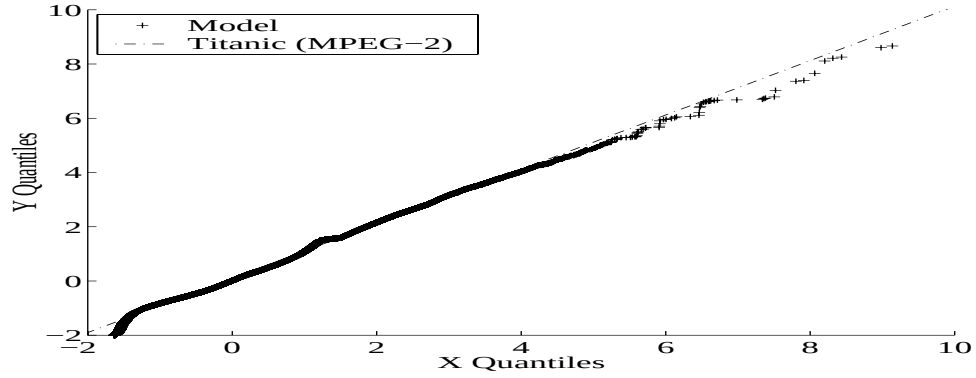


Figure 6.13: Distributions of real *Titanic* MPEG-2 video traffic and traffic from the gamma/Pareto models.

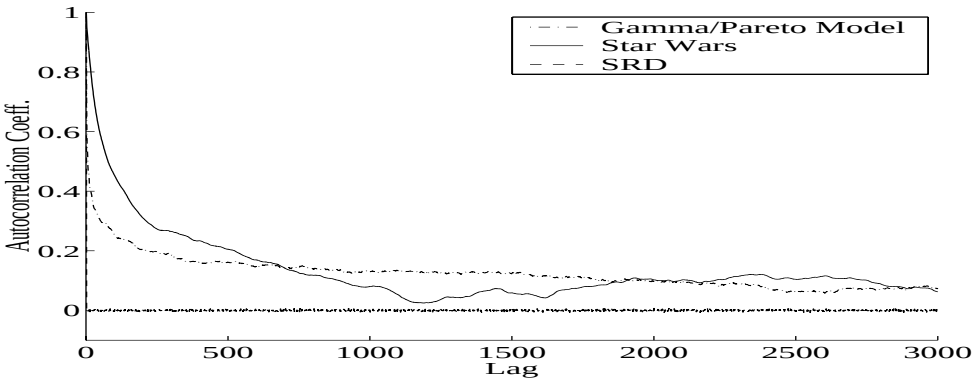


Figure 6.14: ACF of real *Star Wars* JPEG video traffic and traffic from the gamma/Pareto models.

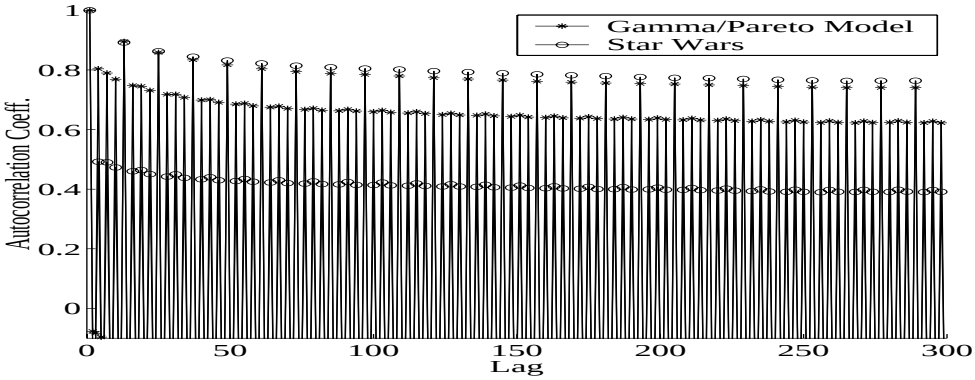


Figure 6.15: ACF of real *Star Wars* MPEG-1 video traffic and traffic from the gamma/Pareto models.

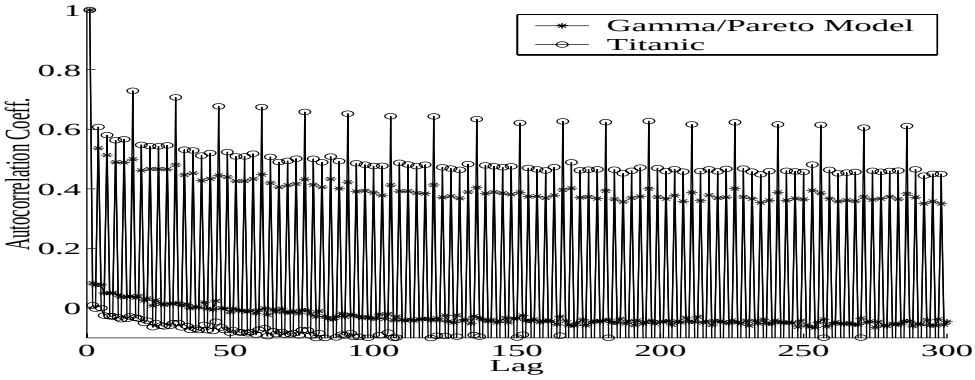


Figure 6.16: ACF of real *Titanic* MPEG-2 video traffic and traffic from the gamma/Pareto models.

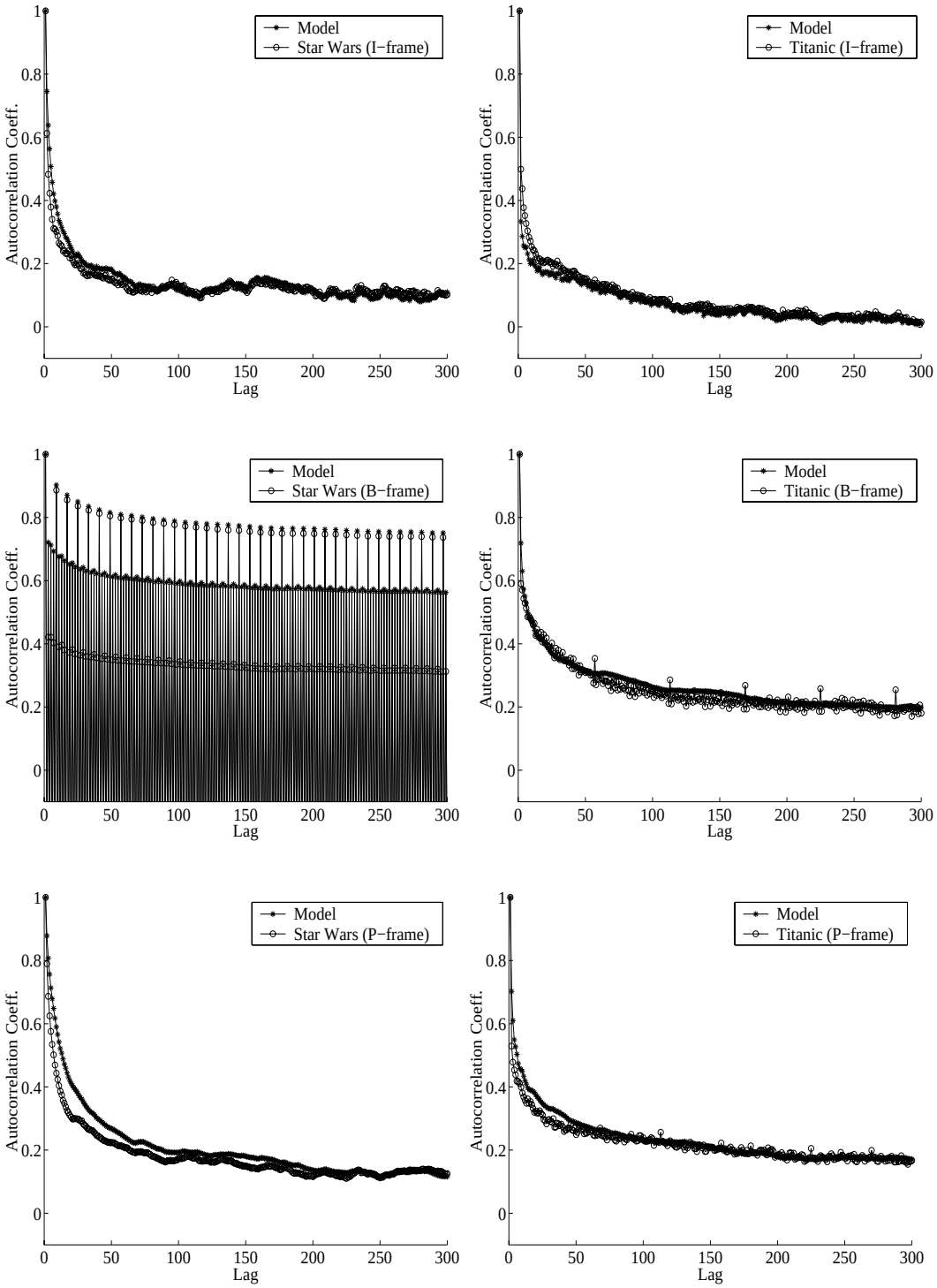


Figure 6.17: ACFs of I-, B-, and P-frames of real *Star Wars* MPEG-1, *Titanic* MPEG-2 video traffic and traffic estimated by gamma/Pareto models.

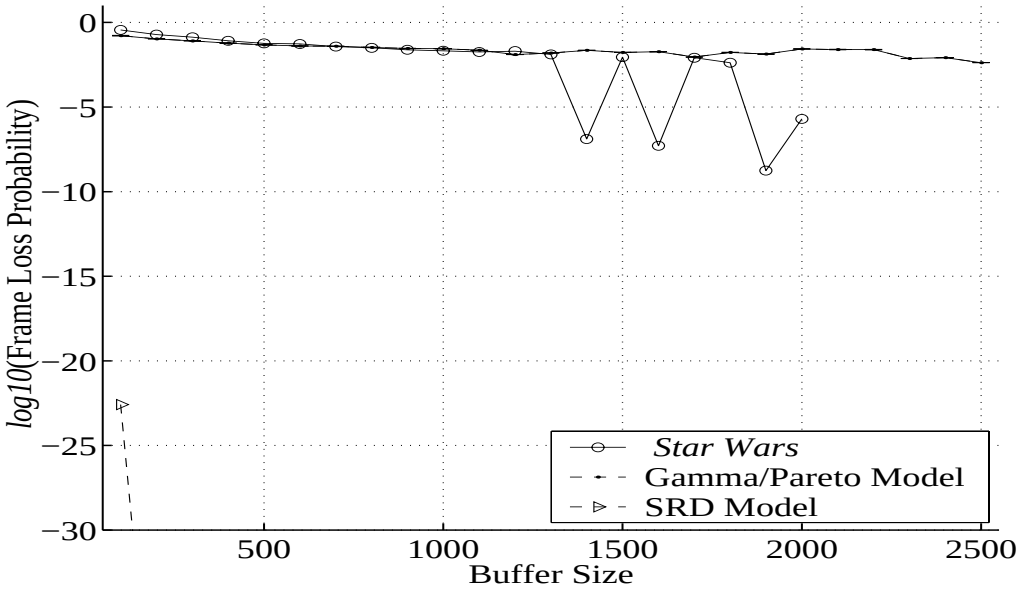


Figure 6.18: Frame loss probabilities of real *Star Wars* JPEG video traffic and the combined gamma/Pareto model ($\rho = 0.6$).

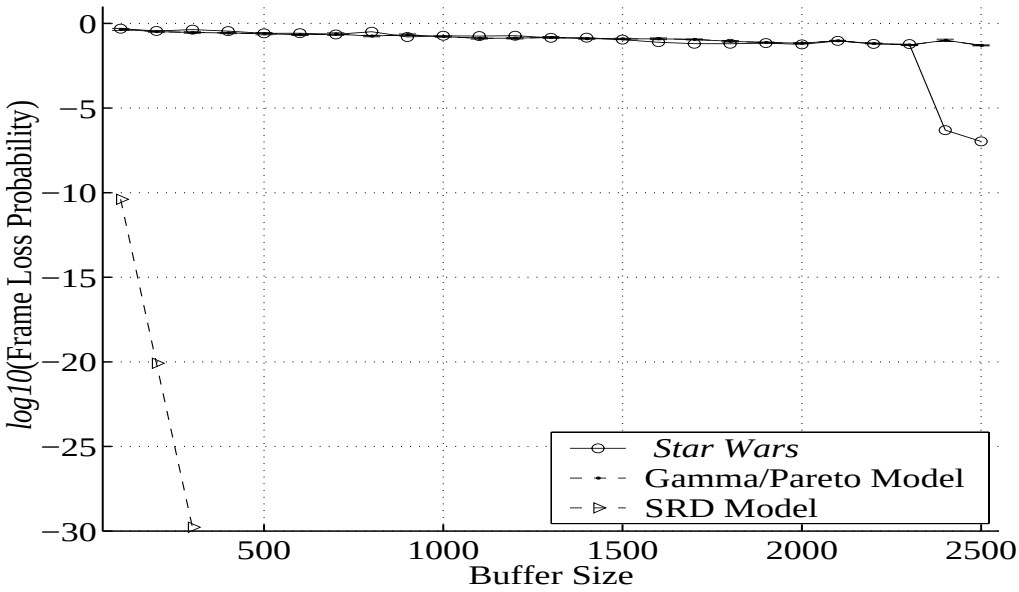


Figure 6.19: Frame loss probabilities of real *Star Wars* JPEG video traffic and the combined gamma/Pareto model ($\rho = 0.8$).

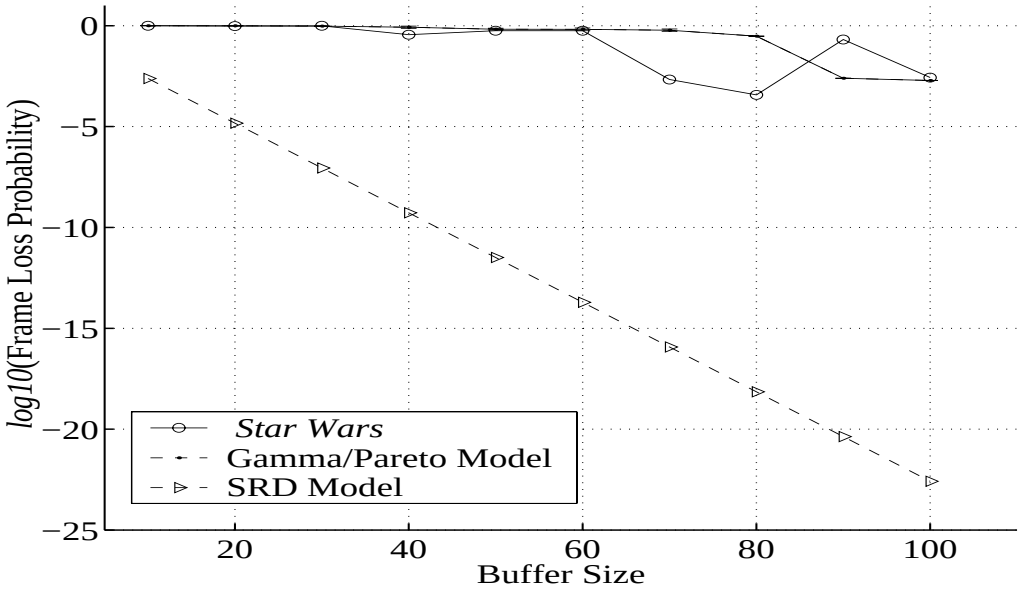


Figure 6.20: Frame loss probabilities of real *Star Wars* MPEG-1 video traffic and the combined gamma/Pareto model ($\rho = 0.6$).

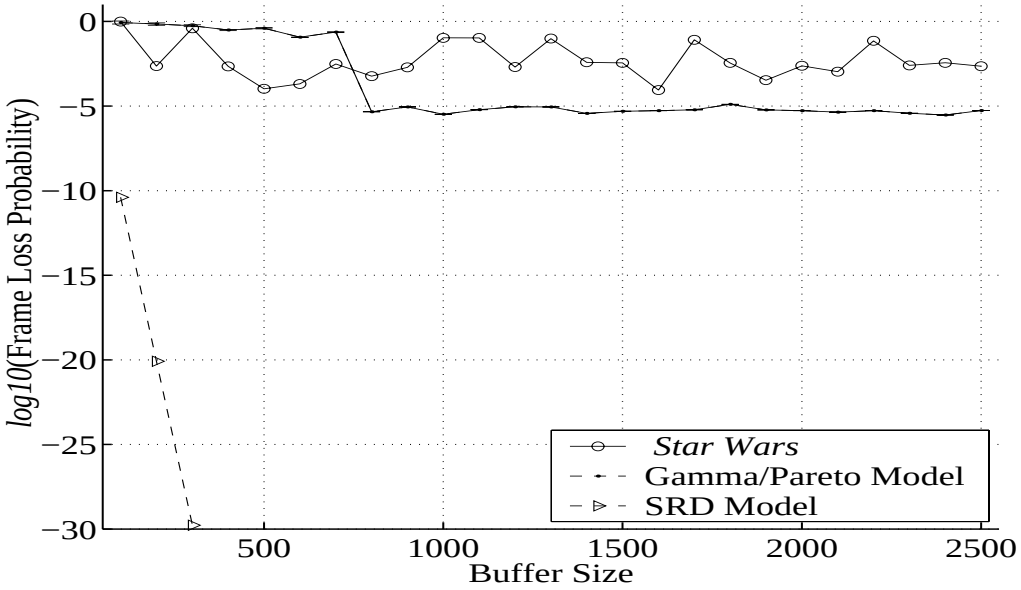


Figure 6.21: Frame loss probabilities of real *Star Wars* MPEG-1 video traffic and the combined gamma/Pareto model ($\rho = 0.8$).

video traffic and the gamma/Pareto model was significantly slower than in the SRD model.

Table 6.4: Mean values of difference between the curves of frame loss probabilities of real *Star Wars* JPEG video traffic and the combined gamma/Pareto model ($\rho = 0.6$) in Figure 6.18.

Buffer Size	Real Traffic - Model	Real Traffic - SRD	Model - SRD
100	1.9533e-01	3.5933e-01	1.6400e-01
200	8.3619e-02	1.9072e-01	1.0710e-01
300	5.3935e-02	1.3564e-01	8.1700e-02
400	2.3107e-02	8.3117e-02	6.0010e-02
500	1.0624e-02	5.8064e-02	4.7440e-02
600	1.3792e-02	5.4392e-02	4.0600e-02
700	1.7120e-03	3.7538e-02	3.9250e-02
800	1.2780e-03	3.2152e-02	3.3430e-02
900	5.1604e-03	2.3860e-02	2.9020e-02
1000	6.6463e-03	2.1164e-02	2.7810e-02
1100	4.6164e-03	1.8094e-02	2.2710e-02
1200	7.2675e-03	2.0018e-02	1.2750e-02
1300	2.6802e-03	1.2930e-02	1.5610e-02
1400	2.2480e-02	1.3112e-07	2.2480e-02
1500	8.1637e-03	8.8363e-03	1.7000e-02
1600	1.8620e-02	5.1811e-08	1.8620e-02
1700	7.0436e-04	8.3216e-03	9.0260e-03
1800	1.2858e-02	4.1521e-03	1.7010e-02
1900	1.3570e-02	1.7601e-09	1.3570e-02
2000	2.7068e-02	1.9911e-06	2.7070e-02
2100	2.5080e-02	0.0000e+00	2.5080e-02
2200	2.4810e-02	0.0000e+00	2.4810e-02
2300	7.3140e-03	0.0000e+00	7.3140e-03
2400	8.3320e-03	0.0000e+00	8.3320e-03
2500	4.1750e-03	0.0000e+00	4.1750e-03

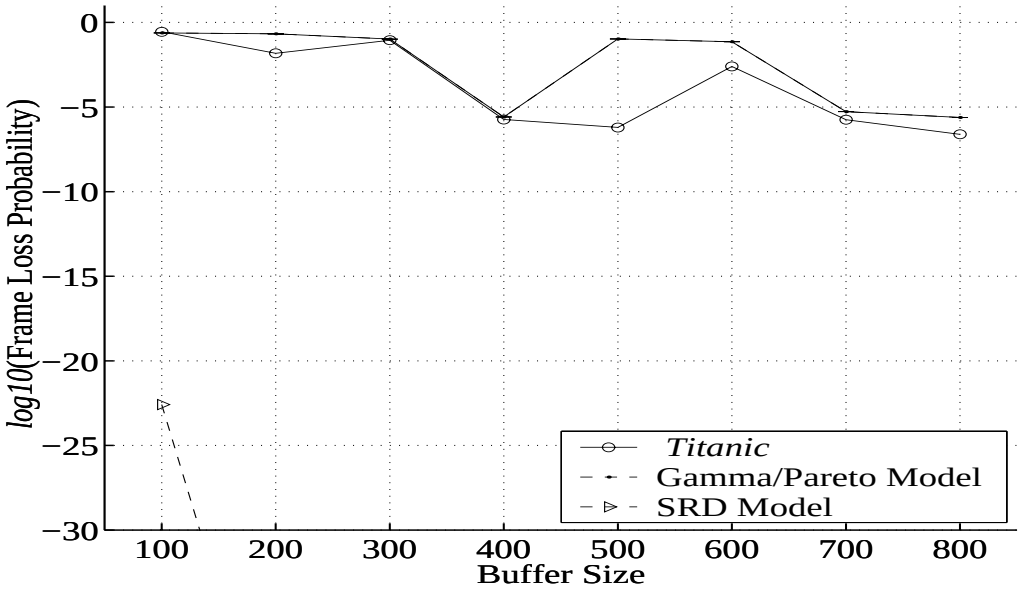


Figure 6.22: Frame loss probabilities of real *Titanic* MPEG-2 video traffic and the combined gamma/Pareto model ($\rho = 0.6$).

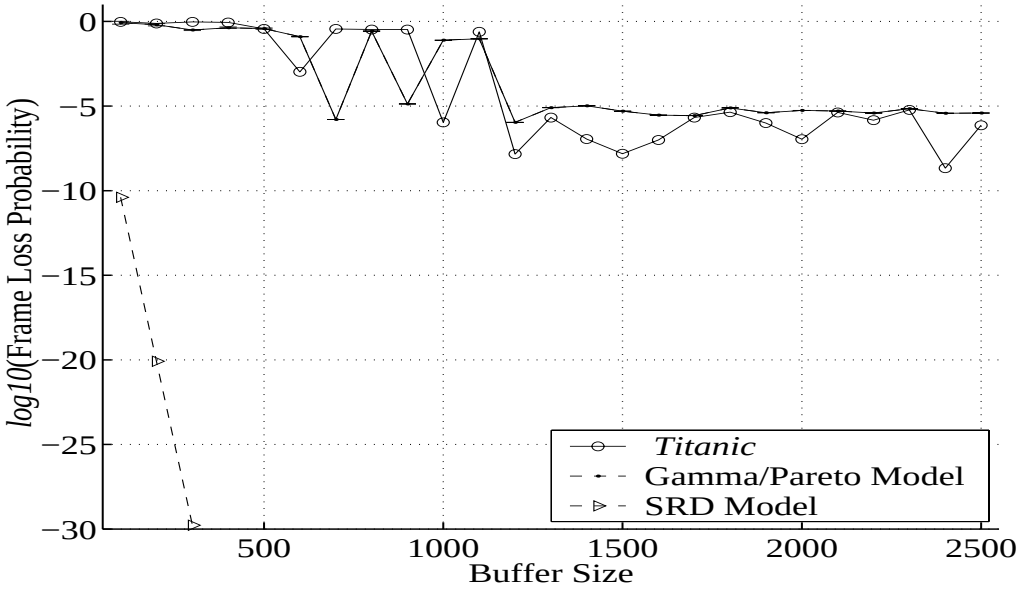


Figure 6.23: Frame loss probabilities of real *Titanic* MPEG-2 video traffic and the combined gamma/Pareto model ($\rho = 0.8$).

6.5 Conclusions

We showed how pseudo-random self-similar sequences can be applied to produce a model of teletraffic associated with the transmission of VBR JPEG/MPEG

Table 6.5: Mean values of difference between the curves of frame loss probabilities of real *Star Wars* JPEG video traffic and the combined gamma/Pareto model ($\rho = 0.8$) in Figure 6.19.

Buffer Size	Real Traffic - Model	Real Traffic - SRD	Model - SRD
100	4.8845e-02	4.9454e-01	4.4570e-01
200	2.4524e-02	3.5592e-01	3.3140e-01
300	1.4122e-01	4.3252e-01	2.9130e-01
400	7.9468e-02	3.5167e-01	2.7220e-01
500	1.4860e-02	2.6226e-01	2.4740e-01
600	4.8633e-02	2.6343e-01	2.1480e-01
700	6.0480e-03	2.2815e-01	2.3420e-01
800	1.4609e-01	3.2509e-01	1.7900e-01
900	5.2323e-02	1.5988e-01	2.1220e-01
1000	1.7216e-02	1.8522e-01	1.6800e-01
1100	4.7065e-02	1.7886e-01	1.3180e-01
1200	5.1855e-02	1.8765e-01	1.3580e-01
1300	7.5770e-03	1.4202e-01	1.4960e-01
1400	1.7397e-02	1.4350e-01	1.2610e-01
1500	1.1219e-02	1.1018e-01	1.2140e-01
1600	4.6842e-02	7.7158e-02	1.2400e-01
1700	4.8925e-02	6.4175e-02	1.1310e-01
1800	2.5853e-02	6.3047e-02	8.8900e-02
1900	6.8608e-03	6.8339e-02	7.5200e-02
2000	1.0820e-02	5.7550e-02	6.8370e-02
2100	1.5807e-03	9.2719e-02	9.4300e-02
2200	3.1493e-03	6.2281e-02	6.5430e-02
2300	6.7497e-03	5.9550e-02	5.2800e-02
2400	1.0190e-01	4.9264e-07	1.0190e-01
2500	5.0820e-02	1.0620e-07	5.0820e-02

video. A combined gamma/Pareto model based on the application of the FGN-DW generator was used to synthesise VBR JPEG/MPEG video traffic.

In the chapter we showed that synthetically generated streams of VBR video, compressed under such standards as JPEG, MPEG-1 and MPEG-2, can be statistically equivalent to real video traces. We investigated how compression algorithms on correlation structure of compressed teletraffic influence real video traffic. Generalisation of findings of Garrett and Willinger, showing that video compression algorithms (MPEG-1 and MPEG-2) lead to self-similar processes was studied. We considered outcomes of MPEG-1 and MPEG-2 in addition to previously studied outcomes of JPEG (Garrett and Willinger [42]), to show that the results of (Garrett and Willinger [42]) do not depend on the compression algorithms.

We also presented that results of a steady-state simulation of a single buffer fed by these synthetic video streams are used to show that simulations based on synthetic streams of teletraffic can provide the same qualitative and quantitative results as simulations based on real traces of VBR video. We showed that all curves of frame loss probabilities obtained from the gamma/Pareto model and three real traffic videos decayed much more slowly than the SRD

Table 6.6: Mean values of difference between the curves of frame loss probabilities of real *Star Wars* MPEG-1 video traffic and the combined gamma/Pareto model ($\rho = 0.6$) in Figure 6.20.

Buffer Size	Real Traffic - Model	Real Traffic - SRD	Model - SRD
10	3.7000e-05	9.9751e-01	9.9747e-01
20	4.0061e-02	9.5582e-01	9.9589e-01
30	2.9347e-02	9.7775e-01	9.4840e-01
40	4.8277e-01	3.5773e-01	8.4050e-01
50	1.0969e-01	5.6181e-01	6.7150e-01
60	9.6904e-02	5.6160e-01	6.5850e-01
70	5.9735e-01	2.1453e-03	5.9950e-01
80	3.0063e-01	3.7352e-04	3.0100e-01
90	2.0432e-01	2.0679e-01	2.4720e-03
100	7.7256e-04	2.7026e-03	1.9300e-03

model over all buffer sizes.

Better GOP and cell-layered modelling is needed for development of integrated MPEG video traffic models. While some general assessment of queueing

Table 6.7: Mean values of difference between the curves of frame loss probabilities of real *Star Wars* MPEG-1 video traffic and the combined gamma/Pareto model ($\rho = 0.8$) in Figure 6.21.

Buffer Size	Real Traffic - Model	Real Traffic - SRD	Model - SRD
100	1.1090e-01	9.7640e-01	8.6550e-01
200	6.9771e-01	2.2890e-03	7.0000e-01
300	1.6064e-01	3.9956e-01	5.6020e-01
400	3.0937e-01	2.2273e-03	3.1160e-01
500	4.0030e-01	1.0429e-04	4.0040e-01
600	1.1660e-01	2.0412e-04	1.1680e-01
700	2.3034e-01	3.0608e-03	2.3340e-01
800	5.7760e-04	5.8221e-04	4.6030e-06
900	1.9297e-03	1.9387e-03	8.9170e-06
1000	1.0817e-01	1.0817e-01	3.2970e-06
1100	1.0697e-01	1.0697e-01	6.0730e-06
1200	1.9873e-03	1.9962e-03	8.9730e-06
1300	9.5353e-02	9.5362e-02	8.8700e-06
1400	3.7845e-03	3.7882e-03	3.7140e-06
1500	3.5627e-03	3.5676e-03	4.8740e-06
1600	8.2918e-05	8.8243e-05	5.3260e-06
1700	8.2120e-02	8.2126e-02	5.9800e-06
1800	3.5222e-03	3.5348e-03	1.2600e-05
1900	3.3017e-04	3.3606e-04	5.8920e-06
2000	2.3930e-03	2.3983e-03	5.3090e-06
2100	1.0678e-03	1.0721e-03	4.3330e-06
2200	7.1066e-02	7.1071e-02	5.3700e-06
2300	2.5466e-03	2.5504e-03	3.7620e-06
2400	3.6267e-03	3.6296e-03	2.9540e-06
2500	2.2401e-03	2.2455e-03	5.4240e-06

Table 6.8: Mean values of difference between the curves of frame loss probabilities of real *Titanic* MPEG-2 video traffic and the combined gamma/Pareto model ($\rho = 0.6$) in Figure 6.22.

Buffer Size	Real Traffic - Model	Real Traffic - SRD	Model - SRD
100	4.6738e-02	2.8524e-01	2.3850e-01
200	1.9590e-01	1.5095e-02	2.1100e-01
300	1.6347e-02	8.7853e-02	1.0420e-01
400	7.4141e-07	1.8476e-06	2.5890e-06
500	1.0570e-01	6.2734e-07	1.0570e-01
600	6.9094e-02	2.4963e-03	7.1590e-02
700	3.5803e-06	1.7686e-06	5.3490e-06
800	2.1596e-06	2.4339e-07	2.4030e-06

performance can be obtained from single-streams, more universal results could be obtained from the queueing performance analysis of multiplexed streams of video traffic. These issues await further investigations.

Table 6.9: Mean values of difference between the curves of frame loss probabilities of real *Titanic* MPEG-2 video traffic and the combined gamma/Pareto model ($\rho = 0.8$) in Figure 6.23.

Buffer Size	Real Traffic - Model	Real Traffic - SRD	Model - SRD
100	1.0800e-01	9.5040e-01	8.4240e-01
200	1.3052e-01	7.6742e-01	6.3690e-01
300	6.2271e-01	9.3441e-01	3.1170e-01
400	4.5359e-01	8.8189e-01	4.2830e-01
500	1.2736e-02	3.6476e-01	3.7750e-01
600	1.2456e-01	1.0411e-03	1.2560e-01
700	3.5738e-01	3.5739e-01	1.6010e-06
800	7.0212e-02	3.3721e-01	2.6700e-01
900	3.3614e-01	3.3615e-01	1.3400e-05
1000	7.7839e-02	1.0726e-06	7.7840e-02
1100	1.4955e-01	2.4345e-01	9.3900e-02
1200	1.0855e-06	1.4485e-08	1.1000e-06
1300	5.8476e-06	2.1234e-06	7.9710e-06
1400	1.0199e-05	1.1085e-07	1.0310e-05
1500	4.9829e-06	1.5102e-08	4.9980e-06
1600	2.8230e-06	1.0002e-07	2.9230e-06
1700	5.8618e-07	2.1258e-06	2.7120e-06
1800	3.3973e-06	4.3517e-06	7.7490e-06
1900	3.0244e-06	9.8364e-07	4.0080e-06
2000	5.3796e-06	1.0938e-07	5.4890e-06
2100	8.7891e-07	4.2141e-06	5.0930e-06
2200	2.4271e-06	1.4529e-06	3.8800e-06
2300	1.0602e-06	5.9548e-06	7.0150e-06
2400	3.7799e-06	2.1306e-09	3.7820e-06
2500	3.1211e-06	7.4387e-07	3.8650e-06

Chapter 7

STEADY-STATE SIMULATION OF SELF-SIMILAR QUEUEING PROCESSES

There are two issues considered in this chapter. Firstly, we show how self-similarity of arrival processes influences the run-length of sequential stochastic simulation of queueing systems. For this purpose, the simulation run-length of $SSM/M/1/\infty$ queueing systems, conducted for estimating steady-state mean response times is compared with the results obtained from simulations of $M/M/1/\infty$ queueing systems. $SSM/M/1/\infty$ means a self-similar inter-arrival process with an exponential marginal distribution, exponentially distributed service times, single server and infinite buffer capacity. In both cases, simulations were stopped when the final estimates were achieved with the relative statistical error[1] not larger than 10%, for 95% confidence level.

Secondly, we use a sequential steady-state simulation of the $SSM/M/1/B$ queueing system ($B < \infty$) (i.e., queueing systems with the finite buffer capacity) to investigate the influence of self-similarity in arrival processes on the buffer overflow probability. This problem has been already addressed in a con-

[1]Having collected n output data (observations), the relative statistical error is defined as the ratio of the half-width of the confidence interval of a given estimate and that estimate obtained on the basis of the n observations.

siderable number of publications. However, claims of different authors can be regarded as being mutually inconsistent.

Leland et al. [89] claimed that a self-similar process is needed to model the behaviour of a time series well enough to capture its impact on queueing performance, while Ryu and Lowen [148] claimed that the self-similarity of the arrival process needs to be taken into account only when the traffic intensity and the buffer size are very large. Heyman [57] examined how the Markov-modulated Poisson process and FBM model proposed in the literature perform on two data sets of LAN traffic. Both models overestimated loss probabilities in the area of interest. The effects of a finite buffer of size B could be estimated by calculating the probability that the infinite buffer has greater than B items in it. This approximation may be fundamentally inaccurate for calculating small loss probabilities and modest values of B. Heyman [58] also found that the approximation is more accurate for heavy-tailed distributions rather than those without power-law tails. Further investigation of the accuracy of this approximation is required.

Resnick and Samorodnitsky [134] and Dahl and Willemain [24] found that LRD arrival processes can significantly affect the performance of queueing systems. Neidhardt and Wang [113] pointed out that arrival processes with larger values of H do not always cause longer queues. On the other hand, the results of Grossglauer and Bolot [53] showed that the behaviour of buffer can be more sensitive to the marginal distribution of arrival processes than to the H value of these processes. Further, Boxma and Cohen [12] studied an approximation for the waiting time distribution for an M/G/1 queueing system with a heavy-tailed service time distribution by using the heavy traffic limit theorem. As the resulting approximation shows, there is a need to develop more accurate models of long-range dependent traffic and studying their impact on queueing systems and networks. Because of that, more work is needed to fully understand all the effects of self-similarity on queueing processes.

One of our aims of our investigations of these problems was to make a quantitative assessment of the influence of H values on buffer overflow probabilities. In Section 7.1 and Appendix F, we assessed the simulation run-lengths when simulating self-similar processes. We looked at the number of observations required in a steady-state simulation of queueing models with self-similar input processes, and compared our results with simulation run-lengths of the

same queueing models fed by SRD processes, such as the $M/M/1/\infty$ queueing system. Section 7.2 describes performance measures of queueing systems with self-similar input and finite buffer sizes (such as buffer overflow probability). The influence of the self-similarity in arrival processes on the buffer overflow probability is also analysed before the final conclusions were formulated.

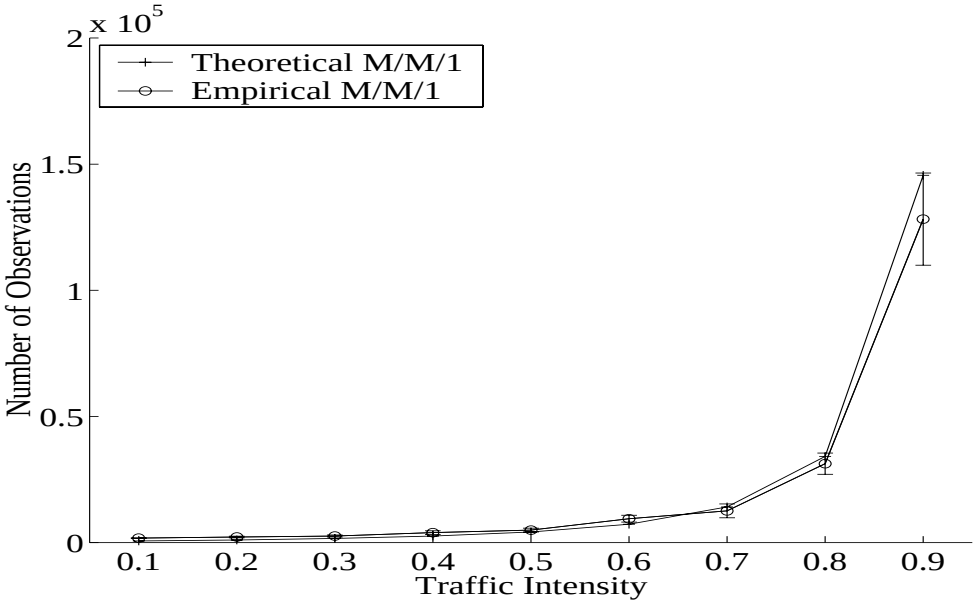
7.1 Influence of Input H Values on the Run-Length of Sequential Steady-State Simulation of $SSM/M/1/\infty$ Queueing Systems

The number of observations required to estimate this parameter can be obtained theoretically. The derivation of a formula for the theoretically required run-length for the $M/M/1/\infty$ queueing system was given by Daley [25]. For more detailed discussions of simulation run-length and defining a way of measuring the relative precision, see Appendix F.

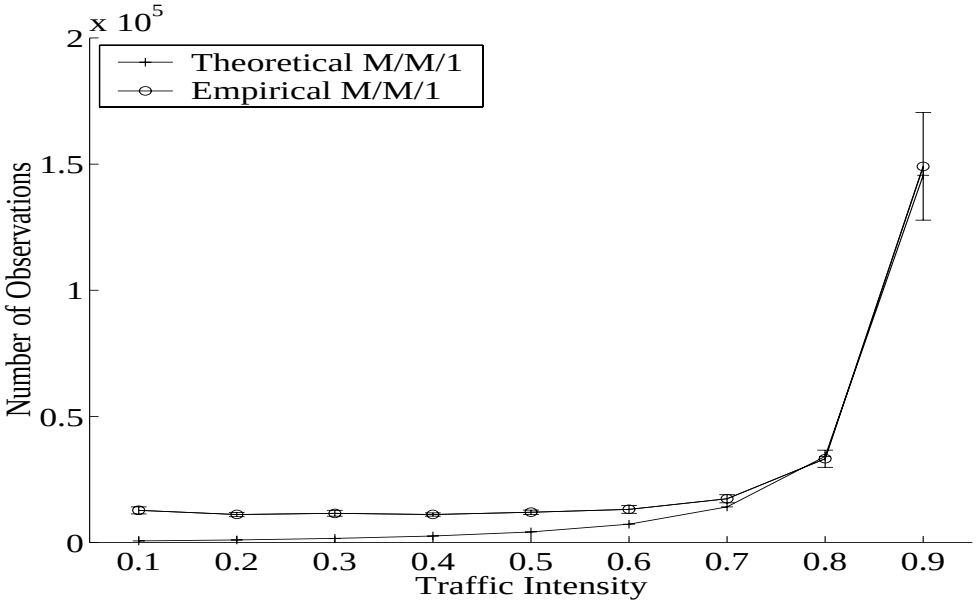
To investigate the influence of self-similarity on inter-arrivals, we need self-similar processes with an exponentially marginal distribution obtained from the SRP-FGN method [160] ($SSM/M/1/\infty$). However, the theoretically required run-length to estimate mean response times in the $SSM/M/1/\infty$ queueing system may not be known. In practice, we compare it with the simulation run-length of the same queueing models fed by Poisson processes.

We have considered a queueing system with a self-similar arrival process with an exponential marginal distribution in a steady-state simulation. We investigated how self-similar input processes influence the required length of simulation for a given relative precision of results. For this purpose we simulated $M/M/1/\infty$ and $SSM/M/1/\infty$ queueing systems in which analysis was stopped when the estimates of mean response times reached 10% (or smaller) relative error, at 95% confidence level. All simulations were executed under control of Akaroa-2 [33], see Appendix G.

Figures 7.1 (a) and (b) show the expected number of observations, and the empirical mean number of observations, necessary to achieve the required relative statistical precision for different values of traffic intensity ρ, when analysing the steady-state mean response time in an $M/M/1/\infty$ queueing



(a) SA in $M/M/1/\infty$.



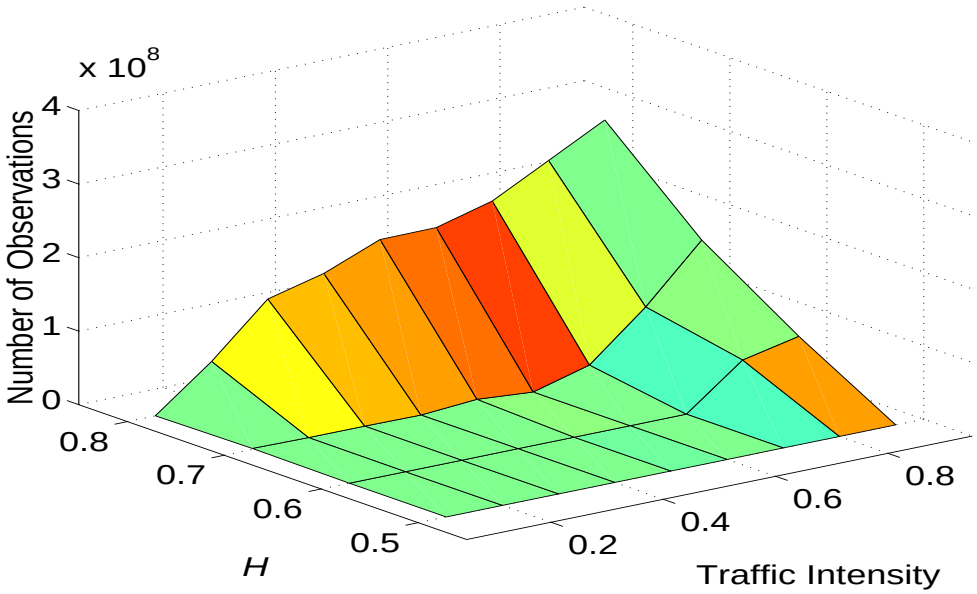
(b) BM in $M/M/1/\infty$.

Figure 7.1: Mean numbers of observations needed in the sequential analysis of a steady-state mean response time in an $M/M/1/\infty$ queueing system with relative precision $\leq 10\%$, (at 95% confidence level) using (a) spectral analysis (SA), and (b) batch means (BM).

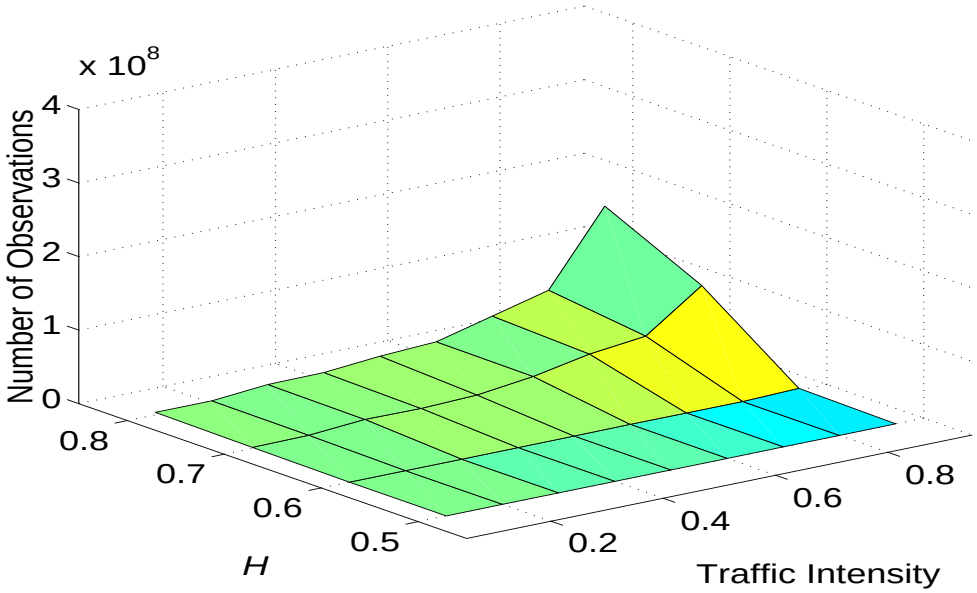
system. Each mean experimental run-length of the simulations was averaged over 30 replications. Confidence intervals of relative half-widths of 10% or less, for 95% confidence level in the methods based on spectral analysis (SA) and batch means (BM)[2] when one simulation engine is used, are also shown. For $\rho > 0.7$, the empirical mean number of observations in the method based on SA was slightly lower than the number of theoretically required observations, while for $\rho \leq 0.8$, the empirical mean number of observations in the method based on BM was slightly higher than the number of theoretically required observations. The range of the empirical mean numbers of observations for SA and BM was between 11,099 and 149,091, and between 1,787 and 128,211, respectively. The results show that the means for both methods gradually increased as ρ increased. There was no significant difference in the theoretical and empirical means for both methods.

In contrast, the results in Figures 7.2 (a) and (b) show that analysis of an $SSM/M/1/\infty$ queueing system with self-similar input processes requires many more observations than an $M/M/1/\infty$ queueing system. The mean number of observations, or simulation run-length, significantly increased when $H > 0.5$. For $\rho = 0.4$; and $H = 0.6$, 0.7 and 0.8, the empirical mean numbers of observations for the $SSM/M/1/\infty$ queueing system were 34,371 (769.1%), 43,544 (1,001.0%), and 146,801,520 (3,711,695.7%) in the SA method and 124,523 (1,020.0%), 7,503,269 (67,387.6%), and 10,546,564 (94,760.3%) in the BM method, respectively. In contrast, the $M/M/1/\infty$ queueing system required only 3,955 observations in the SA method and 11,118 observations in the BM method, and theoretically required 2,603 observations. $(\cdot\ \%)$ are the relative increases of the run-length for the $SSM/M/1/\infty$ against the $M/M/1/\infty$ queueing system. In addition, for $\rho = 0.8$; and $H = 0.6$, 0.7 and 0.8, the empirical mean numbers of observations for the $SSM/M/1/\infty$ queueing system were 58,197,030 (250,166.7%), 84,540,706 (363,453.4%), and 237,796,806 (1,022,506.0%) in the SA method, and 358,994 (980.8%), 43,363,298 (130,453.4%), and 59,902,358 (180,247.3%) in the BM method, respectively. For $\rho = 0.8$, the $M/M/1/\infty$ queueing system required 23,254 observations in the SA method

[2]So far no methods of simulation output data analysis for self-similar processes have been proposed. However, we believe the SA and BM methods can be used for these processes as long as the analysed phenomena have the first two finite moments. The sequential versions of SA and BM (in its version proposed by Heidelberger and Welch [56]) have been implemented in Akaroa-2 [33], [88], [122].



(a) SA in $SSM/M/1/\infty$.



(b) BM in $SSM/M/1/\infty$.

Figure 7.2: Mean numbers of observations required in the sequential analysis of a steady-state mean response time in an $SSM/M/1/\infty$ queueing system with relative precision $\leq 10\%$, (at 95% confidence level) in the method based on (a) spectral analysis (SA), and (b) batch means (BM).

and 33,215 observations in the BM method, and theoretically required 34,190 observations.

Our results show that the length of simulations depends both on the analysed model and on the method of analysis of output data. We found that the length of simulations with data analysed under SA can be much longer than when BM is used. This becomes more evident in simulations with larger H value assumed in arrival processes. The phenomenon needs further investigations.

The results demonstrate that the performance of queueing systems under Poisson arrival processes and self-similar arrival processes are significantly different for both low and high traffic intensity [75]. Since self-similar properties of packet count processes as well as self-similarity of inter-event processes are well documented [5], [89], [126], there is the need of running simulations of computer networks under assumptions that simulation streams of data represent self-similar processes.

7.2 Influence of Input H Values on Buffer Overflow in $SSM/M/1/B$ Queueing Systems

Beginning with the empirical finding of self-similarity in Ethernet LAN traffic data reported in Leland et al. [89], there has been mounting evidence of the practical importance of the Hurst parameter H for traffic engineering purposes. Particularly, Norros [114] demonstrated a significant difference in the queueing performance between traditional models of teletraffic such as Poisson processes and Markovian processes, and those exhibiting self-similar behaviour. More specifically, while tails of queue length distributions in traditional models of teletraffic decrease exponentially, those of self-similar traffic models decrease much slower.

In practice, if self-similarity is not accounted for at the modelling stage, serious overestimations of performance predictions may result, and thus we establish QoS requirements that are impossible to guarantee in a realistic network scenario [126], [173]. Thus, we investigate the potential impacts of traffic characteristics, including the effects of self-similar behaviour on queueing and

network performance, protocol analysis, and network congestion controls [173].

Norros [114] analysed a queueing system fed by fractional Brownian motion (FBM) as a self-similar input process. The queueing system was modelled as a storage with an arrival process $\mathbf{A}(t)$ and a constant leak rate C (unit per second). Then the contents of the storage at instant t is given by

$$\mathbf{V}(t) = \sup_{s \leq t}[\mathbf{A}(t) - \mathbf{A}(s) - (t-s)C], \;\; t \in (-\infty, \infty) \tag{7.1}$$

with $\mathbf{A}(t)$ is assumed to be an FBM process. It was found that the buffer overflow probabilities have a heavy-tailed distribution. The following approximation of probability that the storage contents exceed B unit was proposed:

$$Pr(\mathbf{V}(t) > B) \sim \exp\left(-\frac{1}{2a\lambda(1-H)^2}\left(\frac{(1-\lambda)(1-H)}{H}\right)^{2H} B^{2(1-H)}\right). \tag{7.2}$$

This result was obtained from the analysis of the storage with infinite capacity.

We considered a queueing system with a buffer of a finite size B. To show how self-similarity can effect probability of buffer overflow, we simulated $SSM/M/1/B$ and $M/M/1/B$ queueing systems and compared the steady-state probabilities of overflow in both systems under the same traffic load. The theoretical results for steady-state characteristics in $M/M/1/B$ were also used.

We considered the overflow probability of an $SSM/M/1/B$ queueing system with the finite buffer sizes, $B = 5$, 10, 15, 20, 25 and 30. Figure 7.3 shows the theoretical buffer overflow probability in an $M/M/1/B$ queueing system for buffer size $B = 5$, 10, 15, 20, 25 and 30, as traffic intensity increases. These probabilities gradually become larger as traffic intensity increases and buffer size decreases.

For traffic intensity $\rho = 0.6$ and 0.8 and $H = 0.6, 0.7$ and 0.8, simulation results of the overflow probability are shown in Figures 7.4 and 7.5, respectively, which plot *log_{10}(Overflow Probability)* against *Buffer Size*. Each experimental mean overflow probability was obtained over 30 replications. Confidence intervals of half-widths of 10% or less, for 95% confidence level, are also shown. The numerical results show that the buffer overflow probabilities of $SSM/M/1/B$ queueing systems were much higher, as H values increased, than the $M/M/1/B$ queueing systems, because self-similar traffic resulted in a

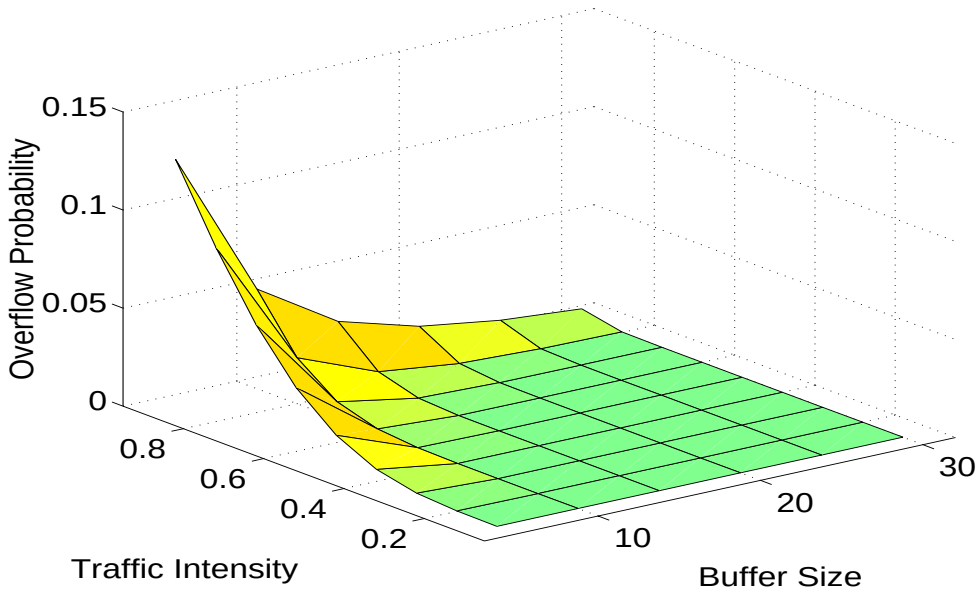


Figure 7.3: Theoretical buffer overflow probability in an $M/M/1/B$ queueing system for buffer size $B = 5$, 10, 15, 20, 25 and 30.

hyperbolic decrease in buffer overflow probabilities rather than an exponential decrease. This queueing behaviour of self-similar traffic results in longer delays through the network and greater packet losses due to buffer overflows in real computer networks.

7.3 Conclusions

We have examined queueing behaviour in steady-state stochastic simulations of $SSM/M/1/\infty$ with self-similar inter-event input and $M/M/1/\infty$ queueing models.

Firstly, we estimated the number of observations required in the sequential steady-state simulation of a queueing system with self-similar input. As we have shown, assuming self-similar inter-event processes (i.e., $SSM/M/1/\infty$ queueing systems), many more observations are required to obtain the final simulation results with a required precision, as H increases, than when assuming Poisson models, exhibiting SRD (i.e., $M/M/1/\infty$ queueing systems).

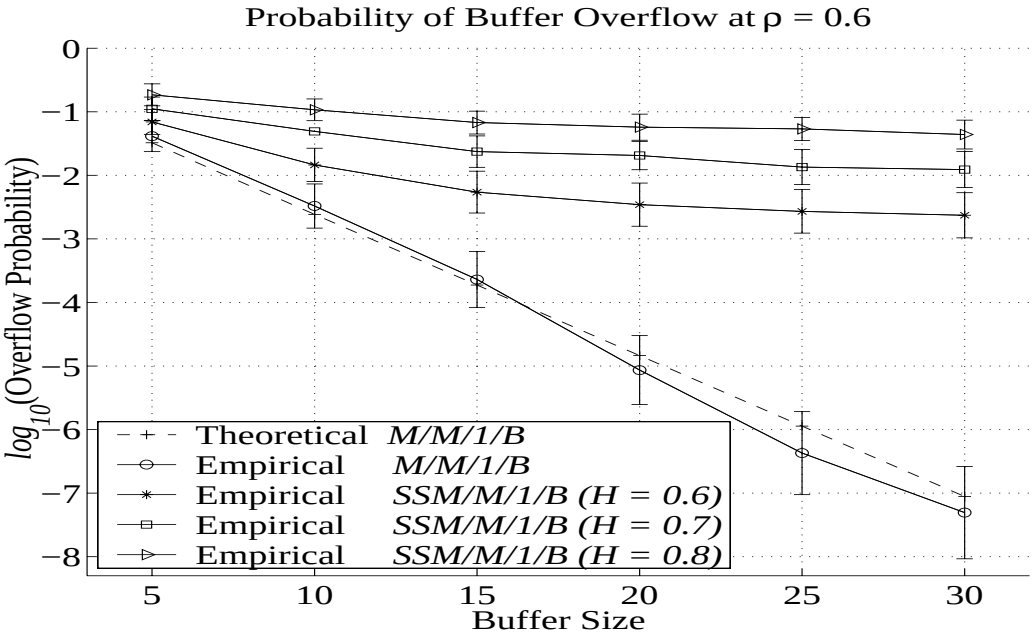


Figure 7.4: Buffer overflow probabilities of $M/M/1/B$ and $SSM/M/1/B$ buffers for traffic intensity $\rho = 0.6$, $H = 0.6$, 0.7 and 0.8.

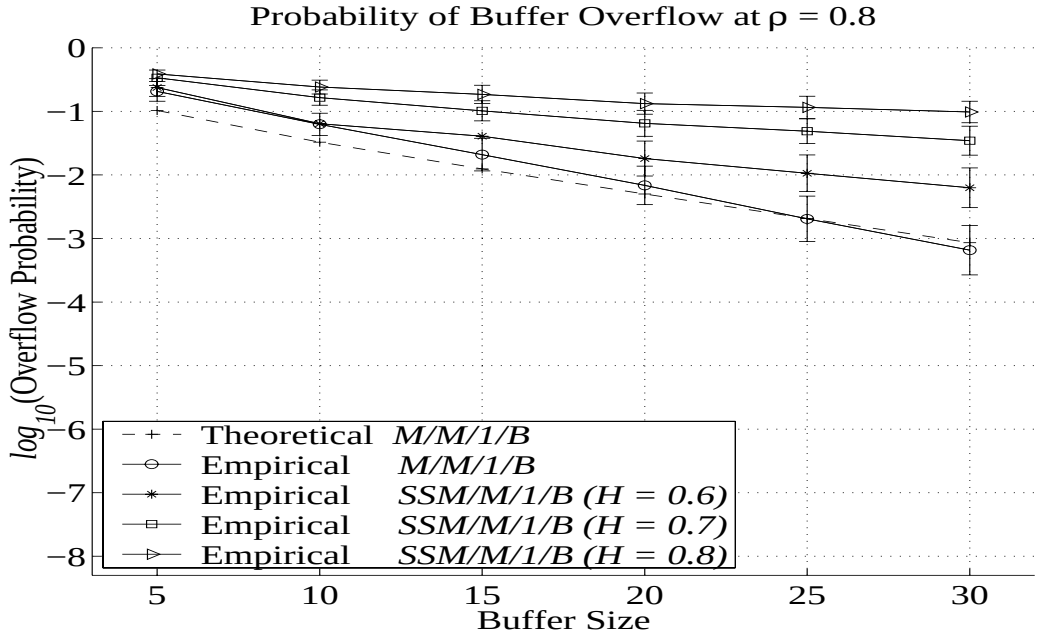


Figure 7.5: Buffer overflow probabilities of $M/M/1/B$ and $SSM/M/1/B$ buffers for traffic intensity $\rho = 0.8$, $H = 0.6$, 0.7 and 0.8.

We also found that the SA method required more observations to reach the required precision than the BM method, because the SA method produced better analysis of coverage. To secure a predefined statistical precision of final simulation results, one must generate arbitrary long sequences of inter-event times, and this must be done sequentially.

Secondly, we investigated the extent to which self-similarity affects the performance of queueing processes in QoS requirements such as buffer overflow probability. The buffer overflow probability of an $SSM/M/1/B$ queueing system was greater than the equivalent queueing system with Poisson or SRD input, and it increased as the Hurst parameter approached one. Therefore, inclusion of self-similarity in arrival processes when studying the performance of queueing systems seems to be mandatory, at least in the cases we considered.

Chapter 8

SUMMARY

8.1 Summary and Conclusions

One of the current research issues of modern computer networks is the need of full understanding of the self-similar nature of teletraffic and its incorporation in performance models of networks. Many recent, high-quality measurement studies of real teletraffic data have shown that several types of teletraffic exhibit self-similarity over a broad range of time scales.

The main contributions of this dissertation have been made in the following areas: estimation techniques of the Hurst parameter, generation of self-similar processes with normal marginal distribution and their transformation to processes with arbitrary marginal distributions, and in steady-state simulation studies of self-similar queueing processes.

- *Critical review of Hurst parameter estimation techniques:*

  We evaluated the most commonly used methods to estimate the self-similarity parameter H using appropriate numbers of long sequences. The estimators considered include the wavelet-based H estimator and Whittle's Maximum Likelihood Estimator (MLE), and estimators obtained from periodogram analysis, R/S-statistic analysis, variance-time analysis and IDC(t) analysis. Their properties were assessed on the basis of mean estimates and other statistical tests to statistically prove which of the estimators should be recommended in practice. The most efficient estimators were the wavelet-based H estimator and Whittle's MLE.

- *Algorithmic generators of self-similar teletraffic:*

  We proposed two new fixed-length pseudo-random generators of self-similar teletraffic (SRA and FGN-DW generators). A comparative analysis of practical pseudo-random generators of self-similar teletraffic was studied. Both sequential and fixed-length generators were considered.

  One of the problems that computer network researchers face is the generation of long, synthetic sequential and fixed-length self-similar sequences in order to conduct simulation studies of computer networks. Two aspects must be considered: (i) how accurately self-similar processes can be generated, and (ii) how quickly the methods generate long self-similar sequences.

  The results of a comparative analysis of six sequential generators of (long) pseudo-random self-similar sequences were presented. All six sequential generators, based on the FBNDP, MGIP, PMPP, SRP-FGN, SAP and SFRP methods, generated approximately self-similar sequences, with SRP-FGN the most accurate. Our results showed, however, that the MGIP and PMPP-based generators are strongly biased.

  The analysis of mean times required to generate sequences of a given length also showed that sequential generators are more attractive for practical simulation studies of computer networks because they are much faster than the F-ARIMA-based generator. However, they require more input parameters than methods that generate fixed-length self-similar sequences controlled by the Hurst parameter alone. Hence, for all sequential generators the problem of selecting appropriate input parameter values remains, and in the case of SAP, of defining the relationship between the Hurst parameter and the two shape parameters of a beta-distribution.

  We also presented the results of a comparative analysis of five fixed-length generators of self-similar sequences. All five generated approximately self-similar sequences, with the relative inaccuracy of the resultant $\hat{H}$ below 9% if $0.6 \leq H \leq 0.9$. Again, the analysis of mean times required to generate sequences of a given length showed that all generators except F-ARIMA are attractive for practical simulation studies of computer networks because they are much faster. When the wavelet-based H estimator and Whittle's MLE (the least biased of the H estimation

techniques) are applied, FFT and FGN-DW produce the most accurate results. Therefore, overall, FFT and FGN-DW are the most practical for simulation studies with self-similar input in both accuracy and speed.

Our results indicated that the fastest and most accurate generators of the six sequential and five fixed-length sequence generators considered were the SRP-FGN, FFT and FGN-DW methods, when evaluated by the wavelet-based H estimator and Whittle's MLE. These three methods each have a strength and a weakness. The FFT and FGN-DW methods are more attractive for non-sequential simulations because they can generate the required number of sequences more accurately and quickly than the SRP-FGN method. If the FFT and FGN-DW methods are used for sequential simulations, sufficient numbers of sequences must be generated before the simulation begins; however, the required number of sequences for sequential simulations is not easy to predict in practical simulations. In contrast, the SRP-FGN method is more attractive for sequential simulations because one does not need to know the required number of sequences beforehand. This method, however, is less accurate and requires more generating time than the FGN-DW method.

Thus, the FGN-DW method was used to synthesise VBR video traffic as described in Chapter 6, and the SRP-FGN method was used to investigate queueing behaviour in steady-state simulation studies of self-similar queueing systems, as outlined in Chapter 7.

- *Generation of a self-similar process with an arbitrary marginal distribution:*

  Before conducting simulation studies of computer networks, self-similar processes with arbitrary marginal distributions must be obtained from sequences of the exact self-similar FGN processes. In order to obtain these results, we transformed given sequences of the exact self-similar FGN process into suitable self-similar processes with arbitrary marginal distributions. We examined a method based on the ICDF transformation, which can produce self-similar processes with arbitrary marginal distributions for the stochastic simulation studies of telecommunication networks.

  We determined that the self-similarity of original self-similar processes

can be preserved by the ICDF transformation if the output process has a finite variance. Then, we tested the resulting processes with the exponential, gamma, Pareto, uniform and Weibull marginal distributions using the ICDF transformation. The relative inaccuracy of estimated mean values of self-similarity obtained using the wavelet-based H estimator and Whittle's MLE was less than 4%.

We provided evidence of ACF preservation after the ICDF transformation was applied. However, for $H = 0.6$, 0.7, 0.8 and 0.9, the ICDF transformation, if applied to self-similar processes with normal marginal distributions, preserves ACFs if the output processes with different marginal probability distributions have a finite variance. In other words, our results provided clear experimental evidence that input self-similarity and ACFs are not preserved in the output process generated by the ICDF transformation if the output process has an infinite variance.

- *Modelling and generation of self-similar VBR video traffic:*

  We investigated how compression algorithms on correlation structure of compressed teletraffic influences real video traffic. Generalisation of findings of Garrett and Willinger, showing that video compression algorithms (MPEG-1 and MPEG-2) lead to self-similar processes was studied. We considered outcomes of MPEG-1 and MPEG-2 in addition to previously studied outcomes of JPEG (Garrett and Willinger), to show that the results of (Garrett and Willinger) do not depend on the compression algorithms.

  We used a gamma/Pareto model to show that synthetically generated streams of VBR video, compressed under such standards as JPEG, MPEG-1 and MPEG-2, are statistically equivalent to real video traces. We also presented that results of a steady-state simulation of a single buffer fed by these synthetic video streams are used to show that simulations based on synthetic streams of teletraffic can provide the same qualitative and quantitative results as simulations based on real traces of VBR video.

- *Simulation studies of queueing processes with self-similar arrival processes:*

  We analysed run-lengths of sequential steady-state simulations of queueing models with self-similar arrivals, under two different methods of simulation output data analysis [spectral analysis (SA) and batch means (BM)]. Steady-state simulations of queueing processes with self-similar input processes were conducted to investigate influence of the degree of self-similarity on run-lengths of sequential simulation and on the overflow probability of finite buffers.

  All simulation studies were conducted under control Akaroa-2. In particular, it was found that the methods of output data analysis that in Akaroa-2 were originally intended for non-self-similar processes were applicable to analysis of self-similar processes as well, as long as the analysed phenomena has the first two finite moments.

  We examined the queueing behaviour in steady-state stochastic simulations of $SSM/M/1/\infty$ with self-similar arrivals and compared it with $M/M/1/\infty$ queueing models. We obtained mean numbers of observations required in the sequential analysis of steady-state mean response times in both an $M/M/1/\infty$ and an $SSM/M/1/\infty$ queueing systems with a relative precision $\leq 10\%$, at 95% confidence level in the SA and BM methods.

  We estimated the number of observations required in the sequential steady-state simulation of a queueing system with self-similar input. We have shown that for different analytical methods, as we have shown, assuming self-similar processes with an exponential marginal distribution (i.e., $SSM/M/1/\infty$ queueing systems), many additional observations are required to obtain final simulation results with a required precision, as H increases, than when assuming Poisson models exhibiting SRD (i.e., $M/M/1/\infty$ queueing systems).

  We also investigated the degree to which self-similarity affects the performance of queueing processes, in regard to QoS requirements such as buffer overflow probability. The buffer overflow probability of an $SSM/M/1/B$ queueing system is greater than the equivalent queueing system with a Poisson or SRD process (i.e., $M/M/1/B$ queueing system) input, and this overflow probability increases as the Hurst pa-

rameter approaches one. Therefore, inclusion of self-similarity in arrival processes when studying the performance of queueing systems seems to be mandatory, at least in the cases we considered.

8.2 Future Work

The results of this research can be extended by the following work:

- Designing more computationally efficient sequential self-similar generator able to construct arbitrary long sequences. Such sequences are necessary in simulation studies of telecommunication networks.
- Developing diverse layered (for example, GOP and cell) modelling for multimedia teletraffic of specific scenarios. Most important measures of queueing performance can be obtained from single-streams, but to obtain more reliable simulation results, the queueing performance for multiplexed streams must also be investigated.
- Studies of multifractal processes that are related to the generalisation of self-similar processes and provide more flexible scaling properties, which seem necessary to capture local irregularities in computer network traffic [36], [111], [168].

References

[1] Abry, P., Flandrin, P., Taqqu, M., and D.Veitch. *Long-Range Dependence: Theory and Applications*. Doukhan, Oppenheim, and Taqqu (eds), 2000, ch. Self-Similarity and Long-Range Dependence Through the Wavelet Lens.

[2] Abry, P., and Veitch, D. Wavelet Analysis of Long-Range-Dependent Traffic. *IEEE Transactions on Information Theory 44*, 1 (1998), pp. 2–15. http://www.emulab.ee.mu.oz.au/∼darryl/.

[3] Addie, R., Neame, T., and Zukerman, M. Modeling Superposition of Many Sources Generating Self-Similar Traffic. In *Proceedings of IEEE International Conference on Communications (ICC'99)* (Vancouver, British Columbia, Canada, 1999), pp. 387–391.

[4] Addie, R., Zukerman, M., and Neame, T. Dimensioning Networks for Fractal Traffic. In *Australian Telecommunication Networks and Applications Conference* (Melbourne, 1994), pp. 341–346.

[5] Addie, R., Zukerman, M., and Neame, T. Fractal Traffic: Measurements, Modelling and Performance Evaluation. In *Proceedings of IEEE INFOCOM'95* (Boston, Massachusetts, 1995), pp. 977–984.

[6] Addie, R., Zukerman, M., and Neame, T. Performance of a Single Server Queue with Self-Similar Input. In *Proceedings of IEEE International Conference on Communications (ICC'95)* (Seattle, WA, 1995), pp. 461–465.

[7] Andersen, A., and Nielsen, B. An Application of Superpositions of Two-State Markovian Sources to the Modelling of Self-Similar Behaviour. In *Proceedings of IEEE INFOCOM'97* (Kobe, Japan, 1997), pp. 196–204.

[8] Beran, J. Statistical Methods for Data with Long Range Dependence. *Statistical Science 7*, 4 (1992), pp. 404–427.

[9] Beran, J. *Statistics for Long-Memory Processes*. Chapman and Hall, New York, 1994.

[10] Beran, J., Sherman, R., Taqqu, M., and Willinger, W. Long-Range Dependence in Variable-Bit-Rate Video Traffic. *IEEE Transactions on Communications 43*, 2 (1995), pp. 1566–1579.

[11] Beran, J., and Terrin, N. Estimation of the Long-Memory Parameter, Based on a Multivariate Central Limit Theorem. *Time Series Analysis 15*, 3 (1994), pp. 269–278.

[12] Boxma, O., and Cohen, J. *Self-Similar Network Traffic and Performance Evaluation*. John Wiley & Sons, Inc., K. Park and W. Willinger (eds), New York, 2000, ch. The Single Server Queue: Heavy Tails and Heavy Traffic, pp. 143–169.

[13] Brockwell, P., and Davis, R. *Time Series: Theory and Methods, Second Edition*. Springer-Verlag, New York, 1991.

[14] Cario, M., and Nelson, B. Autoregressive to Anything: Time-Series Input Processes for Simulation. *Operations Research Letters 19* (1996), pp. 51–58.

[15] Cario, M., and Nelson, B. Numerical Methods for Fitting and Simulating Autoregressive-to-Anything Processes. *INFORMS Journal on Computing 10*, 1 (1998), pp. 72–81.

[16] Costamagna, E., Favalli, L., Gamba, P., and Iacovoni, G. A Simple Model for VBR Video Traffic Based on Chaotic Maps: Validation through Evaluation of ATM Multiplexers QoS Parameters. In *Proceedings of IEEE International Conference on Communications (ICC'98)* (Atlanta, GA, USA, 1998), pp. S16_6.1–S16_6.5.

[17] Cox, D. Long-Range Dependence: a Review. In *Statistics: An Appraisal* (1984), Iowa State Statistical Library, The Iowa State University Press, H.A. David and H.T. David (eds.), pp. 55–74.

[18] Cox, D., and Isham, V. *Point Processes*. Chapman and Hall, New York, 1980.

[19] Cox, D., and Lewis, P. *The Statistical Analysis of Series of Events*. Spottiswoode, Ballantyne & Co. Ltd., UK, 1966.

[20] Crilly, A., Earnshaw, R., and Jones, H. *Fractals and Chaos*. Springer-Verlag, New York, 1991.

[21] Crovella, M., and Bestavros, A. Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes. *IEEE ACM Transactions on Networking 5*, 6 (1997), pp. 835–846.

[22] Crovella, M., Lindemann, C., and Reiser, M. Internet Performance Modeling: The State of the Art at the Turn of the Century. Dagstuhl Castle Workshop, Germany, 1999.

[23] Crovella, M., and Lipsky, L. Long-Lasting Transient Conditions in Simulations with Heavy-Tailed Workloads. In *Proceedings of the 1997 Winter Simulation Conference* (Atlanta, Georgia, 1997), S. Andradottir, K.J. Healy, D.H. Withers, and B.L. Nelson (eds.), pp. 1005–1012.

[24] Dahl, T., and Willemain, T. The Effect of Long-Memory Arrivals on Queue Performance. *Operations Research Letters 29* (2001), pp. 123–127.

[25] Daley, D. J. The Serial Correlation Coefficients of Waiting Times in a Stationary Single Server Queue. *Journal of the Aust. Math. Soc. 8* (1968), pp. 683–699.

[26] Daubechies, I. *Ten Lectures on Wavelets*, vol. 61 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM Press, Philadelphia, Pennsylvania, 1992.

[27] Duffield, N. Queueing at Large Resources Driven by Long-Tailed $M/G/\infty$-Modulated Processes. *Queueing Systems 28* (1998), pp. 245–266.

[28] Erramilli, A., Gordon, J., and Willinger, W. Applications of Fractals in Engineering for Realistic Processes. In *The Fundamental Role of Teletraffic in the Evolution of Telecommunications Networks: Proceedings of the 14th International Teletraffic Congress-ITC 14* (Antibes Juan-les-Pins, France, 1994), vol. 1a, Elsevier, Amsterdam, J. Labetoulle and J.W. Roberts (eds), pp. 35–44.

[29] Erramilli, A., Narayan, O., and Willinger, W. Experimental Queueing Analysis with Long-Range Dependent Packet Traffic. *IEEE ACM Transactions on Networking 4*, 2 (1996), pp. 209–223.

[30] Erramilli, A., Pruthi, P., and Willinger, W. Fast and Physically-Based Generation of Self-Similar Network Traffic with Applications to ATM Performance Evaluation. In *Proceedings of the 1997 Winter Simulation Conference* (Atlanta, Georgia, 1997), S. Andradottir, K.J. Healy, D.H. Withers, and B.L. Nelson (eds.), pp. 997–1004.

[31] Erramilli, A., Singh, R., and Pruthi, P. Chaotic Maps As Models of Packet Traffic. In *The Fundamental Role of Teletraffic in the Evolution of Telecommunications Networks: Proceedings of the 14th International Teletraffic Congress-ITC 14* (Antibes Juan-les-Pins, France, 1994), vol. 1a, Elsevier, Amsterdam, J. Labetoulle and J.W. Roberts (eds), pp. 329–338.

[32] Ewing, G., Pawlikowski, K., and McNickle, D. Akaroa 2.5: User's Manual. Tech. Rep. TR-COSC 07/98, Department of Computer Science, University of Canterbury, Christchurch, New Zealand, 1998.

[33] Ewing, G., Pawlikowski, K., and McNickle, D. AKAROA-2: Exploiting Network Computing by Distributing Stochastic Simulation. In *Proceedings of 13th European Simulation Multiconference, ESM'99* (Warsaw, Poland, 1999), vol. 1, pp. 175–181.

[34] Fahmy, S., Jain, R., Rabie, S., Goyal, R., and Vandalore, B. Quality of Service for Internet Traffic over ATM Service Categories. *Computer Communications 22*, 14 (1999), pp. 1307–1320.

[35] Fan, Y., and Georganas, N. On Merging and Splitting of Self-Similar Traffic in High-Speed Networks. In *Proceedings of ICCC'95* (Seoul, Korea, 1995).

[36] Feldmann, A., Gilbert, A., and Willinger, W. Data Networks as Cascades: Investigating the Multifractal Nature of Internet WAN Traffic. In *Computer Communication Review, Proceedings of ACM SIGCOMM'98* (Vancouver, Canada, 1998), vol. 28 (4), pp. 42–55.

[37] Feller, W. *An Introduction to Probability Theory and Its Applications.* 2nd ed., John Wiley & Sons, Inc., New York, 1966. Volume II.

[38] Feller, W. *An Introduction to Probability Theory and Its Applications.* 3rd ed., John Wiley & Sons, Inc., New York, 1968. Volume I.

[39] Fishman, G. Grouping Observations in Digital Simulation. *Management Science 24*, 5 (1978), pp. 510–521.

[40] Flandrin, P. Wavelet Analysis and Synthesis of Fractional Brownian Motion. *IEEE Transactions on Information Theory 38*, 2 (1992), pp. 910–917.

[41] Frost, V., and Melamed, B. Traffic Modeling For Telecommunications Networks. *IEEE Communications Magazine 32*, 3 (1994), pp. 70–81.

[42] Garrett, M., and Willinger, W. Analysis, Modeling and Generation of Self-Similar VBR Video Traffic. In *Computer Communication Review, Proceedings of ACM SIGCOMM'94* (London, UK, 1994), vol. 24 (4), pp. 269–280.

[43] Geist, R., and Westall, J. Practical Aspects of Simulating Systems Having Arrival Processes with Long-Range Dependence. In *Proceedings of the 2000 Winter Simulation Conference* (Orlando, Florida, USA, 2000), J.A. Joines, R.R. Barton, K. Kang, and P.A. Fishwick (eds.), pp. 666–674.

[44] Geist, R., and Westall, J. Correlational and Distributional Effects in Network Traffic Models. *Performance Evaluation 44* (2001), pp. 121–138.

[45] Georganas, N. Self-Similar ("Fractal") Traffic in ATM Networks. In *Proceedings of the 2nd International Workshop on Advanced Teleservices and High-Speed Communication Architectures (IWACA'94)* (Heidelberg, Germany, 1994), pp. 1–7. Invited Keynote Speaker.

[46] González Sánchez, S., González Prelcic, N., and García Galán, S. *Uvi_Wave. Wavelets Toolbox for use with Matlab.* Departamento de Tecnoloxías das Comunicacións. Universidade de Vigo, Vigo, 1996. `ftp://ftp.tsc.uvigo.es/pub /Uvi_Wave.doc.ps.gz`.

[47] Goudey, C. *A Handbook of Ferns for Australia and New Zealand.* Lothian Publishing Company Pty Ltd, Melbourne, Australia, 1988.

[48] Granger, C. Long Memory Relationships and the Aggregation of Dynamic Models. *Journal of Econometrics 14* (1980), pp. 227–238.

[49] Greiner, M., Jobmann, M., and Klüppelberg, C. Telecommunication Traffic, Queueing Models, and Subexponential Distributions. *Queueing Systems 33* (1999), pp. 125–152.

[50] Greiner, M., Jobmann, M., and Lipsky, L. The Importance of Power-Tail Distributions for Modeling Queueing Systems. *Operations Research 47*, 2 (1999), pp. 313–326.

[51] Gribble, S., Manku, G., Roselli, D., Brewer, E., Gibson, T., and Miller, E. Self-Similarity in File Systems. In *Performance Evaluation Review, Proceedings of ACM SIGMETRICS'98* (Madison, Wisconsin, 1998), pp. 141–150.

[52] Gross, D., and Harris, C. M. *Fundamentals of Queueing Theory.* 2nd ed., John Wiley and Sons, Inc., New York, 1985.

[53] Grossglauser, M., and Bolot, J.-C. On the Relevance of Long-Range Dependence in Network Traffic. *IEEE ACM Transactions on Networking 7*, 5 (1999), pp. 629–640.

[54] Gusella, R. Characterizing the Variability of Arrival Processes with Indexes of Dispersion. *IEEE Journal on Selected Areas in Communications 9* (1991), pp. 203–211.

[55] Hara, S., and Taketsugu, J. Another Cause of Long-Range Time Dependence in Cellular Network Traffic. Kluwer Academic Publisher, Netherlands, 2002.

[56] Heidelberger, P., and Welch, P. A Spectral Method for Confidence Interval Generation and Run Length Control in Simulations. *Communications of the ACM 24*, 4 (1981), pp. 233–245.

[57] Heyman, D. Some Issues in Performance Modeling of Data Teletraffic. *Performance Evaluation 34* (1998), pp. 227–247.

[58] Heyman, D. Performance Implications of Very Large Service-Time Variances. *Performance Evaluation 40* (2000), pp. 47–70.

[59] Heyman, D., and Lakshman, T. *Self-Similar Network Traffic and Performance Evaluation*. John Wiley & Sons, Inc., K. Park and W. Willinger (eds), New York, 2000, ch. Long-Range Dependence and Queueing Effects for VBR Video, pp. 285–318.

[60] Heyman, D., Tabatabai, A., and Lakshman, T. Statistical Analysis and Simulation Study of Video Teleconference Traffic in ATM. *IEEE Transactions on Circuits and Systems for Video Technology 2*, 1 (1992), pp. 49–59.

[61] Higuchi, T. Approach to an Irregular Time Series on the Basis of the Fractal Theory. *Physica D 31*, 2 (1988), pp. 277–283.

[62] Hosking, J. Fractional Differencing. *Biometrika 68*, 1 (1981), pp. 165–176.

[63] Hosking, J. Modeling Persistence in Hydrological Time Series Using Fractional Differencing. *Water Resources Research 20*, 12 (1984), pp. 1898–1908.

[64] Huang, C., Devetsikiotis, M., Lambadaris, I., and Kaye, A. Modeling and Simulation of Self-Similar Variable Bit Rate Compressed Video: A Unified Approach. *Computer Communication Review, Proceedings of ACM SIGCOMM'95 25*, 4 (1995), pp. 114–125.

[65] Jain, R. *The Art of Computer Systems Performance Analysis*. John Wiley & Sons, Inc., New York, 1991.

[66] Jain, R. Current Issues in Telecom Networks: QoS, Traffic Engineering and DWDM. Keynote speaker, http://www.cis.ohio-state.edu/~jain/talks/icon99.htm, 2000.

[67] Jelenković, P. *The Effect of Multiple Time Scales and Subexponentiality on the Behavior of a Broadband Network Multiplexer*. PhD thesis, Graduate School of Arts and Sciences, Columbia University, 1996.

[68] Jeong, H.-D., McNickle, D., and Pawlikowski, K. A Comparative Study of Generators of Synthetic Self-Similar Teletraffic. Tech. Rep. TR-COSC 03/98, Department of Computer Science, University of Canterbury, Christchurch, New Zealand, 1998.

[69] Jeong, H.-D., McNickle, D., and Pawlikowski, K. A Generator of Pseudo-random Self-Similar Sequences Based on SRA. Tech. Rep. TR-COSC 9/98, Department of Computer Science, University of Canterbury, Christchurch, New Zealand, 1998.

[70] Jeong, H.-D., McNickle, D., and Pawlikowski, K. A Comparative Study of Three Self-Similar Teletraffic Generators. In *Proceedings of 13th European Simulation Multiconference, ESM'99* (Warsaw, Poland, 1999), vol. 1, pp. 356–362.

[71] Jeong, H.-D., McNickle, D., and Pawlikowski, K. A Search for Computationally Efficient Generators of Synthetic Self-Similar Teletraffic. In *Proceedings of the Twenty Second Australasian Computer Science Conference* (Auckland, New Zealand, 1999), vol. 21, pp. 75–86.

[72] Jeong, H.-D., McNickle, D., and Pawlikowski, K. Fast Self-Similar Teletraffic Generation Based on FGN and Inverse DWT. Tech. Rep. TR-COSC 03/99, Department of Computer Science, University of Canterbury, Christchurch, New Zealand, 1999.

[73] Jeong, H.-D., McNickle, D., and Pawlikowski, K. Fast Self-Similar Teletraffic Generation Based on FGN and Wavelets. In *Proceedings of the IEEE International Conference on Networks, ICON'99* (Brisbane, Australia, 1999), pp. 75–82.

[74] Jeong, H.-D., McNickle, D., and Pawlikowski, K. Generation of Self-Similar Time Series for Simulation Studies of Telecommunication Networks. In *Proceedings of the First Western Pacific and Third Australia-Japan Workshop on Stochastic Models in Engineering, Technology and Management* (Christchurch, New Zealand, 1999), pp. 221–230.

[75] Jeong, H.-D., McNickle, D., and Pawlikowski, K. Some Problems in Sequential Simulation with Self-Similar Processes. In *Proceedings of the 2000 Summer Computer Simulation Conference* (Vancouver, British Columbia, Canada, 2000), pp. 175–180.

[76] Jeong, H.-D., McNickle, D., and Pawlikowski, K. Generation of Self-Similar Processes for Simulation Studies of Telecommunication Networks. *Mathematical and Computer Modelling* (2001). Accepted.

[77] Ji, C., Ma, S., and Tian, X. Approximation Capability of Independent Wavelet Models to Heterogeneous Network Traffic. In *Proceedings of IEEE INFOCOM'99* (New York, NY, USA, 1999), pp. 170–177.

[78] Kaplan, L., and Kuo, C.-C. Fractal Estimation from Noisy Data via Discrete Fractional Gaussian Noise (DFGN) and the Haar Basis. *IEEE Transactions on Signal Processing 41*, 12 (1993), pp. 3554–3562.

[79] Kleinrock, L. *Queueing Systems, Volume I: Theory.* John Wiley & Sons, Inc., 1975.

[80] Klüppelberg, C. Subexponential Distributions and Integrated Tails. *Journal of Applied Probability 25* (1988), pp. 132–141.

[81] Krunz, M., and Makowski, A. A Source Model for VBR Video Traffic Based on $M/G/\infty$ Input Processes. In *Proceedings of IEEE INFOCOM'98* (San Francisco, CA, USA, 1998), pp. 1441–1448.

[82] Krunz, M., and Matta, I. Analytical Investigation of the Bias Effect in Variance-Type Estimators for Inference of Long-Range Dependence. *Computer Networks 40* (2002), pp. 445–458.

[83] Krunz, M., Sass, R., and Hughes, H. Statistical Characteristics and Multiplexing of MPEG Streams. In *Proceedings of IEEE INFOCOM'95* (Boston, Massachusetts, 1995), pp. 455–462.

[84] Lau, W.-C., Erramilli, A., Wang, J., and Willinger, W. Self-Similar Traffic Generation: the Random Midpoint Displacement Algorithm and its Properties. In *Proceedings of IEEE International Conference on Communications (ICC'95)* (Seattle, WA, 1995), pp. 466–472.

[85] Law, A., and Kelton, W. *Simulation Modeling and Analysis*. 2nd ed., McGraw-Hill, Inc., Singapore, 1991.

[86] Le-Ngoc, T., and Subramanian, S. A Pareto-Modulated Poisson Process (PMPP) Model for Long-Range Dependent Traffic. *Computer Communications 23* (2000), pp. 123–132.

[87] Ledesma, S., and Liu, D. Synthesis of Fractional Gaussian Noise Using Linear Approximation for Generating Self-Similar Network Traffic. *Computer Communication Review, ACM SIGCOMM 30*, 2 (2000), pp. 4–17.

[88] Lee, J.-S. *On Automated Sequential Steady-State Simulation*. PhD thesis, Department of Computer Science, University of Canterbury, 2000.

[89] Leland, W., Taqqu, M., Willinger, W., and Wilson, D. On the Self-Similar Nature of Ethernet Traffic (Extended Version). *IEEE ACM Transactions on Networking 2*, 1 (1994), pp. 1–15.

[90] Leland, W., Willinger, W., Taqqu, M., and Wilson, D. Statistical Analysis and Stochastic Modeling of Self-Similar Data Traffic. In *The Fundamental Role of Teletraffic in the Evolution of Telecommunications Networks: Proceedings of the 14th International Teletraffic Congress-ITC 14* (Antibes Juan-les-Pins, France, 1994), vol. 1a, Elsevier, Amsterdam, J. Labetoulle and J.W. Roberts (eds), pp. 319–328.

[91] Leon-Garcia, A. *Probability and Random Processes for Electrical Engineering*. Addison-Wesley Publishing Company, New York, 1989.

[92] Leroux, H., and Hassan, M. Generating Packet Inter-Arrival Times for FGN Arrival Processes. In *The 3rd New Zealand ATM and Broadband Workshop* (Hamilton, New Zealand, 1999), pp. 1–10.

[93] Leroux, H., Hassan, M., and Egudo, R. Are Inter-Arrival Times of Internet Traffic also Self-Similar? In *Proceedings of IEEE International Conference on Telecommunications (ICT'99)* (Cheju, South Korea, 1999), vol. 1, pp. 549–553.

[94] Leroux, H., Hassan, M., and Egudo, R. On the Self-Similarity of Packet Inter-Arrival Times of Internet Traffic. In *The 3rd New Zealand ATM and Broadband Workshop* (Hamilton, New Zealand, 1999), pp. 11–19.

[95] Likhanov, N., Tsybakov, B., and Georganas, N. Analysis of an ATM Buffer with Self-Similar ("Fractal") Input Traffic. In *Proceedings of IEEE INFOCOM'95* (Boston, Massachusetts, 1995), pp. 985–992.

[96] Liu, B., and Munson, D. Generation of a Random Sequence Having a Jointly Specified Marginal Distribution and Autocovariance. *IEEE Transactions on Acoustics, Speech and Signal Processing ASSP-30*, 6 (1982), pp. 973–983.

[97] Lombardo, A., Morabito, G., Palazzo, S., and Schembra, G. MPEG Traffic Generation Matching Intra- and Inter-GoP Correlation. *Simulation 74*, 2 (2000), pp. 97–109.

[98] López-Ardao, J., López-García, C., Suárez-González, A., Fernández-Veiga, M., and Rodríguez-Rubio, R. On the Use of Self-Similar Processes in Network Simulation. *ACM Transactions on Modeling and Computer Simulation 10*, 2 (2000), pp. 125–151.

[99] Lowen, S., and Teich, M. Estimation and Simulation of Fractal Stochastic Point Processes. *Fractals 3*, 1 (1995), pp. 183–210.

[100] Lu, W. Compact Multidimensional Broadband Wireless: The Convergence of Wireless Mobile and Access. *IEEE Communications Magazine 38*, 11 (2000), pp. 119–123.

[101] Ma, S., and Ji, C. Modeling Video Traffic in the Wavelet Domain. In *Proceedings of IEEE INFOCOM'98* (San Francisco, CA, USA, 1998), pp. 201–208.

[102] Ma, S., and Ji, C. Modeling Video Traffic Using Wavelets. In *Proceedings of IEEE International Conference on Communications (ICC'98)* (Atlanta, GA, USA, 1998), pp. S16_4.1–S16_4.4.

[103] Mandelbrot, B. Long-Run Linearity, Locally Gaussian Processes, H-Spectra and Infinite Variances. *International Economic Review 10* (1969), pp. 82–113.

[104] Mandelbrot, B., and Ness, J. V. Fractional Brownian Motions, Fractional Noises and Applications. *SIAM Review 10*, 4 (1968), pp. 422–437.

[105] Mandelbrot, B., and Wallis, J. Computer Experiments with Fractional Gaussian Noises. *Water Resources Research 5*, 1 (1969), pp. 228–267.

[106] McNickle, D. Private Communication. Department of Management, University of Canterbury, Christchurch, New Zealand, 2000.

[107] Melamed, B. TES: a Class of Methods for Generating Autocorrelated Uniform Variates. *ORSA Journal on Computing 3*, 4 (1991), pp. 317–329.

[108] Melamed, B., and Hill, J. R. A Survey of TES Modeling Applications. *Simulation* (1995), pp. 353–370.

[109] Microsoft Press. *Microsoft Press Computer Dictionary, Third Edition*. Microsoft Press, Redmond, Washington, 1997.

[110] Mitchell, J., Pennebaker, W., Fogg, C., and LeGall, D. *MPEG Video Compression Standard*. Chapman and Hall, New York, 1997.

[111] Molnár, S., and Maricza, I. Source Characterization in Broadband Networks. In *Proceedings of the COST257 Mid-Term Seminar* (Vilamoura, Portugal, 1999), pp. 139–260.

[112] Neame, T., Zukerman, M., and Addie, R. Application of the M/Pareto Process to Modeling Broadband Traffic Streams. In *Proceedings of the IEEE International Conference on Networks, ICON'99* (Brisbane, Australia, 1999), pp. 53–58.

[113] Neidhardt, A., and Wang, J. The Concept of Relevant Time Scales and its Application to Queueing Analysis of Self-Similar Traffic (or Is Hurst Naughty or Nice?). In *Performance Evaluation Review, Proceedings of ACM SIGMETRICS'98* (Madison, Wisconsin, USA, 1998), pp. 222–232.

[114] Norros, I. A Storage Model with Self-Similar Input. *Queueing Systems 16* (1994), pp. 387–396.

[115] Östring, S. *Reactive Traffic Control Mechanisms for Communication Networks with Self-Similar Bandwidth Demands*. PhD thesis, Department of Electrical and Electronic Engineering, University of Canterbury, 2001.

[116] Östring, S., Sirisena, H., and Hudson, I. Sensitivity of ABR Congestion Control Algorithms to Hurst Parameter Estimates. In *Proceedings of Networking 2000* (Paris, France, 2000), pp. 36–48.

[117] Park, K., Kim, G., and Crovella, M. On the Relationship Between File Sizes, Transport Protocols, and Self-Similar Network Traffic. In *Proceedings of IEEE International Conference on Network Protocols* (1996), pp. 171–180.

[118] Park, K., Kim, G., and Crovella, M. On the Effect of Traffic Self-Similarity on Network Performance. In *Proceedings of SPIE International Conference on Performance and Control of Network Systems* (1997), pp. 296–310.

[119] Park, K., and Tuan, T. Performance Evaluation of Multiple Time Scale TCP Under Self-Similar Traffic Conditions. *ACM Transactions on Modeling and Computer Simulation 10*, 2 (2000), pp. 152–177.

[120] Park, K., and Willinger, W. *Self-Similar Network Traffic and Performance Evaluation*. John Wiley & Sons, Inc., K. Park and W. Willinger (eds), New York, 2000, ch. Self-Similar Network Traffic: An Overview, pp. 1–38.

[121] Parulekar, M., and Makowski, A. $M/G/\infty$ Input Processes: a Versatile Class of Models for Network Traffic. In *Proceedings of IEEE INFOCOM'97* (Kobe, Japan, 1997), pp. 419–426.

[122] Pawlikowski, K. Steady-State Simulation of Queueing Processes: a Survey of Problems and Solutions. *ACM Computing Surveys 22*, 2 (1990), pp. 123–170.

[123] Pawlikowski, K., Jeong, H.-D., and Lee, J.-S. On Credibility of Simulation Studies of Telecommunication Networks. *IEEE Communications Magazine 40*, 1 (2002), pp. 132–139.

[124] Paxson, V. Fast, Approximate Synthesis of Fractional Gaussian Noise for Generating Self-Similar Network Traffic. *Computer Communication Review, ACM SIGCOMM 27*, 5 (1997), pp. 5–18.

[125] Paxson, V., and Floyd, S. Wide-Area Traffic: the Failure of Poisson Modeling. In *Computer Communication Review, Proceedings of ACM SIGCOMM'94* (London, UK, 1994), pp. 257–268.

[126] Paxson, V., and Floyd, S. Wide-Area Traffic: the Failure of Poisson Modeling. *IEEE ACM Transactions on Networking 3*, 3 (1995), pp. 226–244.

[127] Peitgen, H.-O., Jurgens, H., and Saupe, D. *Chaos and Fractals: New Frontiers of Science*. Springer-Verlag, New York, 1992.

[128] Peitgen, H.-O., and Saupe, D. *The Science of Fractal Images*. Springer-Verlag, New York, 1988.

[129] Peng, C.-K., Buldyrev, S., Havlin, S., Simons, M., Stanley, H., and Goldberger, A. Mosaic Organization of DNA Nucleotides. *Physical Review E 49*, 2 (1994), pp. 1685–1689.

[130] Press, W., Flannery, B., Teukolsky, S., and Vetterling, W. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, Cambridge, 1986.

[131] Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. *Numerical Recipes in C*. Cambridge University Press, Cambridge, 1999.

[132] Rabbani, M., and Jones, P. *Digital Image Compression Techniques*. SPIE Optical Engineering Press, Washington, USA, 1991.

[133] Rayes, A., and Sage, K. Integrated Management Architecture for IP-Based Networks. *IEEE Communications Magazine 38*, 4 (2000), pp. 48–53.

[134] Resnick, S., and Samorodnitsky, G. Performance Decay in a Single Server Exponential Queueing Model with Long Range Dependence. *Operations Research 45*, 2 (1997), pp. 235–243.

[135] Ribeiro, V., Riedi, R., Crouse, M., and Baraniuk, R. Simulation of nonGaussian Long-Range-Dependent Traffic using Wavelets. In *Performance Evaluation Review, Proceedings of ACM SIGMETRICS'99* (1999), vol. 27(1), pp. 1–12.

[136] Riedi, R., Crouse, M., Ribeiro, V., and Baraniuk, R. A Multifractal Wavelet Model with Application to Network Traffic. *IEEE Transactions on Information Theory 45*, 3 (1999), pp. 992–1018.

[137] Robert, S., and LeBoudec, J.-Y. On a Markov Modulated Chain Exhibiting Self-Similarities Over Finite Timescale. *Performance Evaluation 27-28* (1996), pp. 159–173.

[138] Robert, S., and LeBoudec, J.-Y. New Models for Pesudo Self-Similar Traffic. *Performance Evaluation 30*, 1-2 (1997), pp. 57–68.

[139] Roberts, J. Traffic Theory and the Internet. *IEEE Communications Magazine* (2001), pp. 94–99.

[140] Robinson, P. Gaussian Semiparametric Estimation of Long-Range Dependence. *The Annals of Statistics 23* (1995), pp. 1630–1661.

[141] Rose, O. *Traffic Modeling of Variable Bit Rate MPEG Video and Its Impacts on ATM Networks*. PhD thesis, Bayerische Julius-Maximilians-Universität Würzburg, 1997.

[142] Roughan, M., and Veitch, D. Measuring Long-Range Dependence Under Changing Traffic Conditions. In *Proceedings of IEEE INFOCOM'99* (New York, NY, USA, 1999), pp. 1513–1521.

[143] Roughan, M., Veitch, D., and Abry, P. On-Line Estimation of the Parameters of Long-Range Dependence. In *Proceedings of GLOBECOM'98* (Sydney, Australia, 1998), pp. 3716–3721. http://www.emulab.ee.mu.oz.au/∼matt/oldIndex.html, or http://www.research.att.com/info/roughan.

[144] Roughan, M., Yates, J., and Veitch, D. The Mystery of the Missing Scales: Pitfalls in the Use of Fractal Renewal Processes to Simulate LRD Processes. In *Applications of Heavy Tailed Distributions in Economics, Engineering and Statistics* (American University, Washington, DC, 1999).

[145] Rubinstein, R., and Melamed, B. *Modern Simulation and Modeling*. John Wiley & Sons, Inc., New York, 1998.

[146] Ryu, B. *Fractal Network Traffic: from Understanding to Implications*. PhD thesis, Graduate School of Arts and Sciences, Columbia University, 1996.

[147] Ryu, B. *Implications of Self-Similarity for Providing End-to-End QoS Guarantees in High-Speed Networks: A Framework of Application Level Traffic Modeling*, vol. 1044. Springer-Verlag, B. Plattner (eds), Zurich, Switerland, 1996, ch. Lecture Notes in Comp. Sci. (Proceedings of International Zurich Seminar on Dig. Comm.).

[148] Ryu, B., and Lowen, S. Point Process Approaches to the Modeling and Analysis of Self-Similar Traffic – Part I: Model Construction. In *Proceedings of IEEE INFOCOM'96* (San Francisco, CA, 1996), pp. 1468–1475.

[149] Ryu, B., and Lowen, S. Point Process Models for Self-Similar Network Traffic, with Applications. *Communications in Statistics-Stochastic Models 14*, 3 (1998), pp. 735–761.

[150] Sahinoglu, Z., and Tekinay, S. On Multimedia Networks: Self-Similar Traffic and Network Performance. *IEEE Communications Magazine 37*, 1 (1999), pp. 48–52.

[151] Samorodnitsky, G., and Taqqu, M. *Stable Non-Gaussian Random Processes: Stochastic Models with Infinite Variance*. Chapman and Hall, New York, 1994.

[152] Schiff, S. Resolving Time-Series Structure with a Controlled Wavelet Transform. *Optical Engineering 31*, 11 (1992), pp. 2492–2495.

[153] Sheng, Y. *The Transforms and Applications Handbook*. CRC Press, Inc., A.D. Poularikas (eds), USA, 1996, ch. Wavelet Transform, pp. 747–827.

[154] Stathis, C., and Maglaris, B. Modelling the Self-Similar Behaviour of Network Traffic. *Computer Networks 34*, 1 (2000), pp. 37–47.

[155] Taqqu, M. Self-Similar Processes. In *Encyclopedia of Statistical Sciences*, vol. 8. John Wiley and Sons, Inc., S. Kotz and N. Johnson (eds), New York, 1988.

[156] Taqqu, M., and Levy, J. Using Renewal Processes to Generate Long-Range Dependence and High Variability. *Dependence in Probability and Statistics 11* (1986), pp. 73–89.

[157] Taqqu, M., and Teverovsky, V. Robustness of Whittle-Type Estimators for Time Series with Long-Range Dependence. *Stochastic Models 13* (1997), pp. 723–757.

[158] Taqqu, M., Teverovsky, V., and Willinger, W. Estimators for Long-Range Dependence: an Empirical Study. *Fractals 3*, 4 (1995), pp. 785–788. http://math.bu.edu/people/murad/methods/index.html.

[159] Taqqu, M., Willinger, W., and Sherman, R. Proof of a Fundamental Result in Self-Similar Traffic Modeling. *Computer Communication Review, ACM SIGCOMM 27*, 2 (1997), pp. 5–23.

[160] Taralp, T., Devetsikiotis, M., Lambadaris, I., and Bose, A. Efficient Fractional Gaussian Noise Generation Using the Spatial Renewal Process. In *Proceedings of IEEE International Conference on Communications (ICC'98)* (Atlanta, GA, USA, 1998), pp. 7–11.

[161] Tekalp, A. *Digital Video Processing*. Prentice Hall PTR, NJ, USA, 1995.

[162] Tsybakov, B., and Georganas, N. On Self-Similar Traffic in ATM Queues: Definitions, Overflow Probability Bound, and Cell Delay Distribution. *IEEE ACM Transactions on Networking 5*, 3 (1997), pp. 397–409.

[163] Tsybakov, B., and Georganas, N. Overflow Probability in an ATM Queue with Self-Similar Input Traffic. In *Proceedings of IEEE International Conference on Communications (ICC'97)* (Montreal, Canada, 1997), pp. 822–826.

[164] Tsybakov, B., and Georganas, N. Self-Similar Processes in Communications Networks. *IEEE Transactions on Information Theory 44*, 5 (1998), pp. 1713–1725.

[165] Tsybakov, B., and Georganas, N. Self-Similar Traffic and Upper Bounds to Buffer-Overflow Probability in an ATM Queue. *Performance Evaluation 32* (1998), pp. 57–80.

[166] Tsybakov, B., and Georganas, N. Overflow and Loss Probabilities in a Finite ATM Buffer Fed by Self-Similar Traffic. *Queueing Systems 32* (1999), pp. 233–256.

[167] Veitch, D., and Abry, P. A Wavelet Based Joint Estimator of the Parameters of Long-Range Dependence. *IEEE Transactions on Information Theory 45*, 3 (1999), pp. 878–897. Special Issue on Multiscale Statistical Signal Analysis and its Applications, http://www.emulab.ee.mu.oz.au/∼darryl/.

[168] Veitch, D., Bäckar, J.-A., Wall, J., Yates, J., and Roughan, M. On-Line Generation of Fractal and Multifractal Traffic. In *The PAM2000 Passive and Active Measurement Workshop* (Hamilton, New Zealand, 2000).

[169] Veres, A., Kenesi, Z., Molnár, S., and Vattay, G. On the Propagation of Long-Range Dependence in the Internet. In *Computer Communication Review, Proceedings of ACM SIGCOMM'00* (Stockholm, Sweden, 2000), vol. 30 (4), pp. 243–254.

[170] Wickerhauser, M. *Adapted Wavelet Analysis from Theory to Software*. A. K. Peters, Ltd., Wellesley, Massachusetts, 1994.

[171] Willinger, W., Taqqu, M., and Erramilli, A. *A Bibliographical Guide to Self-Similar Traffic and Performance Modeling for High-Speed Networks*. Oxford University Press, F.P. Kelly, S. Zachary and I. Ziedins (eds), Oxford: Clarendon, 1996, ch. Stochastic Networks: Theory and Applications, pp. 339–366.

[172] Willinger, W., Taqqu, M., Leland, W., and Wilson, D. Self-Similarity in High-Speed Packet Traffic: Analysis and Modeling of Ethernet Traffic Measurements. *Statistical Science 10*, 1 (1995), pp. 67–85.

[173] Willinger, W., Taqqu, M., Sherman, R., and Wilson, D. Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level. *IEEE ACM Transactions on Networking 5*, 1 (1997), pp. 71–86.

[174] Wise, G., Traganitis, A., and Thomas, J. The Effect of a Memoryless Nonlinearity on the Spectrum of a Random Process. *IEEE Transactions on Information Theory IT-23*, 1 (1977), pp. 84–89.

[175] Woolford, S. Vector Quantisation and Wavelet Based Image Compression for Architectural Type Image Data. Master's thesis, Department of Computer Science, University of Auckland, 1995.

[176] Young, R. *Wavelet Theory and Its Applications*. Kluwer Academic Publishers, Massachusetts, USA, 1994.

APPENDICES

Appendix A

Stochastic Preliminaries

The following sections provide definitions of statistical terms. Most of the definitions are taken from Jain [65], Law and Kelton [85], and Leon-Garcia [91].

A.1 Stochastic Processes

In analytical modelling, we use not only several random variables but also several different sequences or families of random variables that are functions of time. For example, let X_t denote the number of jobs at the CPU of a computer system. If we examine several identical systems and observe the number of jobs at the CPU as a function of time, the number X_t is a random variable. To specify its behaviour, we would need to specify the probability distribution function for X_t at each possible value of t. Similarly, the waiting time in a queue (W_t) is a random function of time. Such random functions of time or sequences are called *stochastic processes*, and are useful in representing the state of queueing systems. Some of the common types of stochastic processes used in queueing theory are described as follows:

A.1.1 Discrete-State and Continuous-State Processes

A process is classified as discrete-state or continuous-state depending on the values its state can accept. If the number of possible values is finite or count-

able, the process is called a discrete-state process. For example, the number of jobs in a system, X_t, can only be equal to discrete values, such as $0, 1, 2, \ldots$. Therefore, $\mathbf{X}$ is a discrete-state process. In contrast, the waiting time (W_t) can equal any value on the real line. Therefore, $\mathbf{W}$ is a continuous-state process. A discrete-state stochastic process is also called a *stochastic chain*.

A.1.2 Markov Processes

If the future states of a process are independent of the past and depend only on its present state, the process is called a *Markov process*. The Markov property makes a process easier to analyse, since we do not have to know the complete past trajectory. These processes are named after A.A. Markov, who defined and analysed them in 1907.

Note that to predict the future of a continuous-time Markov process, it is sufficient to know the current state. It is unnecessary to know how long the process has been in the current state. Predicting the future is possible only if the state time has a memoryless distribution.

A.1.3 Poisson Processes

If inter-arrival times are independent and identically, exponentially distributed, the number of arrivals over a given interval $(t, t+s)$ has a Poisson distribution. Then, the arrival process is referred to as a *Poisson process* or a *Poisson stream* of arrivals.

In other words, the stochastic count process $\mathbf{N} = \{N_t, t \geq 0\}$ is said to be a *Poisson process* if

(i) customers arrive one at a time,

(ii) $N_{t+s} - N_t$ (the number of arrivals in the time interval $(t, t+s]$) is independent of N_u, for any u, $0 \leq u \leq t$, and

(iii) the distribution of $N_{t+s} - N_t$ is independent of t for all $t, s \geq 0$;

see for example Law and Kelton [85]. Poisson processes are popular in simulating queueing theory; arrivals are memoryless because the inter-arrival time

is exponentially distributed. In addition, Poisson streams have the following properties:

- Merging of Poisson streams results in a Poisson stream.
- Splitting of a Poisson stream results in Poisson streams.
- Departures from an $M/M/m$ queue, for $m \geq 1$, form a Poisson process.

A.1.4 Stationary Random Processes

The simplest assumption about random processes is that the nature of their randomness does not change over time. It means that a realisation of a process in the time interval (t_0, t_1) exhibits the same type of random behaviour as its realisation in some other time interval $(t_0+\tau, t_1+\tau)$. This leads us to postulate that the probabilities of samples of the process do not depend on the instant at which we began taking observations; that is, probabilities involving samples taken at times $t_1, \ldots, t_i$ will not differ from those taken at $t_1 + \tau, \ldots, t_i + \tau$.

If we are working with random processes that began at $t = -\infty$, then the previous condition can be stated precisely as follows. A discrete-time or continuous-time random process $\mathbf{X}_t$ is *stationary* if the joint distribution of any set of samples does not depend on the placement of the time origin. This means that the joint CDF of $X_{t_1}, X_{t_2}, \ldots, X_{t_i}$ is the same as that of $X_{t_1+\tau}, X_{t_2+\tau}, \ldots, X_{t_i+\tau}$:

$$F_{X_{t_1},\ldots,X_{t_i}}(x_1, \ldots, x_i) \quad = \quad F_{X_{t_1}+\tau,\ldots,X_{t_i}+\tau}(x_1, \ldots, x_i),$$

for all time shifts τ, all i, and all choices of sample times $t_1, \ldots, t_i$. If a process begins at some definite time, i.e., $t = 0$, then we say it is stationary if its joint distributions do not change under time shifts to the right.

Appendix B

Self-Similarity of Inter-Arrival Times

B.1 Graphical Test of Self-Similarity

For a graphical test of self-similarity, we used two real Ethernet traffic series collected at the Bellcore Morristown Research and Engineering facility during the period from August 1989 to October 1989. Leland et al. [89] used these series in their paper on self-similarity as well. However, they looked at the count processes (the number of packets submitted within a time unit). The *pAug.TL*[1] and *pOct.TL* traffic series contain one million arrival times (with time counted from the beginning of recording) and lengths of arriving packets, which were captured for 3,142.82 seconds starting at 11:25 a.m. on 29 August 1989 and for 1,759.62 seconds starting at 11:00 a.m. on 5 October 1989, respectively.

We examined the same time series for self-similarity of inter-arrival times and used a synthetic sequence of one million inter-arrival times with mean 1, variance 10 and Hurst parameter 0.9 to demonstrate a graphical test, using the inverse cumulative density function (ICDF) technique, based on the exact self-similar FGN process. Figures B.1 and B.2 show graphical tests of self-similarity of the *pAug.TL* and *pOct.TL* series over 10, 50, 100, 500 and 1000

[1]Ethernet LAN data traces can be obtained via anonymous ftp://bellcore.com/pub/lan_traffic.

time units. Figure B.3 shows graphical tests of the self-similarity of synthetic traffic generated by the exact self-similar FGN process. Bursts appeared over all five time scales, and provided some evidence that inter-arrival times are also self-similar. Visual comparisons between the different sequences also suggest that the more often bursts occurred in the traffic, the higher the Hurst parameter H. However, Figure B.4 shows that the bursts of the Poisson process did not appear in all five time scales. Thus, the self-similar processes shown in Figures B.1 – B.3 have different characteristics from the Poisson process shown in Figure B.4.

B.2 Comparison of the Exact FGN and Real Ethernet LAN Traffic

Traffic generated by the exact self-similar FGN process was compared with a Poisson model and the real Ethernet LAN traffic series: *pAug.TL* and *pOct.TL* [89] using the index of dispersion for intervals (IDI). IDI was used to measure the number and frequency of traffic bursts because it is relatively straightforward to estimate and conveys more information than simpler indices such as the coefficient of variation.

Inter-arrival times obtained from the exact self-similar FGN process were also compared with inter-arrival times of a real Ethernet LAN traffic series and Poisson models using the IDI.

Figures B.5 and B.6 show that the IDI curves for inter-arrival times of the exact self-similar FGN process, the *pAug.TL* series and the *pOct.TL* series monotonically increased as t increased. The behaviour of self-similar processes can clearly be recognised in Figures B.5 and B.6. The exact self-similar FGN process and the *pAug.TL* series were similar over a wide range of time scales. The exact self-similar FGN process and the *pOct.TL* series also were similar well over a wide range of time scales, as seen in Figure B.6. However, the IDI curve of the *pOct.TL* series was higher than the curve of the exact self-similar FGN process when time scales $log10(t) < 1.9$ because the variation of the consecutive numbers of the *pOct.TL* series was unstable. There were large variances in expected values when time scales were small. Hurst parameter estimates, $\hat{H}$, for the exact self-similar FGN process, the *pAug.TL* series and

the $pOct.TL$ series, were 0.8134, 0.7666 and 0.8090, respectively. In contrast, Figures B.5 and B.6 also show that IDI curves for a Poisson model were a constant value for all t. The Hurst parameter estimate, $\hat{H}$, for the Poisson model, was 0.4949.

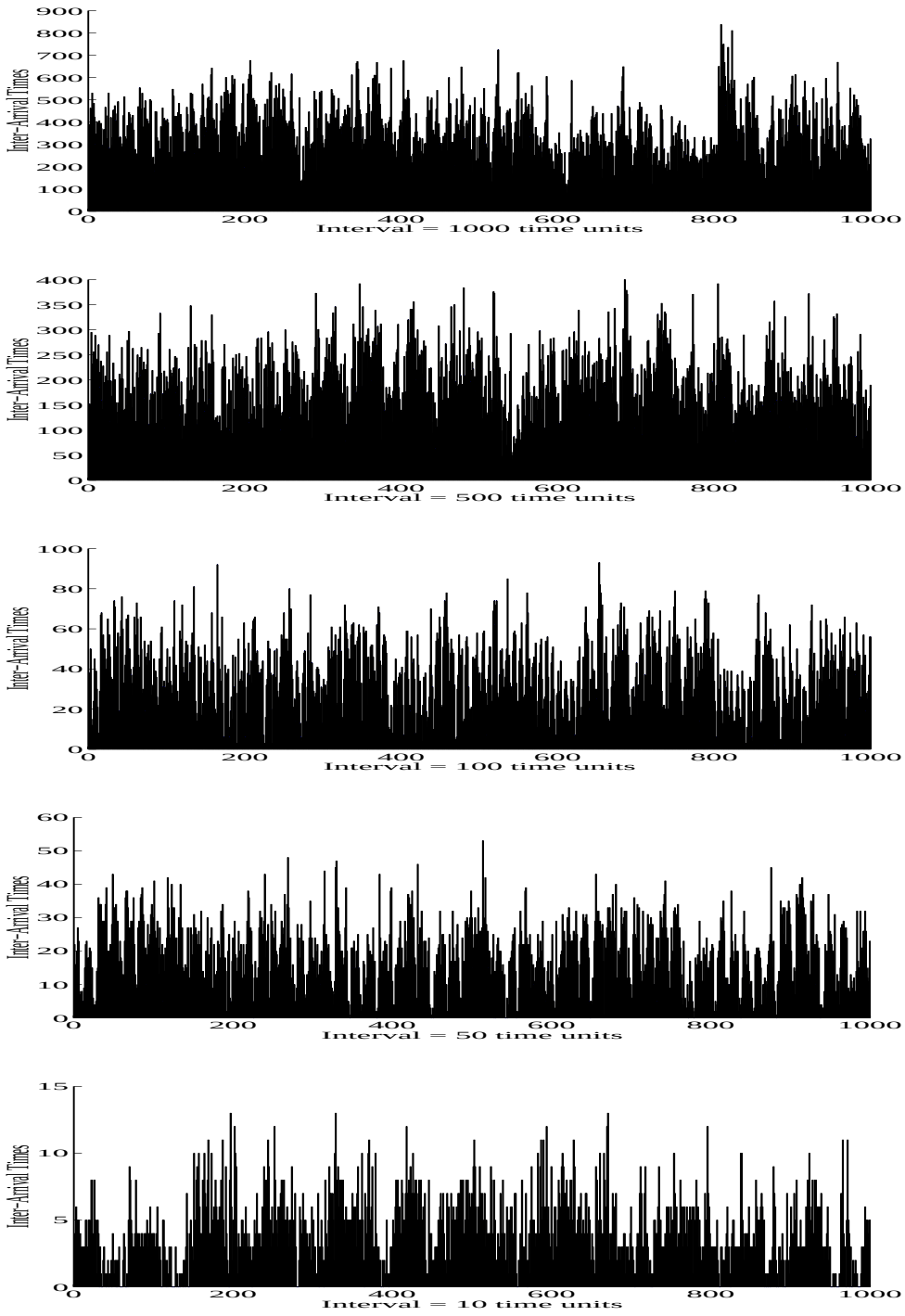


Figure B.1: Graphical test of self-similarity of inter-arrival times: *pAug.TL* series over 10, 50, 100, 500 and 1000 time scales.

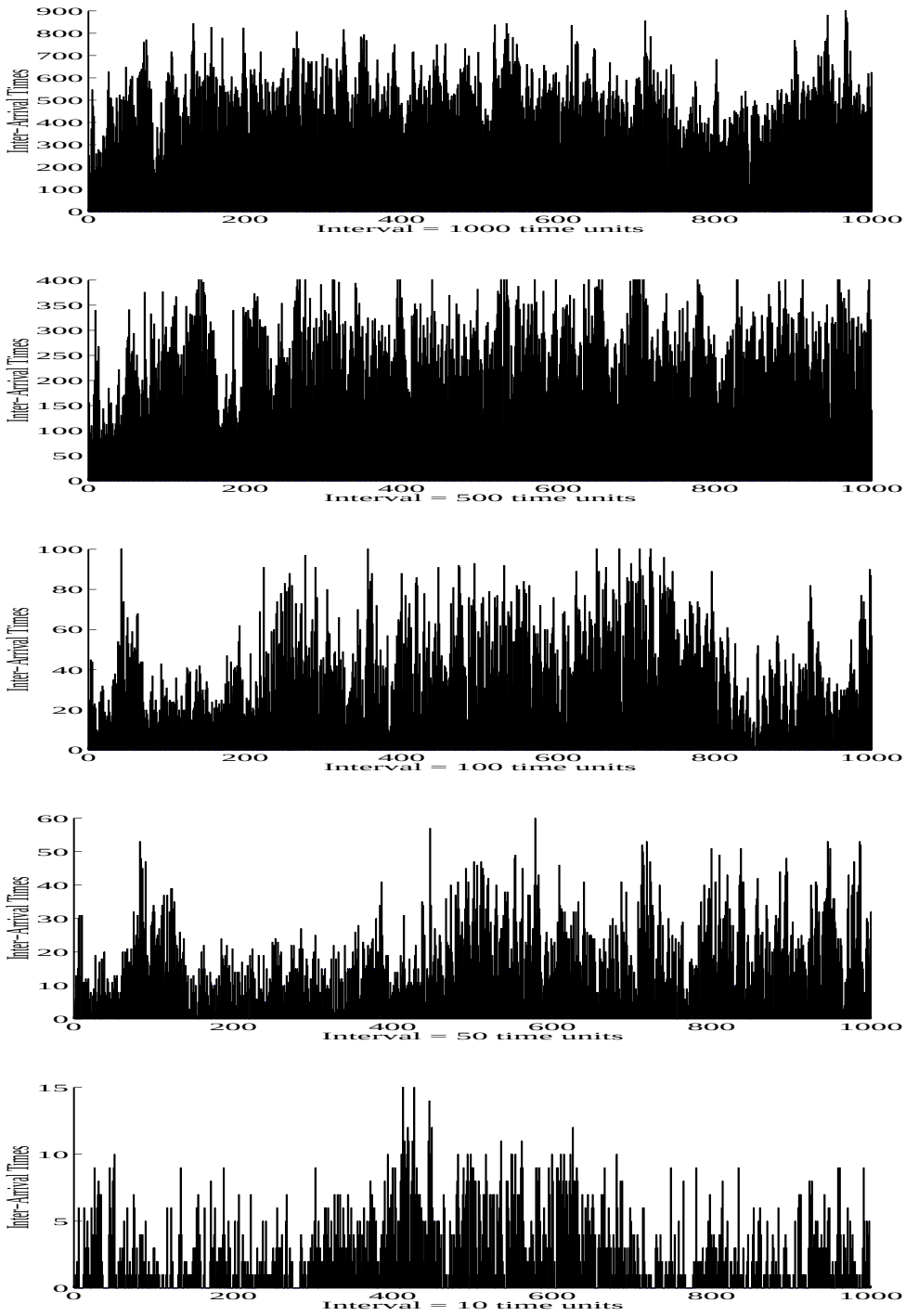


Figure B.2: Graphical test of self-similarity of inter-arrival times: *pOct.TL* series over 10, 50, 100, 500 and 1000 time scales.

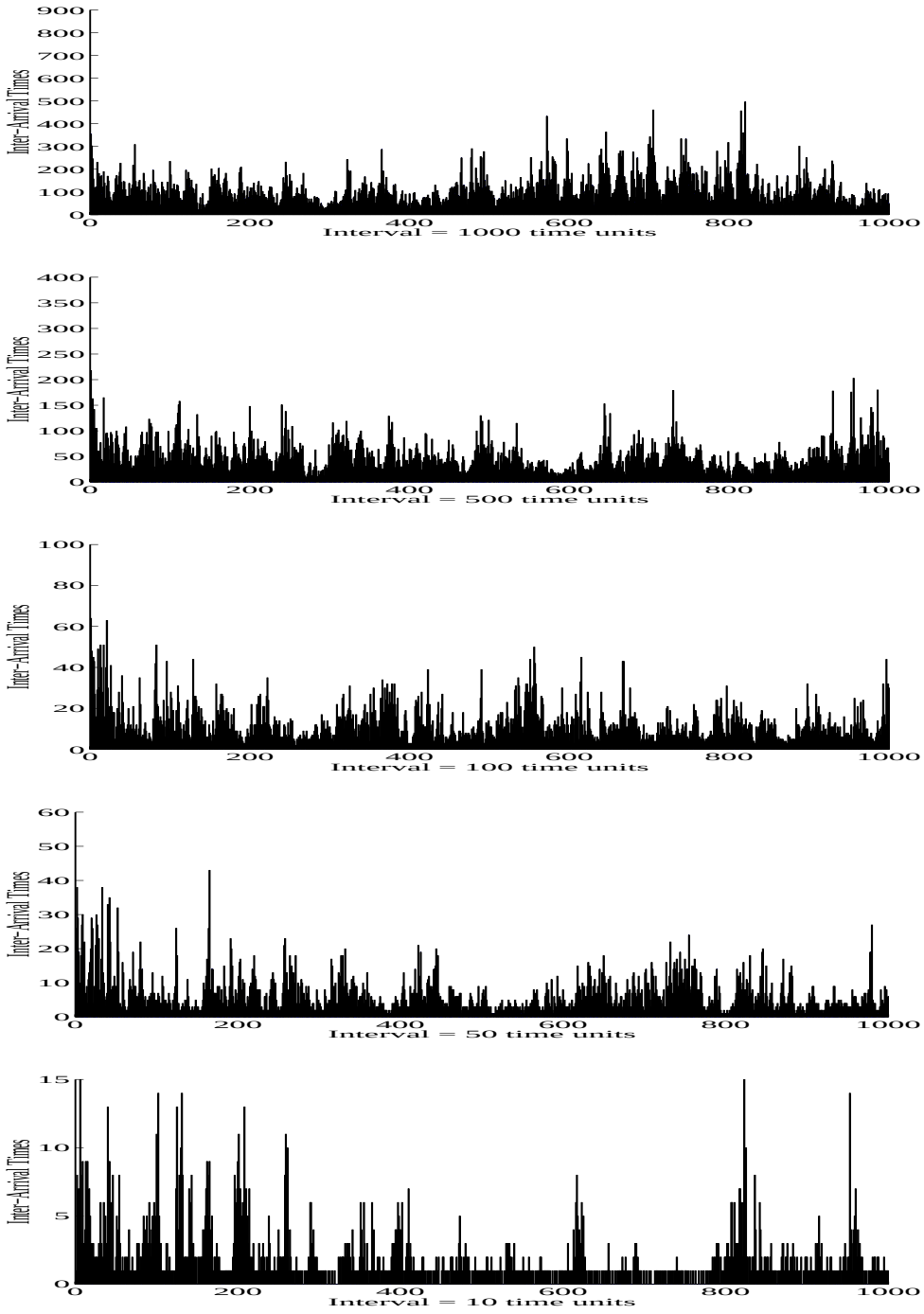


Figure B.3: Graphical test of self-similarity of inter-arrival times: synthetic traffic obtained from the exact self-similar FGN process using the ICDF transformation over 10, 50, 100, 500 and 1000 time scales.

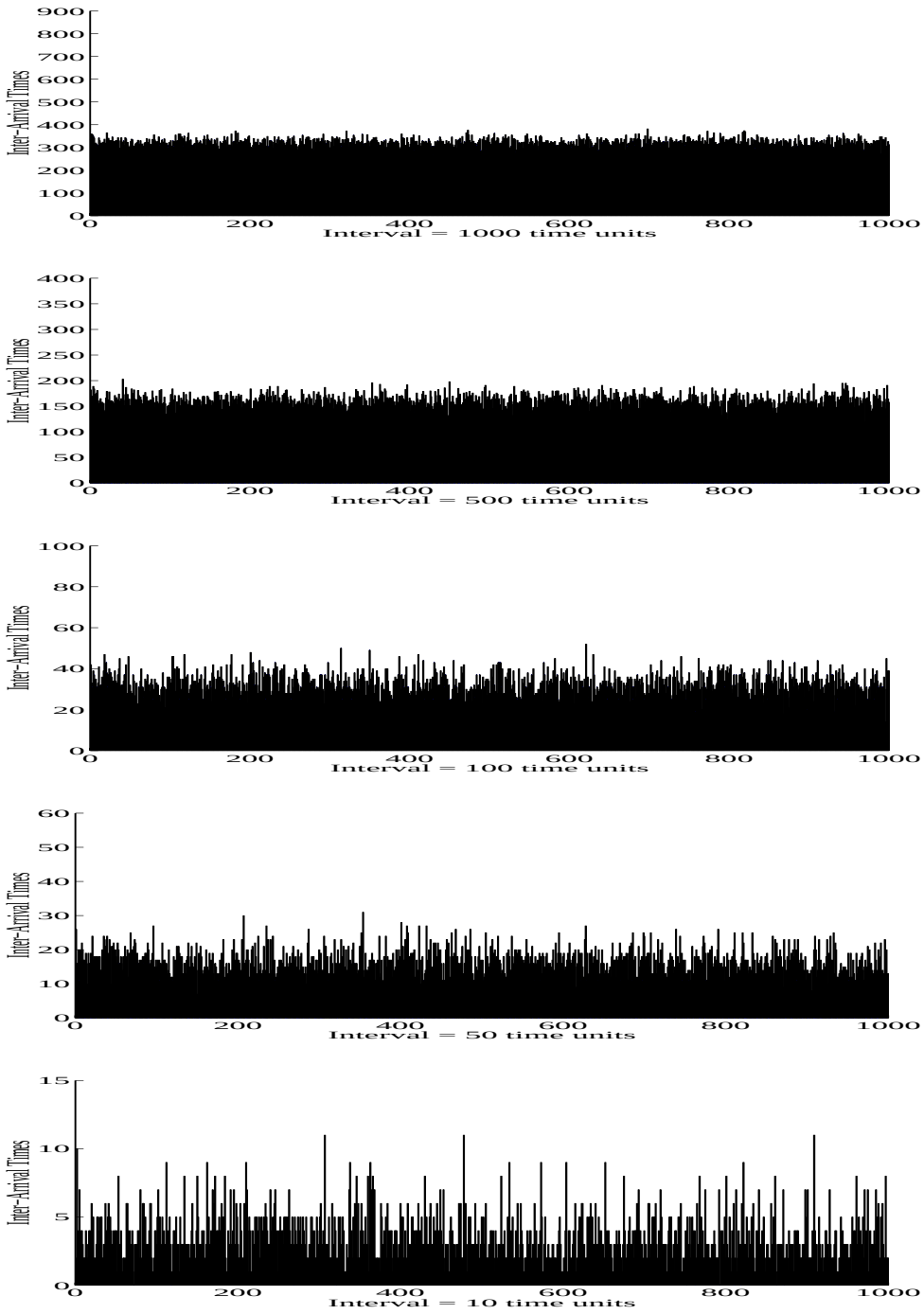


Figure B.4: Graphical test of self-similarity of inter-arrival times: synthetic traffic obtained from the *Poisson process* over 10, 50, 100, 500 and 1000 time scales ($\lambda = 0.9$).

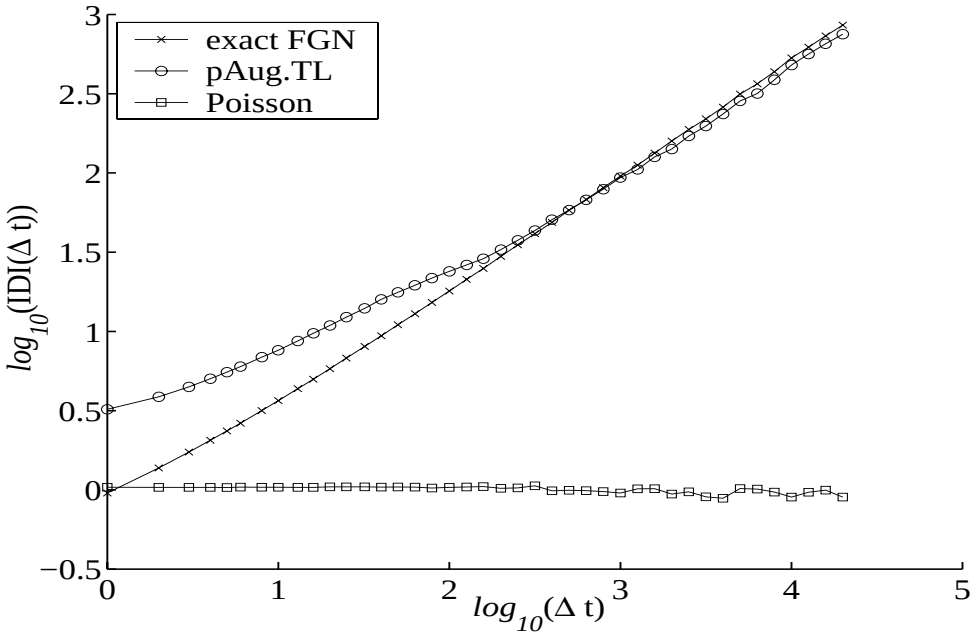


Figure B.5: IDI curves of inter-arrival times for the exact self-similar FGN process, Bellcore Ethernet LAN traffic *pAug.TL* series and the Poisson process.

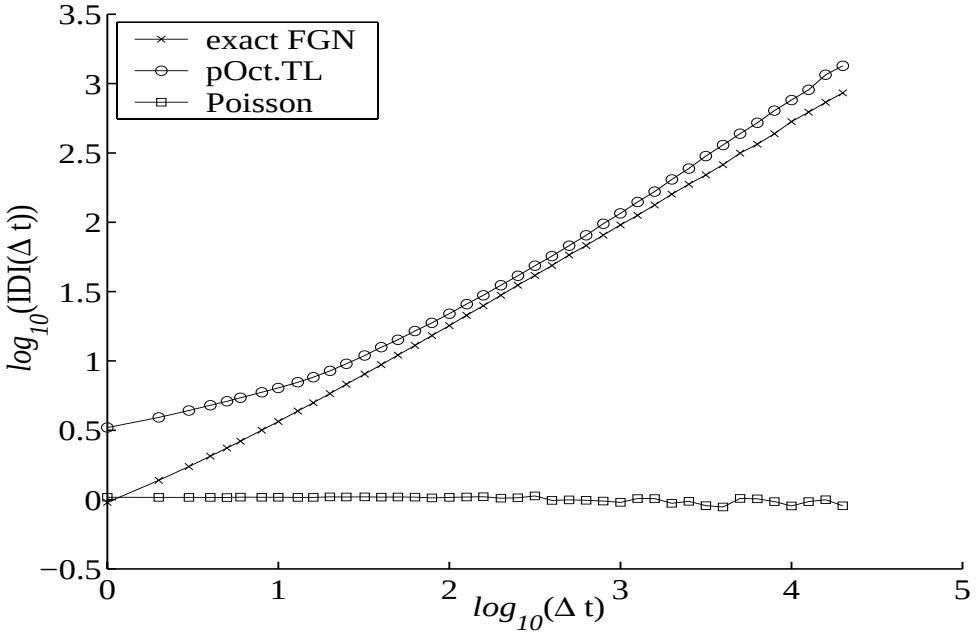


Figure B.6: IDI curves of inter-arrival times for the exact self-similar FGN process, Bellcore Ethernet LAN traffic *pOct.TL* series and the Poisson process.

Appendix C

Exact Self-Similar FGN Process

C.1 Durbin-Levinson Algorithm

An exact self-similar FGN sequence $\{X_1, X_2, \ldots, X_n\}$ with mean zero and autocovariance function given by coefficients $\gamma_k = \frac{\sigma^2}{2}[(k+1)^{2H} - 2k^{2H} + (k-1)^{2H}]$, $k = 0, \pm 1, \pm 2, \ldots$, can be generated using the Durbin-Levinson algorithm [1], [13], [158]. First, generate a sequence $\{Z_1, Z_2, \ldots, Z_n\}$ of IID normal $N(0,1)$ random variables. Then, set $X_1 = \sqrt{\gamma_0} Z_1$ and generate $\{X_1, X_2, \ldots, X_n\}$ from

$$X_{n+1} = \phi_{n,1} X_n + \ldots + \phi_{n,n} X_1 + \sqrt{var_n} Z_{n+1}, n \geq 1,$$

where the variances var_i $(i = 0, \ldots, n)$ and the coefficients $\phi_{n,i}$ $(i = 1, \ldots, n)$ are computed recursively by the following algorithm.

Set $var_0 = \gamma_0$, $\phi_{1,1} = \gamma_1/\gamma_0$, $var_1 = var_0 * (1 - {\phi_{1,1}}^2)$.

For $n = 1, 2, \ldots$,

$$\begin{aligned} \phi_{n,n} &= \left[\gamma_n - \sum_{j=1}^{n-1} \phi_{n-1,j}\gamma_{n-j}\right] / var_{n-1}, \\ \begin{bmatrix} \phi_{n,1} \\ \vdots \\ \phi_{n,n-1} \end{bmatrix} &= \begin{bmatrix} \phi_{n-1,1} \\ \vdots \\ \phi_{n-1,n-1} \end{bmatrix} - \phi_{n,n} \begin{bmatrix} \phi_{n-1,n-1} \\ \vdots \\ \phi_{n-1,1} \end{bmatrix} \end{aligned}$$

and

$$var_n = var_{n-1}[1 - {\phi_{n,n}}^2].$$

We use this algorithm to generate exact self-similar sequences in our comparative analysis of H estimation techniques. This algorithm, written in C, has been adopted from [13] and [158].

```
#include <string.h>
#include <stdio.h>
#include <math.h>
#include "ranlib.h"

double ExactFGN(long n,double H,double std,double *data)
{
  /*Calculate the autocovariance function. */
  std2 = std*std/2.0;
  for (i = 0; i <= n; i++)
      autocov[i] = std2*(pow((double)(i+1),2.0*H) -
                   2.0*pow((double)i,2.0*H) +
                   pow((double)abs(i-1),2.0*H));

  /* Calculate the Durbin-Levinson coefficients. */
  phi1[1] = autocov[1]/autocov[0];
  phi2[1] = phi1[1];
  var[0]  = autocov[0];
  var[1]  = var[0]*(1.0 - phi1[1]*phi1[1]);
  data[1] = sqrt(var[0])*gennor(0.0,1.0);
```

```
  for (i=2; i <= n; i++){
      phi1[i] = autocov[i];
      for (j=1; j <= i-1; j++)
          phi1[i] -= phi2[j]*autocov[i-j];
      phi1[i] = phi1[i]/var[i-1];
      var[i]  = var[i-1]*(1.0-phi1[i]*phi1[i]);
      data[i] = sqrt(var[i-1])*gennor(0.0,1.0);
      for (j=1; j <= i-1; j++){
          phi1[j] = phi2[j]-phi1[i]*phi2[i-j];
          data[i] += phi2[j]*data[i-j];
      }
      for (j=1; j <= i; j++)
          phi2[j] = phi1[j];
  }
} /* ExactFGN */

int main(int argc, char *argv[])
{
    x = (double*) malloc((n+1) * sizeof(double));
    ExactFGN(n,H,1.0,x);
    for (i = 1; i <= n; ++i){
        fprintf(outfp,"%lg\n", x[i]);
    }
    free(x);
}
```

Appendix D

FGN-DW Generator of Self-Similar Sequences

D.1 C Code

```
#include <string.h>
#include <stdio.h>
#include <math.h>
#include "ranlib.h"

extern void DaubechiesFilter(double *h, double *g, double *rh,
            double *rg, int daub_coeff);
extern void InverseWaveletTransform(double *data, double *rh,
            double *rg, int scale, int n, int daub_coeff);
void FGN_spectrum(double *pow_spec, int size, double H);
```

```
int main(int argc,char *argv[])
{
   /* Approximate ideal power spectrum: */
   FGN_spectrum(pow_spec,n,H);

   /* Adjust for estimating power spectrum via periodogram: */
   for(i = 1; i <= n; i++)
       pow_spec[i] *= genexp(1.0);

   /* Construct corresponding complex numbers with random phase: */
   for(i = 0; i < n; i++) {
       z[2*i] = sqrt(pow_spec[i+1]);
       z[2*i+1] = 2.0 * M_PI * ranf();
       }

   /* Calculate complex spectral density and  */
   /* real part/imaginary part form. */
   for(i = 0; i < n; i++) {
       re = z[2*i] * cos(z[2*i+1]);
       im = z[2*i] * sin(z[2*i+1]);
       z[2*i]   = re;
       z[2*i+1] = im;
       }

   /* Calculate filter values using Daubechies' algorithm */
   DaubechiesFilter(h,g,rh,rg,daub_coeff);

   /* Calculate a sequence in time domain */
   /* using inverse discrete wavelet transform (IDWT) */
   InverseWaveletTransform(z,rh,rg,scale,n,daub_coeff);
}/* main */
```

```
/************************************************/
void FGN_spectrum(double *pow_spec, int n, double H)
{
   /* Returns an approximation of the power spectrum */
   /* for fractional Gaussian noise at the given frequencies */
   /* lambda and the given Hurst parameter H.*/
   g = exp(gamma(2.0 * H + 1.0));
   fact1 = (1.0 / (2.0 * M_PI)) * sin(M_PI * H) * g;
   mean1 = 0.0;
   for(i=0; i < n; i++) {
       lambda[i] = (M_PI * (i + 1)) / n;
       mean1 += lambda[i];
       }
   mean1 /= n;
   var1  = 0.0;
   for(i=0; i < n; i++)
       var1 += ((lambda[i] - mean1) * (lambda[i] - mean1));
   var1 /= n;
   for(i = 1; i < n+1; i++) {
       a = pow(lambda[i-1],(1.0 - 2.0 * H));
       pow_spec[i] = fact1 * var1 * a;
       }
}/* FGN_spectrum */
```

D.2 Matlab Code

```
% This function returns a self-similar sequence
% with n numbers and the Hurst parameter H.
function SS = FGNDW(n,H,Scale,VanishingMoment)

% Create n frequencies, then calculate fast,
% approximately power spectrum.
lambda = ((1:n)*pi)/n;
f = FGNDWSpectrum(lambda,H);

% Adjust for estimating power spectrum via periodogram.
r = random('Exponential',1,1,n);
f_r = f.*r;

% Construct corresponding complex numbers with random phase.
re = sqrt(f_r);
im = random('Uniform',0,2*pi,1,n);

% Calculate complex spectral density and
% real part/imaginary part form.
real_part = re.*cos(im);
imag_part = re.*sin(im);
z = real_part + imag_part*i;

% Calculate filter values using I. Daubechies' algorithm.
[h,g,rh,rg] = Daub(VanishingMoment);

% Calculate a sequence in time domain
% using inverse discrete wavelet transform (IDWT).
SS = real(iwt(z,rh,rg,Scale));
```

```
% ------------------------------------------------
% Returns an approximation of the power spectrum
% for Fractional Gaussian Noise at the given
% frequencies lambda and the given Hurst parameter H.
% ------------------------------------------------
function FGNS = FGNDWSpectrum(lambda,H)

cf = (1/(2*pi))*var(lambda)*sin(pi*H).*gamma(2*H+1);
FGNS = cf*(abs(lambda)).^(1-2*H);

% ------------------------------------------------
% Returns coefficients of Daubechies wavelets
% ------------------------------------------------
function [h,g,rh,rg] = Daub(VanishingMoment)

n = VanishingMoment/2;
poly = trigpol(n);
zeros = roots(poly);
modulus = abs(zeros);
j = 1;
for i = 1:(2*(n-1))
    if (modulus(i)<1)
       zerosinside(j) = zeros(i);
       j = j+1;
    end;
end;
Pol = poly(1);
Real = 0;
Image = 0;
```

```
for i = 1:(n-1)
    if (imag(zerosinside(i)) == 0)
       Real = Real+1;
       RealZeros(Real) = zerosinside(i);
    else
       Image = Image+1;
       ImageZeros(Image) = zerosinside(i);
    end;
end;
Norm = 1;
for i = 1:Real
    Norm = Norm*abs(RealZeros(i));
end
for i = 1:Image
    Norm = Norm*abs(ImageZeros(i));
end
Norm = sqrt(abs(Pol)/Norm);
Norm = 0.5^n*sqrt(2)*Norm;
rh = [1 1];
for i = 2:n
    rh = conv(rh,[1 1]);
end
for i = 1:Real
    rh = conv(rh,[1 -RealZeros(i)]);
end
for i = 1:2:Image
 rh=conv(rh,[1 -2*real(ImageZeros(i)) abs(ImageZeros(i))^2]);
end
rh = Norm*rh;
[rh,rg,h,g] = rh2rg(rh);
```

Appendix E

Wavelet Transform

E.1 Introduction

Various modelling approaches for capturing the self-similar nature of LAN and VBR video traffic were outlined in Section 2.4. As mentioned earlier, an important requirement for conducting simulation studies of telecommunication networks is the ability to generate long synthetic stochastic self-similar sequences.

We present a new generator of pseudo-random self-similar sequences based on fractional Gaussian noise (FGN) and a wavelet transform using Daubechies wavelets (DW), called the FGN-DW method. Daubechies wavelets are more accurate than Haar wavelets because they more closely match the self-similar structure of long-range dependent processes (see also Section 4.4.5) [2], [170]. The statistical accuracy and time required to produce sequences of a given (long) length were experimentally studied. This generator showed a high level of accuracy of the output data (in the Hurst parameter) and is fast. Its theoretical algorithmic complexity is $O(n)$ [26], [46]. We begin by describing a wavelet transform for synthesising an approximate FGN.

E.2 Wavelet Transform

The wavelet transform was developed in the mid-1980s. The multi-resolution analysis (MRA) term holds in all the procedures to obtain the decomposition

of a sequence according to the scales occurring in it. In wavelet transforms, the MRA is accomplished using the scaling parameter. The scaling function and mother wavelet obtained from the decomposition of a discrete wavelet transform (DWT) of a sequence are described [77], [101], [102]. Our method for generating synthetic self-similar FGN sequences in a time domain is also based on the discrete wavelet transform (DWT). It has been shown that wavelets can provide compact representations for a class of FGN processes [40], [78]. This is because the structure of wavelets naturally matches the self-similar structure of the long range dependent processes [2], [26], [170].

E.2.1 Description of the Wavelet Transform

The basic algorithm of the wavelet transform is shown in Figure E.1 [46], [153], [176]. This simple filter algorithm performs a one-dimensional one-scale wavelet transform on any one-dimensional input sequence. It uses the pyramidal algorithm shown in Figure E.2. This is a recursive algorithm based on two filters (i.e., G_0 and G_1) that are derived from the scaling function and mother wavelet chosen for the transformation. G_1 is a high-pass wavelet filter and G_0 is the complementary low-pass wavelet filter. The outputs are the low-pass residue for the G_0 filter branch, represented by *Approx* in Figure E.1, and the high-pass sub-band for the G_1 branch, represented by *Detail* [46], [115].

The DWT can be computed efficiently by the pyramidal algorithm. Figure E.2 shows that the output coefficient of the one-dimensional multi-scale DWT is obtained by iterating the basic algorithm on the low-pass residue of each

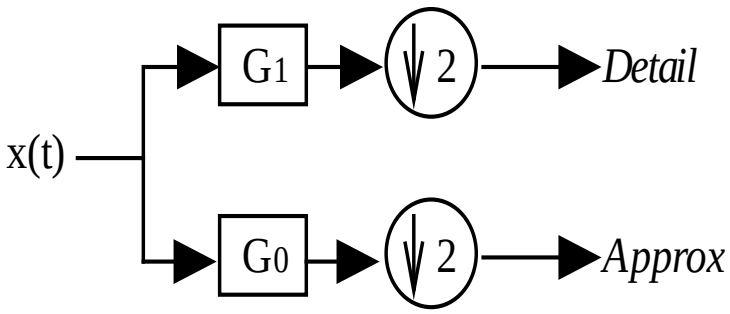


Figure E.1: Basic algorithm for the wavelet transform.

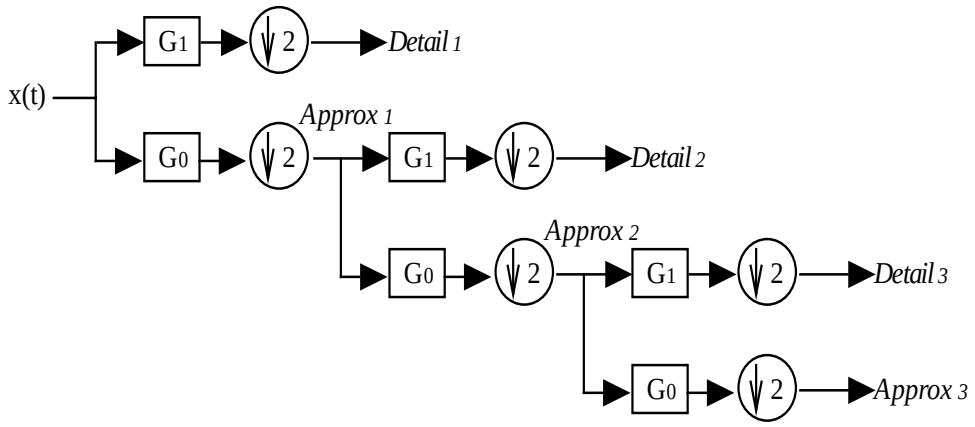


Figure E.2: Recursive pyramidal algorithm for the multi-scale wavelet transform.

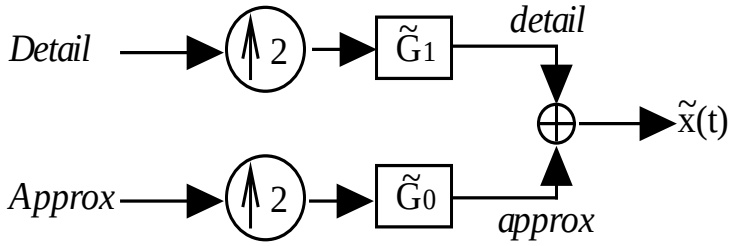


Figure E.3: Basic algorithm for the inverse wavelet transform.

of the previous stages. The computational cost of the pyramidal algorithm is $O(n)$ [2], [124], which is even lower than the cost of the FFT algorithm; $O(n\log n)$.

The synthesis or reconstruction process is achieved by using the basic algorithm shown in Figure E.3. $\tilde{G}_0$ and $\tilde{G}_1$ execute inverse transformations to those executed by G_0 and G_1, and $\tilde{X}(t)$ is the outcome. The multi-scale reconstruction is performed with the same algorithm iterated to obtain the successive low-pass residues.

E.2.2 Multi-Resolution Analysis

The wavelet transform is based on the concept of MRA. It plays a role in mapping the sequence to the time-scale joint representation. A simple definition of MRA is one that considers information at different resolutions (or scales). Fourier transforms are constant resolution based, because they involve single forward/reverse transforms that convert data to/from a different representation. The wavelet transform is an MRA, because it splits information at a given scale into several sub-representations, each of which is a smaller scale representation of part of the original data. This split can be repeated iteratively (until a limit, defined by the size of the original data, is reached).

The MRA consists of a collection of nested subspaces $\{V_i, i \in Z\}$, satisfying the following set of properties [1], [2],:

(i) $\cap V_i = \{0\}, \cup V_i$ is dense in Hilbert space $L^2(R)$.

(ii) $V_i \subset V_{i-1}$.

(iii) $x(t) \in V_i \longleftrightarrow x(2^i t) \in V_0$.

(iv) There exists a function $\phi_0(t)$ in vector V_0, called the *scaling function*, such that the collection $\{\phi_0(t-j), j \in Z\}$ is an orthonormal basis for V_0.

The scaled and shifted functions $\phi_{i,j}(t)$

$$\{\phi_{i,j}(t) = 2^{-i/2}\phi_0(2^{-i}t - j), j \in Z\} \tag{E.1}$$

constitute an orthonormal basis for the vector space V_i.

Performing an MRA of a sequence x means successively projecting the sequence into each of the approximation subspaces V_i, i.e.,

$$\text{approx}_i(t) = (\text{Proj}_{V_i}x)(t) = \sum_j a_x(i,j)\phi_{i,j}(t), \tag{E.2}$$

where the coefficient $a_x(i,j)$ is given by calculating the following inner product of x:

$$a_x(i,j) =< x, \phi_{i,j} > . \tag{E.3}$$

The approximation approx_i is a coarser approximation of x than approx_{i-1} since $V_i \subset V_{i-1}$, and the subsequent loss of information is the detail of the sequence x when moving from one approximation to the next, one $\text{detail}_i(t) = \text{approx}_{i-1}(t) - \text{approx}_i(t)$. The detail sequences, detail_i, can be obtained by projecting x onto a collection of subspaces, called the wavelet subspaces, W_i. Furthermore, the MRA theory shows that there exists a function ψ_0, called the mother wavelet, derived from ϕ_0, such that its templates

$$\{\psi_{i,j}(t) = 2^{-i/2}\psi_0(2^{-i}t - j), j \in Z\} \tag{E.4}$$

constitute an orthonormal basis for the wavelet subspace W_i. Moreover, all base functions $\psi_{i,j}(t)$ have the same shape as the mother wavelet and therefore are self-similar. The detail information of the sequence x is given by

$$\text{detail}_i(t) = (\text{Proj}_{W_i}x)(t) = \sum_k d_x(i,j)\psi_{i,j}(t), \tag{E.5}$$

where the coefficient $d_x(i,j)$ is given by calculating the following inner product of x:

$$d_x(i,j) =< x, \psi_{i,j} > . \tag{E.6}$$

MRA represents the information about sequence x as a collection of details and a low-resolution approximation

$$\begin{aligned} x(t) &= \text{approx}_N(t) + \sum_{i=1}^{N} \text{detail}_i(t) \\ &= \sum_j a_x(N,j)\phi_{N,j}(t) + \sum_{i=1}^{N}\sum_j d_x(i,j)\psi_{i,j}(t). \end{aligned} \tag{E.7}$$

For a scaling function ϕ_0 and mother wavelet ψ_0, the discrete wavelet transform (DWT) is given by a mapping

$$x(t) \longrightarrow \{\{a_x(N,j), j \in Z\}, \{d_x(i,j), i = 1, \ldots, N, j \in Z\}\}. \tag{E.8}$$

Since the function ϕ_0 produces an approximation of the signal x, it must be a low-pass filter. In addition, the mother wavelet ψ_0 must be a high-pass filter, since it performs a differential operation on the input signal to produce the detail version.

A simple and popular example of an orthonormal wavelet basis is the Haar wavelet, where the scaling function ϕ_0 and the mother wavelet ψ_0 are defined by

$$\phi_0(i) \begin{cases} \frac{1}{\sqrt{2}} & i = 0, 1 \\ 0 & otherwise \end{cases} \tag{E.9}$$

and

$$\psi_0(i) \begin{cases} \frac{1}{\sqrt{2}} & i = 0 \\ -\frac{1}{\sqrt{2}} & i = 1 \\ 0 & otherwise. \end{cases} \tag{E.10}$$

The Haar basis functions are the associated decomposition and reconstruction filters that are used in the discrete wavelet transform. The Haar wavelet is not continuous and its Fourier transform decays only by $|\lambda|^{-1}$, which means that its decomposition has poor scale (frequency) localisation. Thus, the Daubechies wavelets produce smoother curves than Haar wavelets by better matching the self-similar structure of long-range dependent processes. The Daubechies wavelets are also a family of orthonormal wavelets. Other examples of wavelets include the Battle-Lemarié, Meyer, Shannon, and Strömberg wavelets [153].

E.2.3 Daubechies Wavelets

Daubechies discovered one of the original wavelet families [26]. This family contains a single wavelet for each possible even valued filter length, beginning with four coefficients. As the filter length grows, the wavelets move from highly localised (due to the small number of non-zero filter coefficients) to highly smooth (for larger numbers of coefficients). The formation of the smallest of this family of wavelets, often referred to as Daub(4), which stands for the Daubechies wavelets with four coefficients, is calculated below [26], [175].

The transformation matrix for a Daubechies wavelet of length four is given by matrix (E.11).

$$\begin{bmatrix}
c_0 & c_1 & c_2 & c_3 & 0 & 0 & 0 & \dots & & \\
c_3 & -c_2 & c_1 & -c_0 & 0 & 0 & 0 & \dots & & \\
0 & 0 & c_0 & c_1 & c_2 & c_3 & 0 & \dots & & \\
0 & 0 & c_3 & -c_2 & c_1 & -c_0 & 0 & \dots & & \\
 & & & \dots & & & & & & \\
 & & & & \dots & & & & & \\
 & & & & & \dots & & & & \\
 & & & & & \dots & c_0 & c_1 & c_2 & c_3 \\
 & & & & & \dots & c_3 & -c_2 & c_1 & -c_0 \\
c_2 & c_3 & 0 & \dots & & \dots & 0 & 0 & c_0 & c_1 \\
c_1 & -c_0 & 0 & \dots & & \dots & 0 & 0 & c_3 & -c_2
\end{bmatrix} \tag{E.11}$$

The odd rows of this matrix represent a convolution of the data vector with the coefficients $[c_0, c_1, c_2, c_3]$. These rows can be efficiently calculated due to its sparsity (requiring four multiplications and three additions per data value). The even rows of this matrix represent a convolution of the data vector with the coefficients $[c_3, -c_2, c_1, -c_0]$. They are also efficiently calculated, requiring the same number of operations. The total effect of this operation is to convolve the data vector with two different four-coefficient filters and reduce the results by a factor of two.

The inverse (reconstruction) transform is the inverse of this matrix, and is equal to its transpose. It can be used to generate the following requirements for the values $[c_0, c_1, c_2, c_3]$ [131]:

$$\begin{cases}
c_0^2 + c_1^2 + c_2^2 + c_3^2 &= 1 \\
c_0 * c_2 + c_1 * c_3 &= 0.
\end{cases} \tag{E.12}$$

For the values of these coefficients to form the desired high- and low-pass filters, we must also require:

$$\begin{cases}
c_3 - c_2 + c_1 - c_0 &= 0 \\
0 * c_3 - 1 * c_2 + 2 * c_1 - 3 * c_0 &= 0.
\end{cases} \tag{E.13}$$

The unique solution to these four equations is:

$$\begin{cases} c_0 &= (1+\sqrt{3})/4 * \sqrt{2} \\ c_1 &= (3+\sqrt{3})/4 * \sqrt{2} \\ c_2 &= (3-\sqrt{3})/4 * \sqrt{2} \\ c_3 &= (1-\sqrt{3})/4 * \sqrt{2}. \end{cases} \tag{E.14}$$

This same general method can be used to form Daubechies wavelet filter values for any even filter length greater than four.

The Daub(4) DWT of an input data vector is calculated as follows (this method assumes a data vector length that is a power of two; the use of other lengths requires special treatment).

Given the data vector:

$$[y_0\ y_1\ y_2\ y_3\ y_4\ y_5\ y_6\ y_7\ y_8\ y_9\ y_{10}\ y_{11}\ y_{12}\ y_{13}\ y_{14}\ y_{15}],$$

transform it with (multiply by) the forward matrix (E.11) to give:

$$[s_0\ d_0\ s_1\ d_1\ s_2\ d_2\ s_3\ d_3\ s_4\ d_4\ s_5\ d_5\ s_6\ d_6\ s_7\ d_7],$$

where s_* are smooth responses (low-pass) and d_* are detail responses (high-pass).

This vector is then permuted to collect the smooth and detail areas as follows:

$$[s_0\ s_1\ s_2\ s_3\ s_4\ s_5\ s_6\ s_7\ d_0\ d_1\ d_2\ d_3\ d_4\ d_5\ d_6\ d_7].$$

For a DWT, this process is then iteratively repeated on the smooth values to obtain the following:

$$[S_0\ D_0\ S_1\ D_1\ S_2\ D_2\ S_3\ D_3\ d_0\ d_1\ d_2\ d_3\ d_4\ d_5\ d_6\ d_7],$$

and then permuted to:

$$[S_0\ S_1\ S_2\ S_3\ D_0\ D_1\ D_2\ D_3\ d_0\ d_1\ d_2\ d_3\ d_4\ d_5\ d_6\ d_7].$$

In this example we must now stop, as the length of the next smooth sub-vector (length four) equals the length of the analysis filter (length four). Reconstruction is an exact reversal of this procedure, using the transpose (inverse) of the forward transform matrix.

For our generator, we chose Daubechies wavelets because they produce more accurate coefficients of wavelets than Haar wavelets (for more detailed discussions, see also [26], [170]; and our results of the comparison in Chapter 4).

Appendix F

$M/M/1/\infty$ Queueing Systems: Simulation Run-Length for Mean Waiting Time

In a typical simulation, a variance or a mean of parameters (such as response times, waiting times and queue lengths) is unknown. Nevertheless, simulation practitioners would like to be able to plan a simulation and, in particular, estimate how long the simulation must be run so as to obtain a credible confidence interval [123]. Depending on which steady-state parameters are estimated in a queueing system, a different number of observations must be collected to reach the required confidence interval. Here, we considered only the mean waiting time from the $M/M/1/\infty$ queueing system. These were estimated with a specified error (or precision) for 95% confidence level. The theoretical number of observations required to estimate these parameters can be obtained as follows. The derivation of a formula that can calculate the theoretically required run-length for the $M/M/1/\infty$ queueing system is ascribed to Daley [25]. We begin by defining a way to measure the relative precision.

Let $\hat{W}_q$ be the estimate for the mean waiting time in the queue W_q. The requirement that the estimate $\hat{W}_q$ of W_q is obtained with a given relative precision of 5%, for 95% confidence level, can be written as:

$$Pr(|\hat{W}_q - W_q| \leq 0.05W_q) = 0.95. \tag{F.1}$$

If $\rho = \lambda/\mu$ is the traffic intensity, where λ is the arrival rate and μ is the

service rate, then for an $M/M/1/\infty$ queue the steady-state mean waiting time in the queue is given by

$$W_q = \frac{\rho^2}{\lambda(1-\rho)}. \tag{F.2}$$

The steady-state variance of the waiting time in the queue is also given by

$$\sigma^2(W_q) = \frac{\rho^3(2-\rho)}{\lambda^2(1-\rho)^2}. \tag{F.3}$$

The Laplace-Stieltjes transform of PDF of response times $W^*(s)$ in the $M/G/1/\infty$ queueing system is defined as:

$$W^*(s) = \frac{(1-\rho)sB^*(s)}{s-\lambda[1-B^*(s)]} \tag{F.4}$$

in Kleinrock [79] (p.199), where $B^*(s)$ is the Laplace-Stieltjes transform of probability density function (PDF) of exponential service times, equal

$$B^*(s) = \int_0^\infty \mu e^{-\mu t} e^{-st} dt = \frac{\mu}{\mu+s}, \tag{F.5}$$

in McNickle [106] and Kleinrock [79] (p.195).

This simplifies Equation (F.5) to:

$$W^*(s) = \frac{\mu(1-\rho)}{s+\mu(1-\rho)} \tag{F.6}$$

as in Kleinrock [79] (p.202). The Laplace-Stieltjes transform of PDF of response times can also be written as

$$W^*(s) = W_q^*(s)B^*(s),$$

where $W_q^*(s)$ is the Laplace-Stieltjes transform of PDF of waiting times in the queue, since $ResponseTime = WaitingTimeInTheQueue + ServiceTime$. Thus, the Laplace-Stieltjes transform of PDF of waiting times in the queue is given by

$$W_q^*(s) = \frac{(1-\rho)s}{s-\lambda[1-B^*(s)]} \tag{F.7}$$

in [52] (p.202), which simplifies Equation (F.5) to:

$$W_q^*(s) = \frac{(s+\mu)(1-\rho)}{s+(\mu-\rho)}. \tag{F.8}$$

Detailed discussions of Laplace-Stieltjes transforms can be found in Gross and Harris [52] and Kleinrock [79].

If the system has been operating for a long time, and one selects n observations of waiting times $W_1, \cdots, W_n$, then the mean waiting time in the queue

$$\hat{W}_q = \sum_{i=1}^{n} W_i/n$$

has, for sufficiently large n,

$$n\sigma^2(\hat{W}_q) \doteqdot \sigma^2(W_q)\left[1 + 2\sum_{j=1}^{\infty}\rho_j(m)\right], \tag{F.9}$$

where

$$1 + 2\sum_{j=1}^{\infty}\rho_j(m) = \frac{1+\rho}{1-\rho} + \frac{\lambda(W_q''' - W_q'W_q'')}{(1-\rho)(W_q'' - W_q'W_q')},$$

where W_q', W_q'' and W_q''' can be calculated by the first, second and third differentiations of the Laplace-Stieltjes transform of PDF of mean waiting time in the queue $W_q^*(s)$, and $\sigma^2(W_q)$ can be calculated by $(W_q'' - W_q'W_q')$ to obtain the number of observations in the queue. For a more detailed discussion, see Daley [25] and Fishman [39].

From Equation (F.1), we assume that $\left(\frac{|\hat{W}_q - W_q|}{\sigma(\hat{W}_q)}\right)$ is a normal $N(0,1)$ distribution, then we have

$$Pr\left(\frac{|\hat{W}_q - W_q|}{\sigma(\hat{W}_q)} \leq \frac{0.05W_q}{\sigma(\hat{W}_q)}\right) \;=\; 0.95$$

or

$$\frac{0.05W_q}{\sigma(\hat{W}_q)} \;=\; 1.96. \tag{F.10}$$

Then, from Equations (F.9) and (F.10), we can obtain the following equation [106]. The number of observations needed to estimate the mean waiting times in an $M/M/1/\infty$ queueing system, with a relative precision of 5% for 95% confidence level, can be calculated theoretically as

$$n \;=\; \left(\frac{1.96}{0.05W_q}\right)^2 A(\rho), \tag{F.11}$$

where

$$\begin{aligned} A(\rho) &= \sigma^2(W_q)\left[\frac{1+\rho}{1-\rho} + \frac{\lambda(W_q''' - W_q' W_q'')}{(1-\rho)(W_q'' - W_q' W_q')}\right] \\ &= \frac{\rho^3(2-\rho)}{\lambda^2(1-\rho)^2}\left[\frac{2\mu^3 + 5\lambda\mu^2 - 4\mu\lambda^2 + \lambda^3}{(2\mu-\lambda)(\mu-\lambda)^2}\right]. \end{aligned} \tag{F.12}$$

Thus, using Equations (F.2) and (F.3) we get:

$$\begin{aligned} n &= \left(\frac{1.96}{0.05}\right)^2 \left(\frac{2+5\rho-4\rho^2+\rho^3}{\rho(1-\rho)^2}\right) \\ &= 1536.64\left(\frac{2+5\rho-4\rho^2+\rho^3}{\rho(1-\rho)^2}\right). \end{aligned} \tag{F.13}$$

Table F.1 shows the numbers of observations required in theory, when estimating mean waiting times in an $M/M/1/\infty$ queueing system with a relative precision of 5% and 10%, respectively; see also [145]. One can note that in this case we need to run the simulation for approximately four times longer if we want to achieve results twice as accurate.

Table F.1: Theoretically required run-length, when estimating mean waiting times in the $M/M/1/\infty$ queueing system with a relative precision of 5% and 10% at 95% confidence level.

ρ	*Relative Precision = 5%*	*Relative Precision = 10%*
0.1	46,687	11,671
0.2	34,190	8,547
0.3	33,105	8,276
0.4	36,357	9,134
0.5	44,562	11,140
0.6	60,441	15,110
0.7	94,710	24,830
0.8	189,775	47,443
0.9	681,072	170,268

Appendix G

Automated Simulation Package: Akaroa-2

A fully automated simulation package, Akaroa-2 (version 2.4.1)[1], was used as a performance evaluation tool in this dissertation. Akaroa-2 is the latest version of a fully automated simulation tool designed to run parallel/distributed stochastic simulations under the Multiple Replications In Parallel (MRIP) scenario in a local area network (LAN) environment [32], [33].

The Akaroa-2 system has three main components: *akmaster*, *akslave* and *akrun*, plus three auxiliary components: *akadd*, *akstat* and *akgui*; more detailed discussion can be found in Ewing et al. [32] and [33]. Figure G.1 shows the relationships between the three main components of Akaroa-2. Each bold-outlined box represents one Unix process, and the connecting lines represent Transmission Control Protocol and Internet Protocol (TCP/IP) stream connections.

The *Akmaster* is the master process that coordinates the activity of all other processes initiated by Akaroa-2. It launches new simulations, maintains

[1]The original version of Akaroa was designed at the Department of Computer Science, University of Canterbury in Christchurch, New Zealand, by Associate Professor Krzysztof Pawlikowski (Computer Science), Dr. Victor Yau (Computer Science) and Dr. Don McNickle (Management). The latest version (Akaroa-2) was re-implemented by Dr. Greg Ewing (Computer Science). The Akaroa-2 package can be freely downloaded for the purpose of teaching and non-profit research activities at universities and research institutes from http://www.cosc.canterbury.ac.nz.

information about the state running simulations, performs global analysis of the data produced by simulation engines, and makes simulation stopping decisions.

The *akslave* processes run on hosts that run simulation engines. The sole function of the *akslave* is to launch simulation engine(s) on its host as directed by the *akmaster*.

Once the *akmaster* and any desired *akslaves* are running, the *akrun* program is used to initiate a simulation. It first contacts the *akmaster* process, obtaining its host name and port number from a file left by the *akmaster* in the user's home directory. For each simulation engine requested, the *akmaster* chooses a host from among those hosts on the LAN that are running *akslave* processes. It instructs the *akslave* on that host to launch an instance of the user's simulation program, passing on any specified arguments. The first time the simulation program calls one of the Akaroa-2 library routines, the simulation engine opens a connection to the *akmaster* process and identifies the simulation to which it belongs, so that the *akmaster* can associate the connection with the appropriate simulation data structure.

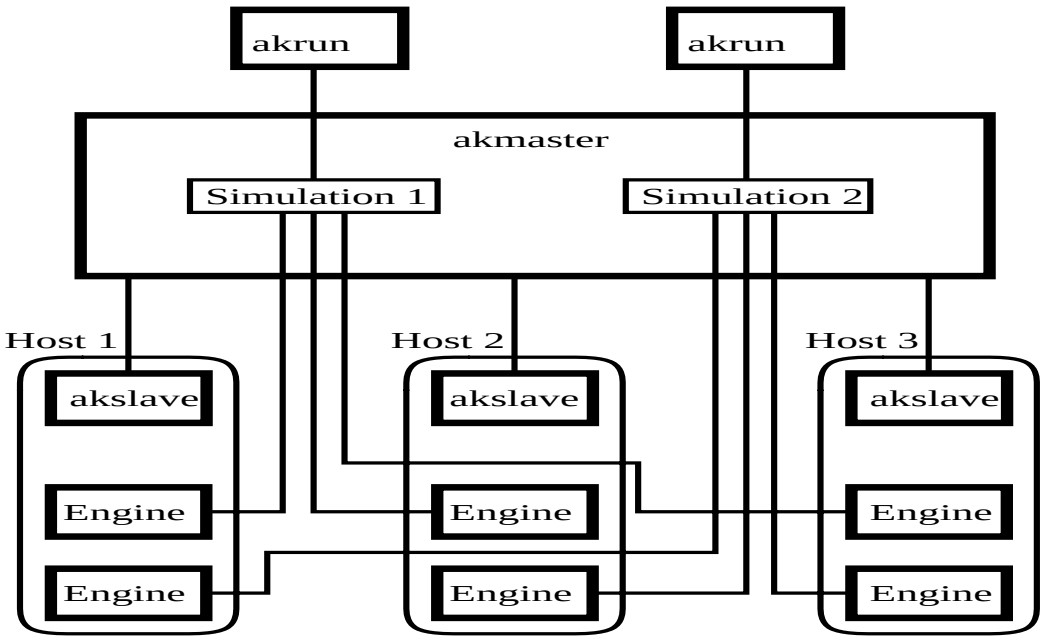


Figure G.1: Architecture of Akaroa-2 (taken from [33]).

Akadd is used to add more simulations to a running simulation. It can be used to replace simulation engines that have been lost for some reason, or to speed up the simulation if more hosts become available. *Akstat* is used to obtain information about the state of the Akaroa-2 system: which hosts are available, which simulations are running, and what progress each simulation is making. *Akgui* provides a graphical user interface to start and monitor simulations that can be used instead of, or in addition to, *akrun* and *akstat*.

Following the principles of sequential simulation [122], each engine in Akaroa-2 performs a sequential analysis of its own data to form a local estimate of each performance measure. At more or less regularly determined checkpoints, the engine sends its local estimates to the *akmaster* process, where the local estimates of each performance measure from all the engines are combined to produce a set of global estimates. Whenever a new global estimate is calculated, the relative half-width of its confidence interval is computed, and compared with the required precision. When the precision of all analysed performance measures becomes satisfactory, the *akmaster* terminates all the simulation engines and sends the final global estimate to the *akrun* process, which in turn reports them to the user.

ACKNOWLEDGEMENTS

I would like to thank my two supervisors, Associate Professor Krzysztof Pawlikowski and Dr. Donald McNickle. Without their enthusiastic support and continuous encouragement, I would never have stood at this point of the dissertation.

I thank Dr. Donald Adjeroh (now at the Department of Computer Science and Electrical Engineering, West Virginia University, USA.), Dr. Tim Bell, Russell Hocken and other members of the Digital Video Laboratory for helping me and allowing me to use their facilities for encoding the *Titanic* video studied in Chapter 6. I also thank Professor Murad Taqqu of Boston University in the USA, Dr. Bong-Kyun Ryu of HRL Labs in the USA, Research Professor Ilkka Norros of VTT in Finland, Dr. Ashok Erramilli of Qnetworx, Dr. Darryl Veitch of the University of Melbourne in Australia, and Professor Dr. Tadao Takaoka and Dr. Easaw Chacko of the University of Canterbury, for valuable discussions of heavy-tailed distributions, fractal renewal processes, finite/infinite queueing models and self-similar processes with marginal distributions to this work. I would also like to thank the members of my defence committee, Dr. Manfred Jobmann, Dr. Don McNickle, Associate Professor Dr. Krzysztof Pawlikowski, Dr. Matthew Roughan and Professor Dr. Moshe Zukerman for their valuable remarks on the final version of the thesis.

I am indebted to all the members of two interdepartmental research groups on Networks and Stochastic Simulation at the University of Canterbury, especially Dr. Greg Ewing, Dr. Irene Hudson, Dr. Sven Östring and Associate Professor Harsha Sirisena, for many productive discussions concerning this work. I thank Gordon Arthur, Jane McKenzie, Stacey Mickelbart, and David Stedman for proofreading drafts of my thesis a number of times. Thanks also go to Tony Dale, Peter Glassenbury, Philip Holland, Dr. Blair McMaster and all other staff members of the Computer Science Department who, in a variety of ways, made my PhD. study at the University of Canterbury a pleasant and memorable experience.

I would like to record my appreciation of the previous President Dae-Hyun Shin of Konyang University, Korea, and Professor Sang-Ok Seo and Visiting Professor Arch Gilchrist of Chungnam National University, Korea, who in-

spired me to come to the University of Canterbury to pursue my professional goals.

During the course of this dissertation, I participated in a number of international conferences with partial financial support from Postgraduate Conference Funding in the Department of Computer Science, Dr. D. McNickle's research grant, the IEEE New Zealand South Section and Telecom New Zealand. A research project on *Algorithmic Generators of Self-Similar Teletraffic and Their Applications in Simulation Studies of Telecommunication Networks* was pursued through the financial support of Telecom New Zealand Ltd. I would like to thank all those who provided any financial support.

Finally, I am deeply thankful to my parents, my brothers and sisters, my family and friends for their unconditional love and support. Thanks so much to Andy, Heather, Raewyn and Derek Chirnside who have helped me to settle in New Zealand. I thank brothers and sisters at the Christchurch Korean assembly, Rowley Avenue and Wairakei Road Bible Chapels for helping my faith to grow. I am thankful to Chris Hanham, Professor Murray McEwan, Murray and Carmen McMillan, and Mary Stedman who have given me confidence, encouragement and support. Dr. Ruth Lee deserves very special thanks for her understanding and assistance throughout all the time of my PhD. study.