University of Alberta

Library Release Form

Name of Author: Shubhankar Chatterjee

Title of Thesis: Long-Range Dependent Traffic Generation and Change Detection

Degree: Master of Science

Year this Degree Granted: 2007

Shubhankar Chatterjee
8/12 Amar Jyoti Society,
Naupada, Thane,
Maharashtra
India 400602.

Date: _____

University of Alberta

Long-Range Dependent Traffic Generation and Change Detection

by

Shubhankar Chatterjee

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Electrical and Computer Engineering

Edmonton, Alberta
Winter 2007

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Long-Range Dependent Traffic Generation and Change Detection** submitted by Shubhankar Chatterjee in partial fulfillment of the requirements for the degree of **Master of Science**.

_____

Stephen Bates
Supervisor

_____

M.H. (Mike) MacGregor
Co-Supervisor

_____

Vincent Gaudet

_____

Heo Giseon
External Examiner

Date: _____

Abstract

Long-range dependence has been shown to be an ubiquitous feature in aggregate network traffic. The degree of long-range dependence is measured by the Hurst parameter. Several techniques have been proposed for estimating this Hurst parameter. In this thesis, we demonstrate the drawbacks of these estimators in the presence of non-stationarities and discuss how some recently developed tools can be used for working around some of these drawbacks.

Simulations are a widely used technique in the study of network architectures and protocols. Such simulation studies require generating synthetic sequences. These synthetic data sets must possess similar features as the original traffic. In this thesis, we present a novel technique for generating traffic traces from a given trace. The technique is based on the use of the stationary bootstrap algorithm in the wavelet domain. The traces generated by our technique have been shown to be capable of capturing the Hurst parameter and the probability distribution function of the parent trace. In addition, we also demonstrate the superiority of our technique over existing algorithms.

The final part of the thesis is aimed towards detecting a change in the Hurst parameter of a data set. This is based on detecting a change in the variance of the wavelet coefficients of the given data set. If a change in the variance of the wavelet coefficients is detected on more than one level, then a change in the Hurst value is signalled.

Contents

List of Figures

List of Tables

List of Acronyms

ACF Autocorrelation Function

ARIMA Autoregressive Integrated Moving Average Model

ARMA Autoregressive Moving Average Model

CLT Central Limit Theorem

DWT Discrete Wavelet Transform

FARIMA Fractionally Integrated Autoregressive Moving Average Model

FBM Fractional Brownian Motion

FFT Fast Fourier Transform

FGN Fractional Gaussian Noise

FPP Fractal Point Process

LRD Long Range Dependent or Long Range Dependence

PDF Probability Distribution Function

SRD Short Range Dependent or Short Range Dependence

SWT Stationary Wavelet Transform

WT Wavelet Transform

Chapter 1

Introduction

Measurement based studies in the 1990's have revealed that traffic behavior in IP networks is self-similar and that aggregate traffic displays long-range dependence (LRD). This LRD behavior has been shown to be present in Local Area Network (LAN) traffic [47], Wide Area Network (WAN) traffic [61], World Wide Web (WWW) traffic [14] and Variable Bit Rate (VBR) Video traffic [9]. Recently, it has been demonstrated that traffic in a broadband network with a high percentage of peer-to-peer traffic is also self-similar [24]. Thus, it has been shown quite conclusively, that presence of LRD in network traffic is ubiquitous.

The presence of LRD in a time series indicates that the autocorrelation function decays hyperbolically. Consequently, while long-term correlations are individually small, their cumulative effect is non-negligible. This behavior is quite different from that observed in traditional short range dependent (SRD) models like the Markovian models in which the autocorrelation function has an exponential decay. Due to the distinct differences between LRD and SRD models, their implications on performance estimation may be significantly different. The presence of LRD in network traffic and its differences from SRD have prompted many researchers to focus on the impact of LRD traffic on network behavior. Simulation-based studies for evaluating the effects on a network system have demonstrated that LRD can affect network performance levels in terms of the network link bandwidth and buffer responses [18, 23]. In [5, 59], it was demonstrated that LRD has a profound effect on queue length and packet loss. In contrast to the negative impacts of the existence of LRD, there are also some positive aspects. For example, in [89], the authors demonstrated that LRD traffic can be forecasted quite accurately. This predictable nature of LRD traffic has been used to propose congestion control algorithms in [32] and [94].

The results above are obtained using simulation based studies. Simulations are also used extensively in the planning process of telecommunication networks. In order to obtain valid conclusions from the simulations it is necessary to use accurate traffic models. There are a number of traffic models proposed in the literature that capture the LRD behavior of aggregate net-

work traffic. The Fractional Gaussian Noise (FGN) and Fractionally Integrated Autoregressive Moving Average (FARIMA) models are currently the most popular models. However, most of the proposed traffic models are targeted towards generating Gaussian traffic, but it has been established that aggregate traffic does not always follow a Gaussian distribution [41, 51], and it is important to account for the Probability Distribution Function (PDF) of the traffic [28, 67, 31]. There have been some models proposed like alpha-stable models [25, 40, 29] or Multifractal models [68] that are capable of generating LRD traffic with the desired PDF. However, most of the techniques are parametric and involve a complicated parameter fitting step. Thus, it is necessary to develop a technique that is automated without the need for any complex parameter fitting procedure.

All the models proposed for capturing LRD have a corresponding long-range dependence parameter called the Hurst parameter, which measures the degree of LRD. There are a number of methods proposed for estimating this parameter, like the R/S estimator [47, 81], the aggregated variance [47, 81], the variance of residuals [47, 81], the absolute moments [47, 81], the periodogram method [47, 81], the Whittle's estimator [81], the detrended fluctuation analysis [62] and the wavelet method [4, 84].

A number of algorithms have been proposed in the literature that make use of the estimates of the Hurst parameter. For example, in [44], the authors have proposed a congestion control mechanism called as *Measurement Based Congestion Control* (MBCC) by monitoring the traffic in real time and reacting to any changes in the characteristics seen. As previously discussed, one of the important characteristics of LRD traffic is the Hurst parameter. In [55] a rate based control algorithm has been proposed that uses the real time estimate of the Hurst parameter. In [56], the authors propose the use of the Hurst parameter for computing the bandwidth requirements of the traffic flow by applying the Norros formula [54] and then use this computed bandwidth for the Random Early Detection (RED) algorithm. In [96], the authors have proposed a method for estimating the effective bandwidth for LRD traffic, which is used to bound the overflow probability. In [97], the authors propose making a real-time estimate of the autocorrelation structure of the network traffic for designing network control schemes. They demonstrate that the precision for the above estimate is linked to the sample length used, which in turn is related to the Hurst parameter. In [75], the authors have proposed an admission control algorithm based on prediction of the traffic by fitting a FARIMA model to traffic at real time.

The estimates of the Hurst parameter required by these algorithms can be obtained by using any of the previously mentioned estimators. However, all these estimators are based on the assumption of stationarity of the Hurst parameter. But real traffic changes its behavior with time; for example, it is well known that there are diurnal variations in the traffic load and preliminary investigations in [47] suggest that there is a correlation between the network load and the Hurst parameter. Thus the Hurst parameter is expected to vary

over time. In [72], the authors speculate that the Hurst parameter depends on the type of applications dominating the traffic, which also changes with time, and thus the Hurst parameter again varies. In [50], the authors use a recurrence plot scheme to demonstrate that the self-similarity seen in measured traffic fluctuates with time. In [57], the authors have shown that the LRD changes with time and also varies with time of day.

Even though the above studies are not able to unambiguously establish the relationship of the Hurst parameter with the underlying traffic, they prove beyond doubt that the Hurst parameter changes with time.

Given the number of algorithms that have been proposed to make use of the Hurst estimate and the demonstrated non-stationarity of the Hurst parameter, it is necessary to develop techniques that are able to detect changes in the Hurst parameter. Once the change point is detected, the Hurst estimate can then be obtained by any of the commonly used techniques.

1.1 Contribution and Outline

The body of this thesis is composed of four chapters. Chapter 2 covers the background required for the rest of the thesis. We discuss self-similarity and long-range dependence. We then cover the background of Wavelet transforms and their use in the analysis of LRD traffic (Section 2.3). In Chapter 3, we discuss in detail the various estimators that are most commonly used for estimating the Hurst parameter, along with a demonstration of some drawbacks of these estimators. We also propose the use of some recently developed tools for working around these drawbacks. In Chapter 4, we propose a novel technique for generating traffic traces which captures the LRD and PDF of traffic. This modeling technique uses the power of bootstrapping coupled with the efficiency of wavelet transforms. The proposed algorithm has been demonstrated to significantly outperform existing algorithms in capturing various characteristics of real traffic traces. In Chapter 5, we propose an algorithm for detecting changes in the Hurst parameter. This algorithm is then compared with another algorithm recently proposed for change detection. Our algorithm is shown to be significantly better in terms of the false positive rate and is also more accurate in detecting the change point. Finally, in Chapter 6, we present our conclusions and further research directions.

Chapter 2

Background

2.1 Self-similarity and Long-Range Dependence

2.1.1 What is Self-similarity?

An object is said to be self-similar if it is similar to a part of itself; i.e parts of the whole can be made to fit to the whole by scaling. Self-similarity, or scale invariance is an attribute of many laws of nature and is the underlying concept of fractals. The concept of self-similarity is related to the occurrence of similar patterns at different time scales. Some examples of self-similar objects are the *Koch snowflake*, *Mandelbrot set* and the *Julia set*.

2.1.2 Definitions and Properties of Self-similar Processes

The standard definition for self-similarity states that a process X_t is self-similar if

$$X(at) \stackrel{d}{=} a^H X(t),\ a > 0,$$

where the equality is in the sense of finite-dimensional distributions, a is a scaling factor, and the self-similarity parameter H is called the Hurst exponent.

A process X(t) is called second-order self-similar with parameter $H = 1 - \beta/2$, $0 < \beta < 1$, if

$$r(k) = \frac{1}{2}[(k+1)^{2-\beta} - 2k^{2-\beta} + (k-1)^{2-\beta}],\ \text{k} \in (1,2,...), \tag{2.1}$$

where r(k) is the autocorrelation function of X(t). Second order self similarity describes the property that an aggregated series will have the same correlation structure (ACF) as the original series. Simply put, a second order self similar time series ACF is the same for either coarse or fine time scales. In cases that we study, we deal only with second order self similarity. Second-order self-similarity manifests itself in a number of ways such as long-range dependence and slowly decaying variances.

2.1.2.1 Long-Range Dependence (LRD)

A stationary process is long-range dependent if its autocorrelations decay to zero so slowly that their sum doesn't converge, i.e. $\sum_{k=1}^{\infty} |\rho(k)| = \infty$. In other words, the autocovariance function has a slow, power-law like decrease at large lags which can be given as:

$$\gamma(k) \sim c_\gamma |k|^{-(2-2H)}, k \to \infty, \tag{2.2}$$

where $c_\gamma > 0$, $0 < H < 1$.

Equivalently in the frequency domain, it can also be defined as the power-law divergence of its spectrum at the origin,

$$f(\nu) \sim c_f |\nu|^{1-2H}, |\nu| \to 0, \tag{2.3}$$

where $c_f > 0$ and $f(\nu) = \sum_k \rho(k) e^{ik\nu}$ denotes the spectral density function. Thus in terms of frequency domain analysis, long-range dependence implies $f(0) = \sum_k \rho(k) = \infty$, or in other words the series has a spectral density which tends to ∞ as the frequency ν approaches 0.

Intuitively, memory is built into the process because the dependence among an LRD process's widely separated values is significant, even across large time shifts. It has been shown that $H > 0.5$ characterizes a series with long range dependence, $H < 0.5$ indicates a series with anti-persistence, while $H = 0.5$ characterizes a series with short range dependence [47]. For our work, we will be concentrating on the LRD property of second-order self-similar processes.

2.1.2.2 Slowly Decaying Variances

Asymptotically second-order self-similar processes have the property of slowly decaying variances, since $\lim_{k\to\infty} \left(r(k)/k^{-\beta}\right) = c$. This property states that the variance of the sample mean decreases slower than the reciprocal of the sample size m and so

$$var[X^{(m)}] \sim cm^{-\beta} \qquad \text{as } m \to \infty \text{ for } 0 < \beta < 1. \tag{2.4}$$

2.2 Modeling Techniques for LRD

In this section, we discuss various techniques that have been proposed for generating self-similar traffic. Such techniques can be broadly classified into two categories; viz. User-oriented models and Black-box models, as described below.

2.2.1 User-oriented Models

In this modeling technique, the behavior of each user is explicitly modeled. This modeling technique is also referred as source level modeling. The different user-oriented models commonly used in the literature are discussed below.

2.2.1.1 On/Off Models

In this technique, traffic is generated by the superposition of a large number of independent ON/OFF sources. Each source transmits data at a constant rate if it is ON and remains silent during the OFF phase. The ON/OFF periods are independent and alternate. The lengths of the ON periods are identically distributed as are the lengths of the OFF periods. This motivation for this technique are the findings in [42, 51, 80, 91, 92], which show that the aggregate flows of many ON/OFF sources with strictly alternating ON and OFF periods, and whose ON and OFF periods exhibit the Noah effect (high variability or infinite variance) can produce aggregate network traffic that exhibits long-range dependence. The observation in [92] that LAN traffic is consistent with an ON/OFF modeling assumption for individual source-destination pairs lends further credibility to this modeling approach. The lengths of the ON and OFF periods have been shown to have finite means and infinite variances. These periods have also been shown to be heavy tailed and Pareto-like with a tail parameter between 1 and 2.

The biggest advantage of this modeling approach is that it is physically meaningful and also offers parsimonious modeling (only one parameter α is used to describe the model). Another benefit of this approach is that since the modeling is at the individual source level, it is possible to use multiple processors for generating the traffic according to the ON/OFF model, with each processor generating the traffic for individual sources with some parameter α and then adding the outputs of all the processors to get the aggregate traffic. This leads to a very fast method for generating long traffic traces.

However, the drawback to this approach is that the number of sources needed for the aggregate traffic to be statistically self-similar is not defined. In addition, the rate of transmission in the ON periods and the lower cut-off of the time scale at which this technique can be applied are not defined.

2.2.1.2 Fractal Point Processes

Another technique of generating source-level traffic is by using Fractal Point Processes (FPPs). These were introduced for traffic modeling in [73], and they incorporate a parameterization method for controlling the time scales over which fractal behavior occurs and thus offer a very attractive approach to modeling packet traffic.

FPPs cover a broad range of stochastic processes which manifest self-similarity and includes the ON/OFF processes discussed in Section 2.2.1.1 [92]. In [74], the authors discuss 8 FPPs and establish the mathematical relationship for the parameters of each model with, what is referred to as the Three Fundamental Parameters (TFPs) quantifying the fractal properties of packet traffic, viz. average arrival rate, Hurst parameter and Fractal Onset Time Scale (FOTS). The different FPPs can be used for modeling different phenomena; for example, one of the models can be used for scenarios where

session or flow arrivals follow a Poisson model and session or flow duration and/or volume are heavy-tailed. Another model represents the case where the session arrivals have a fractal nature irrespective of whether session/flow duration/volume is fractal or not. Thus, in a simulation environment, the right kind of model can be picked, depending on the process to be modeled, and a combination of the different models can also be used. FPPs also provide parsimonious and flexible models for generating fractal traffic. Thus, FPPs seem to possess the required characteristics of a good approach for modeling user-level traffic.

2.2.2 Black-box Models

In these models, the aggregate traffic generated by users is modeled and no attempt is made to distinguish individual users within the population. This modeling technique is also known as aggregate traffic modeling.

In these models, the aggregate traffic of various users is modeled, and no attempt is made to distinguish individual connections within the population. The most widely used family of models used for modeling the aggregate traffic are Gaussian processes with time-stationary increments. The validity of Gaussian processes is justified by the fact that aggregate traffic in the network is obtained by superimposing the contributions of many connections. Thus, by virtue of Central Limit theorem (CLT), it can be argued that the superposition resembles a Gaussian process. Two of the most commonly employed Gaussian processes are Fractional Gaussian Noise (FGN) and Fractional Brownian Motion (FBM). However, as pointed out by the authors in [41], for CLT to be applicable, the individual traffic sources should be independent and that is not always the case. This is specially true when the individual sources are competing for a fixed capacity. Even if the traffic rate is less than the capacity of the network (thus assuming that the sources are not influenced, leading to independence), it has been pointed out in [41], that the use of Gaussian models is justified only if the aggregation (both in time and number of individual sources) is sufficiently large. In [51], it has been pointed out that some traffic measurements do not show an agreement with the Gaussian marginal distribution assumption. The authors show that if connection rates are modest relative to heavy tailed connection length distribution tails, then stable Levy motion is a good approach to modeling of aggregate traffic [51]. In the following sections, different approaches leading to either Gaussian or non-Gaussian aggregate traffic are described.

2.2.2.1 Fractional Brownian Motion and Fractional Gaussian Noise

FBM is a widely used self-similar process for the purpose of traffic modeling. It is a zero mean, non-stationary, Gaussian process, and is commonly represented as $B_H = (B_H(s), s > 0)$ and has Hurst parameter H, $0 < H < 1$. The

correlation function for FBM is given as:

$$corr(s,t) = \frac{1}{2}(s^{2H} + t^{2H} - |s-t|^{2H}) \tag{2.5}$$

The increment process of FBM is stationary as well as self-similar and is called FGN. It is denoted by $X_H = (X_H(k) = B_H(k+1) - B_H(k), k > 0)$. Its autocorrelation function is given as:

$$r(k) = \frac{1}{2}(|k+1|^{2H} - 2|k|^{2H} + |k-1|^{2H}), k > 0 \tag{2.6}$$

which can asymptotically be written as

$$r(k) \sim H(2H-1)|k|^{2H-2}$$

In other words, X_H is self-similar. There are a number of methods for generating either FBM or FGN for the purposes of traffic modeling. Some of the most common referred techniques are explained below.

1. Random Midpoint Displacement (RMD) Algorithm
   This method proposed by Lau et. al. [45] is one of the original methods proposed for generating FBM with a known Hurst value. The process of RMD can be briefly explained as follows. Assume that the FBM process is to be generated in the interval [0,T]. Denoting by X(t), the value computed at time instant t, the algorithm first sets X(0) = 0, and by sampling X(T) from a Gaussian distribution with mean 0 and variance T^{2H}. Then X(T/2) is computed as the average of X(0) and X(T) plus an offset D_1. Then, the 2 intervals from 0 to T/2 and T/2 to T are further subdivided, and so on. At each stage a different offset is added. The whole process can be expressed in terms of equations as follows:

$$X(1/2) = \frac{1}{2}(X(0) + X(T)) + D_1$$

$$X(1/4) = \frac{1}{2}(X(0) + X(1/2)) + D_2$$

$$X(3/4) = \frac{1}{2}(X(1/2) + X(T)) + D_2$$

   and so on. D_n is midpoint displacement that has a variance given by:

$$\sigma_n^2 = \frac{s^2(1 - 2^{2H-2})}{2^{2H_n}} \tag{2.7}$$

   $D_n = (G_n)(\sigma_n)$, where G_n is a Gaussian random number with mean 0 and variance s^2. After computing the FBM process, the corresponding FGN is computed which indicates the number of packets arriving per time instant. Since, the number of arriving packets cannot be negative, the method is truncated so that if the increment process has a negative value at any time instant, it is set equal to 0.

2. Paxson's Algorithm
   In [60], a method based on the Fast Fourier Transform (FFT) is proposed for generating paths corresponding to FGN. This method is based on the generation of a FGN sequence having the same power spectrum as the original series.

   The spectrum of an FGN process is given by:

$$f(\lambda; H) = \mathcal{A}(\lambda; H) \left[|\lambda|^{-2H-1} + \mathcal{B}(\lambda; H) \right] \tag{2.8}$$

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

$$\mathcal{A}(\lambda; H) = 2sin(\pi H)\gamma(2H+1)(1 - cos\lambda)$$

$$\mathcal{B}(\lambda; H) = \sum_{j=1}^{\infty} \left[(2\pi j + \lambda)^{-2H-1} + (2\pi j - \lambda)^{-2H-1} \right]$$

   The steps followed in this method are as follows:

   (a) A signal is generated in the frequency domain that has a power spectrum equivalent to FGN.

   (b) The signal is made random by assigning random phases to each of the samples.

   (c) After randomization, the signal is then made symmetric (even magnitude, odd phase).

   (d) The inverse Fourier transform of this signal yields a real signal.

   The method is shown to be very efficient and accurate in producing samples of FGN with the desired value of H. One of the findings in this paper is that the packet arrival processes in real traffic traces do not have a normal marginal distribution on time scales less than 10 seconds. A logarithmic transformation is suggested to get approximate normal distributions, and then use FGN for modeling the log-transformed processes. The paper also discusses two techniques (uniformly distributing the points in the interval or using the algorithm proposed in [45]) for converting the packet arrival process to inter-arrival times for it to be useful in network simulations. In this paper, the authors point out that their model does not incorporate Short Range Dependence (SRD) that is seen in network traffic and caution about neglecting SRD, specially on small time scales (0.01 seconds). It also highlights the need to develop models incorporating SRD.

3. Wavelet-based Method
   In [35], the authors use Daubechies wavelets for the synthesis of fractional Gaussian noise. The algorithm for the generation is quite similar to the one proposed by Paxson (described above) and has been shown

to be a fast generator of self-similar traffic, with a reasonable level of accuracy for Hurst values up to 0.7. In [19], a wavelet-transform based method is proposed for fast and accurate synthesis of sample paths corresponding to a self-similar FBM processes with known Hurst parameters. The wavelet basis used in this paper is the Haar basis, due to its simplicity. The authors demonstrate that the traffic generated by this technique is more accurate than the RMD algorithm with respect to the Hurst parameter. The drawback of this method is the difficulty in the appropriate selection of the wavelet basis with the appropriate number of vanishing moments. The problem of choosing the right number of vanishing moments is exacerbated by the contradictory benefits from choosing wavelets with more vanishing moments (faster decay of the coefficients' correlation) and that with fewer moments (shorter synthesis filters).

2.2.2.2 Fractional Autoregressive Moving Average Model (FARIMA)

As suggested in [60], network traffic modeling needs approaches which can capture both SRD and LRD. The FARIMA family of models are able to model both the SRD and LRD. In [15], the authors propose a FARIMA model for VBR MPEG video traffic at frame level. The model is used to generate artificial traces of traffic which have been tested using the variances test to verify that they possess the required Hurst value. The autocorrelation function of the artificial traffic is also compared with that of the real traffic to verify the goodness-of-fit.

In [77], the authors propose a FFT based approach for simulating a FARIMA time series for use in modeling the traffic of telecommunication networks.

The marginal distribution of the FARIMA model can be controlled by using the appropriate innovations while generating the model.

2.2.2.3 Alpha-Stable Processes

Another technique for capturing the non-Gaussianity of network traffic is to use models based on alpha-stable distributions.

The characteristic function $\phi(t)$ of the alpha-stable distribution has the form:

$$\phi(t) = exp[it\mu - |ct|^{\alpha}(1 - i\beta sign(t)\theta(t,\alpha))] \tag{2.9}$$

where

$$\theta(t,\alpha) = \begin{cases} tan(\frac{\alpha\pi}{2}) & \text{if } \alpha \neq 1 \\ -\frac{2}{\pi}ln|t| & \text{if } \alpha = 1 \end{cases}$$

In the distribution, μ is called the shift parameter and gives the location of the peak of the distribution. β is a measure of asymmetry and is called the skew parameter. This parameter must lie in the range [-1,1] and when it is zero, the distribution is symmetric about $x = \mu$. c is a scale factor which is a measure of the width of the distribution and α is the exponent of the

distribution. This parameter must lie in the range (0,2]. For $\alpha = 2$, the distribution reduces to a Gaussian distribution with variance $\sigma^2 = 2c^2$ and $mean = \mu$. Thus, the Gaussian distribution is a special case of the α-stable distribution. Other values of α provide different levels of burstiness to match that of real data. A random variable X which follows an α-stable distribution is denoted by $X \sim S^{(\alpha)}_{c,\beta,\mu}$.

In [40], Karasaridis, et. al. proposed a technique for using α-stable processes to model network traffic. Queueing simulations were used to demonstrate that their model gave more accurate results than those produced by Gaussian models. Alpha-stable models have also been proposed to be used for modeling heavy network traffic in [25, 29].

2.2.3 Multifractal Models

The models discussed in Sections 2.2.1 and 2.2.2 are able to capture the self-similarity of the traffic. Self-similarity can be thought of as monofractal scaling; being characterized by a single scaling law that holds globally in time and essentially involves only one parameter, the Hurst parameter. For large time-scales, self-similar models are able to capture the burstiness of traffic. However, at finer time scales; viz. below milliseconds, traffic possess a more complex structure.

Recently [69], multifractal scaling has been proposed as a more suitable technique for modeling network traffic by capturing both the small time scale as well as the large time scale scaling behavior. Multifractal scaling is an extension of monofractal scaling obtained by considering properties higher than second order characteristics. Multifractals allow for time-dependent scaling laws and hence offer greater flexibility in describing irregular phenomenon that are localized in time (caused by network specific mechanisms operating at small time scales).

Multifractality in network traffic was first investigated by Riedi et. al. [69] by performing a statistical analysis of the high frequency part of TCP-traffic. Using this approach they show that TCP traffic is not monofractal, but rather it is multifractal and hence models incorporating monofractality are correct only up to second order statistics. The multifractal nature of network traffic has also been demonstrated in [20]. In addition the multifractal approach offers a parsimonious model that provides a more complete and accurate description of actual data traffic over a wide range of time scales.

In [68], Riedi et. al. propose a wavelet based approach for the multiscale modeling of traffic. They use the Haar wavelet transform for their purpose, and propose a scheme for fitting such models to traffic data. To ensure positive values for the resultant data, the authors suggest a scaling of the wavelet coefficients.

In [20] Feldman et al. model the multifractal behavior of traffic by using cascades, which is a multiplicative process that assigns mass to successively smaller time intervals according to some distribution. The basis behind the use

of this model is that transmitted traffic is constructed through fragmentation at successive network layers, and that the total number of bytes is roughly preserved during this fragmentation process. Use of cascades for generating aggregate traffic with multifractal properties in also discussed in [36].

In [66], the authors propose the construction of a multifractal cascade model for non-Gaussian traffic by using a multiplicative cascade model and the Haar wavelet transform. The key feature of this model is that it avoids the generation of negative values for the synthetic traffic, thereby avoiding artifacts caused by rounding off negative values to zero.

It has been shown in [17], that the fine time scale behavior of multifractal traffic can have a significant effect on queuing behavior at low and intermediate utilization levels, whereas self-similarity is important for high utilization levels. This result strengthens the need to develop traffic models that match the multifractal characteristics of traffic.

2.3 Wavelet Transforms and LRD

2.3.1 Background on Wavelets

Fourier analysis is a classical signal processing tool which breaks down the signal into a possibly infinite series of sines and cosines (which are known as the basis functions). One drawback of Fourier analysis is that it does not work very well for signals with sharp discontinuities, and in such cases a different basis function may be more appropriate. Another drawback is that it has only frequency resolution and no time resolution. As a result, it is possible to determine the different frequencies present in the signal, but there is no way to determine when they are present. This presents a problem in analyzing non-stationary signals. Both these drawbacks can be overcome by using wavelet transform. Wavelet Transform (WT) does not have any fixed basis function, and the appropriate basis function can be chosen based upon the application. In addition, WT presents a time-scale representation of the signal. In other words, it can tell the user when certain features occurred in the signal, and about the scale characteristics of the signal. The term *scale* is related to *frequency* and is a measure of the amount of detail in the signal. Large *scale* means fine details or the big picture, while small *scale* generally means coarse details. Thus going from large scale to small scale is equal to zooming in. By examining a signal over a range of scales, WT offers a method for multi-resolution analysis.

Wavelet transform is based on the principle of expanding the input signal in terms of oscillating functions, called wavelets, which are localized in time and frequency. The wavelets are obtained by scaling and translating a single, mother wavelet function. Wavelet transforms can be broadly classified into the continuous wavelet transforms (CWT) and discrete wavelet transforms (DWT).

2.3.2 Continuous Wavelet Transform (CWT)

To further understand the working of the wavelet transform, consider $X(t)$ to be a square-integrable function (the integral of the square of its absolute value is finite). Its continuous wavelet transform is given by the inner product

$$w(a,\tau) = \int_{-\infty}^{\infty} X(t)\psi^*_{a,\tau}(t)dt, \tag{2.10}$$

where * indicates complex conjugation and

$$\psi_{a,\tau}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-\tau}{a}\right), a \in \Re^+, \tau \in \Re \tag{2.11}$$

are the basis functions of the transformation, called wavelets. $\Re$ indicates the space of real numbers. The variables a and τ are the scaling and translating parameters respectively. The wavelets $\psi_{a,\tau}(t)$ are generated from the mother wavelet $\psi(t)$ is obtained by scaling and translation.

In the above equations, there is no specific basis function specified for the *mother wavelet*, which as mentioned before is one of the differences with the Fourier analysis. Instead, there is freedom of choice of the mother wavelet within certain constraints that define the behavior of the wavelets. Some of these constraints are discussed below.

In order for the transformation to be invertible, the mother wavelet must satisfy the *admissibility condition*, i.e., the mean value of the mother wavelet must be zero:

$$\int_{-\infty}^{\infty} \Psi(t)dt = 0. \tag{2.12}$$

This implies that the mother wavelet must be an oscillating function. In addition, the mother wavelet has a bandpass frequency spectrum with a zero at the origin.

Another property of the *mother wavelet* is that it must have a number of vanishing moments N, defined as the largest N for which

$$\int_{-\infty}^{\infty} t^k\Psi(t)dt = 0, \text{ k=0,1,...,N-1} \tag{2.13}$$

holds. Every mother wavelet has at least one vanishing moment. This can be seen by using $k = 0$ in Eqn. 2.13 which then gives Eqn. 2.12. The number of vanishing moments of the mother wavelet indicates the smoothness of the wavelet function and the flatness of its frequency response. A higher number of vanishing moments leads to a faster decay rate of the wavelet coefficients, and therefore wavelets with higher number of vanishing moments lead to a more compact signal representation. However, the length of filters used to compute the DWT increases with the number of vanishing moments and the complexity of computing the DWT coefficients increases with the size of the wavelet filters. Thus, there is always a tradeoff on the number of vanishing moments to be selected.

2.3.3 Discrete Wavelet Transform (DWT)

The CWT discussed above is highly redundant. To overcome this problem, the discrete wavelet transform has been introduced which is obtained by sampling the time-scale plane on a dyadic grid: $a = 2^j$, $\tau = 2^j k$, $j \in \mathbb{Z}^+$, $k \in \mathbb{Z}$. j is called *octave* or *scale* and k is *translation*.

The characterizing function for the DWT is a low-pass filter known as the scaling function, ϕ(t). It is obtained by solving the recursive equation

$$\phi(t) = \sum_k c_k \phi(2t - k), \tag{2.14}$$

where $\{c_k\}$ is called the scaling sequence. The scaling function is used to create the mother wavelet function ψ(t) as follows

$$\psi(t) = \sum_k (-1)^k c_{1-k} \phi(2t - k). \tag{2.15}$$

For special choices of the wavelet and scaling functions, the shifted and translated version of (2.14) and (2.15) are given by

$$\begin{aligned} \psi_{j,k}(t) &= 2^{\frac{j}{2}} \psi(2^j t - k) \\ \phi_{j,k}(t) &= 2^{\frac{j}{2}} \phi(2^j t - k) \end{aligned} \tag{2.16}$$

and form an orthonormal basis.

The orthonormal basis in 2.16 can be used to create a multi-resolution representation of any signal $X(t)$, by taking the inner product of $x(t)$ with the wavelet and scaling functions as follows:

$$\begin{aligned} d_{j,k} &= \langle X(t), \psi_{j,k}(t) \rangle \\ a_{j,k} &= \langle X(t), \phi_{j,k}(t) \rangle \end{aligned} \tag{2.17}$$

where $d_{j,k}$ are the detail or wavelet coefficients, and $a_{j,k}$ are the scaling or approximation coefficients.

The signal $X(t)$ can now be represented as

$$X(t) = \sum_{j=0}^{\infty} \sum_{k=-\infty}^{\infty} d_{j,k} \psi_{j,k}(t). \tag{2.18}$$

If the sum over j is split in two regions, $j > J$ and $0 \leq j \leq J$, the above equation takes the form

$$\begin{aligned} X(t) &= \sum_{j=J+1}^{\infty} \sum_{k=-\infty}^{\infty} d_{j,k} \psi_{j,k}(t) + \sum_{j=0}^{J} \sum_{k=-\infty}^{\infty} d_{j,k} \psi_{j,k}(t) \\ &= \sum_{k=-\infty}^{\infty} a_{J,k} \phi_{J,k}(t) + \sum_{j=0}^{J} \sum_{k=-\infty}^{\infty} d_{j,k} \psi_{j,k}(t). \end{aligned} \tag{2.19}$$

The first term in the above equation represents an approximation of the signal at the scale J and the second term is a sum of details. Thus the signal X(t) is now represented by a collection of details at different resolutions and a low level approximation.

For our study we have used the Wavelet toolbox in Matlab [1] to compute the wavelet coefficients. The mother wavelet that we have used belongs to the family of Daubechies wavelets. The mother wavelets are referred to as dbN, where db indicates that the wavelet belongs to the Daubechies family and N is a positive integer indicating the number of vanishing moments for the mother wavelet. Thus, $db3$ indicates that the mother wavelet belongs to the Daubechies family and has 3 vanishing moments.

2.3.4 DWT of LRD processes

Network traffic data has been shown to possess LRD in LAN [47], WAN [61], VBR [9], and other kinds of data traffic. In the case of LRD data, traditional statistical techniques cannot be used as shown in [8]. The DWT is very useful for studying this kind of data set. This is due to the fact that DWT de-correlates the long memory data. It has been shown in [22, 82] that the covariance function of the wavelet coefficients behave as

$$\langle d_{j,k}, d_{j',k'} \rangle \sim O\left(|2^j k - 2^{j'} k'|^{2(H-M)}\right), \tag{2.20}$$

where H is the Hurst exponent of the data set. Eqn. 2.20 indicates that the correlation structure of the wavelet coefficients is not LRD, even though the original data $x(t)$ has LRD. This de-correlation of LRD data is illustrated in

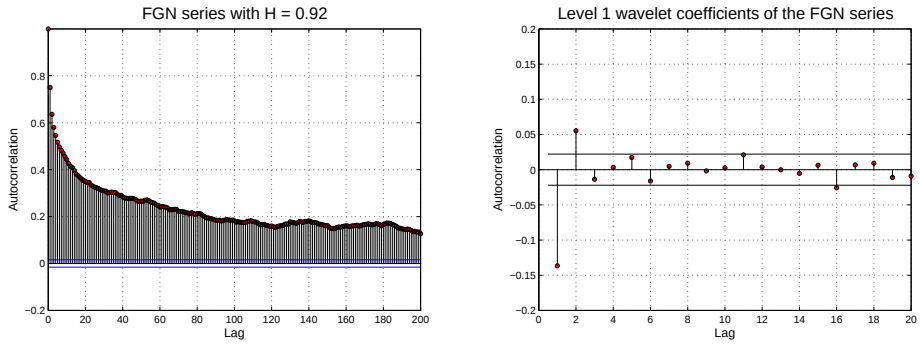

Figure 2.1: Decorrelation of LRD data in wavelet domain.

Fig. 2.1 which shows the correlation structure of a FGN data with $H = 0.92$ and the correlation plot for the wavelet coefficients at scale 1 obtained by using the $db6$ wavelet. As can be seen, the wavelet coefficients at scale 1 have significant correlation only up to a lag of 2, whereas the original data set has significant correlations even up to lag of 200. Similar results are observed for wavelet coefficients at other scales as well. In addition, we tested with

FARIMA data sets, which possess SRD as well as LRD, and the wavelet coefficients are seen to only possess SRD. This demonstrates that the LRD is broken down in the wavelet domain. In addition, it is also shown that the wavelet coefficients are wide-sense stationary [22]. Once the data is decorrelated, traditional statistical techniques can be used.

From (2.20), it is also seen that the correlation structure is controlled by the number of vanishing moments of the wavelet. It has been shown that for $M \geq H + 0.5$, the wavelet coefficients across scales and within the same scale are approximately de-correlated [84]. Abry and Veitch propose using a mother wavelet with at least 3 vanishing moments, i.e. $M = 3$ to obtain sufficient decorrelation between the wavelet coefficients within the same level and across levels. For a FGN data set, the wavelet coefficients at the same scale are normally distributed with zero mean, and when the number of vanishing moments are sufficiently high, the wavelet coefficients at the same scale are independent.

In [4], it has been shown:

$$log_2\left(\frac{1}{n_j}\sum_k |d_{j,k}|^2\right) = (2H-1)j + \hat{c}_f. \tag{2.21}$$

This indicates that the variance of the wavelet coefficients at each level is related to the Hurst value of the parent series. This result is useful for the change detection algorithm discussed in Chapter 5.

We finally conclude this chapter by drawing attention to how the LRD in the time domain is broken down in the wavelet domain by using DWT. This is a useful property that we use repeatedly through this thesis. In addition, the FGN technique of modeling LRD traffic, discussed in Section 2.2.2.1 is also used repeatedly through our thesis.

Chapter 3

Estimation of the Hurst parameter

3.1 Hurst Estimators

In this section, we discuss the most commonly used techniques for estimating the Hurst parameter. For all the techniques discussed below X_t is the original time series for which we are trying to estimate the Hurst parameter.

3.1.1 Time-domain estimators

3.1.1.1 R/S Estimator

Consider the partial sum of the series, i.e.

$$Y(n) = \sum_{t=1}^{n} X_t$$

Let

$$S(n) = \sqrt{\frac{1}{n}\sum_{t=1}^{n} X_t^2 - \left(\frac{1}{n}\right)^2 Y(n)^2}$$

denote the standard deviation of the sample.

Let

$$R(n) = \max_{0\le t\le n}\left(Y(t) - \frac{t}{n}Y(n)\right) - \min_{0\le t\le n}\left(Y(t) - \frac{t}{n}Y(n)\right)$$

Then the fraction $Q(n) = \frac{R(n)}{S(n)}$ denotes the rescaled adjusted range or the R/S statistic.

For fractional Gaussian noise or fractional ARIMA, as $n \to \infty$ we have:

$$E[Q(n)] \sim C_H n^H,$$

where C_H is a positive, finite constant not dependent on n.

To determine H using the R/S statistic, the following method is used [47, 81]:

1. Given a time series X_t of length N, the whole series is subdivided into K non overlapping blocks, each of size N/K.
2. For a number of values of n, compute the rescaled adjusted range Q(n) starting at points k_i = (iN/K) +1, i=1,2,... such that $k_i + n \leq N$. For each value of n, we obtain a number of R/S Samples, say K. This number K decreases for larger values of n because of the limiting condition on k_i values mentioned above.
3. Choose logarithmically spaced values of n, i.e. $n_{l+1} = mn_l$ with $m > 1$, starting with n_0 of about 10 and plot log Q(n) versus log n. This plot is called the Pox plot for R/S statistic.
4. The parameter H can be estimated by fitting a least squares line to the points in the Pox plot.
5. Short range dependence results in a transient zone at the low end of the plot and hence low end of the plot is not used for estmating H, whereas at the higher end there are very few points on the plot to make it reliable and hence they should also not be used. The values of n that lie between the lower and higher cut-off points are used to estimate H.

3.1.1.2 Aggregated Variance

1. Divide the original series into blocks of length m and compute the sample average within each block.

$$X_k^{(m)} = \frac{1}{m} \sum_{t=(k-1)m+1}^{km} X_t, \tag{3.1}$$

   k = 1,2,....,[N/m]

2. For each m, compute the sample variance of $X_k^{(m)}$as:

$$s_m^2 = \frac{1}{N/m} \sum_{k=1}^{N/m} (X_k^{(m)})^2 - \left(\frac{1}{N/m} \sum_{k=1}^{N/m} X_k^{(m)} \right)^2$$

3. Repeat this procedure for different values of m with the value being equidistant on a log scale.
4. Plot log s_m^2 against log m.
5. For sufficiently large values of m, the slope estimates 2H -2 [47, 81]. The slope is estimated by fitting a least squares line to the points obtained from the plots.

3.1.1.3 Variance of residuals

1. Series is divided into blocks of size m. Within each block, the partial sums of series are calculated as:

$$Y(k)^{(m)} = \sum_{t=(k-1)m+1}^{km} X_t,$$

   k = 1,2,.....,[N/m]

2. Fit a least squares line to the partial sums within each block.

3. Compute the sample variance of the residual $s_r^{(m)}$. The variance of the residuals is proportional to m^{2H} for large m and for finite variance LRD series. This variance is computed for each block and the average is computed over the blocks.

4. Plot log $s_r^{(m)}$ vs log m, and this should give a line with slope of 2H [47, 81].

3.1.1.4 The Absolute Moments Method

Consider the series of the averages defined in (3.1), and compute its n^{th} absolute moment as:

$$AM_n^{(m)} = \frac{1}{[N/m]} \sum_{k=1}^{[N/m]} \left| X_k^{(m)} - \bar{X} \right| \tag{3.2}$$

$AM_n^{(m)}$ is asymptotically proportional to $m^{n(H-1)}$. To find an estimate for H

1. Compute $AM_n^{(m)}$ for different values of m.

2. Plot it in a log-log plot against m.

3. The point should be scattered along a line with slope n(H-1) [47, 81].

3.1.1.5 Detrended Fluctuation Analysis (DFA)

The method of Detrended Fluctuation Analysis (DFA) was introduced by Peng et. al. in [62]. It was introduced as a way of measuring LRD behavior for non-stationary time series signals. The method is described as follows:

1. Let x(t) be a 1-D stochastic process. Define the integrated signal y(k) as follows

$$y(k) - \sum_{i=1}^{k} x(i) - \mu, \tag{3.3}$$

   where μ is the mean of x(t).

2. Divide the integrated time series y(k) into boxes of equal length n. Find a least-squares line that fits the data in each box of length n.

3. y(k) is detrended by subtracting the local trends $y_n(k)$ as shown:

$$F(n) = \left[\left(\frac{1}{N} \right) \sum_{k=1}^{N} (y(k) - y_n(k))^2 \right]^{\frac{1}{2}} \tag{3.4}$$

4. The above computation is repeated across a broad range of scales to characterize the relationship between the box size n and the average root-mean-square fluctuation function $F(n)$. A power-law relationship between them indicates the presence of scaling given by $F(n) \sim N^{\alpha}$, which means that the process obeys the scaling law characterized by the scaling exponent α.

5. α corresponds to the value of Hurst parameter, and thus represents the LRD of the signal.

The main advantage of the DFA method lies in its applicability to non-stationary LRD time series, and has also been shown to be superior in estimating degree of LRD than conventional tools like the R/S analysis [90].

3.1.2 Frequency-domain estimators

3.1.2.1 Periodogram

The periodogram is defined as

$$I(\nu) = \frac{1}{2\pi N} \left| \sum_{t=1}^{N} X_t e^{-it\nu} \right|^2 \tag{3.5}$$

where $i = \sqrt{-1}$, ν is the frequency, and N is the length of the series.

As stated in Section 2.1, a series with LRD will have spectral density proportional to $|\nu|^{1-2H}$ for frequencies close to the origin.
Thus, if we plot the log-log plot of the periodogram versus the frequency, it will display a straight line with a slope 1-2H [47, 81].
For this method, we should only use the lowest 10% of the frequencies for plotting the line since the proportionality only holds for ν close to the origin.

3.1.2.2 Whittle's Maximum Likelihood Periodogram

The Whittle estimator is also based on the periodogram [81]. Consider the function:

$$Q(\eta) = \int_{-\pi}^{\pi} \frac{I(\nu)}{f(\nu, \eta)} d\nu \tag{3.6}$$

where $I(\nu)$ is given by (3.5)
$f(\nu, \eta)$ is the spectral density function
η is the vector of unknown parameters.

The Whittle estimator is the value of η which minimizes the function Q in (3.6).

In practice the function which the algorithm tries to minimize is:

$$Q(\eta) = \sum_{j=1}^{[(N-1)/2]} \frac{I(\nu_j)}{f(\nu_j, \eta)}$$

For fractional Gaussian noise or FARIMA(0,d,0), η is the H or d parameter respectively. For FARIMA(p,d,q), η also includes the unknown coefficients in the autoregressive and moving average parts.

The main drawback of this method is that it assumes that the parametric form of the spectral density is known which is very rarely the case.

3.1.3 Wavelet-domain estimators

3.1.3.1 Abry-Veitch Estimator

This method was proposed in [84] and the steps for estimating the parameter H using this approach are as follows:

1. For each *scale* j and *position* k, compute the so-called wavelet detail coefficients d(j,k) as

$$d(j,k) =< X_t, \Psi_{j,k}(n) >= \sum_{n=1}^{\infty} X_t \Psi_{j,k}(n) \tag{3.7}$$

   where $< .. >$ in the above equation indicates the inner-product of the series with the function $\Psi_{j,k}$ and

$$\Psi_{j,k}(n) = 2^{-j/2}\Psi_0(2^{-j}n - k)$$

   Ψ_0 is the (Daubechies) mother wavelet.

2. The coefficient $|d(j,k))|^2$ measures the amount of energy in the analyzed signal about the time instant $2^j k$ and frequency $2^{-j}\nu_0$, where ν_0 is an arbitrary reference frequency selected by the choice of Ψ_0. Compute the wavelet energy μ_j for each scale j as

$$\mu_j = \frac{1}{N_j} \sum_{k=1}^{N_j} |d(j,k)|^2 \tag{3.8}$$

where N_j is the total number of wavelet coeffients at scale j. Essentially $N_j = 2^{-j}N$ where N is the length of the data. (3.8) is a measure of the amount of energy that lies within a given bandwidth around the frequency ν and can therefore be regarded as a statistical estimator for the spectrum of series X_t.

3. Make a plot of $log_2(\mu_j)$ versus scale j and then plot a least squares line through the points. While plotting the line, we have to neglect the points in the lowest part of the plot as well as some points in the higher range as there will be very few transformed wavelet coefficients in this region and the estimation of Hurst parameters using those points will be quite noisy. The slope of this line will be $2\hat{H}$-1. The range of scales over which the linear fit is considered valid is denoted as $[j_1, j_2]$

4. The confidence interval for the parameter is given by:

$$\hat{H} - \sigma_{\hat{H}} z_\beta \leq H \leq \hat{H} + \sigma_{\hat{H}} z_\beta \tag{3.9}$$

where the variance is given by:

$$\sigma^2_{\hat{H}} = var\hat{H}(j_1, j_2) = \frac{2}{n_{j_1} ln^2 2} \frac{1 - 2^J}{1 - 2^{-(J+1)}(J^2 + 4) + 2^{-2J}}$$

where $J = j_2 - j_1$ is the number of octaves involved in the linear fit and $n_{j_1} = 2^{-j_1} n$ is the number of available coefficients at scale j_1.
In (3.9), z_β is the 1 - β quantile of the standard Gaussian distribution.

3.2 Drawbacks of Hurst parameter estimators

In this section, we discuss various drawbacks of the Hurst estimators discussed in the above section.

3.2.1 Single Estimate

All the estimators described in Section 3.1 give an estimate of the Hurst parameter for the complete time series. These estimators do not give any indication of change in the Hurst parameter. However, it is expected that in real life, the Hurst parameter of the aggregated traffic will not remain constant, instead it will vary with time, depending upon the traffic content and traffic intensity [85]. To demonstrate that in such a scenario it is misleading to look at only the Hurst estimates, we generated a time series of length 32768 points where the first 16384 points are from a FGN series of Hurst value 0.5777 and the next 16384 points are from a FGN series with Hurst value of 0.8793. On running the Abry-Veitch estimator on the resultant series, we get a Hurst parameter of 0.7730.

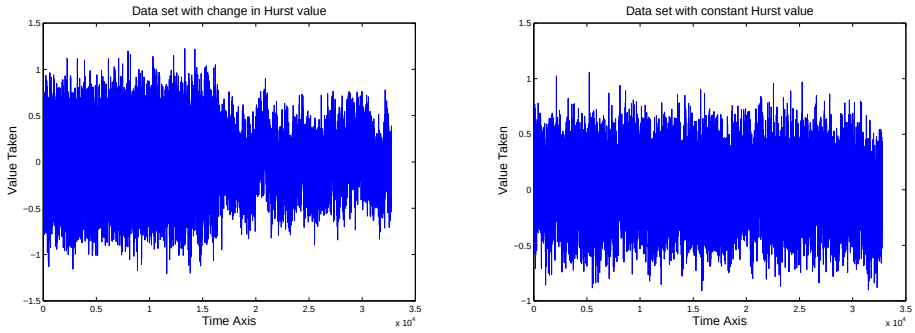

Figure 3.1: Time series plots for the two data sets of length 32768 used for our experiments. The first one has a change in the Hurst parameter at the center of the series and the second has a constant Hurst parameter.

Next, we generate another FGN series with 32768 points and a Hurst value of 0.7793. The time series plot for both the data sets are shown in Fig. 3.1. If we compare the Hurst values of these two series, it will appear that both the data sets are equivalent. However, that is not the case. Thus, it can be seen that the estimators discussed above have a serious drawback in the sense that they give a single estimate for the entire series without accounting for a change within the series.

The way to get around this issue is to first test if the Hurst parameter is constant over the data set. One of the ways to achieve this is to study the multiscaling behavior of the data set. This is achieved by plotting a Linear Multiscale Diagram (LMD) representing the scaling behavior as a function of scaling order. In order to study this behavior we have used the Multiscaling tool developed by Veitch et. al. [86]. We demonstrate the use of this Multiscaling tool in Fig. 3.2 where we plot the Multifractal spectrum for the two series discussed above. From Fig. 3.2, it is clear that the Hurst value is not constant for the first series. However, this method does not indicate the location of the change in the Hurst parameter. We also need to know the location of the change so that the series can then be broken down into sections over which the Hurst parameter can be considered to be constant. We discuss such techniques in Chapter 5.

3.2.2 Impact of non-stationarities

In addition to the drawback of giving a single estimate for the entire series, the Hurst estimators discussed in Section 3.1 also suffer from presence of non-stationarities. It has been shown that non-stationarities have an adverse effect on these estimators [38, 39, 52, 81]. In the sections below, we conduct some tests to demonstrate the impact of non-stationarities on the Hurst parameter estimators discussed in Section 3.1.

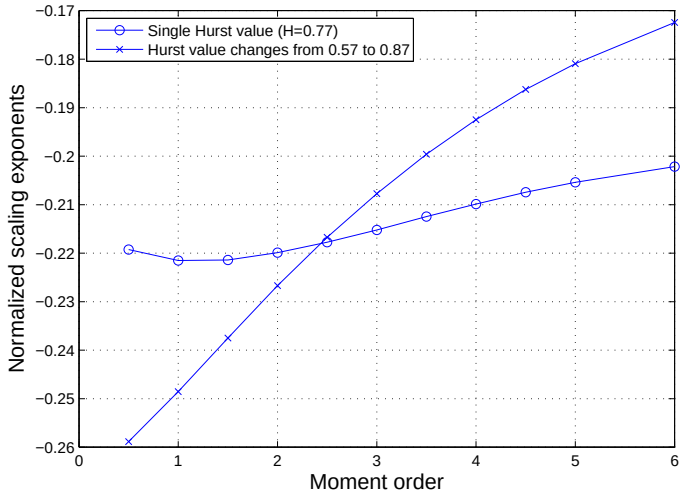

Figure 3.2: Comparison of Linear Multifractal Spectrum of two series with similar Hurst values.

3.2.2.1 Changing Mean

In order to study the impact of changing mean of the series, we generated several FGN data sets of length 16384 with Hurst values ranging from 0.57 to 0.92 in steps of 0.05. From each series, we generate a new series of length 32768 with the first part of the series (16384 points) identical to the original series, and the second part of the series also identical to the original series but with an increase in mean of 100. Thus, the Hurst value for the entire series is constant, but there will be a step change in the mean of the series at time instant 16385. Next, we use the SELFIS tool [37] to get the estimated Hurst values for 7 different estimators; viz. R/S, Aggregated variance, Variance of residuals, Absolute moments, Periodogram, Whittle's estimator and Abry-Veitch estimator. However, for the Abry-Veitch estimator, we do not use the results reported by this tool, but use the results obtained from the Matlab code provided by the authors as that allows us to control the number of vanishing moments and the scale over which the estimation is carried out. We have tabulated the estimated Hurst values for these series for the different estimators in Table 3.1.

From Table 3.1, it can be seen that a change in mean by 100 results in errors for almost all the estimators. Only the Whittle and the Abry-Veitch (Wavelet) estimator are able to give appropriate estimates. However, these two estimators fail when the change in mean is of a higher magnitude. This is clear from Table 3.2 which tabulates the Hurst estimates for series similar to those used for the previous case, except that the second part of the series now has a change in mean of 1000.

As can be seen in Table 3.2, all the estimators fail to give appropriate results. This is a serious drawback of the estimators, as it is expected that in

Estimation Method	Actual H Value							
	0.57	0.62	0.67	0.72	0.77	0.82	0.87	0.92
R/S	0.598	0.637	0.675	0.712	0.746	0.776	0.802	0.822
Absolute Moments	0.755	0.751	0.746	0.741	0.736	0.732	0.731	0.738
Variance of Residuals	0.881	0.888	0.895	0.902	0.910	0.918	0.931	0.954
Aggregate Variance	0.932	0.927	0.923	0.919	0.915	0.913	0.915	0.922
Periodogram	1.037	1.062	1.088	1.113	1.138	1.164	1.189	1.214
Whittle	0.631	0.659	0.693	0.733	0.777	0.824	0.873	0.922
Wavelet	0.594	0.639	0.685	0.732	0.781	0.830	0.880	0.929

Table 3.1: Estimated Hurst Value for FGN series with change in mean of 100.

Estimation Method	Actual H Value				
	0.57	0.67	0.77	0.87	0.92
R/S	0.475	0.553	0.622	0.367	0.368
Absolute Moments	0.773	0.773	0.773	0.692	0.692
Variance of Residuals	1.133	1.151	1.173	1.456	1.483
Aggregate Variance	1.017	1.017	1.016	1.019	1.019
Periodogram	1.036	1.087	1.137	1.187	1.212
Whittle	0.838	0.862	0.898	0.999	0.999
Wavelet	0.7796	0.8189	0.8754	0.9500	0.9977

Table 3.2: Estimated Hurst Value for FGN series with change in mean of 1000.

aggregate traffic traces, the mean might change based on time of day. As a result, it is necessary to detect points where the mean changes. Methods have been proposed for detecting changes in mean in the presence of long-range dependence in [7] and [79].

In [12], the authors propose the use of scale-space techniques to provide a visual display of the goodness-of-fit of an assumed model to the data set. The advantages of this method are that it allows studying time series that will exhibit variations at different scales by carrying out the tests by dividing the data into separate bins, and changing the bin size to change the scale of study. It shows the difference between the assumed model and the synthetic traffic at different scales and across time. The change is detected by checking if the derivative of the smoothed curve is significantly different from zero. The null model that the authors use for the SiZer technique is white Gaussian noise. Since for Internet traffic analysis, white Gaussian noise is not relevant, an improvement has been proposed in [58]. In this technique, the user specifies

the correlation structure of the model that will be used as the base for testing the data set. The tool then tests for changes in the mean of the given data set on the basis of the specified model. This is a useful tool as it was seen from Tables 3.1 and 3.2 that changes in the mean lead to inaccurate estimates of the Hurst parameter, and thus detecting if there is any change in the mean of the given series would help to obtain reliable estimates. We demonstrate the use of this tool by using a FGN data set with a constant mean and a Hurst value of 0.92. Fig. 3.3 gives the Dependent Sizer plot for this series. In the

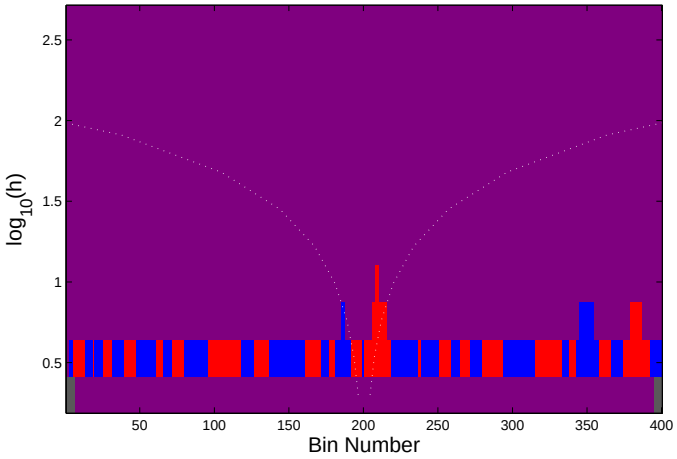

Figure 3.3: Using SiZer to test for a change in the mean of a FGN series with Hurst value of 0.92.

plot, the x-axis represents the bin number and the y-axis corresponds to the logarithm of the bandwidth of the family of smooths. Each pixel in the plot indicates a result of a hypothesis test for the slope of the curve at that point. A red (black) colour indicates that the slope is positive, while a blue (white) color indicates that the slope is negative. When the slope is not significant, a purple (grey) colour is used. Most part of Fig. 3.3 is purple (grey) which indicates that the mean of the series can be considered to be constant.

Next, we use a data set that has a Hurst value of 0.92, with the first 16384 points having a mean of zero and the next 16384 points having a mean of 1000. As was seen in Table 3.2, all the estimators fail to give an accurate estimate for this data set. The Dependent Sizer plot for this case is shown in Fig. 3.4. The model supplied for creating the Dependent Sizer plot is a FGN series with a Hurst value of 0.9977 (estimated Hurst value by using the wavelet estimator as seen in Table 3.2). From Fig. 3.4, it can be seen the data set is flagged to be significantly different from the model supplied. This indicates that the used data set was not generated by a process with constant Hurst value and/or constant mean. By looking at the figure, it is not possible to deduct whether the mean or the Hurst parameter of the process has changed. One way of going about this is to plot the multifractal spectrum for the data

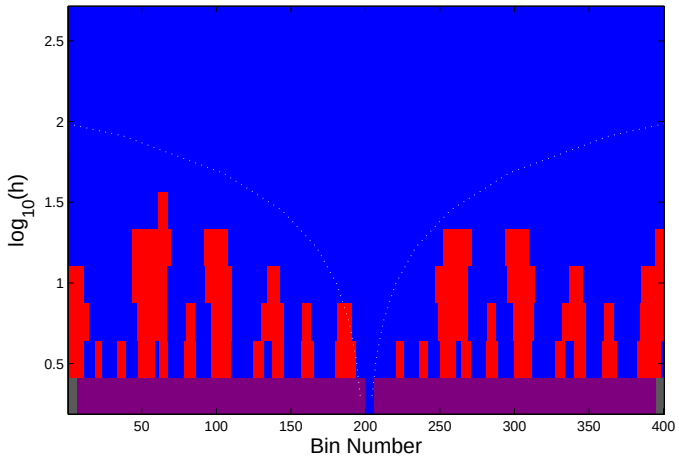

Figure 3.4: Using SiZer to test for a change in the mean of a FGN series with Hurst value of 0.92 and change in mean from 0 to 1000.

set and compare it with the multifractal spectrum for a data set with constant Hurst parameter. Such a plot is shown in Fig. 3.5, where we plot the linear multifractal spectrum for the data set with changing mean and for a data set of the same length, but constant mean. From Fig. 3.5, we can conclude that

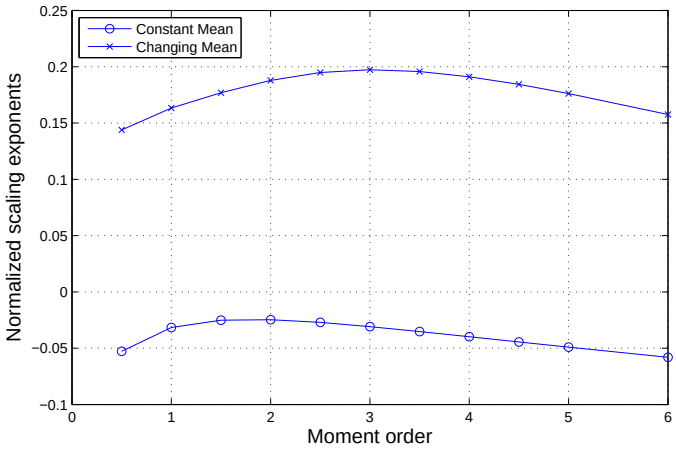

Figure 3.5: Comparison of Linear Multifractal Spectrum of two series with similar Hurst values, but different means.

the linear multifractal spectrum for both the series is nearly similar, and hence it can be concluded that the Hurst value for the data set that we are testing is constant. Thus, on the basis of Fig. 3.4 and Fig. 3.5, it can be concluded that the Hurst value for the data set has remained constant, but the mean has changed, as expected.

After conducting several experiments with many other data sets, we came

up with the following series of steps to be employed for estimating the Hurst parameter of the series:

1. Run one time-domain estimator; say the R/S estimator, the Whittle estimator and the Wavelet estimator. If these estimators give estimates that are significantly different from each other, then go to next step, or else use the average of the three estimates.

2. Plot the linear multifractal spectrum (LMS) of the series, and compare it with the LMS of a series of the same length with constant mean and constant Hurst value.

3. Obtain the Dependent Sizer plot for the data set being tested.

4. On the basis of the two above plots, make a decision on whether the data set being tested can be assumed to be generated by a process with constant Hurst value and constant mean.

5. If it is found that either the mean or Hurst value or both are changing, then further investigation will be needed to figure out the location of change.

3.2.2.2 Impact of Short-Range Dependence

In addition to the non-stationarities discussed above, Internet data also contains SRD coexisting with LRD [15, 75, 76]. In this section, we discuss the negative impact of SRD on the Hurst estimators.

It is shown in [33] that a FARIMA(p,d,q) process has similar LRD behavior to a FARIMA(0,d,0) process with the same value of d. Since H is only related to d, the presence of autoregressive (AR) and moving average (MA) components in the FARIMA model should not impact the estimated H value.

The data sets used in this section were generated by using the "fracdiff" package of R [65] and consist of 65536 data points in each series. The estimates of the Hurst parameter were obtained by using the SELFIS tool discussed earlier.

For the first set of tests, we generated FARIMA(1,d,1) series with the AR and MA components being 0.2 and -0.4 respectively. The estimates obtained by the different estimators for a range of value of actual H are given in Table 3.3. From Table 3.3 it can be seen that the estimates obtained in the presence of SRD are different from the actual H values. The worst estimates are obtained by the *Whittle* and *Wavelet* estimator giving approximately 34% and 39% mean difference respectively from the actual values. In this case, the *Periodogram* method performs the best giving only 2.9% mean difference.

Next, we generate FARIMA(1,d,1) series with the AR and MA components being 0.4 and -0.4 respectively, and the estimates are tabulated in Table 3.4. From Table 3.4 it is seen that the *Variance of Residuals* method performs the best giving a 5.51% mean difference from the actual values. The *Periodogram*

Estimation Method	Actual H Value					
	0.55	0.6	0.7	0.8	0.9	0.95
R/S	0.607	0.629	0.708	0.783	0.820	0.844
Absolute Moments	0.546	0.482	0.671	0.696	0.802	0.807
Variance of Residuals	0.592	0.584	0.729	0.836	0.922	0.962
Aggregate Variance	0.570	0.507	0.701	0.716	0.842	0.850
Periodogram	0.574	0.630	0.728	0.819	0.914	0.953
Whittle	0.908	0.947	0.999	0.999	0.999	0.999
Wavelet	0.860	0.903	0.984	1.072	1.159	1.200

Table 3.3: Estimated Hurst Value for FARIMA(1,d,1) series, AR=0.2, MA=-0.4.

Estimation Method	Actual H Value					
	0.55	0.6	0.7	0.8	0.9	0.95
R/S	0.599	0.666	0.710	0.781	0.788	0.772
Absolute Moments	0.531	0.648	0.583	0.750	0.784	0.803
Variance of Residuals	0.592	0.704	0.710	0.835	0.892	0.963
Aggregate Variance	0.552	0.672	0.613	0.784	0.828	0.852
Periodogram	0.599	0.651	0.745	0.850	0.946	1.001
Whittle	0.999	0.999	0.999	0.999	0.999	0.999
Wavelet	0.975	1.015	1.104	1.196	1.271	1.315

Table 3.4: Estimated Hurst Value for FARIMA(1,d,1) series, AR=0.4, MA=-0.4.

method gives a mean difference of 6.76% from the actual values and is the second best estimator in this case.

We conducted additional experiments by varying the AR and MA parameters for fixed values of d=0.3, p=1, q=1 in order to determine if there is any relationship between the reliability of the estimators and the AR and MA parameters. The estimates obtained for different values of AR and MA parameters are given in Table 3.5. From Table 3.5, it can be seen that as the absolute difference between the AR and MA parameters increases, the accuracy of the estimators reduces. In addition, it is also observed that all the estimators give a much higher estimate when the MA parameter is negative as compared to the estimates when the MA parameter is positive.

On the basis of the above experiments, it can be concluded that if the series has SRD in addition to LRD, then most of the estimators are found to

Estimation Method	SRD Parameter Values				
	AR=0.3 MA=0.4	AR=0.2 MA=0.4	AR=0.2 MA=-0.4	AR=0.3 MA=-0.4	AR=-0.3 MA=0.4
R/S	0.753	0.768	0.783	0.785	0.734
Absolute Moments	0.665	0.670	0.696	0.711	0.692
Variance of Residuals	0.724	0.755	0.836	0.807	0.729
Aggregate Variance	0.688	0.687	0.716	0.747	0.714
Periodogram	0.782	0.766	0.819	0.847	0.755
Whittle	0.683	0.632	0.999	0.999	0.500
Wavelet	0.738	0.674	1.072	1.128	0.400

Table 3.5: Estimated Hurst Value for FARIMA(1,0.8,1) series. Estimation accuracy dependent on difference between AR and MA values.

be highly inaccurate. In addition, it was also seen that in the presence of SRD, there is no single estimator that consistently gives the best estimates for the Hurst parameter. It can also be seen that the reliability of the estimators is dependent on the difference between the AR and MA values. Consequently, if these estimators are used to get estimates of the Hurst parameter in Internet traffic, they may give unreliable estimates and the amount of LRD in the traffic may be incorrectly estimated. In order to avoid this problem, it is possible to filter out the SRD component from the series before estimating the H value. One such technique is to shuffle the series [18, 37], so that the SRD is removed and the LRD is preserved. We tried to employ this technique, but did not achieve any success with it. An alternative approach is to develop new LRD estimation techniques that are resilient to the presence of SRD in the series. We have not found any such techniques in literature and it still remains an open problem.

Chapter 4

Traffic Trace Generation using Bootstrapping

4.1 Approaches for Traffic Trace Generation

As discussed in Section 2.2 the most commonly employed technique for generating traffic traces is the two step procedure of fitting a stochastic model to the observed traffic and then using it to generate traffic traces during simulation. However, the step of model fitting requires user intervention and can be complicated. For instance, for FARIMA models with alpha stable innovations, there are a number of parameters that need to be estimated using complicated estimation procedures spanning multiple steps with continuous user involvement [30]. Using simple models like the FGN model [60], allow the LRD to be matched, but not higher order properties or the marginal distribution. In this Chapter, we propose a technique for generating aggregate traffic traces that does not involve any model fitting, thereby avoiding the complexity of parameter estimation.

4.2 Bootstrapping

The standard bootstrap procedure was introduced in [16] for approximating the sampling distribution and the variance of many statistics under the assumption of independent and identically distributed (i.i.d.) data. The idea of bootstrapping is to re-sample the original data with replacements to obtain a new series. The procedure is repeated a number of times to obtain multiple datasets (which are known as the surrogate series) from the original data. However, when the observations are not independent, the standard bootstrap scheme fails to capture the dependency structure of the data [21]. A number of variants of the original bootstrapping technique have been proposed to address such dependencies in the data. Some examples are the residual bootstrap, sieve bootstrap, moving block bootstrap, stationary bootstrap, threshold bootstrap, etc. (see [21] and the references therein). All the above bootstrap techniques

work for SRD data but not for LRD data.

4.2.1 Bootstrapping for LRD Time Series

To deal with LRD datasets, it has been proposed to transform the data into another domain, by using either the Fourier Transform or Wavelet Transform [63, 83]. In [6], the authors demonstrate that bootstrapping in the wavelet domain (also referred as wavestrapping) works better than in the Fourier domain when attempting to capture the characteristics of non-Gaussian long-memory processes. Aggregate data traffic also falls in this category of data, and hence we employ the Wavelet transform for our case.

4.3 Bootstrap Based Algorithm for Generating Traffic Traces

In this section, we discuss the algorithm that we have proposed for generating traffic traces possessing LRD by using bootstrapping in the wavelet domain. In addition, we also discuss previous work which is related to our approach.

4.3.1 Related Work

As stated previously, the general trend for generating aggregate traffic traces is by fitting a model to the observed data, and then generating traces from this model to be used for simulation. The most common model used for this purpose is the FGN model [35, 46, 60]. Like our proposed algorithm, the process of generating aggregate traffic traces can be automated when using the FGN model. However, the FGN model is not useful for most real traffic traces since they are usually non-Gaussian, whereas our algorithm will be shown to work well for real traffic traces. Traces generated by using the alpha-stable models [25, 29, 40, 93] or FARIMA models with alpha-stable distributions [30] have been shown to perform better than FGN models in capturing the characteristics of Internet data. However, using these models involves complicated parameter estimation techniques requiring significant time and user intervention. In contrast, as will be demonstrated in Section 4.4, traces generated by our technique are able to capture the characteristics of real traces, without any complex parameter estimation process. One model which does not involve very complex parameter estimation is the multifractal wavelet model (MWM) that has been proposed for efficient synthesis of non-Gaussian LRD traffic [68]. This method involves the use of a multiplicative cascade coupled with the Haar wavelet transform. This method ensures the generation of positive output, thereby making it appropriate for traffic modeling which always contains positive data. However, it will be demonstrated in Section 4.4 that our algorithm significantly outperforms this model for real traffic traces in terms of the multifractal and queueing behavior.

One approach for generating aggregate traffic traces without using any specific model has been proposed in [95], where the authors propose a feedforward neural network architecture for generating aggregate self-similar traffic traces. However, the proposed technique involves a learning phase where the number of samples required for training depends on the traffic pattern that is being modelled, with a possibility of the network performance diverging during training, instead of converging. Thus, the procedure is quite complicated and requires a time consuming training phase. This is avoided in our algorithm.

A tool called RAMP, proposed in [43] can be used for the automated generation of traffic traces from live network measurements. However, it is used for source-level modelling of Web and FTP traffic, and thus cannot generate aggregate traffic traces. In addition, this method will also not work with most of the publicly available traces, in which only the aggregate traffic per time slot is available. Our algorithm on the other hand, can be used to generate the traffic traces from direct traffic measurements as well as publicly available traces (as will be demonstrated in Section 4.4).

The use of bootstrapping in the wavelet domain (referred to as wavestrapping), as used by us was first proposed in [63]. It was subsequently employed in [11] for testing the nonlinearity of data sets. However, neither of these two papers on wavestrapping deal with LRD data.

The use of bootstrapping for LRD data was addressed in [27]. In this paper, the authors propose the use of post-blackening moving block bootstrap to generate surrogate datasets for testing the effectiveness of some commonly used estimators of the Hurst parameter. However, in this paper the bootstrapping technique is applied in the time domain, rather than in the wavelet domain as we have proposed here. As a result, the AR (auto regressive) model fitted to the data in [27] has a very high order, making it hard to implement. In addition, the block size selected also needs to be high enough to capture the dependency in the data, making the selection of the block length difficult.

The closest resemblance to our work is seen in [21], in which the authors use the residual bootstrap technique in the wavelet domain to create multiple surrogate series for LRD data. Their algorithm involves fitting a Markov model to capture the SRD in the wavelet domain, and then modeling the residuals by Efron's i.i.d bootstrap [16] to generate the bootstrap residuals. The bootstrap samples of the residuals are then combined with the Markov model to generate the bootstrap wavelet coefficients, and this model is then used to produce bootstrap datasets in time domain. The authors demonstrate that their technique can be used to estimate the sample unit lag autocorrelation and standard deviation for Gaussian datasets. Another similar technique is discussed in [49], where the authors propose the use of an independent model for the wavelet coefficients and capture the variance of these coefficients at each level of decomposition. They demonstrate that their technique is able to capture the autocorrelation function and the queue loss rate for heterogenous traffic possessing both LRD and SRD. In our algorithm, we do not fit any model to the wavelet coefficients, but use the resampling technique to obtain

the new wavelet coefficients. In addition, we are interested in capturing the Hurst parameter, multifractal scaling and the queuing behavior for Gaussian and non-Gaussian data sets, as will be demonstrated in Section 4.4.

4.3.2 Description of Algorithm

The time series data is converted into the Wavelet domain by using the DWT, and the bootstrap scheme will be applied to the wavelet coefficients. However, as evident from Fig. 2.1, the wavelet coefficients within a given level have significant correlation up to lag 2. Hence, we cannot use the standard bootstrap mechanism, which has been designed for independent data. In addition, as stated earlier, the wavelet coefficients at each scale are stationary and hence it is necessary to use a bootstrapping scheme that will also produce a stationary series.

Considering the above factors, the stationary bootstrap algorithm [64] is selected to be used on the wavelet coefficients $d_{j,k}$. This bootstrap algorithm operates as follows.

1. Let the original time series be X_1, X_2, ..., X_T (here X_1, X_2, ...X_T indicate the wavelet coefficients at each level of decomposition).

2. Generate a sequence of i.i.d random variables L_1, L_2, ... having the geometric distribution for n = 1, 2, ..., with the density function $P(n) = (1-p)^{n-1}p$.

3. Generate another i.i.d sequence I_1, I_2, ..., I_Q with discrete uniform distribution.

4. The blocks are represented as $B_{I_i,L_i} = \{X_{I_i}, X_{I_i+1}, ..., X_{I_i+L_i-1}\}$, indicating a block containing L_i observations starting from X_{I_i}.

5. In order to achieve stationarity of the re-sampled time series, the data is wrapped around in a circle, so that X_1 follows X_T.

6. The re-sampled time series is them formed by taking a sequence of blocks B_{I_1,L_1}, B_{I_2,L_2}, ... This process is continued until the required number of data points are generated for the re-sampled series.

We describe the algorithm used to create the surrogate datasets from any given dataset. One of the key requirements for creating surrogate datasets from LRD data is to preserve the scaling behavior. It has been pointed out that re-sampling each wavelet scale independently does not destroy the frequency content and the energy cascade [6]. Therefore, we apply the stationary bootstrap scheme independently to each wavelet decomposition scale and then take the inverse wavelet transform to get new datasets. For data possessing long-tailed distribution, the wavelet coefficients will also possess a long-tailed distribution [78]. As a result, if the wavelet coefficients are randomly sampled,

then the long-tailed nature of the wavelet coefficients will be destroyed and the regenerated series will not possess the long-tailed nature of the parent series; thereby having a different PDF. In order to avoid this, we employ the Box-Cox transformation [10] to convert the original data close to a normal distribution.

The algorithm can be summarized as:

1. Compute the DWT of the dataset to obtain the wavelet and scaling coefficients.

2. Apply the Box-Cox transformation to the wavelet coefficients at each level.

3. Re-sample the wavelet coefficients at each scale via the stationary bootstrap technique. The parameter p defining the geometric distribution is chosen separately for each scale as

$$p_j = 2^{((-log(T_j)/log(2))/3)}, \tag{4.1}$$

   where p_j is value of p at scale j having T_j wavelet coefficients.

   The scaling coefficients obtained at the highest scale of decomposition are not re-sampled.

4. Take the inverse Box-Cox transformation of the resampled wavelet coefficients.

5. Take the inverse DWT of the resampled and transformed wavelet and scaling coefficients to obtain the surrogate dataset.

6. Round off the series obtained in the previous step, and change the negative values to 0 to obtain the required dataset. Since, the scaling coefficients are not re-sampled there is no need to make any adjustments for the mean. The mean of the surrogate series will be close to that of the parent series.

7. Steps 3, 4, 5 and 6 are repeated to obtain as many surrogate datasets as required.

The transformation of the negative values to 0 in Step 6 above might create an artifact in the surrogate data sets. However, after implementation of our algorithm, we found that the percentage of data points having a value of zero ranges from 0.004% to 0.3% which is probably low enough to not cause any impact on the simulations. This observation was made when modeling FGN and the real datasets discussed earlier, with one thousand surrogate series generated for each parent dataset.

There are two variables in our algorithm, for which we need to find appropriate values. The first is the number of vanishing moments of the mother wavelet and the second is the number of levels of decomposition for the DWT.

We determine the values for these variables experimentally by using FGN datasets (generated by using Paxson's algorithm [60]) as the parent dataset.

To find the appropriate number of vanishing moments, we experiment with the db3, db6 and db9 mother wavelets (having 3, 6 and 9 vanishing moments respectively). One thousand surrogate series are created from each FGN series, using each of the three mother wavelets. Table 4.1 gives the Hurst values of the parent and surrogate series using the different mother wavelets. As seen

Hurst Value of Parent Series	Hurst Value of Surrogate Series					
	db3		db6		db9	
	Mean	Variance	Mean	Variance	Mean	Variance
0.5214	0.5210	0.000094	0.5147	0.000093	0.5173	0.000224
0.6244	0.6153	0.000090	0.6170	0.000102	0.6181	0.000086
0.7263	0.7087	0.000100	0.7181	0.000088	0.7175	0.000080
0.8274	0.8010	0.000104	0.8196	0.000094	0.8171	0.000082
0.9282	0.8932	0.000111	0.9204	0.000095	0.9154	0.000081

Table 4.1: Hurst Value of surrogate series with different mother wavelets

from Table 4.1, using the db6 mother wavelet gives better results than the db3 wavelet in terms of the Hurst value. No substantial improvement can be seen with the db9 wavelet, and hence the db6 mother wavelet is selected for our algorithm.

In order to find the appropriate number of decomposition levels, we experiment with the same set of FGN datasets used above, and the db6 mother wavelet. The results indicate that using between seven to ten levels gives enough variability between the surrogate series, while keeping the Hurst parameter of the surrogate series close to the parent series. We use ten levels for our algorithm.

It is possible that the optimum mother wavelet and number of decomposition levels will depend on the type of traffic pattern and the length of the parent series. This will involve conducting multiple experiments and establishing an appropriate criteria for selecting the right values for these variables. We leave this for future work.

4.4 Performance Evaluation of Algorithm

We use a combination of synthetic and real datasets for evaluating our algorithm. The synthetic traces were generated by using the the FGN generator proposed by Paxson in [60]. Each dataset is generated to be of length 16384, with the Hurst value, $H \in [0.52, 0.97]$. The data points generated by the Paxson generator contain fractional as well as negative values. We are interested in representing the aggregate workload (packets or bytes) per time slot, so the data points must be non-negative integers. To achieve this, we add an integer value greater than the absolute of the minimum value of the generated series.

The data points are then rounded to the nearest integer, and this dataset is used in the analysis.

The real datasets are obtained from publicly available trace repositories and a trace recorded at the gateway to our university in 2001. The first two traces belong to the well known set of Bellcore traces analyzed in [47]. From this set, we use the pAug89 and pOct89 traces and generate datasets representing the number of bytes per 12 ms and 10 ms respectively. Next we have the 20030424-000000-0 trace obtained from the Cooperative Association for Internet Data Analysis (CAIDA) [2]. This is a 5-minute trace of packet headers on an OC48 backbone link, made available to us by CAIDA. We created two datasets from this trace; the first containing the number of packets per millisecond for the initial 107622 milliseconds, and the second containing the number of packets per millisecond for the entire trace. The fourth trace was collected on April 09, 2002 at 0300 hours by the Network Data Analysis Study Group at the University of North Carolina at Chapel Hill [3]. This trace contains the number of packets and bytes per millisecond for 732247 milliseconds. From this we created a dataset of the number of packets per 100 milliseconds giving us 73225 data points. The final trace was collected in 2001 on the outgoing link connecting the University of Alberta's campus network to the Internet. The trace is a record of 100000 packets passing the gateway. From this trace, we formed a dataset of length 71391 representing the number of packets per millisecond. The details of the series formed from the traces are summarized in Table 4.2.

Series Name	Data Type	Aggregation Period (ms)	Length of Series
pAug89	Bytes	12	261902
pOct89	Bytes	10	175962
CAIDA	Packets	1	107622
full-length CAIDA	Packets	1	300000
UNC	Packets	100	73225
UofA	Packets	1	71391

Table 4.2: Datasets formed from actual traces

4.4.1 Performance Tests

The criteria we use for evaluation of our algorithm are the Hurst parameter, H, (measuring the second order scaling properties), the probability distribution function (PDF) plots, the Linear Multiscale diagram (used to test the higher order scaling properties) and the queue tail probability.

We compare our algorithm with the FGN model which has been traditionally used for comparison of all traffic models and a multifractal wavelet model that is one of the latest models proposed to capture the multifractal behavior

for real traffic data having a non-Gaussian distribution. The FGN datasets are generated by Paxson's algorithm [60] and the β-multifractal wavelet model (MWM) [68] is used as the multifractal model. For each of the three techniques, we generate one thousand datasets and for comparison we randomly pick one data set generated by each method and run the corresponding tests.

4.4.1.1 Hurst Values

For estimating the Hurst value, we have used the Wavelet based estimator [84] with three vanishing moments for the mother wavelet (It should be noted that the number of vanishing moments used for the estimator has no relation to the number of vanishing moments used in our bootstrap algorithm). The wavelet estimator gives confidence intervals along with the estimates, however we report only the point estimates.

The fourth and fifth columns of Table 4.1 illustrate that the surrogate series generated by using the db6 mother wavelet have H close to that of the parent series. In addition, it is also necessary to examine if the variation of the Hurst value for the surrogate series changes with increasing Hurst values of the parent series. To test this, we computed the difference between the Hurst value of the parent series and each surrogate series. The boxplots of these differences are plotted in Fig. 4.1. The difference between the Hurst value of

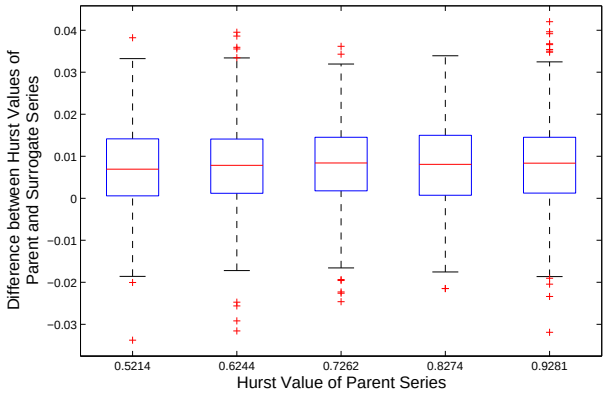

Figure 4.1: Boxplot of difference between Hurst values of surrogate series and parent series for FGN dataset.

the original and surrogate series remains constant over a wide range of values. This demonstrates that the accuracy of our algorithm is constant over a wide range of Hurst values. However, it is observed that there is positive bias in the Hurst parameter of the surrogate datasets.

Next, we evaluate the results obtained with the real datasets. The Hurst values for the parent series and the means and variances of the Hurst values for the surrogate series are reported in Table 4.3. The values in the table indicate that the Hurst values of the surrogate series closely match those of the

original dataset; thereby demonstrating that our algorithm is able to capture the second order self-similarity of the parent dataset (as measured by the Hurst parameter).

Parent Series		Surrogate Series	
		Hurst Value	
Name	Hurst Value	Mean	Variance
pAug89	0.7958	0.7831	0.000052
pOct89	0.7739	0.7630	0.000039
CAIDA	0.5915	0.5869	0.000081
UNC	0.9339	0.9138	0.000563
UofA	0.6112	0.6273	0.000078

Table 4.3: Hurst values for multiple datasets

4.4.1.2 PDF Plots

We examine the match of the distribution for the surrogate series with the parent series by plotting their PDFs. We plot the PDF for only one surrogate series in each case, but similar results are obtained for the other surrogate datasets. Fig. 4.2 is a plot of the PDF for the UofA dataset and one of its surrogate series. We also plot the PDF for a FGN dataset generated by using the Paxson's algorithm. The PDF for both the FGN and surrogate data sets are close to the parent dataset, but the surrogate dataset generated by our algorithm is a better match. This is expected because the UofA dataset does not possess a Gaussian distribution.

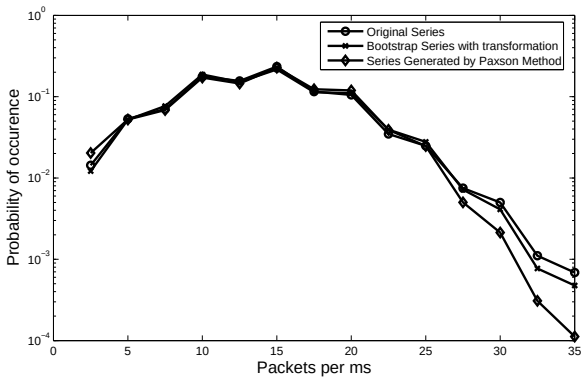

Figure 4.2: PDF of Packets per millisecond of UofA dataset.

In Fig. 4.3, we plot the PDF for the parent series and two bootstrap series; one generated by using the Box-Cox transformation and the other without using the transformation. We also plot the PDF for a FGN dataset generated by the Paxson's algorithm. The PDF for the surrogate series using Box-Cox

transformation is much closer to the original PDF than the series generated without using the Box-Cox transformation, as well as the FGN dataset. This shows that using the Box-Cox transformation provides a better match to the distribution of the parent series and surrogate series generated without using the transformation has a distribution close to that of a FGN dataset.

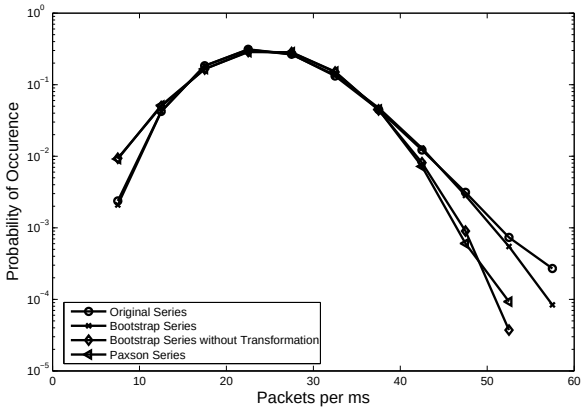

Figure 4.3: PDF of Packets per millisecond of CAIDA dataset.

In Fig. 4.4, we plot the PDF for the pAug89 series. The surrogate series generated by using the Box-Cox transformation provides a closer match to the PDF of the parent series. However, the PDFs are not as close a match

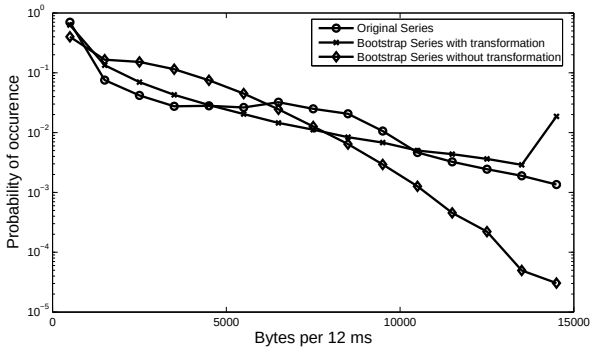

Figure 4.4: PDF of Bytes per 12 milliseconds of pAug89 dataset.

as observed for the UofA and CAIDA datasets. For the pOct89 and UNC datasets, results similar to the pAug89 case are obtained. We are investigating the reason for this divergence in some of the datasets. One possible reason is that the first two datasets are close to Gaussian in terms of the PDF (seen from Fig. 4.2 and Fig. 4.3).

We applied multiple Box-Cox transformations on the original dataset hoping to get a better match of the PDF. However, no improvement was seen even after applying the Box-Cox transformation three times. Another option explored was independently applying the Box-Cox transformation to the wavelet

coefficients at each level. Even this approach does not provide better results. In fact, the PDF of data sets obtained by using this modification is nearly the same as that without any transformation.

4.4.1.3 MultiScaling Behavior

As mentioned above, the Hurst parameter is a measure of the second order scaling properties of the dataset. This is done by studying the variance $S_2(j)$ of the wavelet coefficients at octave j which has the behavior $S_2(j) \sim Cj^{\alpha}$. In addition to the scaling in the second order, the dataset will very often also have scaling for all moments, which can be denoted as:

$$S_q(j) = E[|d(j)|^q] \sim C_q j^{\alpha_q}, \tag{4.2}$$

with q signifying the scaling order.

If α_q is a linear function, the process is said to be monofractal. However, if α_q is not linear, then the process is said to exhibit multiscaling. It has been discovered [69] that data traffic in general and WAN traffic in particular possess multiscaling behavior, and a number of multifractal models have been proposed for capturing the behavior of such traffic [53, 68]. In order to accurately characterize our algorithm we study the higher order scaling behavior of the traces generated by our model and compare them to the original traces as well as to traces generated by Paxson's model and the β-MWM model.

The multiscaling behavior is studied by plotting a Linear Multiscale Diagram (LMD) representing α_q as a function of q. In order to study this behavior we have used the Multiscaling tool developed by Veitch et. al. [86].

In Figs. 4.5 and 4.6 we have plotted the LMD for the UofA and CAIDA datasets respectively. In both the figures, we have plotted the LMD for the parent dataset, the datasets generated by our algorithm (referred to as Bootstrap series in the figures), the FGN model and the β-MWM model. In both the figures, it is seen that the LMD of the dataset generated by our algorithm has a closer match to the LMD of the original trace, as compared to the β-MWM model and the FGN model. Similar results were obtained for the UNC, pAug89 and pOct89 datasets as well.

This demonstrates that our algorithm is able to capture even the higher order scaling properties of the original datasets.

4.4.1.4 Queueing Behavior

The final test is a simple queueing experiment. The setup consists of a single server with infinite buffer size, servicing the incoming data at a rate of 1.1 times the mean arrival rate of the dataset. We then compute the queue length at the end of each interval and from this series compute the queue tail probability, as the probability of the queue length exceeding a certain buffer size, i.e. we compute $P(Q > x)$, where Q is the queue length, and x is the buffer size. Once again, we use the parent dataset, and a randomly picked data set each

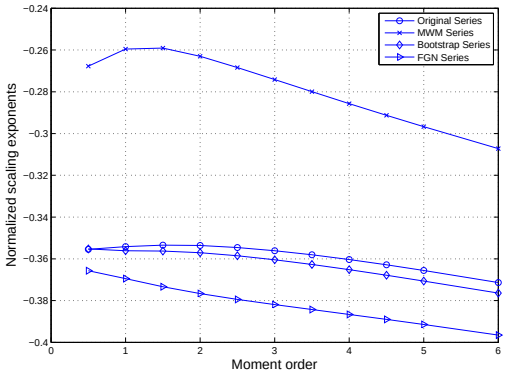

Figure 4.5: Linear Multiscale diagram for UofA dataset.

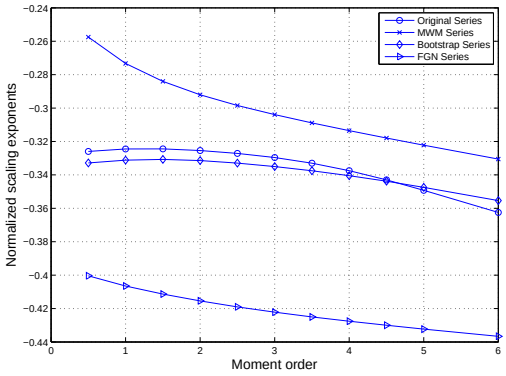

Figure 4.6: Linear Multiscale diagram for CAIDA dataset.

from the set generated by our algorithm (referred to as Bootstrap series) and the FGN and β-MWM model.

Fig. 4.7 is a plot of the queue tail probability for a FGN dataset with Hurst value of 0.7263, and one of its surrogate series. The figure shows that the surrogate dataset has a similar queuing behavior as the parent series.

The queue tail probability for the UofA trace is plotted in Fig. 4.8. It is seen that the dataset generated by our algorithm has behavior that is almost similar to the original data set, while the other two datasets show a very different behavior.

Our algorithm performs better than the other two models even for the UNC and Caida datasets as seen in Fig. 4.9 and Fig. 4.10 respectively.

For the pAug89 and pOct89 datasets, similar results are obtained for the queuing behavior of the surrogate datasets. Thus, it is shown that the surrogate series generated by our algorithm have queuing behavior that is similar to the parent dataset for different data sets.

The results reported in this section conclusively demonstrate that our algorithm is able to retain the Hurst parameter of the parent dataset and performs

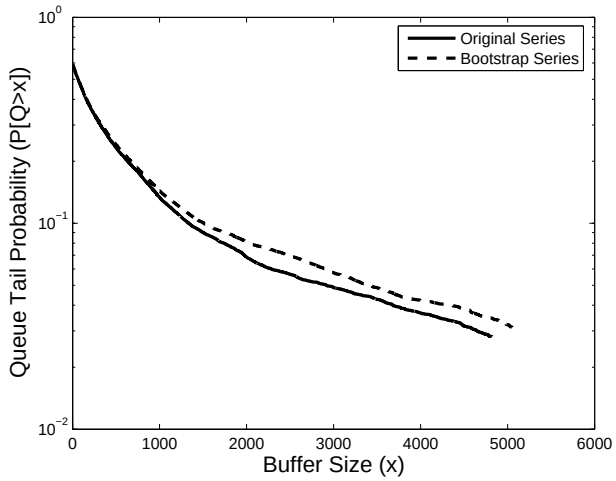

Figure 4.7: Queue Tail Probability for FGN dataset.

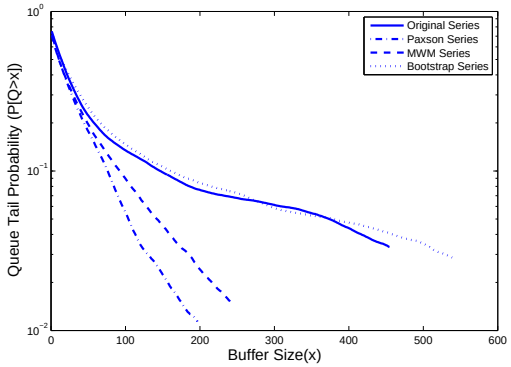

Figure 4.8: Queue Tail Probability for UofA dataset.

better than the FGN and β-MWM models, while considering the multifractal spectrum and the queue tail behavior. This performance is achieved without using any complex parameter fitting procedure and is completed automated, thereby making it easy to use.

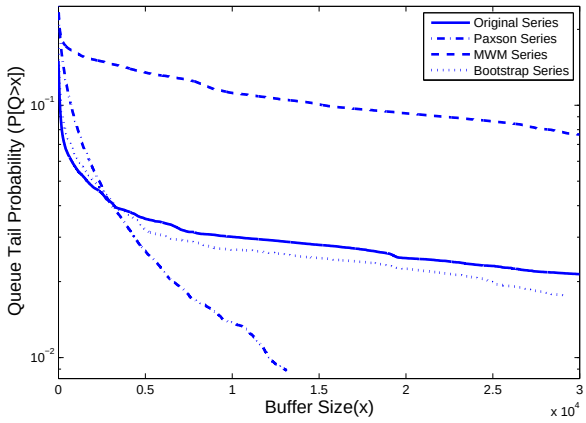

Figure 4.9: Queue Tail Probability for UNC dataset.

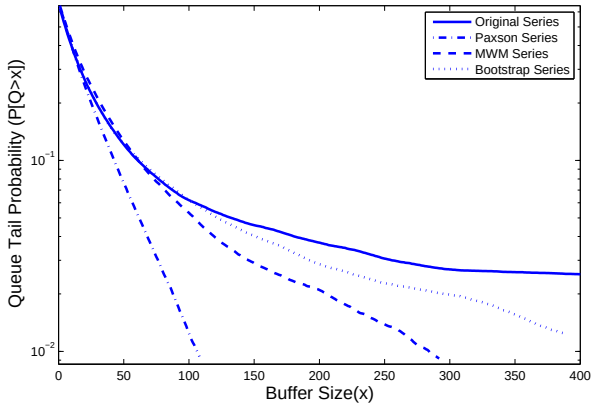

Figure 4.10: Queue Tail Probability for CAIDA dataset.

Chapter 5

Change Detection

5.1 Literature Survey on Change Detection Techniques

There are two different types of techniques in the literature for addressing the change point detection problem. In the first technique, a block size is decided upon and then the estimate of the Hurst parameter over this block is obtained. The Hurst estimates for adjacent blocks are compared and if the values are significantly different, then a change is indicated. The advantage of this method is that along with change detection, the estimates of the Hurst parameter are also readily available. However, the drawback is that the resolution of change detection is limited by the block size. If a very small block size is selected to improve the resolution, then the estimates of the Hurst parameter obtained are not so reliable. If a bigger block size is selected, then the resolution for the change detection is degraded. The second technique is based on monitoring some parameter on a continuous basis and if this parameter changes, then it indicates a change in the Hurst parameter.

In [85], the authors proposed a statistical test to study the stationarity of the scaling exponent over time. Using their test, they have shown that the Hurst value of a traffic trace does not necessarily remain constant. In this method, repeated scans of the stored data are performed by breaking the dataset into different sized blocks and the Hurst parameter estimated for each block. The estimated Hurst values are then subjected to a statistical test to check for their constancy. In [13], a Least Square Criterion is used for detecting changes in the Hurst parameter or the ARMA parameters of a FARIMA model. In [34], the authors propose the use of a Cumulative Sum (CUSUM) like test for the on-line detection of changes for Markov Modulated Poisson Process and Gaussian FARIMA. However, their method works well only when the value of the parameter before and after the change is known in advance. However, in practical scenarios, where this is not the case, they propose running a bank of change detection algorithms with different assumed values for the parameter after the change. This is computationally quite expensive. In [87], a CUSUM

technique is used for detecting change in the Hurst parameter of LRD traffic. However, the drawback of the above method is that it is based on a parametric model of the traffic. In [88], three tests are proposed for detecting changes in the Hurst value. Two of the tests depend on using the Whittle estimator for blocks of non-overlapping data, and the third method is limited to $H < 0.75$.

In [71], the authors propose decomposing the incoming traffic into different scales using wavelets and then use the Schwartz Information Criterion (SIC) to detect when there is a change in the variance of the wavelet coefficients at different scales. If a change is detected in a sufficient number of scales, then it says that the Hurst parameter has changed. A further refinement of this method is proposed in [70], where a combination of the Stationary Wavelet Transform (SWT) and Discrete Wavelet Transform (DWT) is used to decompose the data. The change points indicated by the SIC are then clustered together by the Hough transform to indicate the change point. Their proposal for using the SWT is based on their claim that SWT decomposes the data in a similar fashion as the DWT. However, no theoretical basis is provided for the same. In fact, our experiments indicate that the wavelet coefficients obtained by using the SWT are not as decorrelated as those obtained by the DWT. This can be seen clearly from the figure 5.1 where we have plotted the ACF for the level 2 DWT and SWT coefficients of a FGN series with Hurst value of 0.92. This figure shows that the ACF for the SWT coefficients is significant

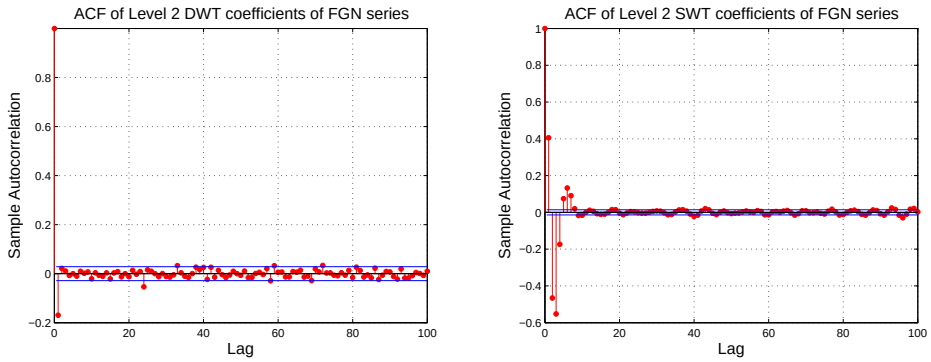

Figure 5.1: Comparison of ACF for DWT and SWT coefficients of FGN series.

upto a lag of 6, while for the DWT coefficients, it is significant only upto a lag of 1. This shows that the DWT decorrelates the data better than SWT. In addition, the wavelet coefficients obtained by SWT do not possess a normal distribution even for a FGN dataset. To test this, we use the same FGN series used above, and decomposed it to 10 levels by using both SWT and DWT. We then test if the wavelet coefficients obtained by both the decomposition techniques at all levels are normal. It is found that the wavelet coefficients at all 10 levels obtained by using DWT are normal, while for the SWT case, the coefficients at only the first three levels are normal, while those at the higher seven levels are not normal.

5.2 Proposed Algorithm

Our algorithm is similar to the one proposed by Rincon and Sallent [70]. It is based on the following equation from [4]:

$$\mu_j = 2^{j(2H-1)} c_f C, \tag{5.1}$$

where $C = \int |\nu|^{1-2H} |\Psi_0(\nu)|^2 d\nu$, and $\Psi_0(\nu)$ is the Fourier transform of the wavelet $\psi(t)$. μ_j is the variance of the wavelet coefficients at level j given by the equation:

$$\mu_j = \frac{1}{n_j} \sum_{k=1}^{n_j} d_{j,k}^2, \tag{5.2}$$

where n_j is the number of available coefficients $d_{(j,.)}$ at level j. Rewriting 5.1 as

$$log_2(\mu_j) = j(2H-1) + log_2(c_f C) \tag{5.3}$$

it is seen that the variance of the wavelet coefficients at any level is proportional to the Hurst value of the original series. Thus, if the Hurst value of the original series changes at any point, the variance of the wavelet coefficients around that location will also change. Thus, by detecting a change in the variance of the wavelet coefficients we can indicate a change in the Hurst value. In the next section, we describe the algorithm used for detecting a change in the variance of the wavelet coefficients.

5.2.1 Variance Change Detection

For detecting a change in the variance of the wavelet coefficients, we have used the algorithm proposed in [26]. The algorithm can be explained as follows: Let X_1, X_2, ... be independent random variables with density $f(X; \theta, \eta)$. We assume $\theta \in \Omega_1 \subset \Re^d$, $d \geq 1$, $\eta \in \Omega_2 \subset \Re^p$, $p \geq 0$. The parameter θ is the parameter of interest, and η will be the nuisance parameter. The following is the hypothesis to be tested:

$$\begin{aligned} H_0 &: \theta = \theta_0, \theta_0 \text{ and } \eta \text{ unknown, for all observations} \\ H_a &: X_1, ..., X_{\tau-1} \text{ have density } f(X; \theta_0, \eta), \ \eta \text{ unknown} \\ &\quad X_\tau, X_{\tau+1}, ... \text{ have density } f(X; \theta_a, \eta), \ \eta, \ \theta_a \text{ unknown,} \end{aligned}$$

where τ, the change point is also unknown.

For our case, the random variables X_i's are replaced by the wavelet coefficients, $d_{j,k}$ for each scale j. The dimension $d = 1$ and $p = 1$. The function f will be the density function of the normal distribution, $\frac{1}{\sqrt{2\pi}\sigma} exp\left[\frac{-1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$, where μ is the mean and σ^2 is the variance. Since, we are interested in monitoring the variance, the hypothesis to be tested becomes:

$$\begin{aligned} H_0 : \sigma^2 &= \sigma_0^2, \quad \sigma_0^2 \text{ and } \mu \text{ are unknown} \\ H_1 : \sigma^2 &\neq \sigma_0^2, \quad \sigma_0^2 \text{ and } \mu \text{ are unknown} \end{aligned} \tag{5.4}$$

This is similar to **Example 2** in [26] in which it has been shown that the test statistic for this case is

$$W_k = \sum_{i=1}^{k} \left[X_i^2 - \left(\sigma_0^2 + \overline{X}_k^2 \right) \right] / \left(2\sigma_0^4 \right)^{1/2}. \tag{5.5}$$

As discussed earlier in Section 2.3, the wavelet coefficients have a mean of zero. Thus, the test hypothesis and the test statistic become:

$$\begin{aligned} H_0 : \sigma^2 &= \sigma_0^2, \quad \sigma_0^2 \text{ is unknown} \\ H_1 : \sigma^2 &\neq \sigma_0^2, \quad \sigma_0^2 \text{ is unknown} \end{aligned} \tag{5.6}$$

$$W_k = \sum_{i=1}^{k} \left[X_i^2 - \sigma_0^2 \right] / \left(2\sigma_0^4 \right)^{1/2} \tag{5.7}$$

The limiting distribution of the statistic $max_{1<k<n} n^{-1/2} |W_k|$ (where n is the length of the dataset) under the null hypothesis is a Brownian bridge [26]. Using this, the critical values C_α have been found to be [48]

α	one-tail	two-tail
0.1	1.073	1.224
0.05	1.224	1.358

We conclude that H_0 is not supported by the data if

$$max_{1<l<n} n^{-1/2} |W_k| \geq C_\alpha, \quad \text{for any } k. \tag{5.8}$$

If no such k, $k \leq n$ exists, then H_0 is not rejected. For our experiments we have used $C_\alpha = 1.224$.

The above discussion applies to a single level of wavelet coefficients. The same procedure is repeated for all the levels of the wavelet coefficients. We use the results obtained at the different levels for making a decision if the Hurst parameter has changed. This decision process is described in the next section.

5.2.2 Decision about Constancy of Hurst Value

By using the algorithm described in the previous section, we are able to find the locations of the variance change points for the wavelet coefficients at the different levels of the DWT of the dataset. The easiest method for concluding that the Hurst value has changed is to test if Eqn. 5.8 is satisfied for any level. In other words, if a change in variance is detected at any level, then a change in the Hurst value is indicated.

However, it is possible that the variance change points at different levels of the DWT might be pointing to different locations in the time domain. Another point to note is that each point in the wavelet domain actually corresponds to a range in the time domain, with the range increasing as we increase the level of

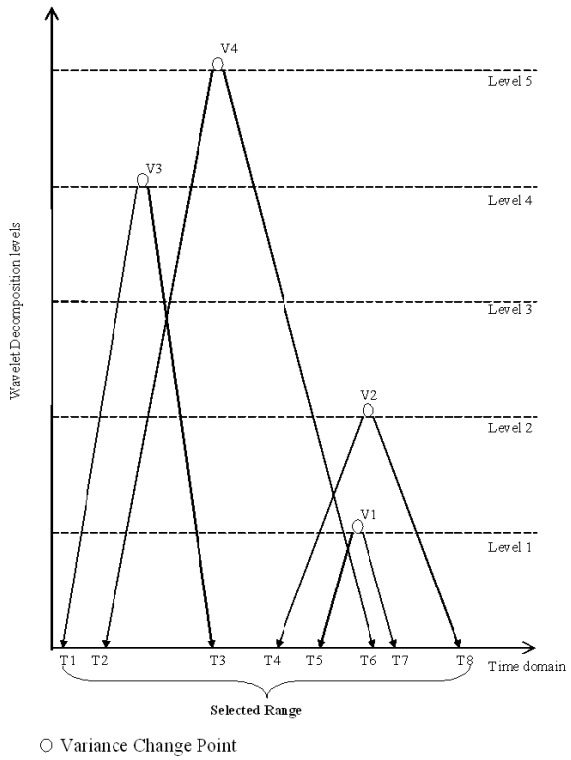

Figure 5.2: Converting change point in variance to Hurst change point.

decomposition. For example, when using the *db*6 wavelet, each point at level 1 of the DWT representation corresponds to 6 points on the time domain, while each point at level 2 corresponds to 16 points, and so on. So, it is necessary to convert each variance change point to its corresponding range along the time domain. This is demonstrated in Figure 5.2, where we have shown 5 levels of wavelet decomposition and their corresponding ranges on the time domain. Thus, a variance change point will actually indicate a range along the time domain for the Hurst change. As can be seen from the figure, variance change points have been indicated at levels 1, 2, 4 and 5, labeled as $V1$, $V2$, $V3$ and $V4$ respectively. Note that, there is no change point indicated at level 3. This happens in many cases, specially when the change in the Hurst value is small. The mapping of each variance change point to its range along the time domain is tabulated below.

Variance Change Point	Time-domain Range
V1	T5-T7
V2	T4-T8
V3	T1-T3
V4	T2-T6

It can be seen from Figure 5.2, that the ranges indicated by the variance change points are not always overlapping. In some cases the ranges indicated

by one level are completely enclosed by the ranges indicated by another level, as for $V1$ and $V2$, while in other cases, there is no overlap as for $V3$ and $V2$. In other cases like for $V3$ and $V4$, there is partial overlap. In order to decide which of the variance change points to select as *the change point*, we look at the number of ranges which intersect or enclose the range indicated by each change point, and then select that point whose range intersects with the maximum number of ranges. The following is a tabulation of the intersecting ranges for each change point.

Variance Change Point	Change Points for which its range intersects
V1	V2,V4
V2	V1,V4
V3	V4
V4	V1,V2,V3

As can be seen, the range indicated by $V4$ intersects with the maximum number of ranges, and hence that is selected as *the change point*. If there are two possible candidates to be selected as *the change point*, then we select the change point at the lower level. One criteria for selecting the variance change point is that its range must intersect with at least one other range. If all the variance change points have mutually exclusive ranges along the time domain, then we consider that as not a valid detection of a Hurst change. Once, the change point is selected, we need to decide the range along the time domain. One way to do that, would be to take just the range indicated by this change point. In this example, that would mean $T2 - T6$. However, we select the range which encompasses all the intersecting ranges. In this case, that becomes $T1 - T8$. Once this "Hurst Change Range" is selected, we take the midpoint of this range as the "Hurst Change Point".

Now that our algorithm has been explained, we demonstrate its performance in the next section.

5.3 Performance Evaluation of Algorithm

In this section, we perform a number of tests to demonstrate the working of our algorithm.

5.3.1 Using FGN Datasets

We tested our algorithm by using FGN datasets of different lengths. The FGN series were generated by using the Matlab function $wfbm$ which generates a Fractional Brownian Motion series with the given Hurst value. The difference between adjacent values of this series gives us the FGN series with the desired Hurst value. For our work, we generated FGN series with the following Hurst

values: 0.52, 0.57, 0.62, 0.67, 0.72, 0.77, 0.82, 0.87, 0.92 and 0.97 of lengths 10000, 20000 and 30000 for each Hurst value. For every Hurst value of a specific length, we generated 1000 series.

We then used the FGN series to create datasets having a single or no change point of the Hurst parameter. The datasets we generated are as follows:

1. Two series of length 20000 each were joined back to back to generate a dataset of length 40000 (referred as $20000 + 20000$, and with change point at 20001).

2. One series of length 30000 was joined with another series of length 10000 to generate a dataset of length 40000. There were two different combinations used in this:

   (a) The series of length 30000 was followed by the series of length 10000 (referred as $30000 + 10000$, and with change point at 30001).

   (b) The series of length 10000 was followed by the series of length 30000 (referred as $10000 + 30000$, and with change point at 10001).

3. One series of length 20000 was joined with another series of length 10000 to generate a dataset of length 30000. There were two different combinations used in this:

   (a) The series of length 20000 was followed by the series of length 10000 (referred as $20000 + 10000$, and with change point at 20001).

   (b) The series of length 10000 was followed by the series of length 20000 (referred as $10000 + 20000$, and with change point at 10001).

For each of the above datasets, we tested for all possible combinations of the following changes of the Hurst parameter: 0 (no change), 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4 and 0.45. Both increases and decreases in the Hurst value were tested.

As discussed previously, our algorithm consists of decomposing the time domain dataset into the wavelet domain by using the DWT. One of the important features of DWT is selecting an appropriate mother wavelet. The most commonly used mother wavelets used in the literature for LRD are the *db*1 (also known as the Haar wavelet) and the *db*3. We use these two wavelets and the *db*6 wavelet for our experiments. We compare the performance of these wavelets for our change detection algorithm by using the $20000 + 20000$ datasets. The corresponding results are shown in the second, third and fourth columns of Tables 5.1, 5.2, 5.3, 5.5, 5.7, 5.9, 5.11, 5.13, 5.15, 5.17 and 5.19. From Tables 5.1 and 5.2, it can be seen that the number of false positives indicated by the change detection algorithm are the least when the *db*1 wavelet is used. However, from Tables 5.3, 5.5, 5.7, 5.9, 5.11, 5.13 and 5.15, it is seen that by using the *db*6 wavelet gives the best detection results. In order to obtain a proper balance between the false positives and correct detections, we decided to use the *db*3 wavelet for all our experiments.

5.3.1.1 False Positives

The first test that we conducted was to check the false positive rate; in other words the number of changes detected when there was no true change. For this purpose, we used the datasets which were formed by joining FGN series having the same Hurst value. The corresponding results are tabulated in Tables 5.1 and 5.2. As seen from Tables 5.1 and 5.2, the number of false positives reduce as the Hurst value increases. We have not yet found any reason for this observation.

Hurst Value	Series				
	$20000 + 20000$			$30000 + 10000$	$10000 + 30000$
	*db*1	*db*3	*db*6	*db*3	*db*3
0.52	0	189	242	230	257
0.57	2	118	195	164	176
0.62	2	61	121	116	119
0.67	3	23	82	80	90
0.72	0	13	57	74	86
0.77	1	4	37	70	91
0.82	1	8	25	65	63
0.87	5	8	21	63	79
0.92	2	8	13	76	85
0.97	1	4	19	59	66

Table 5.1: Number of false positives detected in 1000 series of length 40000. Each series is formed by joining two different FGN series. The Hurst parameter of both the series are the same.

One of the possible reasons for the false positives might be the discontinuity introduced at the point where we join the 2 FGN series. In order to rule out this possibility, we plotted the histogram of the locations where the change was indicated. The histogram for the $20000 + 20000$ dataset with the *db*3 wavelet is shown in Figure 5.3.

If the discontinuity at the point of joining the two FGN series would be the cause for the false positives, we expect to see a peak near the center of the histogram, since that is the point of discontinuity. However, it is seen in Figure 5.3 that this is not the case. In fact, for Hurst values of $0.52, 0.57, 0.62, 0.67$ and 0.72, most of the changes are detected in the initial third of the dataset. Similar results were obtained for the other datasets. This result also proves that the discontinuity introduced by putting FGN series back-to-back does not cause a problem for change detection.

Another observation from Tables 5.1 and 5.2 is that the number of false positives is the least for the $20000 + 20000$ dataset. Even for datasets of the same length; viz. $30000 + 10000$ and $10000 + 30000$, the number of false positives are much greater than the first case. This is another phenomenon for which we have been unable to find any explanation. The worst case behaviors

Hurst Value	Series	
	$10000 + 20000$ *db*3	$20000 + 10000$ *db*3
0.52	310	315
0.57	217	215
0.62	175	164
0.67	143	105
0.72	120	98
0.77	97	94
0.82	104	73
0.87	98	76
0.92	101	92
0.97	87	95

Table 5.2: Number of false positives detected in 1000 series of length 30000. Each series is formed by joining two different FGN series. The Hurst parameter of both the series are the same.

are for the $10000 + 20000$ and $20000 + 10000$ datasets with Hurst value of 0.52 giving a False detection rate of 31%.

5.3.1.2 False Negatives

Here, we test how many true changes are missed by our algorithm; or the number of False Negatives. The results for the three series of length 40000 are tabulated in Tables 5.3, 5.5, 5.7, 5.9, 5.11, 5.13, 5.15, 5.17 and 5.19, and the results for the two series of length 30000 are tabulated in Tables 5.4, 5.6, 5.8, 5.10, 5.12, 5.14, 5.16, 5.18 and 5.20. For each case, the results for all the possible transitions of the Hurst parameter are reported in these tables.

From Tables 5.3 and 5.4, it can be seen that the change detection rate by using the *db*3 wavelet is not very high. The best case is 52.7% for the transition $0.52 - 0.57$ for the $20000 + 10000$ dataset, with the worst case being 23.2% for the $10000 + 30000$ dataset. By looking at these results, it can be concluded that our algorithm is not sensitive enough to detect changes of magnitude 0.05. Another observation that can be made from these tables is that when the change point is not at the center of the dataset, the algorithm has better performance in detecting an increase in the Hurst value if the change point is towards the end of the series, while it is the opposite for detecting decreases in the Hurst value. This is seen for all possible transitions in the Hurst value.

When the change magnitude is 0.1, it can be observed from Tables 5.5 and 5.6 that the algorithm gives reasonably satisfactory results with the best performance observed when the change point is in the center of the series ($20000 + 20000$ dataset). However, when the change point is in the initial quarter of the series ($10000 + 30000$ dataset), the algorithm does not have a very good performance in detecting increases in the Hurst value.

Initial Value - Final Value	Change Point Location				
	20001			30001	10001
	*db*1	*db*3	*db*6	*db*3	*db*3
Increase					
0.52-0.57	141	478	777	448	354
0.57-0.62	135	456	739	407	324
0.62-0.67	133	410	732	366	259
0.67-0.72	133	419	752	301	232
0.72-0.77	140	408	702	347	205
0.77-0.82	139	425	728	358	209
0.82-0.87	156	452	737	328	227
0.87-0.92	178	462	704	362	267
0.92-0.97	196	463	755	376	232
Decrease					
0.57-0.52	166	507	657	358	520
0.62-0.57	192	493	602	322	468
0.67-0.62	183	474	590	302	459
0.72-0.67	190	488	574	259	386
0.77-0.72	185	442	539	253	430
0.82-0.77	218	450	566	255	388
0.87-0.82	218	493	570	265	376
0.92-0.87	201	462	561	258	387
0.97-0.92	219	473	591	270	383

Table 5.3: Number of changes detected for 1000 series of length 40000 each. Each series is formed by joining two different FGN series. The Hurst parameter of the 2 series differ by 0.05.

Initial Value - Final Value	Change Point location		Initial Value - Final Value	Change Point location	
	10001	20001		10001	20001
Increase			Decrease		
0.52-0.57	394	526	0.57-0.52	525	486
0.57-0.62	390	520	0.62-0.57	494	394
0.62-0.67	382	466	0.67-0.62	471	390
0.67-0.72	325	438	0.72-0.67	417	379
0.72-0.77	338	422	0.77-0.72	424	328
0.77-0.82	320	438	0.82-0.77	426	381
0.82-0.87	310	442	0.87-0.82	428	366
0.87-0.92	327	429	0.92-0.87	407	345
0.92-0.97	332	448	0.97-0.92	417	353

Table 5.4: Number of changes detected for 1000 series of length 30000 by using the $db3$ wavelet. Each series is formed by joining two different FGN series; one of length 20000 and the other of length 10000. The Hurst parameter of the 2 series differ by 0.05.

Initial Value - Final Value	Change Point Location				
	20001			30001	10001
	$db1$	$db3$	$db6$	$db3$	$db3$
Increase					
0.52-0.62	425	807	969	760	566
0.57-0.67	463	806	978	736	549
0.62-0.72	433	835	981	714	542
0.67-0.77	478	826	972	726	535
0.72-0.82	470	817	978	728	556
0.77-0.87	534	859	979	722	550
0.82-0.92	515	852	967	741	585
0.87-0.97	541	866	981	759	560
Decrease					
0.62-0.52	515	848	935	625	794
0.67-0.57	535	862	920	626	797
0.72-0.62	551	864	920	614	751
0.77-0.67	557	866	914	598	746
0.82-0.72	553	861	919	597	771
0.87-0.77	605	866	933	633	747
0.92-0.82	571	890	930	593	770
0.97-0.87	612	883	921	593	747

Table 5.5: Number of changes detected for 1000 series of length 40000 each. Each series is formed by joining two different FGN series. The Hurst parameter of the 2 series differ by 0.1.

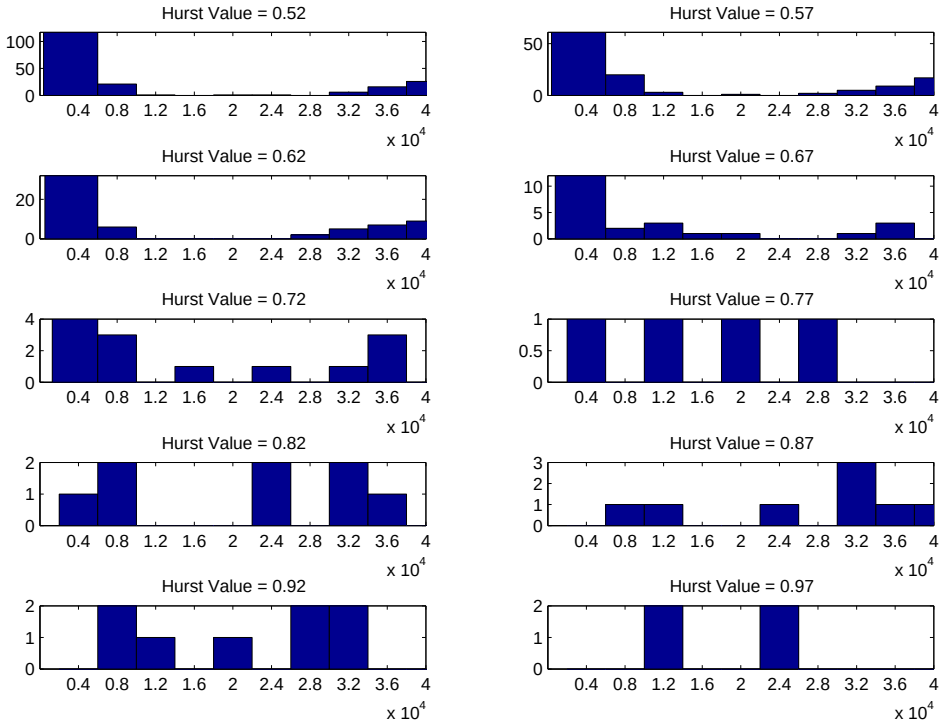

Figure 5.3: Histogram of Change Point Locations for the 20000+20000 dataset using the *db*3 wavelet.

The observations made from Tables 5.3, 5.5, 5.4 and 5.6 show that our algorithm's performance is sensitive to the location of the change point in the dataset. In addition, the algorithm also performs very differently in detecting increases and decreases in the Hurst value if the change point is not in the center of the dataset.

For any transition in the Hurst value with a magnitude of 0.15 or greater, our algorithm detects changes in more than 90% of the cases as can be seen from Tables 5.7, 5.8, 5.9, 5.10, 5.11, 5.12, 5.13, 5.14, 5.15, 5.16, 5.17, 5.18, 5.19 and 5.20. In fact, once the change magnitude is 0.20 or greater changes are detected in more than 98% of the cases.

Initial Value - Final Value	Change Point location		Initial Value - Final Value	Change Point location	
	10001	20001		10001	20001
Increase			Decrease		
0.52-0.62	727	799	0.62-0.52	813	734
0.57-0.67	723	794	0.67-0.57	809	720
0.62-0.72	676	803	0.72-0.62	785	703
0.67-0.77	667	792	0.77-0.67	777	718
0.72-0.82	655	800	0.82-0.72	776	688
0.77-0.87	689	821	0.87-0.77	772	708
0.82-0.92	692	815	0.92-0.82	816	693
0.87-0.97	693	800	0.97-0.87	787	693

Table 5.6: Number of changes detected for 1000 series by using the $db3$ wavelet. Each series is formed by joining two different FGN series; one of length 20000 and the other of length 10000. The Hurst parameter of the 2 series differ by 0.1.

Initial Value - Final Value	Change Point Location				
	20001			30001	10001
	$db1$	$db3$	$db6$	$db3$	$db3$
Increase					
0.52-0.67	787	975	1000	961	839
0.57-0.72	795	974	1000	929	813
0.62-0.77	815	976	999	938	822
0.67-0.82	809	988	1000	955	841
0.72-0.87	855	989	1000	955	840
0.77-0.92	874	990	999	957	834
0.82-0.97	881	994	999	959	849
Decrease					
0.67-0.52	829	975	995	865	965
0.72-0.57	858	986	995	891	952
0.77-0.62	860	990	994	881	960
0.82-0.67	856	990	995	876	950
0.87-0.72	865	990	998	869	956
0.92-0.77	883	987	996	875	959
0.97-0.82	886	993	1000	889	957

Table 5.7: Number of changes detected for 1000 series of length 40000 each. Each series is formed by joining two different FGN series. The Hurst parameter of the 2 series differ by 0.15.

Initial Value - Final Value	Change Point location		Initial Value - Final Value	Change Point location	
	10001	20001		10001	20001
Increase			Decrease		
0.52-0.67	918	953	0.67-0.52	948	925
0.57-0.72	887	945	0.72-0.57	957	938
0.62-0.77	904	963	0.77-0.62	958	918
0.67-0.82	913	951	0.82-0.67	965	926
0.72-0.87	921	962	0.87-0.72	948	930
0.77-0.92	925	969	0.92-0.77	961	932
0.82-0.97	942	979	0.97-0.82	975	938

Table 5.8: Number of changes detected for 1000 series by using the $db3$ wavelet. Each series is formed by joining two different FGN series; one of length 20000 and the other of length 10000. The Hurst parameter of the 2 series differ by 0.15.

Initial Value - Final Value	Change Point Location				
	20001			30001	10001
	$db1$	$db3$	$db6$	$db3$	$db3$
Increase					
0.52-0.72	941	999	1000	990	954
0.57-0.77	963	999	1000	996	950
0.62-0.82	956	998	1000	993	962
0.67-0.87	974	1000	1000	996	971
0.72-0.92	972	1000	1000	996	969
0.77-0.97	972	999	1000	997	976
Decrease					
0.72-0.52	979	999	1000	965	998
0.77-0.57	958	998	1000	977	994
0.82-0.62	970	1000	1000	981	994
0.87-0.67	971	1000	1000	986	995
0.92-0.72	978	1000	1000	979	998
0.97-0.77	979	1000	1000	979	997

Table 5.9: Number of changes detected for 1000 series of length 40000 each. Each series is formed by joining two different FGN series. The Hurst parameter of the 2 series differ by 0.2.

Initial Value - Final Value	Change Point location		Initial Value - Final Value	Change Point location	
	10001	20001		10001	20001
Increase			Decrease		
0.52-0.72	981	990	0.72-0.52	993	989
0.57-0.77	987	994	0.77-0.57	997	990
0.62-0.82	988	998	0.82-0.62	997	987
0.67-0.87	983	998	0.87-0.67	997	993
0.72-0.92	991	997	0.92-0.72	998	998
0.77-0.97	994	999	0.97-0.77	999	990

Table 5.10: Number of changes detected for 1000 series by using the $db3$ wavelet. Each series is formed by joining two different FGN series; one of length 20000 and the other of length 10000. The Hurst parameter of the 2 series differ by 0.2.

Initial Value - Final Value	Change Point Location				
	20001			30001	10001
	$db1$	$db3$	$db6$	$db3$	$db3$
Increase					
0.52-0.77	991	1000	1000	1000	996
0.57-0.82	993	1000	1000	999	996
0.62-0.87	994	1000	1000	1000	994
0.67-0.92	993	1000	1000	1000	994
0.72-0.97	999	1000	1000	999	996
Decrease					
0.77-0.52	983	1000	1000	995	999
0.82-0.57	984	1000	1000	997	1000
0.87-0.62	995	1000	1000	997	1000
0.92-0.67	989	1000	1000	999	1000
0.97-0.72	995	1000	1000	997	1000

Table 5.11: Number of changes detected for 1000 series of length 40000 each. Each series is formed by joining two different FGN series. The Hurst parameter of the 2 series differ by 0.25.

Initial Value - Final Value	Change Point location		Initial Value - Final Value	Change Point location	
	10001	20001		10001	20001
Increase			Decrease		
0.52-0.77	1000	999	0.77-0.52	999	999
0.57-0.82	998	1000	0.82-0.57	1000	999
0.62-0.87	1000	1000	0.87-0.62	999	998
0.67-0.92	1000	1000	0.92-0.67	1000	999
0.72-0.97	1000	1000	0.97-0.72	1000	1000

Table 5.12: Number of changes detected for 1000 series by using the $db3$ wavelet. Each series is formed by joining two different FGN series; one of length 20000 and the other of length 10000. The Hurst parameter of the 2 series differ by 0.25.

Initial Value - Final Value	Change Point Location				
	20001			30001	10001
	$db1$	$db3$	$db6$	$db3$	$db3$
Increase					
0.52-0.82	996	1000	1000	1000	1000
0.57-0.87	1000	1000	1000	1000	999
0.62-0.92	999	1000	1000	1000	999
0.67-0.97	1000	1000	1000	1000	1000
Decrease					
0.82-0.52	996	1000	1000	1000	1000
0.87-0.57	996	1000	1000	1000	1000
0.92-0.62	997	1000	1000	999	1000
0.97-0.67	998	1000	1000	1000	1000

Table 5.13: Number of changes detected for 1000 series of length 40000 each. Each series is formed by joining two different FGN series. The Hurst parameter of the 2 series differ by 0.3.

Initial Value - Final Value	Change Point location		Initial Value - Final Value	Change Point location	
	10001	20001		10001	20001
Increase			Decrease		
0.52-0.82	999	1000	0.82-0.52	1000	999
0.57-0.87	1000	1000	0.87-0.57	1000	1000
0.62-0.92	1000	1000	0.92-0.62	1000	1000
0.67-0.97	1000	1000	0.97-0.67	1000	1000

Table 5.14: Number of changes detected for 1000 series by using the $db3$ wavelet. Each series is formed by joining two different FGN series; one of length 20000 and the other of length 10000. The Hurst parameter of the 2 series differ by 0.3.

Initial Value - Final Value	Change Point Location				
	20001			30001	10001
	$db1$	$db3$	$db6$	$db3$	$db3$
Increase					
0.52-0.87	1000	1000	1000	1000	1000
0.57-0.92	999	1000	1000	1000	1000
0.62-0.97	999	1000	1000	1000	1000
Decrease					
0.87-0.52	998	1000	1000	1000	1000
0.92-0.57	1000	1000	1000	1000	1000
0.97-0.62	997	1000	1000	1000	1000

Table 5.15: Number of changes detected for 1000 series of length 40000 each. Each series is formed by joining two different FGN series. The Hurst parameter of the 2 series differ by 0.35.

Initial Value - Final Value	Change Point location		Initial Value - Final Value	Change Point location	
	10001	20001		10001	20001
Increase			Decrease		
0.52-0.87	1000	1000	0.87-0.52	1000	1000
0.57-0.92	1000	1000	0.92-0.57	1000	1000
0.62-0.97	1000	1000	0.97-0.62	1000	1000

Table 5.16: Number of changes detected for 1000 series by using the $db3$ wavelet. Each series is formed by joining two different FGN series; one of length 20000 and the other of length 10000. The Hurst parameter of the 2 series differ by 0.35.

Initial Value - Final Value	Change Point Location				
	20001			30001	10001
	*db*1	*db*3	*db*6	*db*3	*db*3
Increase					
0.52-0.92	1000	1000	1000	1000	1000
0.57-0.97	1000	1000	1000	1000	1000
Decrease					
0.92-0.52	1000	1000	1000	1000	1000
0.97-0.57	999	1000	1000	1000	1000

Table 5.17: Number of changes detected for 1000 series of length 40000 each. Each series is formed by joining two different FGN series. The Hurst parameter of the 2 series differ by 0.4.

Initial Value - Final Value	Change Point location		Initial Value - Final Value	Change Point location	
	10001	20001		10001	20001
Increase			Decrease		
0.52-0.92	1000	1000	0.92-0.52	1000	1000
0.57-0.97	1000	1000	0.97-0.57	1000	1000

Table 5.18: Number of changes detected for 1000 series by using the *db*3 wavelet. Each series is formed by joining two different FGN series; one of length 20000 and the other of length 10000. The Hurst parameter of the 2 series differ by 0.4.

Initial Value - Final Value	Change Point Location				
	20001			30001	10001
	*db*1	*db*3	*db*6	*db*3	*db*3
Increase					
0.52-0.97	1000	1000	1000	1000	1000
Decrease					
0.97-0.52	999	1000	1000	1000	1000

Table 5.19: Number of changes detected for 1000 series of length 40000 each. Each series is formed by joining two different FGN series. The Hurst parameter of the 2 series differ by 0.45.

Initial Value - Final Value	Change Point location		Initial Value - Final Value	Change Point location	
	10001	20001		10001	20001
Increase			Decrease		
0.52-0.97	1000	1000	0.97-0.52	1000	1000

Table 5.20: Number of changes detected for 1000 series by using the $db3$ wavelet. Each series is formed by joining two different FGN series; one of length 20000 and the other of length 10000. The Hurst parameter of the 2 series differ by 0.45.

5.3.2 Comparison with Rincon-Sallent (R-S) Algorithm

In this section, we compare the performance of our algorithm with that of the Rincon-Sallent (R-S) algorithm discussed previously. This algorithm has been chosen for comparison because it is the latest algorithm proposed for change detection of the Hurst parameter.

For our algorithm, we have used the *db*3 wavelet with 10 levels of decomposition. The critical value selected was the two-tailed value for significance levels of 0.1; i.e. 1.224. For the R-S algorithm, we used the code provided by the authors, and the parameters selected are as follows:

1. Number of levels of decomposition = 8
2. Border scale between the two methods = 3
3. Significance level for DWT = 1E-12
4. Significance level for SWT = 1E-5
5. Minimum segment size = 10000
6. Offset = 0
7. Resolution = 2000
8. Quorum = 3
9. Number of change points = 1

5.3.2.1 False Positives

The "Quorum" selected for the R-S algorithm is the minimum number of levels on which a change should be indicated for the point to be selected as a Hurst change point. This is more than what we have used for our algorithm; we have indicated a change in the Hurst parameter, if a change point is found at any two levels, as discussed previously. It is expected that using a higher number of levels for making the decision should give fewer false positives, and thus the false positive rate of the R-S algorithm should be less than our algorithm. However, as seen in Table 5.21, the R-S algorithm has a false positive Rate of more than 90% for all cases. If "Number of change points" is increased to 2, then the false positive rate further increases. This high false positive rate is a drawback of the R-S algorithm.

5.3.2.2 False Negatives

In this section, we make a comparison of the performance of our algorithm and the R-S algorithm in detecting a true change. The results are tabulated in Tables 5.22, 5.23, 5.24, 5.25, 5.26, 5.27, 5.28, 5.29 and 5.30. In these Tables, we have given the number of instances in which a change is indicated out

Hurst Value	Our algorithm	R-S algorithm
0.52	189	912
0.57	118	935
0.62	61	963
0.67	23	955
0.72	13	977
0.77	4	974
0.82	8	978
0.87	8	972
0.92	8	980
0.97	4	984

Table 5.21: Comparison of the number of false positives indicated by our algorithm and Rincon-Sallent (R-S) algorithm for the 20000 + 20000 series.

of 1000 iterations for each case. We have used the 20000 + 20000 dataset for our comparisons. The accuracy of the change point detection algorithms are compared by computing the difference between the actual change point location (20001) and the indicated change point. The means and standard deviations of the difference is also shown in those tables.

From Table 5.22, it can be seen that the R-S algorithm has a much higher detection rate as compared to our algorithm when the change in Hurst parameter is 0.05. Our algorithm gives a worst case detection rate of 41% and a best case detection rate of 50.7%. The R-S algorithm on the other hand has a worst case detection rate of 98.8% and a best case of 100%. However, on comparing the deviation of the change point indicated, it can be seen that in the cases where our algorithm indicates a change, it is much closer to the actual change location than what is indicated by the R-S algorithm. This can be seen more clearly in Figure 5.4 and Figure 5.5 which are the plots of the histograms of the location of the change points indicated by the R-S algorithm and our algorithm. Figure 5.5 shows a sharp peak near 20000 which is the location of the actual change point, with a few change points indicated in other locations. The change points seen near 4000 are all invalid change points. However, in Figure 5.4, it can be seen that change points are indicated over the entire range of the dataset.

It is seen that the mean of the deviation from the actual change point sometimes has a negative value for our algorithm. This might be considered as an invalid detection, since the change is indicated before the actual change point. However, as discussed previously, in our algorithm we decide on the change point as the mid point of the range along the time domain. For the *db*3 wavelet with 10 levels of decomposition, as used in our algorithm, the maximum range is 5116. Thus, any deviation with an absolute value within this range can be considered as an accurate detection of change point irrespective of its sign. It will be seen that in all the cases, the mean deviation has an absolute value

Transition	Our algorithm			R-S algorithm		
	Number	Deviation		Number	Deviation	
		Mean	Standard Deviation		Mean	Standard Deviation
Increase						
0.52-0.57	478	1618.82	7771.58	988	2178.21	9183.79
0.57-0.62	456	846.12	6273.32	992	1961.73	8856.92
0.62-0.67	410	178.11	4151.43	997	1956.37	9527.59
0.67-0.72	419	-405.65	3081.06	997	1740.51	9839.81
0.72-0.77	408	-310.15	2342.02	996	1964.59	9193.01
0.77-0.82	425	-246.79	2429.20	1000	2075.47	9269.57
0.82-0.87	452	-389.66	2148.97	1000	1458.30	9708.35
0.87-0.92	462	-428.09	2430.18	1000	2051.85	9659.23
0.92-0.97	463	-350.27	1732.83	999	1543.21	10334.85
Decrease						
0.57-0.52	507	1235.53	6936.02	991	2313.07	9168.09
0.62-0.57	493	935.62	5855.80	995	1840.89	9524.70
0.67-0.62	474	488.15	3606.36	997	2329.58	9446.45
0.72-0.67	488	538.63	2558.10	995	2131.32	9609.31
0.77-0.72	442	615.48	2622.39	995	2328.44	9780.62
0.82-0.77	450	272.06	2227.85	1000	2016.51	9349.48
0.87-0.82	493	270.28	2144.51	999	1354.42	9514.67
0.92-0.87	462	367.71	2053.00	999	1955.62	10059.89
0.97-0.92	473	300.84	2163.54	1000	1274.98	10283.36

Table 5.22: Comparison of the number of change points indicated and the deviation of detected change point from actual change point for our algorithm and Rincon-Sallent (R-S) algorithm for the 20000 + 20000 series with the difference between the Hurst parameter of the 2 series being 0.05.

less than 5116, thereby showing that on an average our algorithm indicates the change point quite accurately.

For the case where the Hurst value changes by 0.1, the results are given in Table 5.23. From the table, it can be seen that once again the change detection rate is higher for the R-S algorithm as compared to ours. But our algorithm now has a worst case change detection of 80.7% which can be considered a fair detection rate. In addition, it can be seen that the mean deviation from the actual change point and the indicated change point is lower for our algorithm and also has lower standard deviation as compared to the R-S algorithm.

The above results indicates that our algorithm is not very sensitive to small changes in the Hurst parameter (0.05 and 0.1), as the change in variance for the wavelet coefficients is not very significant. However, when the change is detected, it is much more accurate than the R-S algorithm. The sensitivity of our algorithm can probably be improved by changing the selected critical

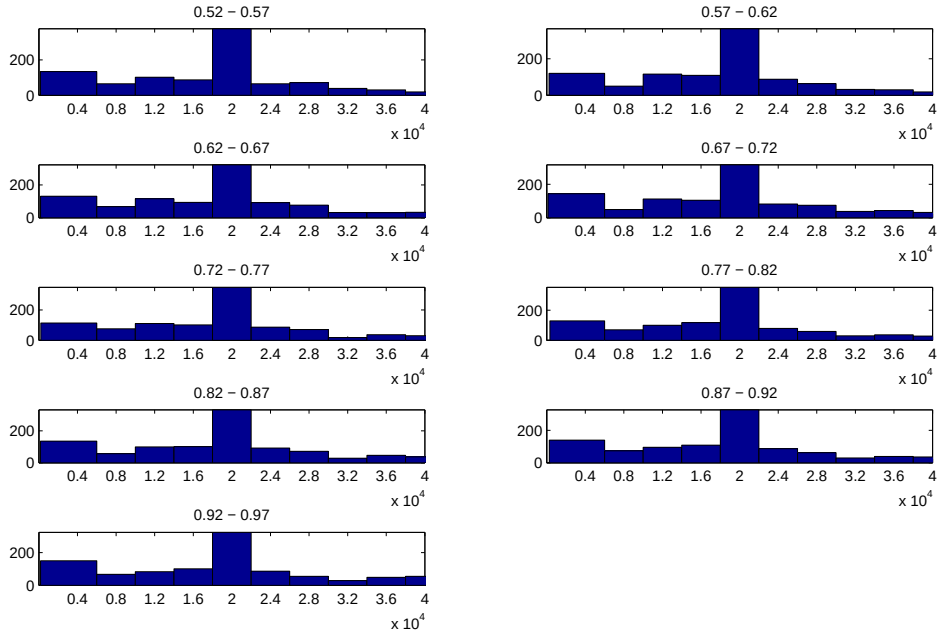

Figure 5.4: Histogram of location of Change point indicated by R-S algorithm for an increase in Hurst value by 0.05

value.

Changes in the Hurst value by a magnitude greater than or equal to 0.15 are detected in at least 97.4% cases by our algorithm, as can be seen in Tables 5.24, 5.25, 5.26, 5.27, 5.28, 5.29 and 5.30. Changes greater than or equal to 0.2 are detected in 99.9% of the cases. On comparing the accuracy of the indicated change point, it can be seen that our algorithm is much more accurate than the R-S algorithm and also has much lower standard deviation in the difference from the actual change point. This again proves that our algorithm is much more accurate than the R-S algorithm in indicating the location of the change point while having the same level of sensitivity for changes of 0.15 or higher.

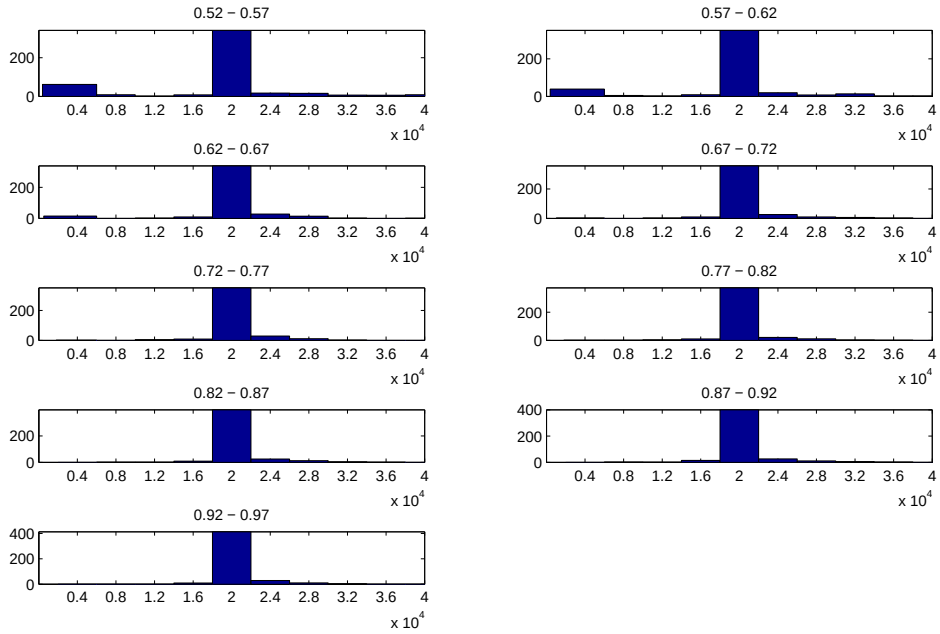

Figure 5.5: Histogram of location of Change point indicated by our algorithm for an increase in Hurst value by 0.05

Transition	Our algorithm			R-S algorithm		
	Number	Deviation		Number	Deviation	
		Mean	Standard Deviation		Mean	Standard Deviation
Increase						
0.52-0.62	807	352.13	3993.36	998	927.65	6514.20
0.57-0.67	806	-73.34	2647.66	1000	960.80	6503.04
0.62-0.72	835	-183.81	2592.01	1000	962.17	7030.32
0.67-0.77	826	-270.72	1496.24	1000	1174.85	7316.95
0.72-0.82	817	-200.49	1226.12	999	986.06	7185.26
0.77-0.87	859	-239.59	1512.26	1000	1332.77	7374.70
0.82-0.92	852	-207.58	1718.70	1000	862.55	7370.26
0.87-0.97	866	-189.65	1488.46	999	-7.86	7945.05
Decrease						
0.62-0.52	848	357.53	3712.78	1000	1797.77	6955.99
0.67-0.57	862	254.26	2424.52	998	2361.35	7153.13
0.72-0.62	864	319.12	1829.32	1000	1889.27	6577.37
0.77-0.67	866	303.89	1801.10	1000	1618.54	7102.65
0.82-0.72	861	267.96	1522.06	1000	1668.61	7524.41
0.87-0.77	866	312.18	1640.39	1000	1466.49	7427.96
0.92-0.82	890	219.46	1504.10	1000	1666.42	7878.45
0.97-0.87	883	306.88	1587.14	1000	922.37	7897.06

Table 5.23: Comparison of the number of change points indicated and the deviation of detected change point from actual change point for our algorithm and Rincon-Sallent (R-S) algorithm for the 20000 + 20000 series with the difference between the Hurst parameter of the 2 series being 0.10.

Transition	Our algorithm			R-S algorithm		
	Number	Deviation		Number	Deviation	
		Mean	Standard Deviation		Mean	Standard Deviation
Increase						
0.52-0.67	975	40.01	1896.91	1000	685.05	5582.62
0.57-0.72	974	-8.73	1417.72	1000	315.29	5664.47
0.62-0.77	976	-117.52	1170.92	1000	496.91	5826.06
0.67-0.82	988	-135.17	1085.63	1000	664.18	6018.67
0.72-0.87	989	-130.63	1088.36	1000	581.65	5985.20
0.77-0.92	990	-191.53	1185.22	1000	865.71	6046.80
0.82-0.97	994	-194.76	1256.14	1000	415.42	6813.14
Decrease						
0.67-0.52	975	172.90	1756.84	1000	1352.59	5994.10
0.72-0.57	986	215.78	1599.02	1000	1444.10	5603.67
0.77-0.62	990	222.69	1085.59	1000	1240.56	6090.68
0.82-0.67	990	225.59	1030.76	1000	1024.49	6197.22
0.87-0.72	990	189.31	1079.73	1000	1443.00	6154.58
0.92-0.77	987	209.06	1077.03	1000	1046.24	6459.13
0.97-0.82	993	198.67	1068.77	1000	1044.52	6203.34

Table 5.24: Comparison of the number of change points indicated and the deviation of detected change point from actual change point for our algorithm and Rincon-Sallent (R-S) algorithm for the 20000 + 20000 series with the difference between the Hurst parameter of the 2 series being 0.15.

Transition	Our algorithm			R-S algorithm		
	Number	Deviation		Number	Deviation	
		Mean	Standard Deviation		Mean	Standard Deviation
Increase						
0.52-0.72	999	-37.73	1304.96	1000	356.01	4906.57
0.57-0.77	999	-76.68	713.43	1000	94.19	4956.41
0.62-0.82	998	-107.96	758.66	1000	351.32	5076.71
0.67-0.87	1000	-81.75	626.50	1000	506.73	5338.87
0.72-0.92	1000	-88.34	694.52	1000	307.26	5067.95
0.77-0.97	999	-81.79	603.67	1000	451.94	5621.55
Decrease						
0.72-0.52	999	151.88	918.11	1000	1434.82	5368.49
0.77-0.57	998	186.75	899.44	1000	1119.38	5196.72
0.82-0.62	1000	130.67	791.02	1000	1162.11	5688.18
0.87-0.67	1000	132.19	723.87	1000	1022.27	5538.30
0.92-0.72	1000	115.32	694.65	1000	820.10	6090.31
0.97-0.77	1000	164.72	833.75	1000	906.07	5680.95

Table 5.25: Comparison of the number of change points indicated and the deviation of detected change point from actual change point for our algorithm and Rincon-Sallent (R-S) algorithm for the 20000 + 20000 series with the difference between the Hurst parameter of the 2 series being 0.20.

Transition	Our algorithm			R-S algorithm		
	Number	Deviation		Number	Deviation	
		Mean	Standard Deviation		Mean	Standard Deviation
Increase						
0.52-0.77	1000	-75.64	640.78	1000	252.13	4781.00
0.57-0.82	1000	-75.05	568.52	1000	82.14	4660.24
0.62-0.87	1000	-32.40	334.31	1000	365.21	4690.48
0.67-0.92	1000	-32.11	421.65	1000	276.60	5298.52
0.72-0.97	1000	-32.74	346.43	1000	45.07	5341.69
Decrease						
0.77-0.52	1000	90.43	590.19	1000	1271.04	5400.51
0.82-0.57	1000	73.80	450.68	1000	1440.06	5456.25
0.87-0.62	1000	87.00	505.81	1000	1233.47	5389.78
0.92-0.67	1000	65.98	449.90	1000	1093.06	5725.81
0.97-0.72	1000	70.92	456.12	1000	533.14	5164.68

Table 5.26: Comparison of the number of change points indicated and the deviation of detected change point from actual change point for our algorithm and Rincon-Sallent (R-S) algorithm for the 20000 + 20000 series with the difference between the Hurst parameter of the 2 series being 0.25.

Transition	Our algorithm			R-S algorithm		
	Number	Deviation		Number	Deviation	
		Mean	Standard Deviation		Mean	Standard Deviation
Increase						
0.52-0.82	1000	-18.05	330.01	1000	150.33	5162.04
0.57-0.87	1000	-18.51	330.20	1000	-62.72	4798.96
0.62-0.92	1000	-24.32	379.21	1000	132.32	5070.06
0.67-0.97	1000	-38.33	376.35	1000	-88.12	4960.17
Decrease						
0.82-0.52	1000	23.71	301.20	1000	1352.49	5575.78
0.87-0.57	1000	33.23	286.78	1000	1070.43	5039.57
0.92-0.62	1000	66.44	457.48	1000	1183.55	5252.98
0.97-0.67	1000	41.23	346.10	1000	731.70	4841.55

Table 5.27: Comparison of the number of change points indicated and the deviation of detected change point from actual change point for our algorithm and Rincon-Sallent (R-S) algorithm for the 20000 + 20000 series with the difference between the Hurst parameter of the 2 series being 0.30.

Transition	Our algorithm			R-S algorithm		
	Number	Deviation		Number	Deviation	
		Mean	Standard Deviation		Mean	Standard Deviation
Increase						
0.52-0.87	1000	-5.86	204.14	1000	-22.16	4658.21
0.57-0.92	1000	5.03	137.38	1000	-233.52	4877.48
0.62-0.97	1000	-12.85	233.27	1000	-180.16	5008.24
Decrease						
0.87-0.52	1000	27.58	302.62	1000	1255.93	5155.16
0.92-0.57	1000	24.59	282.84	1000	1135.56	5079.73
0.97-0.62	1000	29.02	313.64	1000	544.02	5187.97

Table 5.28: Comparison of the number of change points indicated and the deviation of detected change point from actual change point for our algorithm and Rincon-Sallent (R-S) algorithm for the 20000 + 20000 series with the difference between the Hurst parameter of the 2 series being 0.35.

Transition	Our algorithm			R-S algorithm		
	Number	Deviation		Number	Deviation	
		Mean	Standard Deviation		Mean	Standard Deviation
Increase						
0.52-0.92	1000	5.48	121.89	1000	-115.39	4755.69
0.57-0.97	1000	2.12	133.76	1000	-492.24	4899.18
Decrease						
0.92-0.52	1000	3.65	115.98	1000	1097.76	5433.80
0.97-0.57	1000	13.57	198.32	1000	904.12	5174.34

Table 5.29: Comparison of the number of change points indicated and the deviation of detected change point from actual change point for our algorithm and Rincon-Sallent (R-S) algorithm for the 20000 + 20000 series with the difference between the Hurst parameter of the 2 series being 0.40.

Transition	Our algorithm			R-S algorithm		
	Number	Deviation		Number	Deviation	
		Mean	Standard Deviation		Mean	Standard Deviation
Increase						
0.52-0.97	1000	-1.07	188.01	1000	-326.82	4747.17
Decrease						
0.97-0.52	1000	5.65	235.44	1000	1011.33	4837.80

Table 5.30: Comparison of the number of change points indicated and the deviation of detected change point from actual change point for our algorithm and Rincon-Sallent (R-S) algorithm for the 20000 + 20000 series with the difference between the Hurst parameter of the 2 series being 0.45.

Chapter 6

Conclusions and Future Directions

6.1 Conclusions

The methods commonly used for estimating the Hurst parameter assume that it is stationary over the entire range of the data set. However, it has been demonstrated that this is not always true. We used the tool proposed by Abry and Veitch to show that it is possible to detect such non-stationarities in the Hurst parameter. However, their tool is unable to detect the location of any change, and we propose a solution to this in Chapter 5. From the results, in this chapter it is seen that the proposed algorithm for change detection performs quite well when used on FGN data sets. It has very low rate of false positives, and a high rate of correct detection with high accuracy of the change point indication when the change magnitude is 0.1 or more.

In addition to the drawback of giving only a single estimate, the estimators also suffer from the presence of non-stationarities like changing mean and presence of SRD. We have demonstrated how to use some recently proposed tools like the "Dependent Sizer" and the "Linear Multifractal Spectrum" for deciding if the given data set has constant mean and constant Hurst value. A sequence of steps have been proposed to be used for estimation of the Hurst parameter for practical data sets. In the case of presence of SRD, the estimators are shown to perform quite poorly and there is still no existing technique for negating its effect on the estimates obtained. This remains an area for further research.

We also propose a technique for generating multiple traffic traces from a parent trace by using the bootstrapping technique in the wavelet domain. We have shown that our technique can produce data sets matching the original trace in terms of its Hurst parameter and multifractal spectrum both for synthetic and real data sets. The queuing behavior of the traces generated by our algorithm is also shown to be close to that of the actual trace. The behavior of our algorithm is compared to that of the traditional FGN model and the

more recent β-MWM model and is demonstrated to perform better than both of them.

6.2 Future Work

The way our change detection algorithm is currently implemented, only one change point can be detected at a time in a given series. This needs to be extended to be able to detect multiple change points in a given series. One way of doing this would be to run the algorithm and if any change point is found, then the data set is split into two parts around that change point, and the change detection algorithm is run on both these parts. This process is repeated recursively till no changes are found.

Secondly, the method used for deciding if a change has occurred based on results at different DWT levels (Section 5.2.2) needs to be made more sophisticated. One of the options is to use the Hough Transform as discussed in [70].

In addition, the proposed technique is useful only when the wavelet coefficients have a normal distribution. However, if the original data set does not have a Gaussian distribution, as is the case for most real life traffic traces, then the wavelet coefficients will also not have a Gaussian distribution and the algorithm might not give useful results. One of the techniques for using our change detection algorithm with such data sets is to use the Box-Cox transformation as used in our trace generation algorithm in Section 4.3.2.

Finally, the algorithm needs to be adapted to be able to be run online for detecting changes in the Hurst value in real time.

Bibliography

[1] The Mathworks - Wavelet Toolbox - Analyze and synthesize signals and images using wavelet techniques. http://www.mathworks.com/products/wavelet, 2004.

[2] Cooperative Association for Internet Data Analysis. http://www.caida.org/data/passive/index.xml, 2006.

[3] University of North Carolina - Network Data Analysis Study Group. http://www-dirt.cs.unc.edu/unc02_ts, 2006.

[4] P. Abry and D. Veitch. Wavelet Analysis of Long-Range Dependent Traffic. *IEEE Transactions on Information Theory*, 44:2–15, 1998.

[5] C.N. Aduba and M.N.O Sadiku. Performance of a queueing model with self-similar input traffic. *IEEE Proceedings of SouthEastCon 2001*, pages 40–43, 2001.

[6] C. Angelini, D. Cada, G. Katul, and B. Vidakovic. Resampling hierarchical processes in the wavelet domain: A case study using atmospheric turbulence. *Physica D: Nonlinear Phenomena*, 207(1-2):24–40, 2005.

[7] S. Ben Hariz and J.J. Wylie. Convergence rates for estimating a change-point with long-range dependent sequences. *Comptes Rendus Mathematique*, 341:765–768, 2005.

[8] J. Beran. *Statistics for Long-Memory Processes*. Chapman & Hall, 1994.

[9] J. Beran, R. Sherman, M.S. Taqqu, and W. Willinger. Long-Range Dependence in Variable-Bit-Rate Video Traffic. *IEEE Transactions on Communications*, 43:1566–1579, 1995.

[10] G. E. P. Box and D. R. Cox. An analysis of transformations. *Journal of Royal Statistical Socienty, Series B*, 26(2):211–252, 1964.

[11] M. Breakspear, M. Brammer, and P. A. Robinson. Construction of multivariate surrogate sets from nonlinear data using the wavelet transform. *Physica D: Nonlinear Phenomena*, 182(1-2):1–22, 2003.

[12] P. Chaudhuri and J.S. Marron. Sizer for exploration of structures in curves. *Journal of the American Statistical Association*, 94(447):807–823, 1999.

[13] M. Coulon and A. Swami. Least squares detection of multiple changes in fractional arima processes. *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'01*, 5:3177–3188, 2001.

[14] M.E. Crovella and A. Bestavros. Self-similarity in world wide web traffic: Evidence and possible causes. *IEEE/ACM Transactions on Networking*, 6:835–846, 1997.

[15] L.J. De la Cruz, E. Pallares, J.J. Alins, and J. Mata. Self-Similar Traffic Generation Using A Fractional ARIMA Model. Application To the VBR MPEG Video Traffic. *SBT/IEEE International Symposium '98*, 1:102–107, 1998.

[16] B. Efron. Bootstrap methods: Another look at the jackknife. *Annals of Statistics*, 7:1–26, 1979.

[17] A. Erramilli, O. Narayan, A. Neidhardt, and I. Saniee. Performance impacts of multi-scaling in wide area TCP/IP traffic. *Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer Societies, INFOCOM'96*, 1:362–369, 2000.

[18] A. Erramilli, O. Narayan, and W. Willinger. Experimental queueing analysis with long-range dependent packet traffic. *IEEE/ACM Transactions on Networking*, 4:209–223, 1996.

[19] Z. Fan and P. Mars. Self-Similar Traffic Generation And Parameter Estimation Using Wavelet Transform. *IEEE Global Telecommunications Conference '97*, 3:1419–1423, 1997.

[20] A. Feldmann, A.C. Gilbert, and W. Willinger. Data networks as cascades: Investigating the multifractal nature of internet wan traffic. *Computer Communication Review, Proceedings of the ACM/SIGCOMM '98*, 28:42–55, 1998.

[21] H. Feng, T. R. Willemain, and N. Shang. Wavelet-based bootstrap for time series analysis. *Communications in Statistics: Simulation and Computation*, 34(2):393–413, 2005.

[22] P. Flandrin. Wavelet analysis and synthesis of fractional brownian motion. *IEEE Transactions on Information Theory*, 38:910–917, 1992.

[23] H. Furuya, M. Fukushima, H. Nakamura, and S. Nomoto. Modeling of aggregated tcp/ip traffic on a bottleneck link based on scaling behavior. *IEICE Transactions on Communications*, E85-B:1756–1765, 2002.

[24] R. Garcia, V. Garcia, X.G. Paneda, D. Melendi, and J. Perez. Analysis and modelling of a broadband fiber access network with high peer-to-peer traffic load. *Simulation Modelling Practice and Theory*, 14:506–526, 2006.

[25] R.G. Garroppo, S. Giordano, M. Pagano, and G. Procissi. Testing α-stable processes in capturing the queuing behavior of broadband teletraffic. *Signal Processing*, 82(12):1861–1872, 2002.

[26] E. Gombay. Sequential Change-Point Detection and Estimation. *Sequential Analysis*, 22(3):203–222, 2003.

[27] P. Grau-Carles. Tests of long memory : A bootstrap approach. *Computational Economics*, 25(1-2):103–113, 2005.

[28] M. Grossglauser and J.-C. Bolot. On the relevance of long-range dependence in network traffic. *IEEE/ACM Transactions on Networking*, 7:629–640, 1999.

[29] Z. Guangxi, Ge. Xiaohu, Z. Yaoting, and S. Cunyan. A New Modeling for Self-Similar Network Traffic Based on the Alpha-Stable Processes. *IEEE Internation Conference on Neural Networks and Signal Processing*, 2:1689–1693, 2003.

[30] F. Harmantzis and Hatzinakos D. Heavy network traffic modeling and simulation using stable farima processes. *Proceedings of the 19th International Teletraffic Congress*, 2005.

[31] F.C. Harmantzis, D. Hatzinakos, and I. Lambadaris. Effective Bandwidths and Tail Probabilities for Gaussian and Stable Self-Similar Traffic. *IEEE International Conference on Communications, ICC'03*, 3:1515–1520, 2003.

[32] G. He, Y. Gao, J.C. Hou, and K. Park. A case for exploiting self-similarity of network traffic in tcp congestion control. *Proceedings of the 10th IEEE International Conference on Network Protocols*, pages 34–43, 2002.

[33] J.R.M. Hosking. Fractional Differencing. *Biometrika*, 68:165–176, 1981.

[34] R. Jana and S. Dey. Change detection in teletraffic models. *IEEE Transactions on Signal Processing*, 48(3):846–853, 2000.

[35] H.-D.J. Jeong, D. McNickle, and K. Pawlikowski. Fast Self-Similar Teletraffic Generation Based on FGN and Wavelets. *IEEE International Conference on Networks*, pages 75–82, 1999.

[36] K. Kant. On aggregate traffic generation with multifractal properties. *IEEE Global Telecommunications Conference, 1999. GLOBECOM '99*, 2:1179–1183, 1999.

[37] T. Karagiannis and M. Faloutsos. SELFIS: A Tool for self-similarity and Long-Range Dependence Analysis. *1st Workshop on Fractals and Self-Similarity in Data Mining: Issues and Approaches*, 2002.

[38] T. Karagiannis, M. Faloutsos, and R. Riedi. Long-Range Dependence: Now you see it, now you don't! In *IEEE GLOBECOM, Global Internet Symposium*, pages 2165–2169, 2002.

[39] T. Karagiannis, M. Molle, and M. Faloutsos. Long-range dependence: Ten years of Internet Traffic modeling. *IEEE Internet Computing*, 8:57–64, 2004.

[40] A. Karasaridis and D. Hatzinakos. Network Heavy Traffic Modeling Using α-Stable Self-Similar Processes. *IEEE Transactions On Communications*, 49:1203–1214, 2001.

[41] J. Kilpi and I. Norros. Testing the Gaussian appoximation of aggregate traffic. *Proceedings of the second ACM SIGCOMM Workshop, Marseille, France*, pages 49–61, 2002.

[42] T. Konstantopoulos and S.-J. Lin. Macroscopic Models for Long-Range Dependent Network Traffic. *Queueing Systems*, 28:215–243, 1998.

[43] K.-C. Lan and J. Heidemann. Rapid model parameterization from traffic measurements. *ACM Transactions on Modeling and Computer Simulations*, 12(3):201–229, 2002.

[44] N. Larrieu and P. Owezarski. Towards a measurement based networking approach for Internet QoS improvement. *Computer Communications*, 28(3):259–273, 2005.

[45] Wing-Cheong. Lau, A. Erramilli, J.L. Wang, and W. Willinger. Self-Similar Traffic Generation: The Random Midpoint Displacement Algorithm and Its Properties. *IEEE International Conference on Communications '95*, 1:466–472, 1995.

[46] S. Ledesma and D. Liu. Synthesis of fractional Gaussian noise using linear approximation for generating self-similar network traffic. *ACM SIGCOMM Computer Communication Review*, 30:4–17, 2000.

[47] W. Leland, M. Taqqu, W. Willinger, and D. Wilson. On the Self-Similar Nature of Ethernet Traffic (Extended Version). *IEEE/ACM Transactions on Networking*, 2:1–15, 1994.

[48] Shuangquan Liu. *Nonparametric tests for change-point problems with random censorship*. PhD thesis, University of Alberta, 1998.

[49] S. Ma and C. Ji. Modeling heterogeneous network traffic in wavelet domain. *IEEE/ACM Transactions on Networking*, 9(5):634–649, 2001.

[50] M. Masugi. Recurrence plot-based approach to the analysis of ip-network traffic in terms of assessing nonstationary transitions over time. *IEEE Transactions on Circuits and Systems - I: Regular Papers*, 53:2318–2326, 2006.

[51] T. Mikosch, S. Resnick, H. Rootzen, and A. Stegeman. Is Network Traffic Approximated by Stable Levy Motion or Fractional Brownian Motion. *The Annals of Applied Probability*, 12:23–68, 2002.

[52] S. Molnar and T.D. Dang. Pitfalls in Long Range Dependence Testing and Estimation. In *IEEE GLOBECOM*, 2000.

[53] A. Nogueira, P. Salvador, and R. Valadas. Modeling network traffic with multifractal behavior. *10th International Conference on Telecommunications*, 2:1071–1077, 2003.

[54] I. Norros. A Storage Model with Self-Similar Input. *Queueing Systems*, 16:387–396, 1994.

[55] S.A.M. Ostring, H.R. Sirisena, and I. Hudson. Rate control of elastic connections competing with long-range dependent network traffic. *IEEE Transactions on Communications*, 49(6):1092–1101, 2001.

[56] Y.-C. Ouyang and L.-B. Yeh. Predictive bandwidth control for MPEG video: A wavelet approach for self-similar parameters estimation. *IEEE International Conference on Communications*, 5:1551–1555, 2001.

[57] C. Park, F. Hernandez-Campos, J.S. Marron, and F.D. Smith. Long-range dependence in a changing internet traffic mix. *Computer Networks*, 48(3):401–422, 2005.

[58] C. Park, J.S. Marron, and V. Rondonotti. Dependent sizer: Goodness-of-fit tests for time series models. *Journal of Applied Statistics*, 31(8):999–1017, 2004.

[59] K. Park, G. Kim, and M.E. Crovella. On the effect of traffic self-similarity on network performance. In *Proceedings of the SPIE International Conference on Performance and Control of Network Systems*, pages 296–310, 1997.

[60] V. Paxson. Fast, Approximate Synthesis of Fractional Gaussian Noise for Generating Self-Similar Network Traffic. *Computer Communications Review*, 27:5–18, 1997.

[61] V. Paxson and S. Floyd. Wide-Area Traffic: The Failure of Poisson Modeling. *IEEE/ACM Transactions on Networking*, 3:226–244, 1995.

[62] C.-K. Peng, S.V. Buldrev, S. Havlin, M. Simons, H.E. Stanley, and A.L. Goldberger. Mosaic organization of DNA nucleotides. *Physics Review, E.*, 60:1390–1400, 1999.

[63] D. B. Percival, S. Sardy, and A. C. Davison. Wavestrapping time series: Adaptive wavelet-based bootstrapping. In W. J. Fitzgerald, R. L. Smith, A. T. Walden, and P. C. Young, editors, *Nonliner and Nonstationary Signal Processing*, pages 442–470. Cambridge University Press, Cambridge, England, 2001.

[64] D.N. Politis and J.P. Romano. The stationary bootstrap. *Journal of the American Statistical Association*, 89:1303–1313, 1994.

[65] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2004. ISBN 3-900051-07-0.

[66] V. J. Ribeiro, R. H. Riedi, M. S. Crouse, and R. G. Baraniuk. Simulation of nonGaussian long-range-dependent traffic using wavelets. *Proceedings of the 1999 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, pages 1–12, 1999.

[67] V.J. Ribeiro, R.H. Riedi, M.S. Crouse, and R.G. Baraniuk. Multiscale queuing analysis of long-range-dependent network traffic. *Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer and Communication Societies, INFOCOM 2000*, 2:1026–1035, 2000.

[68] R. Riedi, M. Crouse, V. V. Riberiro, and R. Baraniuk. A multifractal wavelet model with application to network traffic. *IEEE Transactions on Information Theory*, 45:992–1019, 1999.

[69] R. Riedi and J. Vehel. Multifractal properties of tcp traffic: a numerical study. Technical Report RR-3129, INRIA, 1997.

[70] D. Rincon, F. Minerva, and S. Sallent. Segmenting LRD Traffic with Wavelets and the Schwarz Information Criterion. *Proceedings of Passive and Active Measurement Conference, PAM 2006*, pages 1–10, 2006.

[71] D. Rincon and S. Sallent. Segmentation of fractal network traffic with wavelets and log-likelihood statistics. *IEEE International Conference on Communications, ICC 2005*, 1:11–15, 2005.

[72] R. Roughan, D. Veitch, and P. Abry. Real-time estimation of the parameters of long-range dependence. *IEEE/ACM Transactions on Networking*, 8:467–478, 2000.

[73] B. Ryu and S. Lowen. Point Process Models for Self-Similar Network Traffic, with Applications. *Stochastic Models*, 14:735–761, 1998.

[74] B. Ryu and S. Lowen. Fractal Traffic Models for Internet Simulation. *Proceedings of Fifth IEEE Symposium on Computers and Communications*, pages 200–206, 2000.

[75] Y. Shu, Z. Jin, L. Wang, and O.W.W. Yang. Prediction-based admission control using FARIMA models. *IEEE International Conference on Communications*, 3:1325–1329, 2000.

[76] Y. Shu, Z. Jin, L. Zhang, L. Wang, and O.W.W. Yang. Traffic prediction using FARIMA models. *IEEE International Conference on Communications*, 2:891–895, 1999.

[77] S. Stoev and M. Taqqu. Simulation methods for linear fractional stable motion and farima using the fast fourier transform. *Fractals*, 12:95–121, 2004.

[78] S. Stoev and M. S. Taqqu. Asymptotic self-similarity and wavelet estimation for long-range dependent fractional autoregressive integrated moving average time series with stable innovations. *Journal of Time Series Analysis*, 26(2):211–249, 2005.

[79] A. Suarez-Gonzalez, J.C. Lopez-Ardao, C. Lopez-Garcia, M. Rodriguez-Perez, M. Fernandez-Veiga, and M.E. Sousa-Vieira. A batch means procedure for mean value estimation of processes exhibiting long range dependence. *Winter Simulation Conference Proceedings*, 1:456–464, 2002.

[80] M. Taqqu, W. Willinger, and R. Sherman. Proof of a Fundamental Result in Self-Similar Traffic Modeling. *ACM SIGCOMM Computer Communication review*, 27:5–23, 1997.

[81] M.S. Taqqu, V. Teverovsky, and W. Willinger. Estimators for Long-Range Dependence: An Empirical Study. *Fractals*, 3:785–788, 1995.

[82] A. H. Tewfik and M. Kim. Correlation structure of the discrete wavelet coefficients of fractional brownian motion. *IEEE Transactions on Information Theory*, 38:904–909, 1992.

[83] J. Theiler, S. Eubank, A. Longtin, B. Galdrikian, and J. Farmer. Testing for nonlinearity in time series: The method of surrogate data. *Physica D*, 58:77–94, 1992.

[84] D. Veitch and P. Abry. A wavelet based joint estimator of the parameters of long-range dependence. *IEEE Transactions on Information Theory*, 45:878–897, 1999.

[85] D. Veitch and P. Abry. A statistical test for the time constancy of scaling exponents. *IEEE Transactions on Information Theory*, 49(10):2325–2334, 2001.

[86] Darryl Veitch. Matlab code for the estimation of MultiScaling Exponents. http://www.cubinlab.ee.mu.oz.au/∼darryl/MS_code.html, 2004.

[87] R.A. Vesilo. Cumulative sum techniques in atm traffic management. *IEEE Global Telecommunications Conference, GLOBECOM 98*, 5:2970–2976, 1998.

[88] R.A. Vesilo and A. Chan. Detecting change points in long range dependency traffic. *Proceedings of Australian Telecommunication Networks and Applications Conference, Melbourne*, pages 567–572, 1996.

[89] X. Wang, Y. Ren, and X. Shan. Forecasting internet traffic with wavelet method. In *Proceeding of the International Conference on Telecommunications*, pages 1192–1195, 2002.

[90] R. Weron. Estimating long-range dependence: Finite sample properties and confidence intervals. *Physics A.*, 312.

[91] W. Willinger, V. Paxson, and M.S. Taqqu. Self-Similarity and Heavy Tails: Structural Modeling of Network Traffic. *A Practical Guide to Heavy tails: Statistical Techniques and Applications*, pages 27–53, 1998.

[92] W. Willinger, M.S. Taqqu, R. Sherman, and D.V. Wilson. Self-similarity through high-variability: Statistical Analysis of Ethernet LAN traffic at the source level. *IEEE/ACM Transactions on Networking*, 5:71–86, 1997.

[93] G. Xiaohu, Z. Guangxi, and Z. Yaoting. A New Model for Network Traffic Based on Alpha-Stable Self-Similar Processes. *The 11th IEEE International Conference on Networks*, pages 1–5, 2003.

[94] H. Yin, C. Lin, G. Min, and X. Chu. Effective congestion control for QoS enhancement of self-similar multimedia traffic. *IEE Proceedings - Communications*, 153(5):675–682, 2006.

[95] H. Yousefi'zadeh. A neural-based technique for estimating self-similar traffic average queueing delay. *IEEE Communications Letters*, 6(10):419–421, 2002.

[96] X. Yu, L.J. Thng, and Y. Jiang. Measurement-based effective bandwidth estimation for long range dependent traffic. *IEEE Region 10 International Conference on Electrical and Electronic Technology*, pages 359–365, 2001.

[97] R. Yunhua. Evaluation and estimation of second-order self-similar network traffic. *Computer Communications*, 27(9):898–904, 2004.