Modelling Computer Network Traffic Using Wavelets and Time Series Analysis

Mbulelo Brenwen Ntlangu

Supervisor: Prof.A.Baghai-Wadji

Submitted to the Department of Electrical Engineering in fulfillment of the academic requirements for the degree of Master of Science in Electrical Engineering at the University of Cape Town

February 2019

Declaration

I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own. This dissertation has been submitted to the Turnitin module (or equivalent similarity and originality checking software) and I confirm that my supervisor has seen my report and any concerns revealed by such have been resolved with my supervisor.

Mbulelo Brenwen Ntlangu
November 2018

Signed by candidate

Supervisor's Declaration

As the candidate's supervisor, I have approved this dissertation for submission.

Prof.A.Baghai-Wadji
February 2019

Acknowledgements

And I would like to acknowledge ...

Abstract

Modelling of network traffic is a notoriously difficult problem. This is primarily due to the ever-increasing complexity of network traffic and the different ways in which a network may be excited by user activity. The ongoing development of new network applications, protocols, and usage profiles further necessitate the need for models which are able to adapt to the specific networks in which they are deployed. These considerations have in large part driven the evolution of statistical profiles of network traffic from simple Poisson processes to non-Gaussian models that incorporate traffic burstiness, non-stationarity, self-similarity, long-range dependence (LRD) and multi-fractality. The need for ever more sophisticated network traffic models has led to the specification of a myriad of traffic models since. Many of these are listed in [91, 14]. In networks comprised of IoT devices much of the traffic is generated by devices which function autonomously and in a more deterministic fashion. Thus in this dissertation the activity of building time series models for IoT network traffic is undertaken.

In the work that follows a broad review of the historical development of network traffic modelling is presented tracing a path that leads to the use of time series analysis for the said task. An introduction to time series analysis is provided in order to facilitate the theoretical discussion regarding the feasibility and suitability of time series analysis techniques for modelling network traffic. The theory is then followed by a summary of the techniques and methodology that might be followed to detect, remove and/or model the typical characteristics associated with network traffic such as linear trends, cyclic trends, periodicity, fractality, and long range dependence. A set of experiments is conducted in order determine the effect of fractality on the estimation of AR and MA components of a time series model. A comparison of various Hurst estimation techniques is also performed on synthetically generated data.

The wavelet-based Abry-Veitch Hurst estimator is found to perform consistly well with respect to its competitors, and the subsequent removal of fractality via fractional differencing is found to provide a substantial improvement on the estimation of time series model parameters.

Table of contents

Acronyms

ABIDS	Anomaly Based Intrusion Detection System.
DDoS	Distributed Denial of Service.
DNS	Domain Name Server.
DoS	Denial of Service.
EWMA	Exponentially Weighted Moving Average.
IDS	Intrusion Detection System.
NIDS	Network Intrusion Detection System.
NTP	Network Time Protocol.
OS	Operating System.
SBIDS	Signature Based Intrusion Detection System.

List of figures

List of tables

Chapter 1

Introduction

1.1 Background

It is indisputable that the development of the Internet is chief among the contributing technologies responsible for the rapid development society is experiencing today. A quick search on the definition of the "Internet" yields the following: "A global computer network providing a variety of information and communication facilities, consisting of interconnected networks using standardized communication protocols[34]." Noting that the definition of the internet has at its core a computer network, we find that a computer network is defined as follows: "A computer network is a group of computer systems and other computing hardware devices that are linked together through communication channels to facilitate communication and resource-sharing among a wide range of users [11]."

As a result of the above-mentioned technologies, knowledge and ideas are being communicated ubiquitously. People are being connected to each other more of the time and to increasingly wider social, and business networks. Services and commodities are being traded with diminishing barrier to market entry. From small start-ups to multi-national conglomerates, banks, and government institutions, all have become unavoidably dependent on computer networks for the processing, storage, transportation, and sharing of resources within their respective organisations as well as for the provision of vital services core to their business or essential to the general public [33, 25].

The increasing ubiquity of web-based applications along with the advent of Internet of Things (IoT) technology has also led to a rise in the need for tools which facilitate the development and management of computer network based services. Issues such as network security and quality of service are no longer just the concern of ISPs and big corporations,

but have now become the intimate concern of private users with implications that directly affect the personal lives and businesses. At the heart of addressing these concerns lies the problem of modelling the networks on which interconnected devices now operate. While the activity of provisioning the networks and responding to threats may still lie in the hands of networking specialists, the ability to at least know when something is amiss remains instrumental in establishing the confidence and peace of mind of network users. Furthermore, by increasing user confidence and reducing the overhead of network management, it is expected that an increase in the adoption of web-based solutions will be facilitated. This is particularly critical for developing economies looking to leverage on online technologies and compete in global markets.

As with any powerful technology, it is not immune to abuse. In capable and malicious hands, the internet possesses the ability in equal measure to cripple organizations and compromise the security of entire nations. The threat is so great in fact that the US department of Defence has defined cyber-warfare as "the fifth battle-space" (after the land, sea, air, and submarine domains). For this reason network security has become essential to the protection of civil freedoms, economic opportunity, and national security[55]. In general, an attack on a computer network may be defined as an attempt to destroy, disable, disrupt, expose, alter or gain access to any unauthorised asset. Some of the most popular reasons for attacks on computer networks are:

- Revenge - disgruntled (ex)employees, customers, competitors
- Diversion/Cloaking - to cause distraction so as to mask other ongoing illegal activity
- War - governments' new alternative to physical war
- Politics - groups looking to make political statements or silence opposition
- Intellectual - individuals looking to either hone their skills or flaunt their hacking-prowess[64]

The major concern in recent times has been that the community of hackers or people with the skills to perform such attacks is growing rapidly. This is aided particularly by the fact that the tools employed for such exploits are becoming trivial to use and very easily available. Furthermore, the techniques implemented by these tools are constantly improving and becoming increasingly sophisticated. A critical component in the securing and defending of a computer network is the detection of attacks or intrusions on the network. To this

end, an Intrusion Detection System (IDS) can be employed to monitor operating system or network activities. By capturing and analysing audit data, gathered from network traffic or an application log, the IDS is then able to determine whether or not the system/network is presently under attack[87, 1].

As noted above, computer networks have become vital components in the operations of corporations and government institutions as well as in the daily lives of citizens. Attacks on computer networks pose a great threat to all stakeholders. While there are many ways in which computer networks can be attacked, by far the most commonly occurring attacks on the Internet today are Denial of Service (DoS) attacks. It is estimated that on average about 28 DoS attacks occur everyday[12]. These DoS attacks are capable of crippling the critical services of institutions and businesses, sometimes costing them millions in revenue, and disrupting the lives of many users. The emergence of ever more sophisticated and severe attacks, as well as the ease with which these attacks can be executed makes the Internet an increasingly unsafe place, thereby threatening its very existence. This necessitates the development and implementation of ever more capable network security solutions, in order to preserve the integrity of computer networks.

Generally speaking, network security is often implemented in a tiered approach, where multiple complementary tasks are performed in order to address various aspects of securing the network. Typical examples of such tasks would be user/machine authentication, firewalls, traffic/IP filtering, and intrusion detection[70, 64]. These are all different and important measures that can be implemented simultaneously in order to secure a network asset in a more comprehensive manner. A particularly important and somewhat tricky task in network security is being able to detect when a system is under attack, in order to implement counter-active measures. Systems that perform the detection of attacks or intrusions on a computer or its network can typically be divided into two classes: signature-based and anomaly-based intrusion detection systems.

Signature Based Intrusion Detection System (SBIDS)s make use of known patterns of "bad behaviour", i.e. signatures, to detect attacks and every variation of the same attack. While this type of detection system can detect many and all known attacks with very few false alarms, they are simply incapable of detecting unknown or "zero-day" attacks[44]. Anomaly Based Intrusion Detection System (ABIDS)s on the other hand make the assumption that all intrusive activities are necessarily anomalous. Thus if a model of "normal" activity can be formulated, any significant deviation from the model is signalled to be an attack. While this

approach is capable of detecting unknown attacks it is also very liable to flag unobserved or anomalous instances of legitimate usage as attacks too, that is to say ABIDS are prone to a high rate of false positives[44]. SBIDSs are highly effective and are in large part a solved problem, one good example is SNORT. However, they do not form a complete solution to network security in themselves. ABIDSs, despite all their shortcomings, remain very much an active research topic due to their attractive attributes of being relatively non-intrusive and thus easy to deploy, as well as having the ability to detect unknown attacks. For these reasons this research project focuses on model building in order to facilitate the development of ABIDSs.

A further division can be made between network based and host based detection systems. A Network based IDS would typically be inserted in the network, like a device, and examine every packet it sees on the wire. A host based IDS on the other hand would typically run as an application on the host machine, correlating the complex array of system specific parameters used to identify attacks[44].

It is a point that cannot be over-emphasized that there is no "cure-all" solution to network security. There are pros and cons to every approach. It is impossible to build an impregnable network system[63]. However, it is generally agreed upon that the best defence for a network is achieved when complementary techniques are employed in an appropriate fashion such as combining network-based and host-based IDSs and/or combining signature-based and anomaly-based IDSs.

This research project focuses on network-based intrusion detection. Note that no attempt is made to address the mechanisms of responding to or defending against an attack. The goal of this is to strive towards the creation of a tool that will give a network administrator awareness and fair warning of an imminent threat, with the hope that counter-active measures can be taken by the administrator to mitigate the attack. The reader is referred to the literature [9][64] for guidelines and ideas with regards to defence. Among the detection techniques studied, only those that fall within the category of anomaly detection are concentrated upon. While the study will not go as far as to specify any anomaly detection techniques, it will deal with the development of models which form the foundation for such activity to occur. These choices are further motivated by the following global outcomes:

1. To develop an IDS that is platform independent.

2. To develop an IDS that is deployable with the minimal amount of effort.

3. To develop an IDS that is as unobtrusive as possible.
4. To develop an IDS that is as widely applicable and hence adaptable as possible.
5. To develop an IDS that requires a minimal amount of human intervention (at least as much as is feasible).

1.2 Problem statement and hypothesis

It has been observed from literature that there are a plethora of algorithms that can be used to perform the modeling and detection of anomalies in network traffic. Each algorithm has its own strengths and weaknesses. Each brings to the table a different set of characteristics and behaviour which may make it more or less suitable in a given context than other competing algorithms. The issue of choosing an appropriate algorithm is thus a critical one. Unfortunately this choice is complicated by the fact that network traffic is highly complex. The nature of network traffic is such that it is always changing and cannot be relied upon to consistently conform to a specific statistical model. This means that any algorithm chosen to perform an anomaly detection task on network traffic is invariably going to come across instances of traffic that it is inept to deal with.

It is the hypothesis of this research project that there exists a subset of network traffic characteristics which tend to govern the observed behavior of traffic on computer networks. By identifying these dominant or common traffic characteristics at a subnet or host level, the traffic signal might become more amenable to analysis and could subsequently be modelled using time series analysis techniques.

1.3 Research Questions and Objectives

The research questions that are to be addressed in this study are provided in the section that follows.

The selection of an appropriate network modelling scheme, given the observed ambient conditions of a network, is a task that must be performed automatically and in real time, ideally, to be useful. Consequently, the use of time series analysis techniques to accomplish this task seems a promising avenue to investigate. Time series analysis in itself however is a very broad field, presenting the researcher with a large number of options with regards to approach and implementation. In order to attempt to make the most effective use of this processing tool, the following questions will be considered:

1. How are time series analysis techniques classified in the literature? The objective here is to gain some insight about the various classes of time series analysis techniques in order to focus on the set of classes that are most potentially useful to this research project, given the application and knowledge gained from preceding question sets.

2. What aspects of time series analysis algorithms limit their applicability in various scenarios? Ultimately the objective is to obtain a hard yes or no as to whether a given class of time series analysis can be used for the current application.

3. Which time series analysis techniques have been successfully applied in the past to network modelling? Is it possible to intelligently switch between modelling techniques based on the observed traffic conditions?

4. Which tests and/or metrics would be useful for discovering patterns in the performance figures obtained via experimentation? The goal is to determine which analysis techniques are able to find meaningful/significant patterns in results arising from repeated experimentation.

Chapter 2

Literature Review

In this section, a survey of the relevant literature is presented. The study commences by looking at general network security concerns leading to the study of denial-of-service attacks, which form the primary case study for many anomaly detection techniques. Having studied the domain of application a few of the early landmark techniques are presented followed by a coverage of the historical development of modelling in network traffic.

A secure network is characterised as having the following three attributes:

1. Data confidentiality - Data transferred through the network is only accessible to authorised parties.
2. Data integrity - Data received should be consistent with data sent i.e. without loss or corruption from either random events or malicious activity.
3. Data availability - The network should be resilient to denial of service attacks.

2.1 Denial of Service Attacks

There are a myriad of attacks and intrusion types that can be carried out, but by far the most common and most difficult to solve is the DoS attack (and all its variations). DoS attacks on computer networks are designed to render the victim network incapable of providing normal services. This is typically accomplished by compromising or choking resources on the victim network such as communication, network bandwidth, sockets, CPU usage, memory, router processing, database or disk I/O etc.[64]. DoS attacks can be particularly severe as they cannot be addressed with software fixes and they affect the target with little or no warning, and within a short period of time exhaust the computing and communication resources of the

victim[19, 31]. This also makes any attempts at a counter-active measure nearly impossible short of disconnecting. The effects of such attacks vary from minor inconveniences to major financial losses to businesses relying on on-line availability. This poses a huge threat, with high profile web-servers like search engines, social websites, commercial servers, payment gateways, root-name servers all being particularly attractive targets for perpetrators[4].

Launching these powerful attacks has become a trivial task, and unfortunately the detection of and response to these attacks tends to be slow and far from the desired level of competence that would allow counter-active measures to be taken in time[31]. The seriousness of the threat of denial of service attacks as well as the ease with which these powerful attacks can be launched has caused this study to focus on addressing the issue of detecting DoS attacks with a particular emphasis on exploring ways in which to improve the performance of IDSs in this regard.

There are numerous ways in which to classify DoS attacks. In [64] DoS attacks are categorised according to their attack vector i.e. the means by which the attack is carried out. These can be:

1. Volumetric - The attacker attempts to overwhelm the target system by sheer volume of traffic, consuming its bandwidth and leading to congestion. This can also be termed "bandwidth depletion" or "brute force". A further distinction within this category can also be made between flooding and amplification. While flooding requires the attacker to generate all the traffic used in the attack, amplification enables the attacker to reap traffic at least an order of magnitude more than the traffic they have to generate themselves. The technique makes use of reflectors i.e. hosts that respond to packets or requests with replies. Common reflectors used in DoS attacks are Domain Name Server (DNS) servers. In a typical case an attacker would send packets with a spoofed IP address to the broadcast address on the reflector's network[31]. This in turn solicits responses from all the hosts on that network. These responses are then routed to the spoofed address which belongs to the victim or target of the attack. Not only is this effective in amplifying the flood of packets in the direction of the victim, but it also hides the identity of the attacker.

2. Protocol exploitation also known as resource depletion is where various features or bugs of some protocol are exploited in order to consume resources. This is usually done by misusing the protocol or by sending malformed packets. A popular example of protocol misuse is the TCP-SYN attack (TCP state exhaustion). This attack exploits the feature within TCP's three-way handshake which allocates substantial memory in a connection queue immediately when TCP-SYN packets arrive. Essentially, a half

open connection is made and placed in this queue. The connection is completed once the client sends an acknowledgement packet back to the server. The attacker, instead, initiates multiple connections with the target server without ever completing them, thus filling up the queue and inhibiting the victim server from making connections to legitimate clients.

3. Malformed packet attacks attempt to solicit undefined behaviour in the target system by sending deliberately erroneous packets. Typical examples of this form of attack are the IP address attack, where the attacker sends packets with overlapping IP fragments. And the IP packet options attack, where the attacker sends packets with invalid/illegal TCP flags.

4. In application layer (OSI layer 7) attacks, the attacker attempts to exploit weaknesses in the application layer protocols such as HTTP, SMTP, DNS, SIP/VoIP etc. Some of the common examples of simple application layer DoS attacks are: HTTP GET flood and HTTP POST flood.(This category of attack is particularly difficult to detect)

5. All of the above-mentioned attacks can be further extended by emanating from multiple sources. In this setting the attack would typically be conducted in four distinct phases as follows:

   (a) Recruiting or selection of agents or zombies. These are machines with the required resources for conducting the attack. Often these are inadvertent participants in the process.

   (b) Compromise or exploitation of security holes on the identified agents.

   (c) Infection. In this phase the tools and code required to manipulate the agent and conduct the attack are planted.

   (d) The attack is conducted.

   This is what is called a Distributed Denial of Service (DDoS) attack[64]. In more recent times a modern version DoS attack has emerged which makes use of the Internet's Network Time Protocol (NTP). NTP is a protocol designed to synchronize the clocks of computers over a network using coordinated universal time. Once again the attacker, with a spoofed request as small as 234 bytes, is able to solicit responses from NTP servers, which may be located in dozens or even hundreds of locations all over the world. When this attack is coordinated with a botnet (network of compromised machines), the net effect is a flood of traffic towards the target that can exceed 100 Gbps[18].

One of the earliest costly incidents of denial of service attack was on 7 February 2000, when Yahoo, at that point the most visited site on the internet, was crippled by a relatively simple DoS attack. The attack resulted in massive revenue losses to Yahoo[64, 10]. In October 2002, root servers providing DNS to internet users were targeted by DoS attacks which shut-down 9 out of 13 of these servers. In September 2012 the websites of Bank of America (BAC), JPMorgan Chase (JPM), Wells Fargo (WFC), U.S. Bank (USB) and PNC Bank all fell victim to a series of denial of serve attacks, suffering day-long slowdowns and being sporadically unreachable to their customers. The Islamist group Izz ad-Din al-Qassam Cyber Fighters publicly claimed responsibility for the attacks terming their exploits "Operation Ababil"[17]. In March 2013, Bit-Coin exchanges Mt. Gox and Dwolla were hit with DoS attacks to the extent that they had to be taken off-line temporarily[17]. The largest observed DDoS attack was launched against Spamhaus in 2013. The attack reached peak levels of 300 Gbps[64]. Recently, in Jan 2014, gaming sites, such as League of Legends, have become the targets of 100 Gbps DDoS attacks[18].

These are of course but a few high profile examples of the havoc being reeked by denial of service attacks. It is estimated that as much as 7000 DoS attacks are launched daily[61]. These attacks are also growing, with an average size of 1.77 Gbps in 2013, a 19 percent increase on the previous year. The majority of attacks, 62.4 percent (as at 2013), are still under 1 Gbps, but this proportion is dropping rapidly.

In addressing the defence of a network against attack, intrusion detection systems form a vital component of any system that aims to provide security in a network system. Intrusion detection systems perform the function of monitoring and analysing system activity, performing statistical analysis in order to identify abnormal activity patterns, which deviate from existing models and ultimately recognise activity that is indicative of an attack. These objectives are usually achieved in mainly two different and somewhat complimentary fashions. These approaches by and large can be distinguished by the input data they make use of.

In section 2.2 and 2.3 host based and network based intrusion detection systems will be discussed respectively in more detail. This will be followed by a similiar discussion of signature based and anomaly based intrusion detection systems in 2.4 and 2.5 respectively. A brief discussion of the various approaches to network defense is presented in section 2.6.

2.2 Host Based Intrusion Detection

Host-based intrusion detection systems make use of data recorded by operating system mechanisms, called audit trails, to collect information about the activity of a particular single system or host. These audit trails are essentially logs written in text files a few lines at a time

as events occur and operations are executed on the system. This type of intrusion detection attempts pick up on subtle patterns of misuse. Indeed attacks which exploit vulnerabilities that are specific to certain operating systems can be very difficult to detect. Thus the ability of host-based IDSs to correlate complex arrays of system specific parameters that might constitute the signature of an attack is of great benefit[44].

The audit trails also allow the IDS to trace users and processes associated with a given event, enabling it to identify compromised users and inside jobs[63], where authorised users use local resources in a manner that violates the security policy[44]. There is also some comfort to be had in the fact that the audit trails are somewhat protected by the operating system itself, as Operating System (OS)s have an existing aim to protect their audit layers[63]. The use of host-based IDSs has the added advantage of being scalable, as the load associated with monitoring the network is evenly distributed among the available hosts on the network.

Host-based intrusion detection does, however have a number of drawbacks. Host-based IDSs are intimately tied to the OSs on which they operate. This dependence means that any vulnerability on the host OS weakens the integrity of the IDSs. Not only this, but the IDS becomes another application that must be maintained by the system and migrated when required. Needless to say the IDS must be compatible with the platform on which it is to run. This presents quite a burden especially in environments that are constantly being updated[63, 44]. Host-based IDSs do not see network traffic at all[63]. This makes them completely unable to protect against basic network layer attacks[44]. It can also be argued that host-based IDS have high set up and deployment costs, given that each host requires an individual sensor, and management[63].

2.3 Network Based Intrusion Detection

Network-based IDSs operate by collecting data from the network itself. This is achieved by scanning the contents of network packets i.e. "packet sniffing". Network based IDSs are inserted in the network as a device, which then promiscuously examines every packet it sees on the wire[44, 63]. This has a number of attractive attributes. Network-based IDS are easy to deploy, as they have no dependence on any operating system, making them highly portable. They can be inserted as part of the network, and data can be collected by the mere configuration of a network card. They may even be integrated onto ports on ethernet switches which provide visibility to all packets on the wire. This is particularly beneficial in network topologies that are liable to change. They also run without degrading the performance of any applications running on the network.

Network-based IDSs do have the rather ominous drawback of not being scalable. This is because network monitors must inspect every packet that passes through its segment. It has been noted in [63] that traffic volumes of around 100 Mbps are sufficient to cause most IDSs to struggle. This makes network based IDSs particularly susceptible to denial of service attacks, especially if it is single-handedly servicing an entire subnet.

2.4 Signature Based Intrusion Detection

When monitoring a network for security threats, an IDS can implement one or both of two detection techniques, namely signature(misuse)-based detection and/or anomaly(behaviour)-based detection.

Signature or misuse-based detection makes use of a database of defined signatures for matching against well-known attack patterns. A signature generally refers to a set of conditions that characterises the pattern of an intrusive activity. Because of the ability of signature-based IDSs to detect many if not all known attacks with very few false detections, historically they have been the more commonly implemented method of intrusion detection. However, in the case of novel or unknown attacks, they are virtually useless. This is a massive drawback, especially considering the amount of time and resources hackers dedicate to evading such systems[20, 44].

2.5 Anomaly Based Intrusion Detection

Anomaly-based intrusion detection is a technique whereby actual system activity is compared to an established model of "normal" behaviour. It can be seen as a two step process that involves:

1. Training a system with data to establish some notion of normality. This can be achieved via off-line learning and research and subsequently programmed into the system. Or it can be achieved on-line while processing the network traffic.

2. Using the established profile on real data to flag instances of behaviour that deviates from the "normal" profile.

If we consider that anomaly-based detection tries to detect the complement of "good" behaviour, we can see that it allows for the detection of a much broader range of novel or unknown attacks[44]. Also, since it can detect any abnormal traffic behaviour, it makes an excellent candidate for early warning of potential threats[20].

One of the notable flaws of anomaly-based detection with regards to intrusion detection is that it assumes all intrusive activities are anomalous and all anomalous behaviour is necessarily intrusive. In reality, of course, we know that the set of intrusive activity only intersects the set of anomalous activity and they are not exactly the same. This presents the following challenges:

1. anomalous activity that is not intrusive is flagged as an intrusion, i.e. false alarm, and that
2. intrusive activity that is not anomalous is not detected, i.e. missed detection

These difficulties are further exacerbated by the ever changing nature of networks and applications, with network traffic being extremely complex and having dynamically changing statistics. It is clear to see then that the profile relied upon by the anomaly detection system is most key to its success. Consequently, some of the most desirable characteristics that make up a good profile are that it must:

1. be stable and consistent in tracking "normal behaviour",
2. b sensitive to events that pose security threats,
3. be adaptive, to account for the normal changes in the network without raising false alarms.
4. be self-learning, to ensure successful deployment under a wide spectrum of conditions[20].

It is also quite evident that understanding the characteristics of the traffic itself in the target system is critical in being able to select or synthesise a model that is successful.

2.6 Defence Mechanisms

Among the defence mechanisms implemented in network security, a distinction can be made between them based on the location of the implemented defence mechanism. Source-based IDSs basically focus on restricting network customers from generating or participating in denial of service attacks[64]. Such defence mechanisms would be deployed at the source router that servers a gateway between the source network and the rest of the internet. In this way, by being deployed close to the source, the attack flows can be mitigated before they enter the core internet where they would blend in with other flows[36].

Egress filtering (i.e. filtering of outgoing packets) attempts to combat denial of service attacks by specifying filters that only allow packets with valid source IP addresses for the originating network. This is of course based on the assumption that denial of service attacks almost always make use of spoofed IP addresses. Even if this doesn't stop the attack from occurring, in the case of legitimate addresses being used, it should give a network administrator the information they need to block the offending network(s)[37].

D-Ward makes use of a set of addresses whose outgoing traffic must be policed. It monitors the two-way traffic flows between the policed addresses and the rest of the internet. Online traffic statistics are then generated and compared to predefined models of normal traffic. Any flows detected to be non-compliant are rate limited, and every subsequent re-confirmation of non-compliance results in further rate limiting. D-WARD assumes it can identify the police address set, and that all machines in the list use the source router where D-WARD is deployed as the exit point[36].

MULTOPS proposes a MUlti Level Tree for On-line Packet Statistics as a data structure to be used in conjunction with a heuristic in order to detect denial of service attacks. The tree consists of nodes that contain packet rate statistics for subnet prefixes at different aggregation levels. An attack is detected when there is a disproportional difference between incoming and outgoing packet rates in a given flow. Packets are subsequently dropped within that flow. The assumption made here is that under normal conditions, the rate of packets from point A to point B should be proportional to the rate of packets from point B to point A. This is motivated by the fact that an acknowledgement is expected for every packet or set thereof that is sent[24].

These approaches require many routers, at different network entry points, to each independently deploy the given source-based defence mechanism. This represents a major challenge in deploying these source-based defences, since the immediate benefit of these is felt by the victim and not the deploying network[36].

Destination-based defence mechanisms are those that are deployed close to the victim, at the edge/access router of the destination. IP trace-back is a technique in this category which is used to identify spoofed users. Packet marking and filtering mechanisms can also be used to make a distinction between legitimate and attack traffic. This can be achieved in a number of ways including history-based IP filtering, hop-count filtering, path identifier, and packet dropping based on congestion[64].

There are also network-based defences that can be deployed inside networks and on the routers of autonomous systems. These typically use defences such as route-based packet filtering, and filtering of malicious routers[64].

2.7 Change-Point Detection

In the sections that follow some of the early algorithms used in anomaly detection are introduced. These can alll be broadly grouped into the category of change point detection algorithms. In particular, section 2.7.1 summarises the exponentially weighted moving average algorithm (EWMA), section 2.7.2 introduces the likelihood ratio, which leads to the of the hypothesis testing based CUSUM algorithm in 2.7.3.

The problem of detecting when a system is under attack can be presented thus: Given an incoming stream of packets, extract a set of observables (e.g. traffic rates, TCP-flag counts, UDP packets, etc.) and generate a time series of measurements for each. Assume that should an attack occur within this stream, the statistics of some or all of the observables will change. If an attack begins at unknown time, T, detect the change in statistical nature of the observables as close as possible to time point T. This can be considered a classic change-point detection problem i.e. discovering time points at which properties of time series data change[42]. It is noteworthy that choosing the relevant network parameters to be observed is a crucial aspect of the development of an ABIDS.

Within the statistics community change-point detection has been an active field of research for several decades now. A typical formulation in change-point detection is to consider the probability distributions from which the observed data is being generated[42]. Say we have observations $X_1, X_2, ..., X_n$ generated by a certain process according to distribution F_0. At unknown time, T, something happens to the process so that the samples generated are now distributed according to F_1. The objective is then to detect this change in distribution "as soon as possible" after its occurrence. A few of the pertinent algorithms that are typically employed in change-point detection are presented in the sub-sections that follow.

2.7.1 Adaptive Threshold Algorithm

One of the simplest criteria for detecting a change in the mean of a distribution is a weighted sum of the last few, k say, observations, i.e. a moving average[26]. This is a simple technique which is based on estimating the mean of the observed values of some time series generating process. With each new observation, x_n, the value of x_n is compared to an estimate of the mean, u, which is based on previous observations. If x_n should exceed u by a certain threshold, a, i.e. $x_n > (1+a) * u$, then an alarm is signalled at time n[75]. Following the comparison of the latest observation, x_n, with the previously estimated mean, u, the mean is then re-estimated incorporating the current observation in the estimate, thus making the

threshold adaptive or sensitive to the latest trend in the data. The algorithm further takes the seasonality of trends in the data into account by employing an Exponentially Weighted Moving Average (EWMA) to estimate the mean of the previous observations. The EWMA procedure is given by:

$$u_{\mathrm{n}} = b * x_{\mathrm{n}} + (1-b) * u(n-1) \tag{2.1}$$

where b is the $EWMA$ factor.

Expanding this formula over p iterations shows that the current observation at time n receives a weight of b, while the observation from earlier time $n-p$ receives a weight of $b(1-b)^p$. It is evident then that should one be interested in preserving the influence of observations for a longer period of time a small value for b should be used, while the converse is true should only the short term trend be of interest[94]. This technique is known to generate a high number of false alarms, thus it is common that an additional parameter is specified. This parameter, k, is the number of consecutive threshold violations required to trigger an alarm. The false alarm is inherently affected by the threshold and smoothing factors as well. We would like to minimise this false rate, or at least keep it constant, while maintaining a high detection rate so as to remain useful. We would also like to detect the change as soon after its occurrence as possible. All these objectives have competing interests and thus are all at a trade-off to one another. A combination of these parameters that best satisfies these objectives for the given application must be found. This technique, consequently, leaves us with three parameters, a, b, and k to tune or optimise[94].

2.7.2 Likelihood Ratio

Say some samples, y_n, are observed of a process distributed according to a density function, p_0, with a parameter P_0. Should a change occur in the process the samples would be distributed according to some other distribution, p_1, with parameter P_1. We would like to know whether a given sample is still distributed according to the original distribution or if the distribution of the samples has changed. Stating this as a hypothesis test:

$$\begin{aligned} H_0: &\quad P = P_0. \textbf{ The data is distributed according to } p_0. \\ H_1: &\quad P = P_1. \textbf{ The data is distributed according to } p_1. \end{aligned} \tag{2.2}$$

The ratio of the likelihood functions, $p_0(y)/p_1(y)$, can be used to make a decision as to whether the null hypothesis still holds or not. It turns out that it is convenient to take the ratio of the logarithms of the likelihood functions. This gives the following:

$$s(y) = \ln\left(\frac{p_0(y)}{p_1(y)}\right). \tag{2.3}$$

This is referred to as the log-likelihood ratio. A key property of this ratio is that if we take the expectation under the two distributions p_0 and p_1 respectively, then $E(p_0)s < 0$ and $E(p_1)s > 0$. In other words, a change in the parameter P is reflected as a change in the sign of the mean value of the log-likelihood ratio [3]. It is commonly observed that the behaviour of the log-likelihood ratio shows a negative drift before change, and a positive drift after change. This gives us the intuition that the difference between the log-likelihood ratio and its current minimum value is the most relevant quantity with regards to change detection [3]. Use of the likelihood ratio test can be further justified by the Neyman–Pearson lemma, which shows that the log-likelihood ratio test has the highest power among all other tests, assuming all the parameters of the two distributions being compared are known. The log-likelihood ratio has a particularly restrictive feature, however, of requiring prior knowledge of the pre and/or post change distributions [78]. This is problematic as in most practical situations this information is simply not available.

2.7.3 CUSUM

CUSUM belongs to the family of change-point detection algorithms that are based on hypothesis testing[75]. It assumes the conventional change-point setting where a process produces a random variables $X(1), X(2), \ldots$ which are independent and identically distributed (i.i.d.). This is assumed to be the case until a change occurs at unknown time, T, at which point the observations are again i.i.d, however now, with a different distribution [78]. If we let p_0 and p_1 denote the probability density functions before and after the change respectively, the Page's CUSUM procedure is then to compute:

$$\frac{\max(1 < k < n) p_1(x(k)) * p_1(x(k+1)) * \ldots * p_1(x(k+n))}{p_0(x(k)) * p_0(x(k+1)) * \ldots * p_0(x(k+n))}. \tag{2.4}$$

for every time instant observed[26]. A change can then be declared to have occurred if the above statistic exceeds some threshold, h. It was shown by Moustakidis (1986) that if f1 was in fact the true probability density of the post change distribution, then the CUSUM procedure is the quickest to detect the change in distribution among all other known procedures, for a given bound on the false alarm rate[26].

An alternative statistic was proposed by Shiryayev (1963) and Roberts (1966), in which

$$\sum_{k=1}^{n} \frac{p_1(x(k)) * p_1(x(k+1)) * ... * p_1(x(k+n))}{p_0(x(k)) * p_0(x(k+1)) * ... * p_0(x(k+n))}. \tag{2.5}$$

was computed, once again, for every observed time instant. The change in distribution being asserted, once again, when the statistic exceeded some threshold. This procedure, called the Shiryayev-Roberts procedure, was shown by Pollak (1985) to be asymptotically optimal, if p_1 was indeed the true probability density of the post change distribution, with a detection delay (speed of detection) that has been shown to be comparable to that of Page's CUSUM [26].

Of course we see from the above that at the core of both procedures is the likelihood ratio $p_1(x)/p_0(x)$. Thus taking the log we obtain the following for the log-likelihood ratio, Z_n:

$$Z_{n,i} = \sum_{k=1}^{n} \ln \frac{p_0(X(n)|X(1),...,X(n-1))}{p_1(X(n)|X(1),...,X(n-1))}, \text{ with } i \leq n. \tag{2.6}$$

While both Page's CUSUM and Shiryayev-Roberts' procedures have desirable properties, their dependence on the log-likelihood ratio, means that they inherit its inherent restrictions. Namely that either one or both of the initial and post change distributions must be known. Furthermore, the requirement that the observations be i.i.d is extremely restrictive as this just simply is not the case in most real-world applications, and especially in network intrusion detection.

Nonetheless, many attempts have been made to circumvent these shortcomings. In [78] the likelihood ratios $p_0(X(i))/p_1(X(i))$ are replaced with a score function. The score function can then be specified as a function of some other statistic of the data that can readily be estimated, for example the mean. A similar approach is used in [75] where an exponentially weighted moving average is used to estimate the mean. Other approaches used to address the i.i.d. restriction make use of the Holt-Winters algorithm to remove trends and seasonality in the data, while autoregressive algorithms are used to remove correlations. These techniques, however tend to add substantial cost with regards to processing with very little gain in performance[78].

2.8 Historical development

Early work on network traffic modelling, based on principles from telephony, proposed the Poisson process as a model [58, 91, 14] for the packet arrival process. The inter-arrival times

of network packets were considered to be exponentially distributed, with the density function of the model given by $(t > 0)$:

$$f(t) = \lambda e^{-\lambda t}. \tag{2.7}$$

The Poisson model is considered suitable when the arrivals are assumed to emanate from a large number of independent Poisson sources. The Poisson distribution is such that the superposition of these Poisson sources gives rise to another Poissson process whose rate is the sum of the rates of the independent Poisson processes. The mean and variance of the Poisson distribution are also given by the rate parameter, $\lambda > 0$ [14].

In time, the emergence of modern high speed networking technologies as well as the plethora of applications and services that came into use meant that the Poisson model was no longer able to fully capture the complexity of network traffic [91]. In the seminal work of [47], the authors performed an empirical study of ethernet LAN traffic to provide evidence of the hypothesised self-similar, fractal, and scaling behaviour of network traffic. The work gives a mathematical presentation of the properties of self-similar processes as well as two approaches for modelling network traffic exhibiting these behaviours. In their discussion regarding the significance and application of self-similarity to network traffic, Leland et al. posit that self similarity presents itself in ethernet traffic due to the absence of a "natural" burst length, where instead burstiness manifests on a wide range of time scales such that "traffic spikes ride on long term ripples, which ride on longer term swells, etc." [47]. This is what is said to give ethernet traffic its self-similar or "fractal quality" [47].

In their subsequent work, Paxson and Floyd (1995)[58] used traces of wide area network traffic in an empirical study of the error that is produced when using the Poisson process to model TCP, FTP and Telnet sessions, connections and packet arrivals. These authors also concluded that "ethernet traffic is better modelled as a self similar process..."[58]. Willinger et al. (1995) responded to the findings of [47] and [58] by providing a physical explanation for the self similarity that is observed in modern network traffic. In this work, the authors employed the ON/OFF source description of network traffic, also known as the "packet train model", to show that if each individual ON/OFF source is characterised by the "Noah Effect" (high variability or infinite variance i.e. very long ON or OFF periods occuring with non-negligible probability), then effectively these individual sources have characteristics which manifest on a wide range of time scales. The aggregate effect of these sources is then to produce network traffic that is self-similar and/or Long Range Dependent (LRD) - termed

the "Joseph Effect" [92].

Traditional models, that is up until the discovery of self-similarity in network traffic, had always been assumed to be finite variance models; like the exponential and geometric distributions. The result of this assumption is that aggregated traffic was then modelled as having no significant correlations in the long term, which is contrary to what is observed in reality [92]. This self-similar and LRD model of network traffic is seen to have provided a significant advancement to our understanding of network traffic dynamics [65]. Infinite variance or long range dependent processes are typically modelled using the class of heavy tailed distributions. These distributions, which include some Pareto and stable distributions, are usually parameterised by a heaviness-of-the-tail parameter α. Alpha is also related to the Hurst parameter, which represents a measure of the degree of self-similarity in a signal [92, 47].

In the works of Simross-Wattenburg et al. [74, 73] the alpha-stable distribution was used to model the marginals of binned network packet counts. Based on these marginals a generalised likelihood ratio test (GLRT) was applied towards classifying the observed traffic into normal and anomalous traffic patterns and hence the detection of flash crowds and denial of services attacks. By modelling network traffic according to the alpha-stable family of distributions, which are statistical distributions with heavy tails, the authors posit that they are able to account for the high variability which manifests from the bursty nature of network traffic. The use of the alpha-stable family of distributions is further motivated by the fact that they are the limiting distribution in the Generalised Central Limit Theorem (GCLT) which states that the sum of random variables, which are distributed according to heavy tailed distributions i.e. distributions with tails that decay as: $|x|^{-\alpha-1}$, where $0 < \alpha < 2$, tend to be distributed according to a stable distribution as the number of random variables becomes large. The Gaussian distribution is a special case where $\alpha = 2$.

Following the discovery of the fractal nature of network traffic, a popular approach became to use fractional Brownian motion (fBm) or rather its increment process, fractional Gaussian noise (fGn) as a means of modelling network traffic[91, 54, 81]. In addition to the long term characteristics of network traffic, Riedi and Willinger [65] studied the small time scaling behaviour of network traffic. This lead to the proposal of multifractality, which was attributed to the network protocol mechanisms. See [67, 66, 86] for some of the multifractal models applied to network traffic. The case of multifractality in network traffic was however disputed in [84], where the authors cite the multifractal behaviour that had been observed as

"...a misinterpretation due to a lack of power in the statistical methodology[84]".

In the work of Scherrer et al. (2007) [69], network traffic is modelled using a Gamma distribution to fit the marginals and an ARFIMA process to fit the temporal characteristics of the data. The Poisson distribution is said to represent one extreme where the data is not very aggregated, while the Gaussian distribution emerges for highly aggregated data, as suggested by the Central Limit Theorem (CLT). Gamma distributions are chosen in [69] because they are able to describe data that has a distribution which lies within the transitionary area between the Poisson and Gaussian distributions. This, in turn, allows them to provide a description of network traffic that is applicable on a wider range of scales. The ARFIMA process introduced to the model is particularly well suited to describing the covariance structure of the data, including its short and long range dependence characteristics. The ARFIMA process, in fact, represents a whole family of fractionally integrated processes, which [69] chooses to restrict to the subclass of ARFIMA $(1,d,1)$ processes, where d is the fractional order of differencing determined from the data. In the next section the basic precepts of time series modelling are introduced, which will lead into a discussion of existing implementations of time series analysis based models for network traffic.

2.9 Related Work

In this section the work which has been performed towards automating the task of time series analysis and the building of seasonal models is presented.

In [71] a seasonal ARIMA model is used to model wireless GSM traffic. The authors proceed by performing a spectral analysis of the data in order to identify cyclical patterns in the data. This is used to compose a traditional multiplicative seasonal ARIMA model. The authors extend this framework by providing an expression which allows for two periodicities to be incorporated into the model. The proposed procedure then proceeds along the same lines as method 2 of [93] with the additional assumption that the order of the fitted models can be limited to the range $[0,2]$.

Procedures for automatically determining the parameters p and q are proposed in [93]. Therein the authors propose, firstly, exploring every possible autoregressive, AR(p), representation of the data choosing the combination of p, d and s that results in the lowest value of the Aikake Information Criterion (AIC). Secondly, a grid search of all possible combinations of p and q may be conducted, where once again, the minimisation of the AIC is used as the objective criterion. This can be extended to include candidate values of s and d. This is

however somewhat of a brute force approach, which may be expensive in terms of memory and computation.

In [95], the instability of model selection is discussed and a number of metrics to quantify the stability of model selection are provided. In the event of having multiple candidate models for a prediction task, instead of attempting to find the "true model", the Aggregated Forecast Through Exponential Re-weighting(AFTER) algorithm is proposed. This entails running multiple competing models at the same time and weighting their output according to their performance in previous time instances. In this way the stability and accuracy of the resulting forecasts is improved. The success of this scheme is of course still dependent on having at least identified suitable candidate models to combine.

Laner et al. in [45, 46] motivate an approach for modelling and simulating network traffic based on three sequential transformations of a zero mean, unit variance, i.i.d, Gaussian noise process. These transformations act to modulate the cumulative distribution, autocorrelation and cross-correlation functions of I independent Gaussian noise processes in order generate output processes that resemble real network traffic. The first transformation forms a weighted sum of the I input processes in order introduce cross-correlation between them, thus simulating the cross-correlation that occurs between the traffic streams of separate applications running concurrently on a network. The second transformation performs a linear time invariant (LTI) filtering of each process, in the form of an ARMA transfer function, in order to introduce the desired autocorrelation structure to the samples. The third transformation is a memoryless polynomial transformation, which shapes the distribution of the output processes. Laner et al. advocate that treating the modelling and simulation problem according to these three separate concerns results in analytically tractable, parsimonious, efficient and low complexity means of generating synthetic network traffic samples. This however remains to be proven.

In [35], Iqbal et al. study the power and performance of online, one-step ahead, traffic predictors. The authors posit that by accurately predicting traffic and identifying periods of idling or low traffic, a given system can be placed into a low-power state thereby using processing resources more efficiently. To this end, predictors from three categories are studied, namely time series analysis based predictors, artificial neural networks(ANNs), and wavelet transform based predictors. The results of the study show that the Double Exponentially Weighted Smoothing (DES) and ARMA models incurred some of the lowest computational overhead among the compared techniques, while also performing the best in terms of prediction accuracy over the Caida(DES was the best), Auckland and Bellcore (ARMA was the best) datasets.

In their paper Stadnytska et al. (2008) evaluated the Minimum Information Criterion (MINIC), smallest canonical correlation method (SCAN), and the extended sample autocorrelation function(ESACF) procedures for automatic ARMA model identification. The details of these procedures can be found in [76] and the references therein. While the study exposed the ineptitude of these techniques, they still retain utility as evidenced by their inclusion in SAS. This shows that the development of automated model identification algorithms, however crude, provides significant value.

In their paper, Tran and Reed (2004)[79] investigate the application of time series analysis to the prediction of temporal I/O arrival paterns. Of particular value to the discussion in this work are their contributions towards automatic detection of non-stationarity and seasonality for model identification. The authors devise an approach that makes use of the fact that after sufficient differencing, the significant spikes that remain in the autocorrelation and partial autocorrelation functions, if positioned at regular intervals, are indicative of seasonality. The distances (in lags) between successive spikes is computed, taking the distance that occurs most frequently as the period of the suspected seasonal component [79].

In the paper by Huang and Shih (2003) [30], ARMA models are proposed for use in making short term forecasts of load in power grid systems. The proposed method incorporates the use of higher order cumulants to determine the order of AR and MA processes that model a given time series. The premise of the procedure is that the higher order cumulants (> 2) of a Gaussian process are zero. The non-Gaussian process can thus be distinguished from the Gaussian component [30]. The authors go on to prescribe the use of the bispectrum, which is a 2D Fourier transform of the third order cumulant, to test for Gaussianity. The subsequent model identification then follows two paths. If the stationary time series is found to be Gaussian, then model identification proceeds in the usual fashion via analysis of the ACF and PACF of the series. Alternatively, the model identification is performed via a cumulant-based order determination procedure, the details of which can be found in [50].

Chapter 3

Theory

In this chapter some of the basic theory required to formulate the modelling approach undertaken in this work is laid out.

In the past many schemes have been proposed to perform anomaly detection on network traffic based on statistical tests such as the log-likelilood ratio tests [3, 78], CUSUM [57, 75, 78, 26], EWMA [94, 26], Kulback-Leiblar (KL) divergence test[13] and others. These tests however are employed based on the underlying assumption that the samples on which they are applied are identically and independently distributed (i.i.d). It has been shown in the literature [91, 92, 47, 58] that the assumption of independence in particular does not hold when it comes to network traffic. There have been many attempts to address the dependence of network traffic by employing ever more sophisticated techniques to accommodate the complex nature of network traffic which includes features such as non-stationarity, long range dependence, and self-similarity[91, 92, 47, 58, 54, 81, 65].

In the present work, the well established tools of time series analysis will be levereged in order to attempt to model network traffic as parsimoniously as possible while taking into account the dependent nature of network traffic as well as addressing its characteristic of non-stationarity. It will be shown that, using some of the basic tools of model building in time series analysis, the network traffic signal can be modelled sufficiently to reproduce the characteristic features of the network traffic signal in terms of its dependence structure, covariance, and power spectrum. The suitability of the identified models can then be assessed using some of the prescribed diagnostic tools in time series analysis. This chapter proceeds by introducing the concepts of a stochastic process, the joint probability function and its moments in section 3.1. These definitions are then built upon in section 3.2 to define the autocovariance and autocorrelation functions of a stochastic process. Section 3.3 presents

an initial formulation of the modelling problem in terms of a linear filter. The stationarity and invertibility of the transfer function corresponding to the linear filter is then derived in sections 3.4 and 3.5. Using the linear filter formulation established in the previous section, the autoregressive (AR) and moving-average (MA) models are introduced in sections 3.6 and 3.7 respectively. In section 3.8, the AR and MA models are combined to form the mixed autoregressive moving-average (ARMA) model. In sections 3.9, 3.10 and 3.11 the autocorrelation functions of the AR, MA and ARMA models are derived.

3.1 Stochastic Processes

A stochastic process may be considered to be some phenomenon or system which generates values that are non-deterministic. Such processes typically give rise to time series, which are sequential observations of the generating process taken at equidistant points in time. Denoting the N successive time instances when observations of the process are made $t = 1,2,...,N$, we obtain the time series $\{z(t)\} = \{z(1),z(2),...,z(N)\} = \{z_1,z_2,...,z_N\}$. It is noted that this set of observations indexed by time (a time series) is but a single realisation of the infinitely many that could have been realised from the generating stochastic process [6].

Each observation in the time series is a specific manifestation of a single random variable drawn from a population of possible values that are distributed according to some probability density function $f(z_t)$. In the aforementioned time series, z_t, we thus have N random variables distributed according to N probability density functions $f_1(z_1), f_2(z_2), ..., f_N(z_N)$. The N observations $z_1, z_2, ..., z_N$ are what is referred to as a sample realisation.

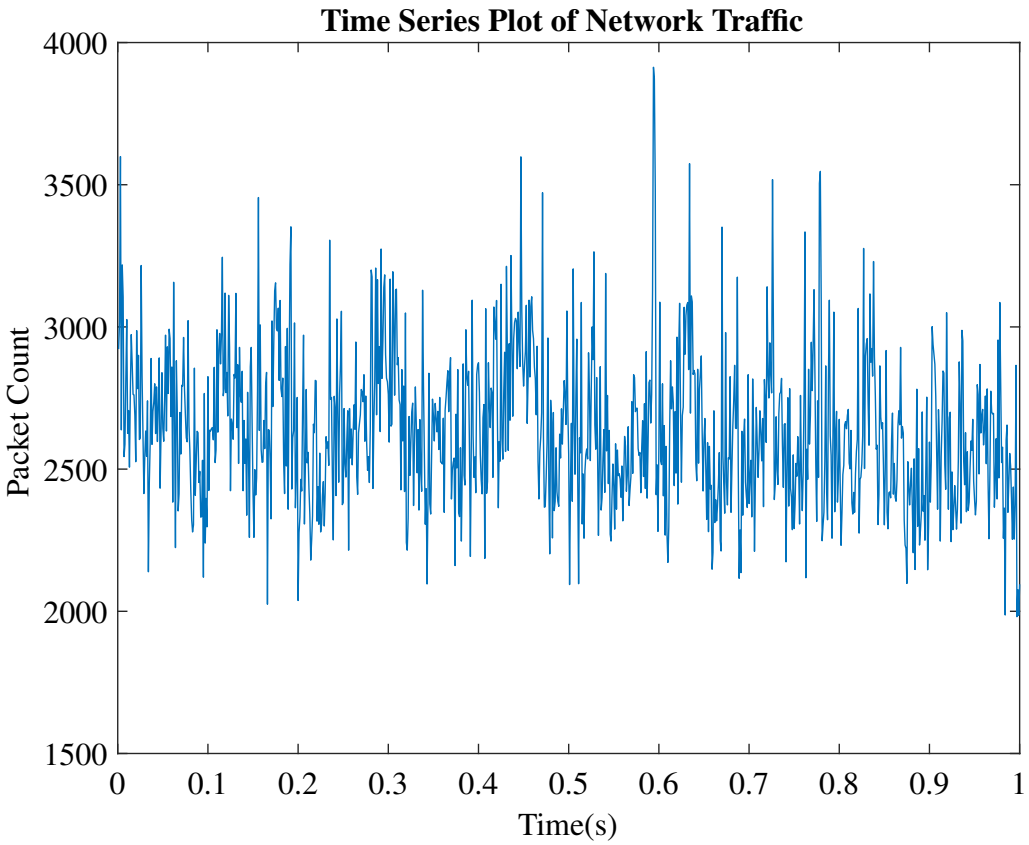

Fig. 3.1 Plot of 1000 observations of a network traffic process. The observations are taken at equidistant points in time, with all observations being separated by $\Delta t = 0.001s$.

3.2 Deriving the Moments of the Joint Distribution

Each pair of random variables z_k, z_{k+j}, corresponding to the sample realisation, are subject to a joint distribution of the two random variables $f_{k,k+j}(z_k, z_{k+j})$. By extension, the entire set of random variables leading to the sample realisation is subject to a joint distribution $f_{1,2,...,N}(z_1, z_2, ..., z_N)$.

The first step in attempting to model an unknown stochastic process is to study the characteristics of a realisation of that process. The most basic way of characterising a time series is by virtue of its mean values. Considering the random variables z_t to be distributed according to a multivariate probability density function, we obtain the following expressions for the moments of the distribution about the origin:

$$\mu_{n_1,n_2,...,n_N} = E[Z_1^{n_1} Z_2^{n_2} ... Z_N^{n_N}]. \tag{3.1}$$

where $n_1, n_2, \ldots, n_N$ denote the n^{th} order moment for $Z_1, Z_2, \ldots, Z_N$ respectively, and can take on any integer values $1, 2, 3, ...$ The discrete case, amounts to

$$\sum_{i_1}\sum_{i_2}\cdots\sum_{i_N} z_1^{n_1} z_2^{n_2} ... z_N^{n_N} f_{1,2,...,N}(z_1, z_2, ..., z_N). \tag{3.2}$$

Setting $n_1 = 1$, which corresponds to the first moment of the random variable Z_1, and set the rest of the moment indices to zero, $\mu_{1,0,0,...}$ the mean of Z_1, $E[Z_1] = \mu_{Z_1}$ is obtained. Likewise, for $\mu_{0,1,0,...}$, the mean of Z_2, $E[Z_2] = \mu_{Z_2}$ is obtained. For $\mu_{0,0,1,...}$, the mean of Z_3, $E[Z_3] = \mu_{Z_3}$ is obtained and so on [60]. This in effect provides the equivalent of determining the first moment of the individual marginal distribution of each random variable independently.

$$E[x] = \int_{-\infty}^{\infty} x f(x) dx. \tag{3.3}$$

where $f(x)$ is the probability density function of the random variable x.

A particular case, which is of great value to time series analysis, is when the expected values of the random variables Z_t at all time instances are the same, i.e.

$$\mu = E[Z_1] = E[Z_2] = \cdots = E[Z_N]. \tag{3.4}$$

Then, the stochastic process from which the realisations z_t are generated is said to be mean-stationary. Furthermore, the expected value μ can be estimated by the sample mean given by

$$\mu = \frac{\sum_{t=1}^{N} z_t}{N}. \tag{3.5}$$

Now consider the $(n_1, n_2, ..., n_N)^{th}$ moments of the joint distribution about the means. The following general expression is obtained:

$$\mu_{n_1,n_2,...,n_N} = E[(Z_1 - \mu_{Z_1})^{n_1}(Z_2 - \mu_{Z_2})^{n_2}...(Z_N - \mu_{Z_N})^{n_N}]. \tag{3.6}$$

which, in the discrete case, amounts to

$$\sum_{i_1}\sum_{i_2}...\sum_{i_N}(z_{i_1} - \mu_{Z_1})^{n_1}(z_{i_2} - \mu_{Z_2})^{n_2}...(z_{i_N} - \mu_{Z_N})^{n_N} * f_{1,2,...,N}(z_1, z_2, ..., z_N). \tag{3.7}$$

where the indices $i_1, i_2, ..., i_N$ run over the entire set of possible values for the realisations of the random variables $Z_1, Z_2, ..., Z_N$ respectively. Note that on setting all the indices $n_1, n_2, ..., n_N$ to zero except for one, we obtain the moments of the univariate marginal distribution of the selected variate. As an example, it can be seen from above that setting all the indices $n_2, n_3, ..., n_N$ to zero and selecting the index n_1 to be one the first moment of the marginal distribution of Z_1 is obtained. Similarly, setting n_1 to be 2, the second moment of Z_1 is obtained as

$$\mu_{2,0,0,...} = E[(Z_1 - \mu_{Z_1})^2]. \tag{3.8}$$

which is the variance $\sigma^2_{Z_1}$ of Z_1.

3.3 Covariance and Autocorrelation

A characteristic of a time series that is readily computed and forms the basis of subsequent analysis is the covariance function. The covariance measures the linear association between variates X and Y. A particular case of covariance, which finds widespread application in time series analysis, is when the association between variates from the same stochastic process is being considered. This covariance is then referred to as the autocovariance. To simplify the discussion, the linear association between only two variates Z_t and $Z_{t+\tau}$ is considered, where t is the time index of the variate and τ is any number of time lags $1, 2, ...$ Consequently, there is need only to deal with the $(a, b)^{th}$ moments of the bivariate distribution $f(Z_t, Z_{t+\tau})$ for Z_t and $Z_{t+\tau}$. The covariance (or more specifically autocovariance) is then obtained by setting

$a = b = 1$ where a and b are now the indices of the moments of Z_t and $Z_{t+\tau}$ respectively. The autocovariance is then given by

$$\mu_{1,1} = E[(Z_t - \mu_{Z_t})(Z_{t+\tau} - \mu_{Z_{t+\tau}})]. \tag{3.9}$$

which, in the discrete case, amounts to

$$\mu_{1,1} = \sum_i \sum_j (z_i - \mu_{Z_t})(z_j - \mu_{Z_{t+\tau}}) * f_{i,j} = \gamma_{Z_t, Z_{t+\tau}}. \tag{3.10}$$

The (auto)correlation between the variates Z_t and $Z_{t+\tau}$ is obtained by normalising the (auto)covariance using the product of the variances $\sigma^2_{Z_t}$ and $\sigma^2_{Z_{t+\tau}}$ as follows:

$$\rho_{Z_t, Z_{t+\tau}} = \frac{\gamma_{Z_t, Z_{t+\tau}}}{\sqrt{\sigma^2_{Z_t} \sigma^2_{Z_{t+\tau}}}}. \tag{3.11}$$

If the autocovariance between variates which are separated by the same number of lags is the same, regardless of when they occur in time, then the stochastic process is said to be covariance stationary. This is to say that the autocovariances $\gamma_{Z_t, Z_{t+\tau}}$ are not dependent on the time index t, but rather on the number of lags, τ, separating the random variables. Consequently, the autocovariance between all variates of the stochastic process separated by τ lags is denoted γ_τ and the corresponding autocorrelations ρ_τ.

3.4 The Linear Filter Transfer Function

Time series modelling may be thought of as defining operations for the purpose of transforming a highly dependent, and possibly nonstationary process $\{z_t\}$, into an uncorrelated white noise process. Conversely, time series modelling may be equivalently considered as the act of determining the linear filter which, when operating on independent random shocks, produces the process $\{z_t\}$.

The random shocks $a_t, a_{t-1}, a_{t-2}, \ldots$ are typically considered to be independent random variables that are normally distributed with mean equal to zero and some fixed variance σ^2_a. Usually one does not observe these shocks directly, but rather observes the realisations of the process itself, $\{z_t\}$. The linear filter ψ simply produces a weighted sum of previous shocks as follows:

$$z_t = \mu + a_t + \psi_1 a_{t-1} + \psi_2 a_{t-2}, \ldots \tag{3.12}$$

where μ represents the level of the process. In practise the filter ψ must be determined from the observations of the process,$\{z_t\}$. The filter may also be considered to be constituted of the ratio of two polynomials $\phi(B)$ and $\theta(B)$ given as

$$\psi(B) = \frac{\theta(B)}{\phi(B)}. \tag{3.13}$$

$$\psi(B) = \frac{1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q}{1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p}. \tag{3.14}$$

where B is the backshift operator. The backshift operator is defined in 3.14.1. The determination of these two polynomials is in fact one of the primary concerns of time series analysis. $\psi(B)$ is called the transfer function of the linear filter relating z_t to a_t [6] as follows:

$$z_t = \psi(B) a_t. \tag{3.15}$$

Seeking to relate a_t in terms of z_t, the following is obtained:

$$a_t = \psi(B)^{-1} z_t. \tag{3.16}$$

Denoting $\psi(B)^{-1}$ as $\pi(B)$, the result is

$$\pi(B) = \frac{1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p}{1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q}, \tag{3.17}$$

where the transfer function $\pi(B)$ is

$$\pi(B) = \frac{\phi(B)}{\theta(B)}. \tag{3.18}$$

$\pi(B)$ can then be referred to as the transfer function of the linear filter relating a_t to z_t [6] as follows:

$$a_t = \pi(B) z_t. \tag{3.19}$$

3.5 Stationarity of the Transfer Function

In [6] it is shown that the autocovariance γ_k of a process may be computed from the transfer function as

$$\gamma_k = \sigma_a^2 \sum_{j=0}^{\infty} \psi_j \psi_{j+k}, \tag{3.20}$$

and in particular, the variance is given by

$$\gamma_0 = \sigma_a^2 \sum_{j=0}^{\infty} \psi_j^2. \tag{3.21}$$

For the process to have finite variance, it is thus required that sequence of weights ψ_j, which may be infinite in extent, converge. In general, it can be shown that, if the infinite sequence of numbers $\{\psi_j\}_{j=0}^{\infty}$ is absolutely summable, i.e.

$$\sum_{j=0}^{\infty} |\psi_j| < \infty, \tag{3.22}$$

then the linear filter it specifies is stable and the resulting process is covariance-stationary [27]. This translates to the condition that the transfer function $\psi(B)$ converge on or inside the unit circle for $B \leq 1$. This is in fact a basic requirement stemming from filter theory.

3.6 Invertibility of Transfer Function

Given a process described as $z_t = \psi(B)a_t$, invertibility requires that a valid representation of the random shocks a_t be obtainable by inverting the transfer function $\psi(B)$,

$$a_t = \pi(B)z_t, \tag{3.23}$$

where $\pi(B) = \psi(B)^{-1}$.

Since $\pi(B)$ may be infinite in extent, it is required that the sequence of weights π_j converge in order for the specified random shocks a_t to be well defined [27]. If in fact the weights π_j are divergent, a situation may arise where the current process value z_t depends on its previous values z_{t-k} with increasing weight as k, the number of lags, increases. In this situation a convergent expansion of a_t in terms of z_t may still be achieved, however it would require the use of present and future values of z_t. This presents a problem for the task of prediction as present and past values of the process are required for this task. The property of invertibility thus ensures that present events can be associated with past events in a sensible way. In turn, ensuring invertibility translates to the condition that the transfer function $\pi(B)$ converges on or inside the unit circle for $B \leq 1$.

3.7 The Autoregressive AR(p) Model

An autoregressive process is one in which the current value of a process, z_t, is represented as a weighted sum of previous values of the process and the current random shock. This is essentially a regression of the process against itself:

$$z_t = \phi_1 z_{t-1} + \phi_2 z_{t-2} + ... + \phi_p z_{t-p} + a_t. \tag{3.24}$$

Collecting all the terms involving the values of the process on the left hand side and factoring out the polynomial $\phi(B)$, the following is obtained:

$$(1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p) z_t = a_t, \tag{3.25}$$

which can be written economically as $\phi(B) z_t = a_t$. The equivalent representation in terms of $\psi(B)$ is then

$$z_t = \frac{a_t}{(1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p)} = \psi(B) a_t, \tag{3.26}$$

where

$$\psi(B) = \frac{1}{\phi(B)} \tag{3.27}$$

$$\psi(B) = (1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p)^{-1} \tag{3.28}$$

$$\psi(B) = (1 + \psi_1 B + \psi_2 B^2 + \psi_3 B^3 + ...). \tag{3.29}$$

An autoregressive process is defined by a set of p weights, which are the coefficients of the polynomial in B, $\phi(B)$. The autoregressive process of order p is denoted $AR(p)$. Note in particular that the polynomial $\phi(B)$ is finite in order and hence in the extent of its weights, as opposed to the corresponding transfer function $\psi(B) = \phi(B)^{-1} = \sum_{j=0}^{\infty} \psi_j B^j$, which is infinite in its extent.

In order for the autoregressive process to be stationary, it is required that the weights ψ_j of the transfer function $\psi(B)$ form a convergent series, as stated above. This in turn imposes a restriction on the autoregressive polynomial $\phi(B)$ - that the roots of the characteristic equation $(1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p) = 0$ should lie outside the unit circle.

3.8 The Moving-Average MA(q) Model

A moving-average process is one in which the current value of a process, z_t, is represented as a weighted sum of previous innovations or random shocks a_t that are input to the process. This amounts essentially to an averaging of the most recent shocks.

$$z_t = \theta_1 a_{t-1} - \theta_2 a_{t-2} - ... - \theta_q a_{t-q} + a_t. \tag{3.30}$$

If the polynomial $\theta(B)$ is factored out, the following is obtained

$$z_t = (1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q) a_t, \tag{3.31}$$

which can be written economically as $z_t = \theta(B) a_t$. The equivalent representation in terms of $\psi(B)$ becomes

$$z_t = (1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q) a_t = \psi(B) a_t, \tag{3.32}$$

where

$$\psi(B) = \theta(B) \tag{3.33}$$

$$\psi(B) = (1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q). \tag{3.34}$$

As seen above, a moving-average process is defined by a set of q weights, which form the polynomial in B, $\theta(B)$. The moving-average process of order q is denoted $MA(q)$. Note in particular that the polynomial $\theta(B)$ is finite in order and hence in the extent of its weights. Naturally then, since $\psi(B) = \theta(B)$, the moving-average's corresponding transfer function $\psi(B)$ is also finite in its extent. The corresponding inverse transfer function $\pi(B) = \psi(B)^{-1}$ by contrast is infinite in its extent.

$$\pi(B) = (1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q)^{-1} \tag{3.35}$$

$$\pi(B) = (1 + \pi_1 B + \pi_2 B^2 + \pi_3 B^3 + ...). \tag{3.36}$$

It can be seen that in order for the moving average process to be stationary, there is no requirement on the transfer function $\psi(B)$ as it naturally forms a convergent series. If we however wish to express the moving average process as an $AR(\infty)$ process by simply

inverting the moving average operator $\theta(B)$, and hence have the moving average process be invertible [27],

$$\frac{z_t}{(1-\theta_1 B-\theta_2 B^2-...-\theta_q B^q)} = a_t \tag{3.37}$$

$$(1+\pi_1 B+\pi_2 B^2+\pi_3 B^3+...)z_t = a_t \tag{3.38}$$

$$\pi(B)z_t = a_t. \tag{3.39}$$

It is required that the weights of the inverse transfer function $\pi(B)$ be convergent on or within the unit circle, as stated above. This in turn imposes a restriction on the moving-average polynomial $\theta(B)$ - that the roots of the characteristic equation

$$(1-\theta_1 B-\theta_2 B^2-...-\theta_q B^q) = 0. \tag{3.40}$$

lie outside the unit circle. That is to say that if we factorise the polynomial $\theta(B)$

$$(1-\theta_1 B-\theta_2 B^2-...-\theta_q B^q) = (1-\lambda_1 B)(1-\lambda_2 B)...(1-\lambda_q B). \tag{3.41}$$

then, if all the $|\lambda_i| < 1$ for $i = 1,2,...,q$, all the roots of $\theta(B)$ will be outside of the unit circle [27].

3.9 The Auto-Regressive Moving-Average ARMA(p,q) Model

Owing to the fact that in practice a given process may not be parsimoniously represented as a purely moving average or autoregressive process, it may be necessary to incorporate both autoregressive and moving-average components into the expansion of the process as follows:

$$z_t = \phi_1 z_{t-1}+...+\phi_p z_{t-p}-\theta_1 a_{t-1}+...+\theta_q a_{t-q}+a_t. \tag{3.42}$$

which, if all the terms involving the process values are collected on the left hand side, may be written as

$$\begin{aligned} -\phi_1 z_{t-1}-\cdots-\phi_p z_{t-p}+z_t &= -\theta_1 a_{t-1}+\cdots+\theta_q a_{t-q}+a_t \\ (1-\phi_1 z_{t-1}-\cdots-\phi_p z_{t-p})z_t &= (1-\theta_1 a_{t-1}+\cdots+\theta_q a_{t-q})a_t \\ \phi(B)z_t &= \theta(B)a_t \\ z_t &= \phi(B)^{-1}\theta(B)a_t. \end{aligned} \tag{3.43}$$

The mixed autoregressive moving-average process with order p autoregressive polynomial $\phi(B)$ and order q moving-average polynomial $\theta(B)$ is denoted $ARMA(p,q)$. The transfer function of the $ARMA(p,q)$ process $\psi(B)$ relates the process values z_t to the random shocks a_t according to $z_t = \psi(B)a_t$, thus $\psi(B)$ is obtained as

$$\begin{aligned} \psi(B) &= \phi(B)^{-1}\theta(B) \\ \psi(B) &= \frac{1-\theta_1 a_{t-1} + \cdots + \theta_q a_{t-q}}{1-\phi_1 z_{t-1} - \cdots - \phi_p z_{t-p}}. \end{aligned} \tag{3.44}$$

Since $\theta(B)$ is of finite order q, to ensure stationarity we simply require that the infinite sequence $1/\phi(B)$ is convergent. This is to say that stationarity is soley dependent on the $AR(p)$ component of the $ARMA(p,q)$ process. This then necessitates that the roots of the characteristic equation of the $AR(p)$ process,

$$(1-\phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p) = 0, \tag{3.45}$$

lie outside the unit circle.

Similarly, since $\phi(B)$ is of finite order p, to ensure invertibility we simply require that the infinite sequence $1/\theta(B)$ is convergent. This is to say that invertibility is soley dependent on the $MA(q)$ component of the $ARMA(p,q)$ process. This then necessitates that the roots of the characteristic equation of the $MA(q)$ process,

$$(1-\theta_1 B - \theta_2 B^2 - \cdots - \theta_q B^q) = 0, \tag{3.46}$$

lie outside the unit circle.

3.10 Derivation of the Autocorrelation Function for AR(p) Models

Consider the $AR(p)$ process

$$z_t = \phi_1 z_{t-1} + \phi_2 z_{t-2} + \cdots + \phi_p z_{t-p} + a_t. \tag{3.47}$$

Recall from section 3.2 that the autocovariance between any two variates of a process, say z_t and z_{t-k}, is given by

$$\mu_{1,1} = E[(Z_t - \mu_{Z_t})(Z_{t-k} - \mu_{Z_{t-k}})], \tag{3.48}$$

where μ_{Z_t} and $\mu_{Z_{t-k}}$ are the means of the random variables Z_t and Z_{t-k} respectively. Denoting the mean corrected process $(Z_t - \mu_{Z_t}) = \widetilde{Z}_t$, computation of the autocovariance invovles multiplying the mean corrected random variable at some time t by the random variable some k lags later, and taking the expectation of the result. Applying this to equation 3.47 the following is obtained:

$$\begin{aligned} E[\widetilde{Z}_t\widetilde{Z}_{t-k}] &= E[(\phi_1\widetilde{z}_{t-1}\widetilde{z}_{t-k} + \phi_2\widetilde{z}_{t-2}\widetilde{z}_{t-k} + \cdots + \phi_p\widetilde{z}_{t-p}\widetilde{z}_{t-k} + a_t\widetilde{z}_{t-k}] \\ E[\widetilde{Z}_t\widetilde{Z}_{t-k}] &= E[(\phi_1\widetilde{z}_{t-1}\widetilde{z}_{t-k}] + E[\phi_2\widetilde{z}_{t-2}\widetilde{z}_{t-k}] + \cdots + E[\phi_p\widetilde{z}_{t-p}\widetilde{z}_{t-k}] + E[a_t\widetilde{z}_{t-k}] \\ \gamma_k &= \phi_1\gamma_{k-1} + \phi_2\gamma_{k-2} + \cdots + \phi_p\gamma_{k-p}. \end{aligned} \tag{3.49}$$

The term $E[a_t\widetilde{z}_{t-k}]$ dissappears since $\widetilde{z}_{t-k}$ is comprised of previous random shocks up to time $t-k$; all of which are uncorrelated with with a_t as per the definition of the noise process $\{a_t\}$. Therefore, if $k > 0$, then $\widetilde{z}_{t-k}$ is independent of a_t. It is worth noting that the autocovariance between the two variates, $\widetilde{z}_t$, and $\widetilde{z}_{t-k}$, incorporates the combined effects of the random variables at the intervening lags, $\{z_{t-1}, z_{t-2}, ..., z_{k-1}\}$. The autocorrelation of the $AR(p)$ process at lag k is obtained by simply normalising the autocovariance by the variances of the two variates under consideration resulting in

$$\rho_k = \frac{\gamma_k}{\sqrt{E[(Z_t - \mu_{Z_t})^2]E[(Z_{t-k} - \mu_{Z_{t-k}})^2]}}. \tag{3.50}$$

Since the process is covariance-stationary, the variance of all the random variables $\{z_1, z_2, ...\}$ is the same. As a result, equation 3.50 becomes

$$\rho_k = \frac{\gamma_k}{\gamma_0}. \tag{3.51}$$

where $\gamma_0 = E[(Z_t - \mu_{Z_t})]E[(Z_t - \mu_{Z_t})]$, which is the autocovariance at lag zero. Dividing equation 3.49 by γ_0 on both sides, we obtain the following expression for the autocorrelation:

$$\rho_k = \phi_1\rho_{k-1} + \phi_2\rho_{k-2} + \cdots + \phi_p\rho_{k-p}. \tag{3.52}$$

From equation 3.52 it can be seen that the computation of the autocorrelation at lag k depends on the p previous autocorrelations. Note, however, that this computation is recursive in nature, meaning that there is some influence from all previous lags (even beyond p lags) that is propagated into the future values of the autocorrelation. As a result, the autocorrelation of an autoregressive process, if not sufficiently attenuated, can potentially be infinite.

3.11 Derivation of the Partial-Autocorrelation Function for AR(p) Models

Taking equation 3.50, now enumerate the set of equations to compute the autocorrelation at each of the lags $k = 1, 2, \ldots, p$, where p is the number of coefficients (or order) of the autoregressive polynomial, and noting that $\rho_j = \rho_{-j}$, the following is obtained:

$$\begin{aligned}
\rho_1 &= \phi_1\rho_0 + \phi_2\rho_1 + \cdots + \phi_p\rho_{p-1} \\
\rho_2 &= \phi_1\rho_1 + \phi_2\rho_0 + \cdots + \phi_p\rho_{p-2} \\
&\vdots \\
\rho_p &= \phi_1\rho_{p-1} + \phi_2\rho_p + \ldots + \phi_p\rho_0
\end{aligned} \tag{3.53}$$

The above set of equations are also known as the Yule-Walker equations; and can be represented in matrix format as $\rho_p = P\phi_p$ where,

$$\rho_p = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \cdot \\ \cdot \\ \cdot \\ \rho_p \end{bmatrix}, \phi_p = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \cdot \\ \cdot \\ \cdot \\ \phi_p \end{bmatrix}, P_p = \begin{bmatrix} 1 & \rho_1 & \rho_2 & \ldots & \rho_{p-1} \\ \rho_1 & 1 & \rho_1 & \ldots & \rho_{p-2} \\ \cdot & \cdot & \cdot & \ldots & \cdot \\ \cdot & \cdot & \cdot & \ldots & \cdot \\ \cdot & \cdot & \cdot & \ldots & \cdot \\ \rho_{p-1} & \rho_{p-2} & \rho_{p-3} & \ldots & 1 \end{bmatrix}$$

Now consider the coefficients of each of the autoregressive processes $AR(k)$, where k is the order of the autoregressive polynomial and varies between $k = 1$ to p; and denoting the index of the coefficients of that autoregressive polynomial $j = 1$ to k; the j^{th} coefficient of the order k autoregressive process can be denoted as $\phi_{k,j}$. The following realisations of the Yule-Walker equations are obtained for the first and second order autoregressive process as a result:

$AR(1)$

$$\rho_p = \begin{bmatrix} \rho_1 \end{bmatrix}, \phi_p = \begin{bmatrix} \phi_{1,1} \end{bmatrix}, P_p = \begin{bmatrix} 1 \end{bmatrix}$$

$$\rho_1 = \phi_{1,1} \tag{3.54}$$

$AR(2)$

$$\rho_p = \begin{bmatrix} \rho_1 \\ \rho_2 \end{bmatrix}, \phi_p = \begin{bmatrix} \phi_{2,1} \\ \phi_{2,2} \end{bmatrix}, P_p = \begin{bmatrix} 1 & \rho_1 \\ \rho_1 & 1 \end{bmatrix}$$

$$\begin{aligned} \rho_1 &= \phi_{2,1} + \rho_1 \phi_{2,2} \\ \rho_2 &= \phi_{2,1}\rho_1 + \phi_{2,2} \end{aligned} \tag{3.55}$$

And in general, the autocorrelations are determined according to

$$\rho_j = \phi_{k,1}\rho_{j-1} + \cdots + \phi_{k,k-1}\rho_{j-k+1} + \phi_{k,k}\rho_{j-k}. \tag{3.56}$$

Now, if instead of looking at the autocorrelations ρ_j, we look at the coefficients $\phi_{k,j}$ that are generated by solving the Yule-Walker equations above for $\phi_{k,j}$ for each set of equations that is produced when $k = 1, 2, ..., p$, we can extract the partial-autocorrelation function as the last coefficient, $\phi_{k,k}$, at each value of k. The partial-autocorrelation function of an $AR(p)$ process is thus given by the set of coefficients $\{\phi_{1,1}, \phi_{2,2}, ..., \phi_{p,p}\}$.

The partial-autocorrelation function $\phi_{k,k}$ can be formed as a function of the autocorrelation function ρ_k and is in fact defined for any stationary process. The quantity $\phi_{k,k}$ is interpreted as the partial-autocorrelation of the process $\{z_t\}$ at lag k, which measures the direct correlation between the variables z_t and z_{t-k} when the effects of the variates between them $\{z_{t-1}, z_{t-2}, ..., z_{t-k+1}\}$ are removed [6]. Another way to understand the meaning of the partial-autocorrelation function $\phi_{k,k}$ is - as we're fitting $AR(k)$ models of successively greater and greater order $\{1, 2, ...\}$, the last coefficient, computed from the Yule-Walker equations that arise at each order, measures the excess correlation at lag k, which is not accounted for by the preceding $AR(k-1)$ model[15].

Notice that the Yule-Walker equations in 3.53 express the autocorrelation of an $AR(p)$ process at lag t as a linear combination of the p autocorrelations going back to lag $(t-p)$. Regardless of how many autocorrelations are available or at which lag they are observed, they will always be combined according to the same p coefficients of the AR(p) polynomial, $\{\phi_1, \phi_2, ..., \phi_p\}$. Thus the partial-autocorrelation function $\{\phi_{k,k}\}$ of an autoregressive process is finite in its extent. In particular, if we have an $AR(p)$ process, then the corresponding partial-autocorrelation function $\{\phi_{k,k}\}$ cuts off after p lags [27]. This is in contrast with corresponding autocorrelation function of an $AR(p)$ process, which may decay over a potentially infinite number of lags.

3.12 Derivation of the Autocorrelation Function for MA(q) Models

Following the same preamble defined in 3.10, computation of the autocovariance function of an $MA(q)$ process proceeds as follows:

$$\gamma_j = E[(a_t + \theta_1 a_{t-1} + \theta_2 a_{t-2} + \cdots + \theta_q a_{t-q}) x (a_{t-j} + \theta_1 a_{t-j-1} + \theta_2 a_{t-j-2} + \cdots + \theta_q a_{t-j-q})]. \tag{3.57}$$

Upon multiplying out, there are terms involving the product of shocks generated at different times. Since the shocks are defined as being independent from each other, the expectation of such terms would amount to zero. This leaves the expectation of the remaining terms to be:

$$\gamma_j = E[(\theta_j a_{t-j}^2 + \theta_{j+1}\theta_1 a_{t-j-1}^2 + \theta_{j+2}\theta_2 a_{t-j-2}^2 + \cdots + \theta_q \theta_{q-j} a_{t-j-q}^2)]. \tag{3.58}$$

Note that, in the above equation, we can only start collecting terms from lag $t-j$ going backwards in time since there would be no common observations of the process at lags later than that. This can be seen from the fact that for the "later" sequence z_{t-j} we have to go j lags back in time. Consequently, the first term with correlation between the two sequences involves a_{t-j} and the j^{th} coefficient of the MA polynomial θ_j.

$$\gamma_j = E[(\theta_j a_{t-j}^2 + \theta_{j+1}\theta_1 a_{t-j-1}^2 + \theta_{j+2}\theta_2 a_{t-j-2}^2 + \cdots + \theta_q \theta_{q-j} a_{t-j-q}^2)] \tag{3.59}$$

Using the fact that the process is covariance stationary, i.e. $E[a_t^2] = E[a_{t-1}^2] = E[a_{t-2}^2] = \cdots = \sigma_a^2$, we obtain the autocovariance γ_j of the $MA(q)$ process as

$$\gamma_j = \begin{cases} \sigma_a^2(\theta_j + \theta_{j+1}\theta_1 + \theta_{j+2}\theta_2 + \cdots + \theta_q\theta_{q-j}), & \text{for } j = 1,2,\ldots,q. \\ 0, & \text{for } j > q. \end{cases} \tag{3.60}$$

Once again, scaling by the variance (also the autocovariance at lag 0, γ_0), we obtain the autocorrelation function $\{\rho_k\}$.

$$\rho_j = \begin{cases} \frac{\sigma_a^2(\theta_j+\theta_{j+1}\theta_1+\theta_{j+2}\theta_2+\cdots+\theta_q\theta_{q-j})}{1+\theta_1^2+\cdots+\theta_q^2}, & \text{for } j = 1,2,\ldots,q. \\ 0, & \text{for } j > q. \end{cases} \tag{3.61}$$

where $\gamma_0 = 1+\theta_1^2+...+\theta_q^2$.

Since the moving average process is essentially a linear combination of independent random shocks going back a finite number of q lags, we see from equation 3.60 that the autocorrelation function of a moving-average process is finite in its extent. In particular, if we have a $MA(q)$ process, then the extent of its autocorrelations is q lags after which it cuts off.

3.13 Derivation of the Partial-Autocorrelation Function for MA(q) Models

If we were now to enumerate the q equations stemming from 3.61 and attempt to solve for the coefficients $\{\theta_1,\theta_2,...,\theta_q\}$, it would quickly become apparent that, unlike the Yule-Walker equations for an $AR(p)$ process, the equations obtained are non-linear [6]. Another way to try and derive the partial-autocorrelation of an $MA(q)$ process would be to use its equivalent representation as an $AR(\infty)$ process. In this instance we obtain a representation of the $MA(q)$ process in terms of the infinite transfer function $\pi(B) = \psi(B)^{-1} = \theta(B)^{-1}$ as follows:

$$\begin{aligned} z_t &= (1-\theta_1 B-\theta_2 B^2-...-\theta_q B^q)a_t \\ (1-\theta_1 B-\theta_2 B^2-...-\theta_q B^q)^{-1} z_t &= a_t \\ (1-\pi_1 B-\pi_2 B^2-...)z_t &= a_t \\ z_t &= a_t+\pi_1 z_{t-1}+\pi_2 z_{t-2}+... \end{aligned} \tag{3.62}$$

At this point we can apply the same methodology as in section 3.11 for the computation of the partial-autocorrelation of an autoregressive process. For the $MA(q)$ process we see that the equivalent $AR(\infty)$ representation in 3.62 gives rises to an infinite number of coefficients, $\{\pi_k\}$, which ultimately translates to a partial-autocorrelation function $\{\phi_{k,k}\}$ of infinite extent that decays to zero rather than cutting off after q lags as is the case with the autocorrelation function of the $MA(q)$ process.

3.14 Time Series Operators

Consider the familiar notation of a function $y = f(x)$. It denotes the operation of accepting an input, x, and producing an output y. The operation performed by the function $f(x)$ could also be applied to multiple inputs $\{x_1, x_2, ..., x_n\}$ in order to produce the output y. In this case, the operation could be more explicitly denoted as $y = f(x_1, x_2, ..., x_n)$.

Time series operators can be considered in the same light as being functions which transform a given sequence of observations $\{x_t\}$, where t is a time index $0 < t < N$, into a new sequence $\{y_t\}$, which is in terms of the corresponding elements of $\{x_t\}$ [27]. As a simple example, define an operator β which performs a scalar multiplication on an input sequence. We thus have $y_t = \beta x_t$. This, in fact, defines a sequence of N multiplications of the values that x_t takes on at each time instance t with the scalar β.

3.14.1 The back-shift operator

An operator which is fundamental and probably the most widely used in time series analysis is the backward shift or lag operator. Denoting the backward shift operator as B, it ican be defined as the operation of taking an input variable x_t and shifting it backwards in time, so that the output y_t is the value that the input variable had in the previous time instance $t-1$

$$y_t = Bx_t \tag{3.63}$$
$$= x_{t-1} \tag{3.64}$$

Thus applying the backward shift operator to the sequence of observations $\{x_t\}$ results in the delayed by one sequence $\{x_{t-1}\}$

$$B\{x_t\} = \{x_{t-1}\} \tag{3.65}$$

If we now apply the backward shift operator a second time, we obtain

$$B(B\{x_t\}) = B\{x_{t-1}\} = \{x_{t-2}\} \tag{3.66}$$

And in general, applying the backward shift operator m times will result in the input sequence that has been shifted m time instances back or delayed by m time steps.

$$B^m(\{x_t\}) = \{x_{t-m}\} \tag{3.67}$$

3.14.2 The forward-shift operator

The forward shift operator is simply the inverse operator of the backshift operator i.e. it takes an input sequence and shifts it forward in time. Denoting the forward shift operator F, then $(F = B^{-1})$. The output y_t of the forward shift operation on the input variable x_t is

$$y_t = Fx_t \tag{3.68}$$

$$y_t = x_{t+1} \tag{3.69}$$

Similarly to the backward shift operator, applying the forward shift operator m times will result in the input sequence being shifted m time instances forward.

$$F^m(\{x_t\}) = \{x_{t+m}\} \tag{3.70}$$

3.14.3 The differencing operator

An operation that comes up when working with non-stationary time series is the backward difference operator. Given an input time series $\{x_t\}_{t=1}^{N}$, the backward difference or simply the difference operator produces an output $\{y_t\}$ such that each variable y_t is the difference between the variable x_t and the variable at the preceding lag x_{t-1}.

$$\begin{aligned} y_t &= x_t - x_{t-1} \\ y_{t-1} &= x_{t-1} - x_{t-2} \\ y_{t-2} &= x_{t-2} - x_{t-3} \\ &\vdots \\ y_{t-N+1} &= x_{t-N+1} - x_{t-N} \end{aligned} \tag{3.71}$$

Note that in the process of applying a difference operation to a sequence of say N numbers, the output sequence is reduced in number by one to $N-1$ values. From the above, it is apparent that the difference operator can be defined in terms of the backward shift operator as $(1-B)$. Applying the differencing operator thus defined to the input series $\{x_t\}$ we obtain

$$(1-B)x_t = x_t - x_{t-1} \tag{3.72}$$

As the backward difference operation occurs fairly frequently in time series analysis, it is designated the capital delta symbol $\Delta = (1 - B)$. As with the backshift operator, the difference operator can be applied repeatedly to a time series. For example, if the differencing operator is now applied to the time series $\{x_t\}$ twice we obtain:

$$\begin{aligned} y_t &= \Delta^2 x_t \\ y_t &= \Delta(\Delta x_t) = \Delta((1-B)x_t) \\ y_t &= \Delta(x_t - x_{t-1}) = (1-B)(x_t - x_{t-1}) \\ y_t &= (x_t - x_{t-1}) - (x_{t-1} - x_{t-2}) \end{aligned} \tag{3.73}$$

where we've allowed x_t to denote the entire set of N observations $\{x_t\}_{t=1}^{N}$ and y_t the entire set of outputs. In general, denoting the number of times the differencing is applied as d, the d^{th} differencing operator can be represented as $\Delta^d = (1 - B)^d$.

The natural definition of the d^{th} difference operator in terms of integer values of d can be extended to allow for non-integer values. In order to derive such an operator, we consider the definition of the d^{th} difference in terms of the polynomial in B, $(1 - B)^d$. To compute the expansion of this polynomial for an arbitrary non-integer value of d, a binomial expansion is used. Recalling the general representation for time series that can be modelled as stationary *ARMA* processes, $\phi(B)z_t = \theta(B)a_t$, we observe that the polynomials $\phi(B)$ and $\theta(B)$ can also be thought of as operators being applied to the process values z_t and the random shocks a_t respectively. If the process values observed are initially operated on by the differencing operator Δ^d, the new input sequence to the stationary autoregressive operator $\phi(B)$ becomes

$$\begin{aligned} w_t &= \Delta^d z_t \\ w_t &= (1-B)^d z_t \end{aligned} \tag{3.74}$$

The resulting *ARMA* model is then given by

$$\begin{aligned} \phi(B)w_t &= \theta(B)a_t \\ \phi(B)(1-B)^d z_t &= \theta(B)a_t \end{aligned} \tag{3.75}$$

Recall that for the model to be considered stationarity it is required that the roots of the autoregressive polynomial $\phi(B)$ lie outside the unit circle. In effect, the relaxation of this restriction allows for the realisation of non-stationary processes. In particular, models where the roots of $\phi(B)$ lie inside the unit circle describe processes which are non-stationary and explosive in nature. Note that there are legitimate cases where such models are useful. A case of particular interest to us however, is when the roots of the autoregressive characteristic

equation lie on the unit circle. Such models are able to describe processes exhibiting homogenous non-stationarity.

Looking at equation 3.75, we can define the non-stationary autoregressive operator $\omega(B) = \phi(B)(1-B)^d$, which is comprised of the stationary autoregressive operator $\phi(B)$ and the non-stationarity unit roots that are defined by the differencing operator $(1-B)^d$. The process described by $\omega(B)z_t = \theta(B)a_t$ is then said to be a unit root process. That is - the autoregressive polynomial of the process contains unit roots and is thus homogenous non-stationary. By contrast, preparing the non-stationary sequence z_t by applying the differencing operator to it, we obtain the stationary sequence w_t. This can be modelled in the usual way as a stationary process according to $\phi(B)w_t = \theta(B)a_t$. A process, such as z_t, that requires differencing in order to attain stationarity may also be described as being difference stationary.

3.14.4 The Integrated Autoregressive Moving-Average ARIMA(p,d,q)

In equation 3.75 we see that the differenced time series $\{w_t\}$ is assumed to be a stationary process which can then be modelled as a stationary and invertible $ARMA(p,q)$ process. Assume that we want to model a process $\{z_t\}$ that requires a differencing operation to be performed before it can be modelled as a stationary $ARMA(p,q)$ process; if we now incorporate integer values of d into the definition of the model, then the model for the process $\{z_t\}$ is called an Integrated Autoregressive Moving-Average model $ARIMA(p,d,q)$, where p is the order of the autoregressive polynomial, d is the integer order of differencing to be performed in order to make the process stationary, and q is the order of the moving-average polynomial. The term "integrated" refers to the fact that the time series being modelled $\{z_t\}$ is an integrated version of the stationary time series $\{w_t\}$, which the $ARMA(p,q)$ model describes. This type of model will become critical when we consider non-stationary processes or processes with roots close to unity.

3.14.5 The fractional differencing operator

The natural definition of the d^{th} difference operator in terms of integer values of d can be extended to allow for non-integer values. In order to derive such an operator, we consider the definition of the d^{th} difference in terms of the polynomial in B, $(1-B)^d$. The binomial theorem expands a binomial raised to a certain power, $(a+b)^k$ according to the formula

$$(a+b)^k = \sum_{n=0}^{k} \binom{k}{n} a^{k-n} b^n \tag{3.76}$$

where $\binom{k}{n}$ is the binomial coefficient, which represents the number of ways n elements can be chosen from a set containing k elements. To illustrate the rationale behind the use of the combination formula, we multiply out the binomial $(a+b)^k$. From the resulting terms, we know that there will only be one term containing b^k. By contrast, there will be a number of different ways in which the term a^2b^{k-2} will emerge; one for each way of choosing exactly 2 elements from an k element set. The coefficients of the terms that arise from the expansion are obtained by computing the series of binomial coefficients from the combination formula:

$$\begin{aligned} \binom{k}{n} &= \frac{k!}{n!(k-n)!} \\ \binom{k}{n} &= \frac{k(k-1)(k-2)...(k-n+1)}{n!} \end{aligned} \tag{3.77}$$

The definition of the binomial series can be extended to work with non-integer powers. Thus the expansion of the fractional difference operator by the binomial expansion results in the following infinite sequence [29]:

$$(1-B)^d = \sum_{n=0}^{\infty} \binom{d}{n} (-B)^n = 1 - d - B - \frac{1}{2}d(1-d)B^2 - \frac{1}{6}d(1-d)(2-d)B^3 - ... \tag{3.78}$$

where d is the order of differencing and n is an integer. A time series which requires a fractional differencing in order to induce stationarity is said to be fractionally integrated. The $ARMA(p,q)$ model formulated based on the fractionally differenced time series, $\nabla^d z_t$, is then referred to as a fractionally integrated $ARMA$ model or a $FARIMA(p,d,q)$ model, where d is allowed to take on fractional values and p and q are the orders of the stationary operators $\phi(B)$ and $\theta(B)$ respectively.

3.14.6 The infinite summation operator

The original process $\{z_t\}$ can be retrieved from the differenced process $\{w_t\}$ by inverting the difference operator to obtain the infinite summation operator [6].

$$\begin{aligned} w_t &= \Delta^d z_t \\ z_t &= \Delta^{-d} w_t \\ z_t &= S^d w_t \end{aligned} \tag{3.79}$$

The infinite summation operator S is defined as

$$S = (1-B)^{-1} = \Delta^{-1} \tag{3.80}$$

Applying this operator to a time series, say $\{w_t\}$, results in the following operation

$$Sw_t = \sum_{h=-\infty}^{t} w_h = (1 + B + B^2 + ...)w_t \tag{3.81}$$

This amounts to a cumulative sum of the process $\{w_t\}$. The process $\{z_t\} = S\{w_t\}$ is thus said to be an integrated process. Once again this operator may be applied multiple times to a given time series. As an example, applying the infinite summation operator twice on some series $\{x_t\}$ results in

$$\begin{aligned} S^2 x_t &= Sx_t + Sx_{t-1} + +Sx_{t-2} + ... \\ S^2 x_t &= \sum_{i=-\infty}^{t} \sum_{h=-\infty}^{i} (1 + 2B + 3B^2 + ...)x_t \end{aligned} \tag{3.82}$$

3.15 Spectral Analysis

In trying to discover the features and characteristics of a given process, it is often beneficial to explore numerous different representations and transformations of the process. Such tranformations are particularly valuable if they are able to bring to the fore some distinctive feature of the process or to represent such features in a more simple and parsimonious manner. In this chapter, thus far, most of the analysis performed has been in the time domain. In this section a representation of a given process is developed in the frequency domain. It will be shown that the frequency domain representation of a process is, in fact, intimately related to the covariance of the process. It is fitting then that a useful entry point to our analysis in the frequency domain be the autocovariance generating function.

3.15.1 Autocovariance generating function

Considering the sequence of autocovariances $\{\gamma_j\}_{j=0}^{\infty}$ computed from 3.20, if the condition in equation 3.22 is indeed satisfied, the autocovariances may be represented as function of the lag operator B

$$\gamma(B) = \sum_{k=-\infty}^{\infty} \gamma_k B^k \tag{3.83}$$

This is called the autocovariance generating function. The absolute summability of the sequence $\{\gamma_j\}_{j=-\infty}^{\infty}$ becomes evident when we substitute 3.20 into 3.83 to obtain

$$\gamma(B) = \sigma_a^2 \sum_{k=-\infty}^{\infty} \sum_{j=0}^{\infty} \psi_j \psi_{j+k} B^k \tag{3.84}$$

Letting $h = j+k$ and hence $k = h-k$, and noting that $\psi_h = 0$ for $h < 0$, as done in [6], we can write

$$\begin{aligned}
\gamma(B) &= \sigma_a^2 \sum_{j=0}^{\infty} \sum_{h=0}^{\infty} \psi_j \psi_h B^{h-j} \\
\gamma(B) &= \sigma_a^2 \sum_{j=0}^{\infty} \sum_{h=0}^{\infty} \psi_j B^{-j} \psi_h B^h \\
\gamma(B) &= \sigma_a^2 \sum_{j=0}^{\infty} \psi_j B^{-j} \sum_{h=0}^{\infty} \psi_h B^h \\
\gamma(B) &= \sigma_a^2 \psi(B) \psi(B^{-1})
\end{aligned} \tag{3.85}$$

We thus obtain a convenient expression for the autocovariance generating function $\gamma(B)$ in terms of the transfer function $\psi(B)$ of the linear filter that models a given process.

3.15.2 Population spectrum

The autocovariance generating function derived above provides a useful tool for performing spectral analysis, if we now allow B to represent a complex valued variable. Concretely, to obtain the power spectrum of the linear process specified by the transfer function $\psi(B)$, we substitute $B = e^{-i2\pi f}$ into $\gamma(B)$,

$$\begin{aligned}
p(f) &= 2\sigma_a^2 \psi(e^{-i2\pi f}) \psi(e^{i2\pi f}) \\
p(f) &= 2\sigma_a^2 |\psi(e^{-i2\pi f})|^2 \qquad 0 \le f \le \frac{1}{2}
\end{aligned} \tag{3.86}$$

The theoretical or population spectrum of an $AR(p)$ process with transfer function $\psi(B) = (1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p)^{-1}$ is thus given by

$$p(f) = 2\sigma_a^2 \frac{1}{|1 - \phi_1 e^{-i2\pi f} - \phi_2 e^{-i4\pi f} - \cdots - \phi_p e^{-i2p\pi f}|^2} \quad 0 \leq f \leq \frac{1}{2} \tag{3.87}$$

The theoretical or population spectrum of an $MA(q)$ process with transfer function $\psi(B) = 1 - \theta_1 B - \theta_2 B^2 - \cdots - \theta_p B^q$ is thus given by

$$p(f) = 2\sigma_a^2 |1 - \theta_1 e^{-i2\pi f} - \theta_2 e^{-i4\pi f} - \cdots - \theta_q e^{-i2q\pi f}|^2 \quad 0 \leq f \leq \frac{1}{2} \tag{3.88}$$

To derive the theoretical or population spectrum of a general $ARMA(p,q)$ process, we recall equation 3.44 and thus obtain

$$p(f) = 2\sigma_a^2 \frac{|1 - \theta_1 e^{-i2\pi f} - \theta_2 e^{-i4\pi f} - \cdots - \theta_q e^{-i2q\pi f}|^2}{|1 - \phi_1 e^{-i2\pi f} - \phi_2 e^{-i4\pi f} - \cdots - \phi_p e^{-i2p\pi f}|^2} \quad 0 \leq f \leq \frac{1}{2} \tag{3.89}$$

The condition for stationarity can once again be derived from the variance of the process as computed from its spectrum. The variance is given by

$$\sigma_z^2 = \int_0^{\frac{1}{2}} p(f) df \tag{3.90}$$

For the integral in 3.90 to converge, it is evident that the sequence $\psi(B)$ must converge, which implies the condition in 3.22. Assuming that the covariances $\{\gamma_k\}_{k=-\infty}^{\infty}$ are indeed absolutely summable and hence represent a stationary process, it can be shown that $p(f) \geq 0$ for all frequencies f [27]. Given a stationary process, a rudimentary interpretation of the spectrum, when considering a frequency range of say f_1 to f_2,

$$\int_{f_1}^{f_2} p(f) df \tag{3.91}$$

is that it is the portion of variance in the process $\{z_t\}$ that is contained in the frequency range $[f_1, f_2]$ where the frequencies are limited to the range $[0, \frac{1}{2}]$. If we base the definition of the spectrum on the autocorrelation function of the process instead of the autocovariance function, which is equivalent to simply normalising the spectrum $p(f)$ by the variance of the process σ_z^2, we obtain the spectral density function

$$g(f) = p(f)/\sigma_z^2 \tag{3.92}$$

The spectral density function has the property, similar to a probability density function, that the area underneath its curve is 1.

$$\int_0^{\frac{1}{2}} g(f)df = 1 \tag{3.93}$$

It may be noted that the frequency f has an upper bound of $\frac{1}{2}$. This is due to the Nyquist frequency, which specifies the highest frequency that can be represented given a set of N observations taken at constant time intervals of Δt. The frequency at which the observations are captured, or the sampling rate, is $f_s = 1/\Delta t$ samples per unit of time. Naturally we cannot say anything about frequencies f higher than this. One can think of this as taking a temperature measurement at a particular place at noon every day. We might be able to make inferences about the temperature variation from day to day, but we cannot say anything about the variation of temperature over shorter cycles, for example, between noon and midnight(i.e. 12 hours) [15].

Unfortunately we also cannot say much about the signal at the frequency $f = f_s$ either. This is due to the fact that there is a potentially infinite number of continuos functions that may fit the sampled points, if we only evaluate one data point per cycle. Thus the Nyquist frequency states that, at the very least, we should have 2 observations per cycle. Equivalently, the highest frequency that can be represented from a set of observations sampled at a rate of f_s is $f = f_s/2$. If our unit of time is taken to be the same as the sampling interval $\Delta t = 1$, then we have that $f < \frac{1}{2}$.

3.15.3 Sample periodogram

Assume a set of N observations $\{z_t\}$, where it is supposed that the data can be modelled as a sum of some unknown periodic components, as per the Fourier series model

$$z_t = \mu + \sum_{i=1}^{N/2} \alpha_i cos(2\pi f_i t) + \beta_i sin(2\pi f_i t) + \varepsilon_t \tag{3.94}$$

where f_i is the i^{th} frequency component given by i/N for $i = 1, 2, ..., k$, ε_t is a random noise process and μ, α_i, β_i are parameters to be estimated from the data. Since this model is linear with respect to the parameters $\mu, \alpha_i, and \beta_i$, we can make use of the least squares estimates of these parameters, which minimize $\sum_{t=1}^{N} (z_t - \mu - \sum_{i=1}^{N/2} \alpha_i cos(2\pi f_i t) + \beta_i sin(2\pi f_i t))^2$[15]. The least squares estimates of $\hat{\mu}, \hat{\alpha}_i, and \hat{\beta}_i$ are given by the following :

$$\begin{aligned}\hat{\mu} &= \sum z_t/N \\ \hat{\alpha}_i &= \frac{2}{N}\sum_{t=1}^{N} z_t cos(2\pi f_i t) \\ \hat{\beta}_i &= \frac{2}{N}\sum_{t=1}^{N} z_t sin(2\pi f_i t)\end{aligned} \tag{3.95}$$

where $i = 1,2,...,k$, and $k = N/2$ if N is even or $k = (N-1)/2$, if N is odd. In the event that N is even, we modify the above estimates for $i = k = N/2$ to

$$\begin{aligned}\hat{\alpha}_i &= \frac{1}{N}\sum_{t=1}^{N} z_t(-1)^t \\ \hat{\beta}_i &= 0\end{aligned} \tag{3.96}$$

The periodogram of the observations is then given by the plot of the intensity values $I(f_i)$, computed as shown below, against the k frequencies f_i.

$$I(f_i) = \frac{N}{2}(\hat{\alpha}_i{}^2 + \hat{\beta}_i{}^2) \tag{3.97}$$

It will be shown, during the analysis of the residuals produced by applying the inverse of the linear filter $\psi(B)$ to the observed data $\{z_t\}$, that the total variance remaining is composed of the residual sum of squares and the sum of squares that is explained by the periodic components at frequencies f_i [15]. These are given by the intensity values $I(f_i)$. If the signal being analysed were truly random (without periodicities) it would be repsented by

$$z_t = \alpha_0 + e_t \tag{3.98}$$

where α_0 is some fixed mean and e_t is a Normal noise process, $e_t \sim N(0, \sigma_2)$.

3.15.4 The merit of modelling based on white noise

In this subsection we take the time to motivate the merits of building models for a process based on the assumption of an uncorrelated gaussian noise process as an input.

Consider a system whose impulse response $y(t)$ is given. Naturally, from the definition of impulse response, we know that the input $x(t)$ is an impulse. Denoting the transfer function of the system $h(t)$, the relationship between the input and output of the system is

$$y(t) = x(t) \circledast h(t) \tag{3.99}$$

Denoting the fourier transforms of the input, output and tranfer function as

$$\begin{aligned} x(t) &\leftrightarrow X(\omega) \\ y(t) &\leftrightarrow Y(\omega) \\ h(t) &\leftrightarrow H(\omega) \end{aligned} \tag{3.100}$$

We recall that in the frequency domain the equivalent of the convolution operation $\circledast$ is multiplication. Thus we obtain the following frequency domain expression for the transfer function of the system

$$H(\omega) = \frac{Y(\omega)}{X(\omega)} \tag{3.101}$$

Because the Fourier transform of the delta function is constant at all frequencies, 3.101 reduces to

$$H(\omega) = CY(\omega) \tag{3.102}$$

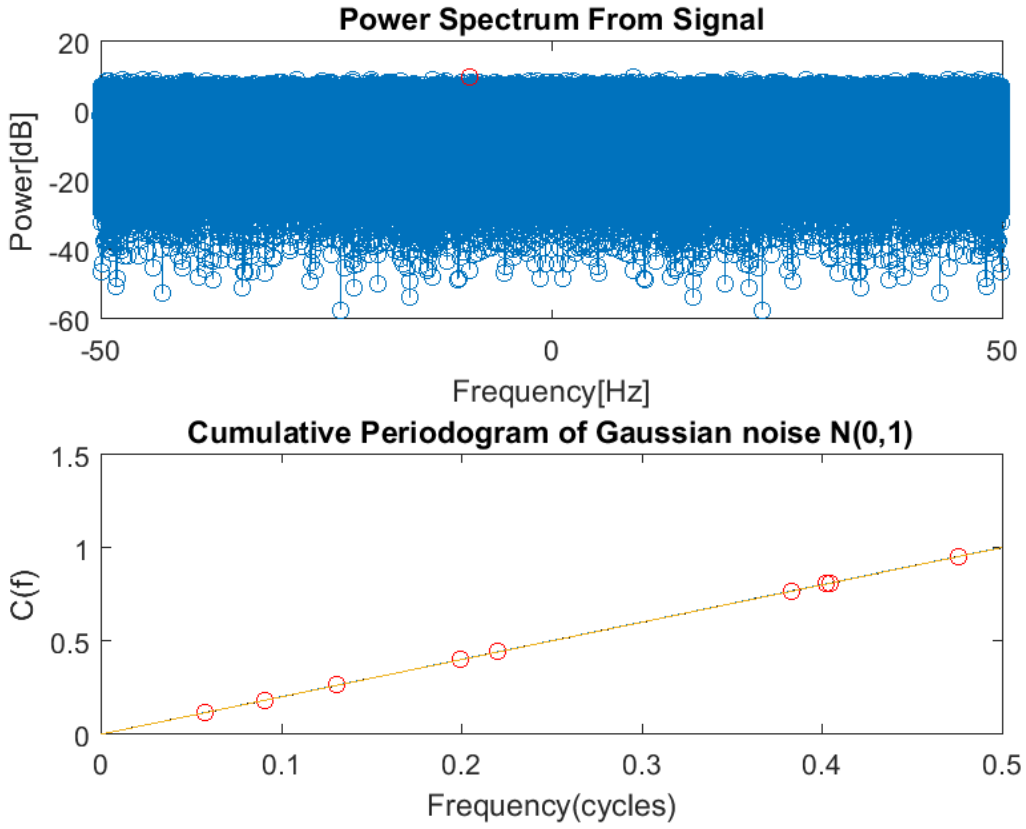

Fig. 3.2 Top: Frequency spectrum of an uncorrelated gaussian noise process. Below: Cumulative periodogram showing a linear accumulation of intensity over all frequencies.

where C is a constant representing the magnitude of the frequency spectrum of the delta function. Having thus found $H(\omega)$, we can obtain $h(t)$ by computing the inverse Fourier transform. We see then that having observed the output $y(t)$, by determining $h(t)$ based on the assumption of a delta function input, we have essentially modelled $y(t)$.

Considering a stochastic process, instead of a deterministic system, we assume that the process is being driven by random uncorrelated shocks or innovations, $\varepsilon(t)$. Such a formulation is typical when modelling, for example, electronic circuits and acoustic systems. If, having observed the output of the process $z(t)$, we wish to determine the transfer function $\psi(t)$ relating the output $z(t)$ to the input $\varepsilon(t)$, we follow the same procedure as above. Computing the Fourier tranforms we obtain

$$\begin{aligned} \varepsilon(t) &\leftrightarrow \varepsilon(\omega) \\ z(t) &\leftrightarrow Z(\omega) \\ \psi(t) &\leftrightarrow \Psi(\omega) \end{aligned} \tag{3.103}$$

$$\begin{aligned} z(t) &= \varepsilon(t) \circledast \psi(t) \\ \Psi(\omega) &= \frac{Z(\omega)}{\varepsilon(\omega)} \end{aligned} \tag{3.104}$$

Once again, it turns out, the Fourier transform of the Gaussian noise sequence, $\varepsilon(\omega)$, is constant at all frequencies. Equation 3.104 thus reduces to

$$\Psi(\omega) = KZ(\omega) \tag{3.105}$$

where K is the constant representing the magnitude of the frequency spectrum of the Gaussian noise. We see again that having observed the output $z(t)$ of the stochastic process, by determining $\psi(t)$ based on the assumption of a white noise input $\varepsilon(t)$, we have essentially modelled $z(t)$ directly.

3.16 Modelling Non-stationary Time Series

The theoretical framework established thus far provides us with a simple, yet powerful, set of tools to be able to model stationary processes of many varieties. Unfortunately, the assumption of stationarity cannot be made for most of the processes encountered in real world applications and, in particular, network traffic. The first task, therefore, is to perform some form of preparation of the data in order to make the data amenable to modelling via our tools. Concretely, we wish to model a given process according to the general mixed $ARMA(p,q)$ model i.e. the model comprised of the stationary operators $\phi(B)$ and $\theta(B)$,

which are the autoregressive and moving average polynomials respectively. Our goal is to manipulate the data into a form such that it can be sufficiently described by an $ARMA(p,q)$ model of the form in equation 3.44. For this goal to be attained, the non-stationarity of a given time series must be addressed.

A non-stationarity process can be defined as one whose mean and covariance do not remain constant over time. Non-stationarity can arise in numerous different ways. In this work, however, the discussion will be restricted to three of the most common sources of non-stationarity, namely the presence of:

- trend,
- seasonality, and
- unit roots.

3.16.1 Unit roots and homogenous non-stationarity

The particular type of non-stationarity that is of interest in this study is homogenous non-stationarity. That is, despite the fact that the process is observed to be non-stationary, it displays behaviour that is similar at different time points. For example a process which shows similar behaviour over time, but with its mean level changing, or a process whose variance might be changing, but the process looks otherwise similar over time. In such situations, a simple differencing, by applying the operator $(1-B)$ to the data, is able to bring the process to stationarity [6]. In general, if a non-stationary process can be made stationary by applying a differencing operation $(1-B)$ or any number d of repeated differencing operations $(1-B)^d$, then it may be referred to as a homogenous non-stationarity process [51]. The stationary process $w_t = (1-B)^d z_t$ resulting from the differencing operation can then be modelled as an $ARMA(p,q)$ process in the usual fashion $\phi(B)w_t = \theta(B)a_t$, where a_t is a white noise process and $\phi(B)$ and $\theta(B)$ are the autoregressive and moving-average polynomials of order p and q respectively; and the said polynomials satisfy the conditions for stationarity and invertibility. Expanding the differencing operator in terms of B next to the autoregressive polynomial, it can be seen that applying a differencing operation is tantamount to adding unit roots to the autoregressive polynomial,

$$\begin{aligned}
\phi(B)(1-B)z_t &= \theta(B)a_t \\
(1-\phi_1 B-\phi_2 B^2-\ldots\phi_p B^p)(1-B)z_t &= (1-\theta_1 B-\theta_2 B^2-\ldots-\theta_q B^q)a_t \\
(1-\lambda_1 B)(1-\lambda_2 B)\ldots(1-\lambda_p B)(1-1B)z_t &= (1-\theta_1 B-\theta_2 B^2-\ldots-\theta_q B^q)a_t,
\end{aligned} \tag{3.106}$$

where $(1-\lambda_1 B)(1-\lambda_2 B)...(1-\lambda_p B)$ is the factorisation of the stationary autoregressive operator $\phi(B)$ and $|\lambda_i| < 1$ for $i = 1,2,..,p$, so that the roots of the transfer function,

$$\psi(B) = \frac{(1-\theta_1 B-\theta_2 B^2-...-\theta_q B^q)}{(1-\phi_1 B-\phi_2 B^2-...\phi_p B^p)}, \tag{3.107}$$

lie outside of the unit circle. We can thus define a general unit root or difference stationary process (named because of the root that is unity contributed by the differencing operator) in terms of the transfer function $\psi(B)$ as

$$(1-B)^d z_t = \delta + \psi(B)a_t, \tag{3.108}$$

where δ is the mean of the stationary process $(1-B)^d z_t$.

An initial plot of the autocorrelation of the data may assist in identifying unit root processes. Unit root processes are typically characterised by an autocorrelation function that decays slowly towards zero. In addition, a trend may be visible in the plot of the original data itself, where the mean of the process appears to vary over time. To demonstrate this, we consider the fundamental unit root process - a random walk. We define a random walk as

$$(1-B)z_t = \delta + a_t z_t \quad = \delta + z_{t-1} + a_t, \tag{3.109}$$

where a_t and δ are defined as before.

The random walk process z_t, $t = 1,2,...$, is a sum of t independent gaussian variables at time t. Each of the random variables are independently distributed as $N(0,\sigma^2)$. Consequently, z_t has a variance of $t\sigma^2$. Thus it can be seen that the variance of the random walk process is also time dependent. As seen in figures 3.3 and 3.5 the differencing operation is particularly well suited to removing this source of non-stationarity. While it is tempting to want to apply differencing in all situations, which indeed may be effective for most trends, one should be aware that in the instance of trend-stationary processes (defined in the next section) this might result in a non-invertible transfer function $(1-B)\psi(B)$. See chapter 15 of [27].

3.16.2 Trend

Trend may informally be considered as a long-term change in the mean level of a process i.e. where the mean level of the process is a function of time. It may indeed turn out that an observed trend is in fact a cyclic variation that only becomes visible when the process is observed over a very long time period, nevertheless it remains worthwhile to consider such cyclic components as trends when the actual wavelength is far greater than the observed time

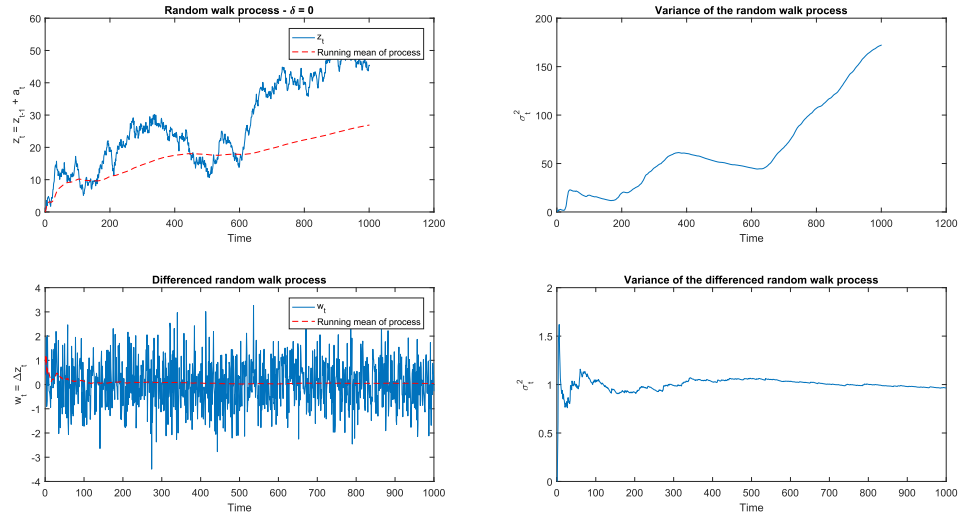

Fig. 3.3 Plot of 1000 observations of a random walk process with no addtional drift i.e. $\delta = 0$. The plot reveals that the mean of the process varies significantly over time. This, however, is not due to a deterministic trend in the process, but rather it is a result of the fact that the random walk process is cumulative or integrated. The variance of the random walk process also shows a significant dependence of the variance on time. By contrast, the differenced random walk, w_t, shows a fairly constant mean and variance.

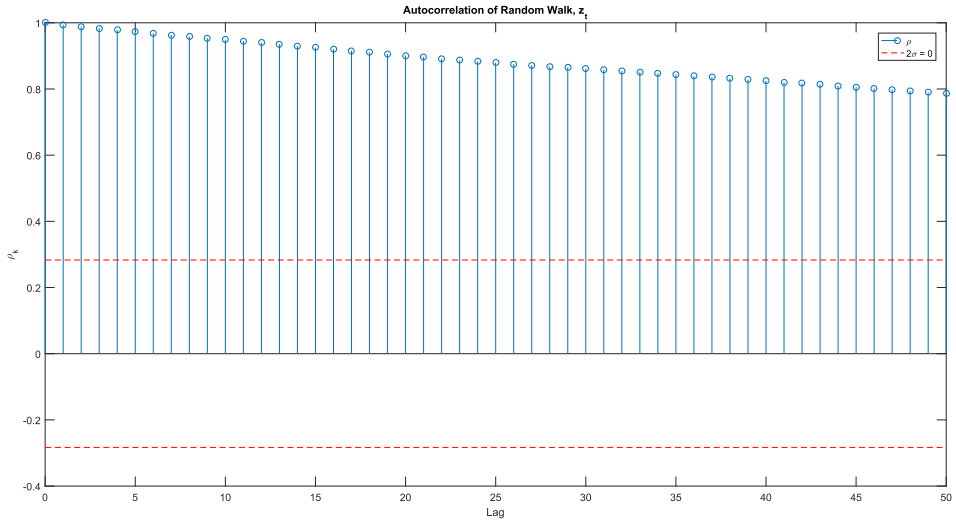

Fig. 3.4 Plot of the autocorrelations of the above random walk process over 50 lags. The plot exhibits the typical slow decay of the autocorrelations indicative of a unit root process.

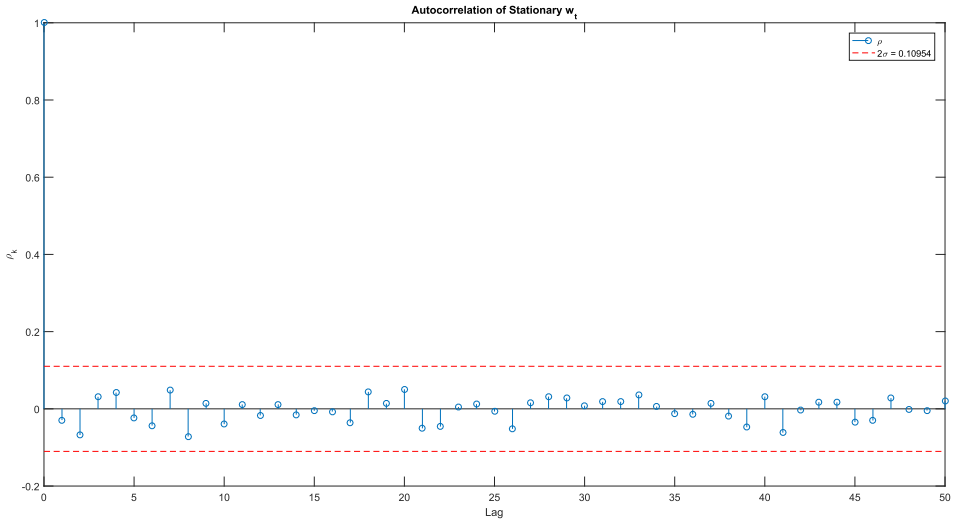

Fig. 3.5 Plot of the autocorrelations of the once differenced random walk process $w_t = (1-B)z_t$ over 50 lags. The plot shows no significant autocorrelations for lags > 0. This is what is to be expected from an uncorrelated and stationary white noise process, $w_t = a_t$.

series [51]. Assuming an observed process $\{x_t\}$ is comprised of a linear trend added to white gaussian noise ε_t, $\{x_t\}$ may formulated as

$$\begin{aligned} x_t &= m_t + \varepsilon_t \\ x_t &= \alpha + \beta t + \varepsilon_t, \end{aligned} \tag{3.110}$$

where mean level of the process is then given by $m_t = \alpha + \beta t$. This may be referred to as the "trend term". Ideally the trend term would be a deterministic function of time describing the global trend of the process. However, in reality, the scope of the trend term may be reduced to describing only local trends; and consequently the parameters α and β may be required to vary over time. There are typically two ways in which the trend in data may be dealt with. Firstly, the trend may be modelled with the view to either use the model to remove the trend from the data or to use the model forecasting. Secondly, a transformation may be applied to the data to simply eliminate the trend. A general method of modelling the trend in a time series is to fit polynomials to the data. For example, one might attempt to fit a parametric family of curves to the data of the form

$$m_t = a_0 + a_1 t + a_2 t^2, \tag{3.111}$$

The parameters a_0, a_1, and a_2 can then be obtained by a least squares estimation procedure where the function $\sum_t (x_t - m_t)^2$ is minimised [8].

Another useful and rather general way of handling trend is to consider it as the task of defining a linear filter whose output has all the long term variation removed i.e. a high pass filter [51]. This typically takes the form of a smoothing or averaging operation on the data such as

$$Sm(x_t) = \sum_{\tau=-m}^{n} a_\tau x_{t+\tau}, \tag{3.112}$$

where $\{a_\tau\}$ is a set of weights (or filter coefficients), and $Sm(x_t)$ represents the smoothing operation applied to x_t. The smoothed time series can then be used for forecasting or subtracted from the original time series to obtain residuals, where the long term trend is now no longer present.

$$Res(x_t) = x_t - Sm(x_t). \tag{3.113}$$

A non-stationary time series that can be made stationary via the removal of trend is said to be stationary about a trend or trend-stationary. Note here that subtracting the time dependent mean, m_t, from the process is indeed an effective way of treating a trend-stationary process in order to make it stationary. However, this treatment is not entirely effective if the process is characterised by time dependent variance. To treat this type of non-stationary process, the differencing operation is more suitable [27]. To see this, consider the random walk with linear drift process $x_t = m_t + x_{t-1} + \varepsilon_t$. Assume the trend component has been successfuly modelled as $m_t = \alpha + \beta t$. If the trend term is now subtracted from the time series the following is obtained:

$$x_t - m_t = x_0 + (\varepsilon_1 + \varepsilon_2 + ... + \varepsilon_t), \tag{3.114}$$

where x_0 is the starting value of the random walk and $\varepsilon_t \sim N(0, \sigma^2)$. The expected value of $(x_t - m_t)$ is then x_0, which is constant over time. The variance of $(x_t - m_t)$ however is $t\sigma^2$, which is still time dependent. By contrast taking the difference of the random walk with linear drift the following is obtained:

$$\begin{aligned}
\nabla x_t &= \nabla(m_t + x_{t-1} + \varepsilon_t) \\
\nabla x_t &= (m_t - m_{t-1}) + (x_{t-1}x_{t-2}) + \varepsilon_t - \varepsilon_{t-1} \\
\nabla x_t &= \beta + \varepsilon_{t-1} + \varepsilon_t - \varepsilon_{t-1} \\
\nabla x_t &= \beta + \varepsilon_t.
\end{aligned} \tag{3.115}$$

where

$$\begin{aligned}
(m_t - m_{t-1}) &= (\alpha + \beta t) - (\alpha + \beta(t-1)) \\
(m_t - m_{t-1}) &= \alpha + \beta t - \alpha - \beta t + \beta 1 \\
(m_t - m_{t-1}) &= \beta
\end{aligned} \tag{3.116}$$

The expected value of ∇x_t, $E[\beta + \varepsilon_t]$ is β, which is constant in time. The variance of ∇x_t, $E[(\beta + \varepsilon_t - \beta)^2] = E[\varepsilon_t^2]$, is σ^2, which is also constant in time. Thus we see that the differencing operation is able to handle variance non-stationarity, whereas the trend estimation and removal techniques are only applicable when the variance of the process is assumed to be constant. Hence we make the distinction between trend stationary processes and difference stationary processes.

3.16.3 Seasonality

Seasonality is a phenomenon that arises when the observations of a process typically manifest patterns which are cyclic or periodic in nature. This can often be seen in data that is collected on a daily, weekly, monthly etc. basis. When dealing with non-stationary data that is seasonal, it is important to note that the analysis of the data must be considered at two (or more) timescales; namely from observation to observation and from season to season. Both the non-stationarity of the data and the corresponding stationary $ARMA(p,q)$ model, which describes the dependence between observations, must be dealt with at each identified timescale. The observation to observation timescale may be defined as the "one lag" timescale, where each sample is separated by one time lag. The season to season timescale be defined as the "s lag" timescale, where each sample is separated by s lags; and s is the suspected period of the seasonal component in the data.

In section 3.16.1, the differencing operator defined as $\nabla = (1-B)$ was used to make a non-stationary process z_t stationary by applying the operator ∇ d times to the time series resulting in the stationary process $w_t = \nabla^d z_t = (1-B)^d z_t$. This operation was in fact dealing with the non-stationarity at the observation to observation scale. This can be made more explicit by defining the operator as $\nabla_1 = (1-B)$, where ∇_1 denotes the fact that the difference

is computed between observations separated by one time lag. Using this notation, we can define the seasonal differencing operator as $\nabla_s = (1 - B^s)$ where ∇_s denotes the fact that the difference is now computed between observations separated by s time lags. This operation can in turn be interpreted as dealing with the non-stationarity at the season to season or s lag timescale.

To facilitate the discussion, we make use of the airline data provided as "Series G" in [6]. The observations in series G are the passenger totals takens on a monthly basis from January 1949 to December 1960.

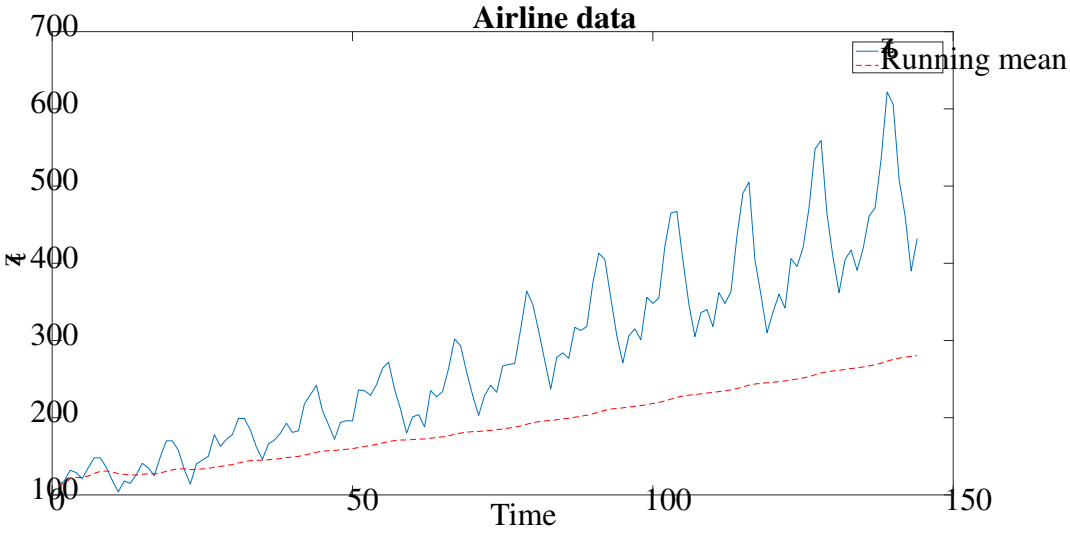

Fig. 3.6 Time series plot of the original airline data of Series G in Box and Jenkins. The mean of the process as it evolves in time is superimposed.

From the mean and variance plots of the airline data we can conclude that the process is non-stationary as both the mean and variance appear to increase with time. Additionally, the behaviour of the process appears to repeat itself every so often, albeit at different levels and with greater magnitude of variation. This suggests that the data may contain a seasonal component. This can be confirmed with a plot of the autocorrelation of the data.

The autocorrelations of the airline data reveal two noticable features of the data. Firstly, the slow decay in autocorrelation of the data, which is indicative of non-stationarity. In general, the presence of a trend in a time series tends to cause the autocorrelations to decay slowly to zero, due to the fact that an observation on one side of the mean tends to be followed by many more observations on the same side of the mean [51]. Secondly, there are fairly significant spikes or kinks located at lags 12, 24, 36 and 48, which is indicative of a seasonal component with a period of 12 lags. In general, time series with a seasonal component

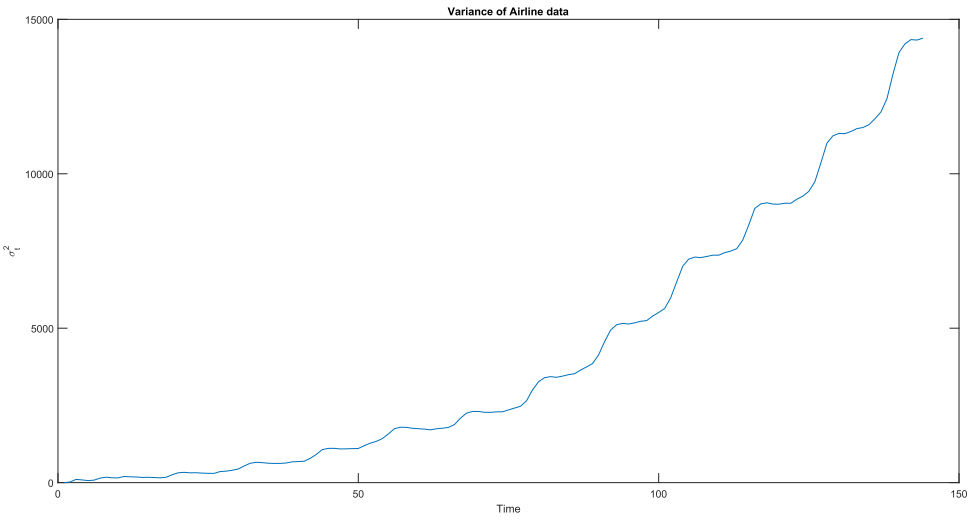

Fig. 3.7 Time series plot of the variance of the airline data as it evolves in time.

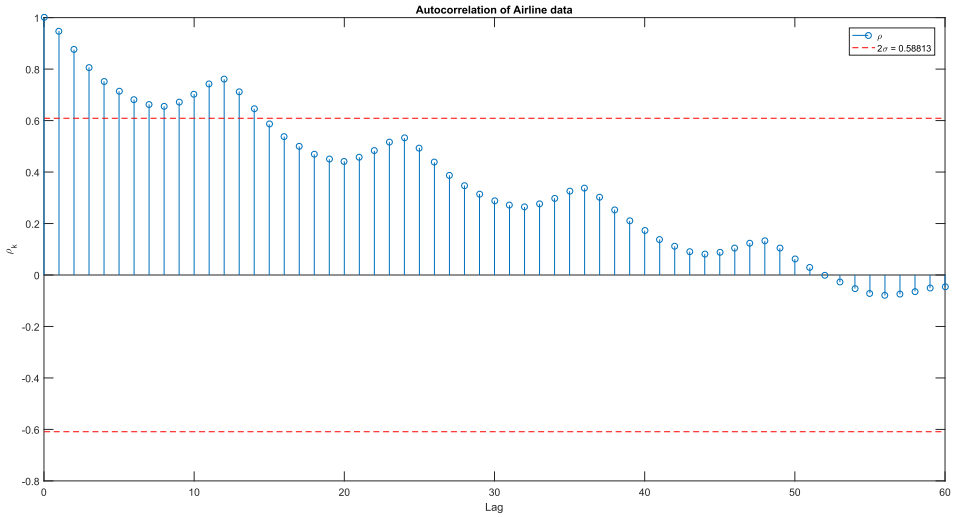

Fig. 3.8 A plot of the autocorrelation between the observations of the airline data at different time lags.

will tend to produce strong autocorrelations at lags corresponding to the oscillation period [51]. In addition, when taken separately, the autocorrelations between observations at 12 lag intervals also appear to decay rather slowly, indicative of a trend that is present at the "12 lag" timescale. Based on the plots of mean and variance, our intuition from section 3.16.1 is that a differencing of the airline data may be in order, since both the mean and variance appear to be time dependent.

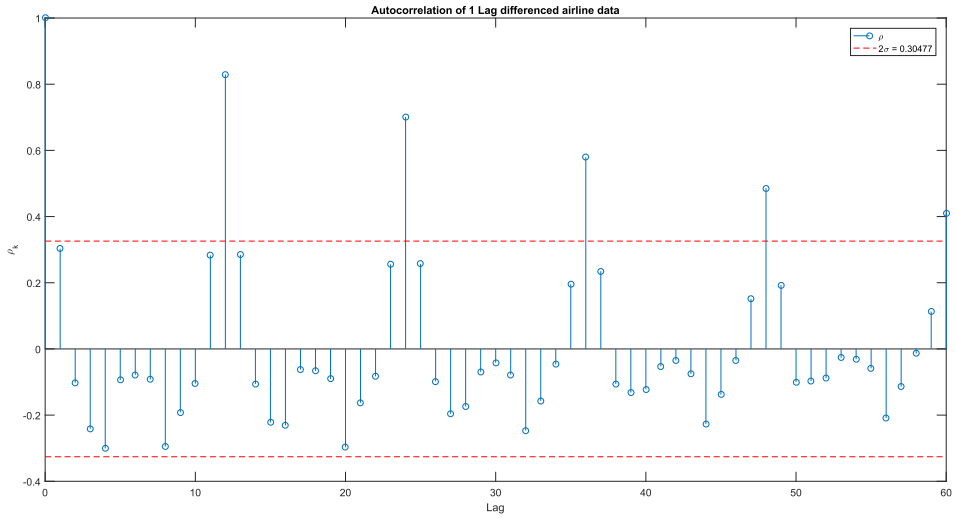

Fig. 3.9 A plot of the autocorrelation between the one lag differenced observations of the airline data.

From the autocorrelation plot (or correllogram) of the airline data, which has been operated on by ∇_1, the slow exponential decay of the autocorrelations has now given way to sinusoidally decaying autocorrelations. Thus it can be seen that the seasonal component of the data is still very much present, and made even more evident. This prompts us to apply the seasonal differencing operator ∇_s to the data instead, where $s = 12$ in this instance.

The resulting autocorrelations from having applied the ∇_{12} operator to the airline data shows a significant reduction in the autocorrelations at the identified period $s = 12$. Nevertheless, there appears to be significant correlation between observations at one lag intervals, which decays fairly slowly (about 10 lags) suggesting the possible presence of a remaining trend at the one lag timescale. This intuition is further supported by the plot of the variance shown in figure 3.11.

Now, applying both the ∇_1 and ∇_{12} operators in succession to the data, we obtain the correlogram as shown in figure 3.13. The resulting autocorrelations no longer appear to be significant, and the plot of the running mean in figure 3.14 reveals a stabilisation of the mean.

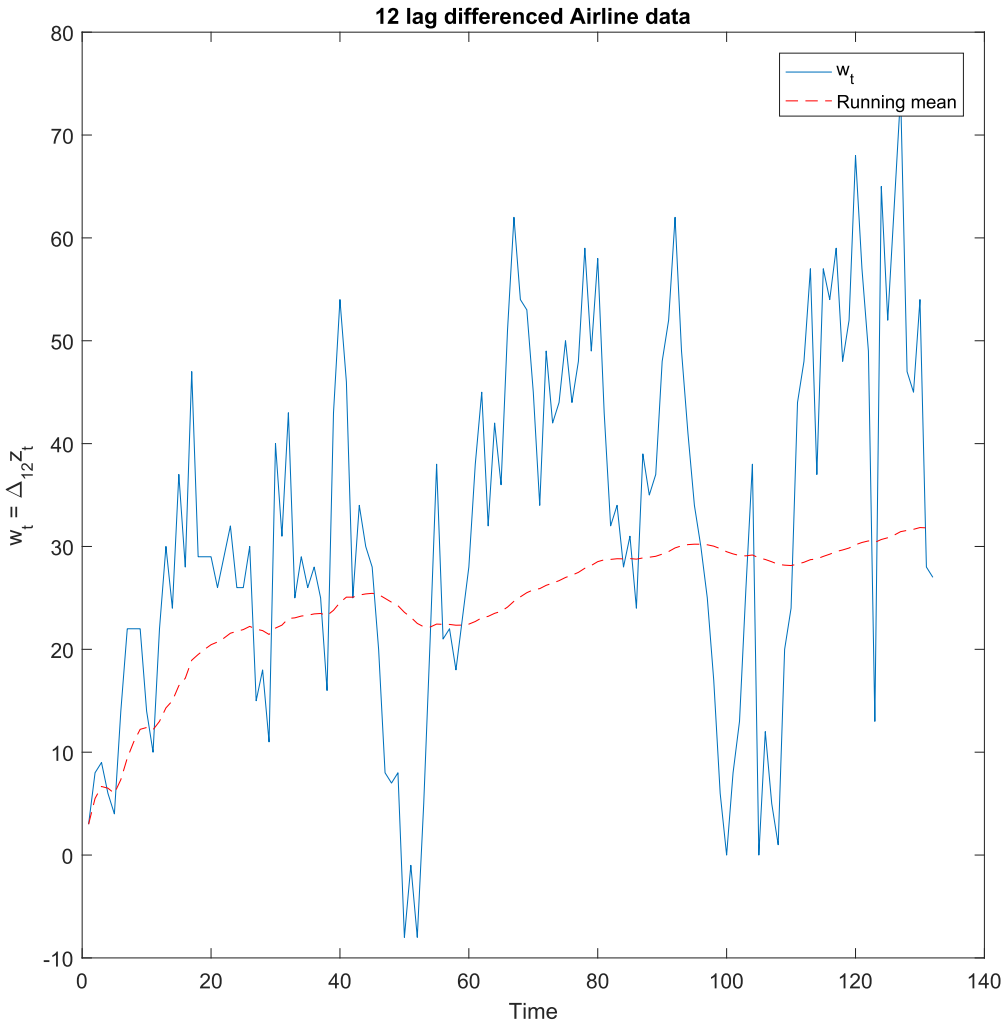

Fig. 3.10 A plot of the 12 lag differenced observations of the airline data, with the running mean superimposed in red.

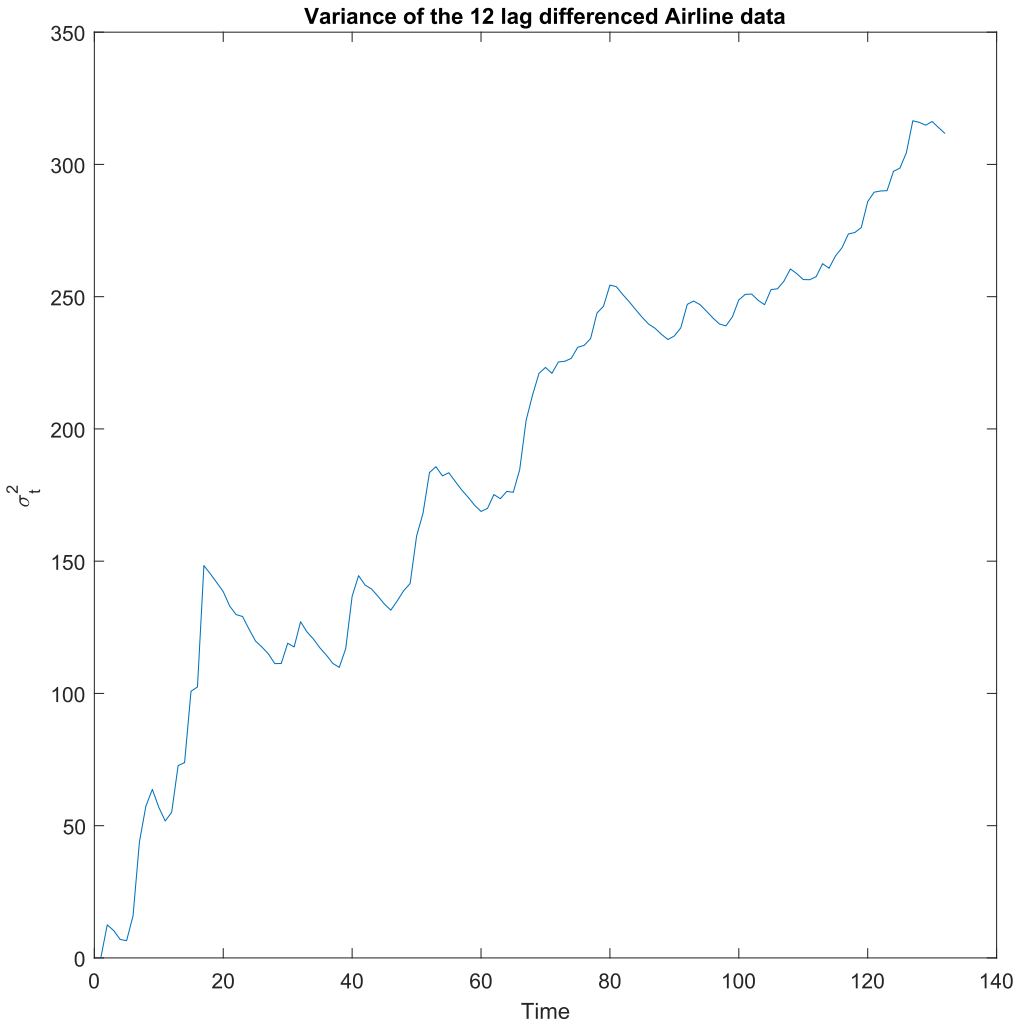

Fig. 3.11 Time series plot of the variance of the seasonally differenced airline data as it evolves in time.

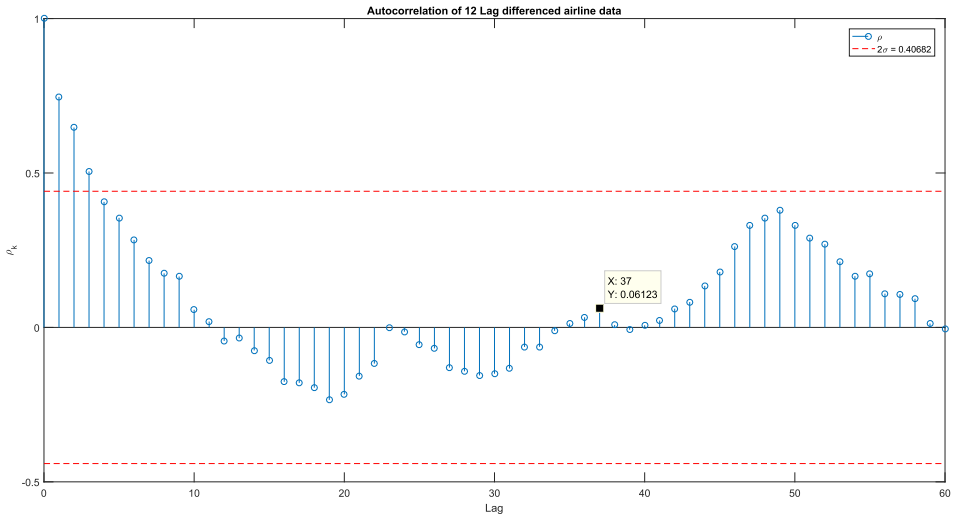

Fig. 3.12 A plot of the autocorrelation between the 12 lag defferenced observations of the airline data.

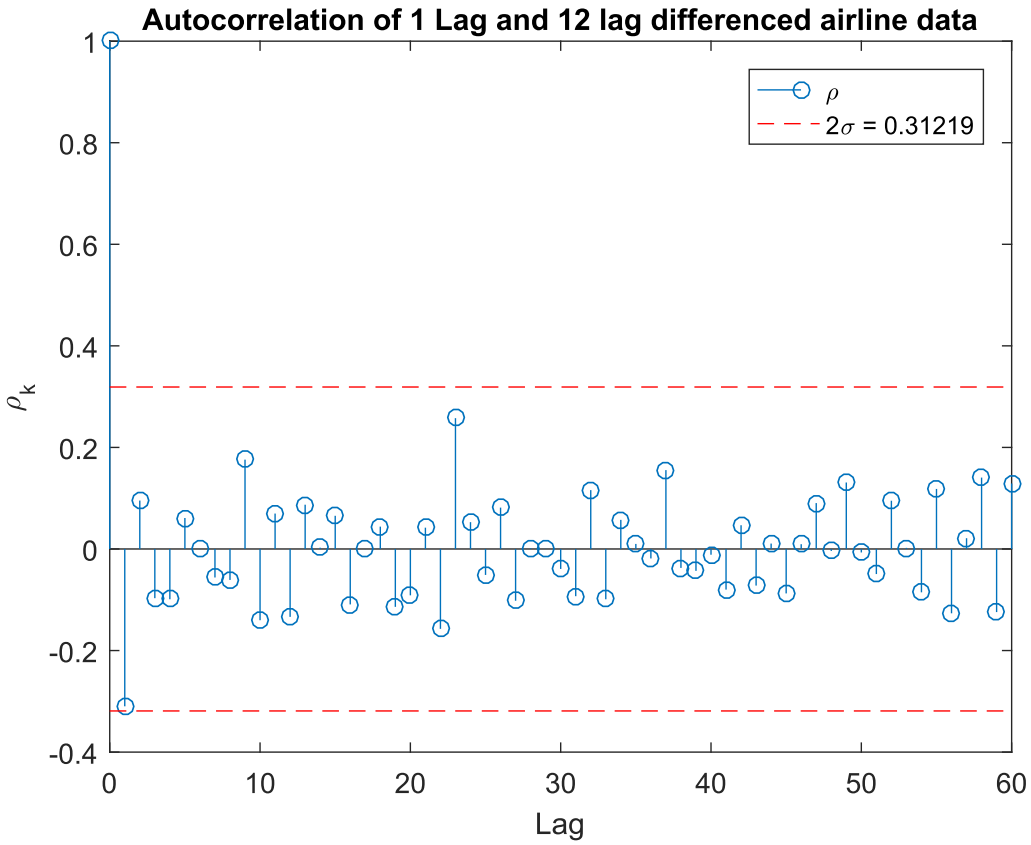

Fig. 3.13 A plot of the autocorrelation between the twice differenced (1 and 12 lag) observations of the airline data.

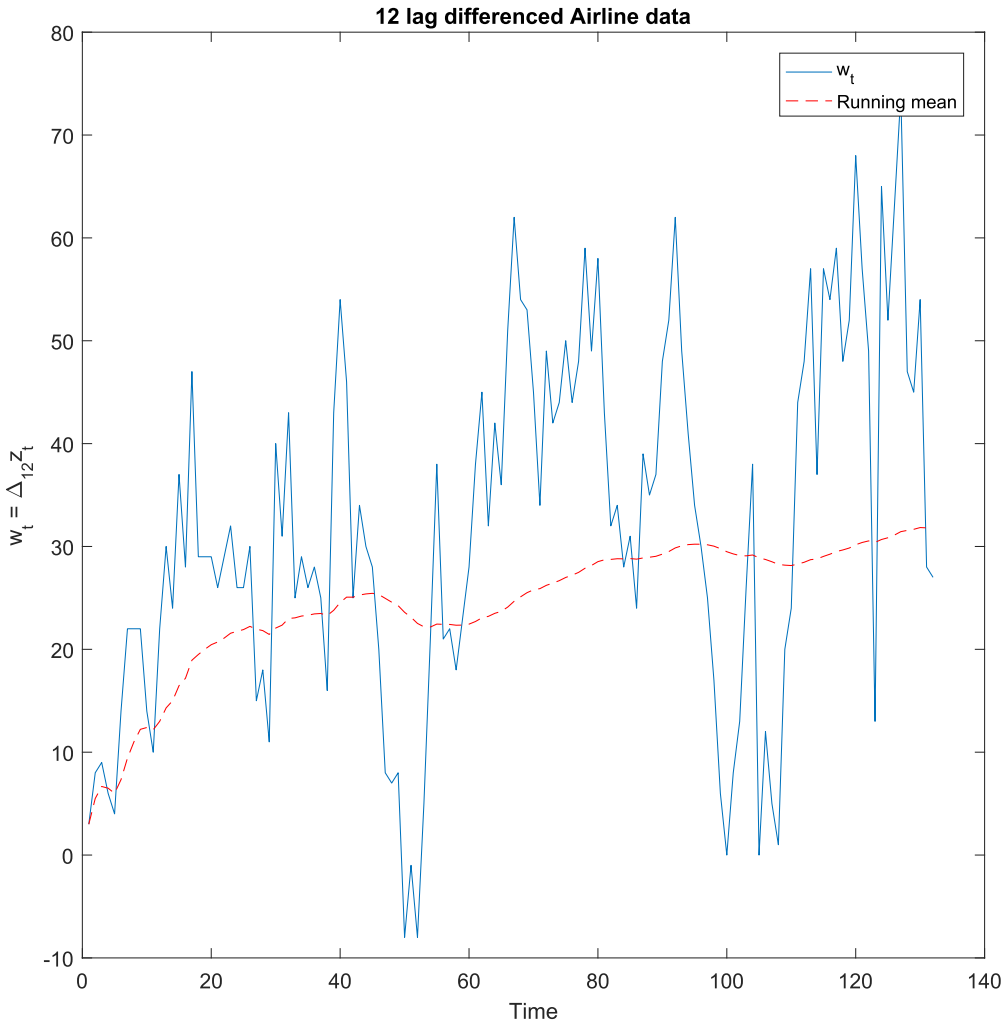

Fig. 3.14 A plot of the twice differenced (1 and 12 lag) observations of the airline data, with the running mean superimposed in red.

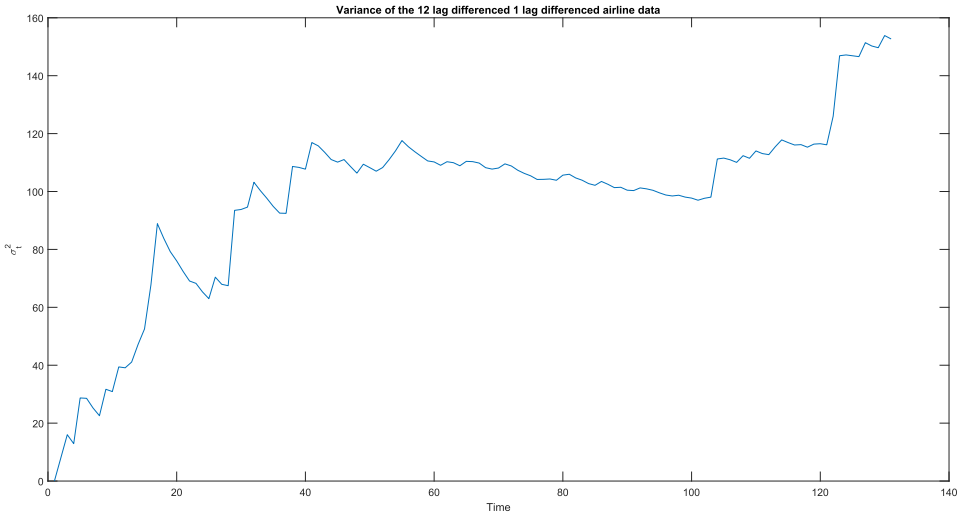

Fig. 3.15 Time series plot of the variance of the twice differenced (one and twelve lag) airline data as it evolves in time.

Comparing the original airline data time series z_t with the twice differenced airline data $\nabla_1\nabla_{12}z_t$ in terms of their variances, we see a significant reduction and stabilisation in the variance of the $\nabla_1\nabla_{12}z_t$ as opposed to the exponentially increasing variance of z_t over time shown in figure 3.7. While evidently not perfect, we may now tentatively attempt to formulate models for the two timescales based on the now stationary time series $\nabla_1 z_t$ and $\nabla_{12}z_t$. Keeping in mind that there are two time scales that are of interest, we may formulate the overall model as two separate models each operating at a different time scale, the one lag timescale and the s lag timescale. The two models may then be multiplied together to form the overall model. This is what is called a multiplicative model.

The fundamental fact about seasonal time series is that, if the time series has a seasonal component with period s, then the series will exhibit strong correlation at lags equal to s and possibly multiples thereof [8, 27]. Tabulating the seasonal data by stacking the observations in rows of length s, where each new row is a continuation of the original time series, a possibly more intuitive representation of the two time scales of the data emerges. The observations running across each row from left to right are then separated by one lag, while the observations running down each column from top to bottom are separated by s lags.

Year	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1949	112	118	132	129	121	135	148	148	136	119	104	118
1950	115	126	141	135	125	149	170	170	158	133	114	140
1951	145	150	178	163	172	178	199	199	184	162	146	166
1952	171	180	193	181	183	218	230	242	209	191	172	194
1953	196	196	236	235	229	243	264	272	237	211	180	201
1954	204	188	235	227	234	264	302	293	259	229	203	229
1955	242	233	267	269	270	315	364	347	312	274	237	278
1956	284	277	317	313	318	374	413	405	355	306	271	306
1957	315	301	356	348	355	422	465	467	404	347	305	336
1958	340	318	362	348	363	435	491	505	404	359	310	337
1959	360	342	406	396	420	472	548	559	463	407	362	405
1960	417	391	419	461	472	535	622	606	508	461	390	432

In order to model the seasonal component of the data, we then focus on the columns of table and make the assumption that all of the columns conform to the same seasonal *ARMA* model. Looking at any single column, we recognise that the observations under consideration are all separated by s lags. Consequently, the autoregressive and moving-average polynomials used to model the seasonal component will be defined in terms of $\tilde{B} = B^s$ as opposed to B.

$$\begin{gathered} \Phi*(\tilde{B})z_t = \Theta*(\tilde{B})u_t \\ (1-\Phi_1^*B^s-\Phi_2^*B^{2s}-...-\Phi_P^*B^{Ps})z_t = (1-\Theta_1^*B^s-\Theta_2^*B^{2s}-...-\Theta_Q^*B^{Qs})u_t \end{gathered} \tag{3.117}$$

where $\{u_t\}$ is an input process which may not necessarily be uncorrelated. The effect of defining the *ARMA* polynomials in terms of B^s is seen in equation 3.117, where the coefficients of $\Phi*(\tilde{B})$ and $\Theta*(\tilde{B})$ are not applied to adjacent observations as in $(1+B+B^2+...)z_t$, but rather to observations separated by s lags as in $(1+B^s+B^2s+...)z_t$, where B is the backshift operator. In this way, the $ARMA(P,Q)$ model specified by $\Phi*(\tilde{B})$ and $\Theta*(\tilde{B})$ captures the correlation between every s^{th} observation of the process, which essentially models the seasonal component of the process. P and Q in this instance denote the order of the autoregressive and moving-average polynomials in $\tilde{B}$, where $\tilde{B} = B^s$. The model obtained here can be seen to be describing the column-wise data in table 3.16.3, which pertains to the airline passenger counts of the same month year on year; e.g. the correlation between passenger counts in January 1949 and January 1950. It is often the case that $\{u_t\}$ itself may be described by an *ARMA* model as

$$\begin{aligned} \phi(B)u_t &= \theta(B)\varepsilon_t \\ (1-\phi_1 B-\phi_2 B^2-\ldots-\phi_p B^p)u_t &= (1-\theta_1 B-\theta_2 B^2-\ldots-\theta_q B^q)\varepsilon_t \end{aligned} \tag{3.118}$$

where ε_t is an uncorrelated guassian noise process $\varepsilon \sim N(0,\sigma^2)$. The $ARMA(p,q)$ model that describes $\{u_t\}$ essentially models the observation to observation (separated by one lag) correlation in the overall process $\{z_t\}$. Looking at the example airline data, we can interpret this model as describing the month to month variation in passenger counts over the course of a number of years; e.g. the correlation in passenger counts from January to December of 1950. Substituting the expression we get for $\{u_t\}$ in terms of $\phi(B)$, $\theta(B)$ and ε into equation 3.117, the resulting multiplicative model takes the form

$$\begin{aligned} \Phi*(\tilde{B})z_t &= \Theta*(\tilde{B})\frac{\theta(B)}{\phi(B)}\varepsilon_t \\ \Phi*(\tilde{B})\phi(B)z_t &= \Theta*(\tilde{B})\theta(B)\varepsilon_t. \end{aligned} \tag{3.119}$$

Equation 3.118 can be generalised to include the case where a differencing is required to remove a trend from the process $\{u_t\}$, so that its $ARMA(p,q)$ model can be formulated based on the stationary time series $u'_t = \nabla u_t$. In general, d number of differences may be required to bring about the stationarity of the time series, thus $u'_t = \nabla^d u_t$ is obtained. The $ARMA(p,q)$ model formulated based on a d times differenced time series may be referred to as a non-stationary ARIMA(p,d,q) model, where it is made more explicit that the model describes a non-stationary time series u_t that is a d times integrated version of a stationary time series u'_t.

If the time series contains a seasonal component which includes a seasonal trend, then a seasonal differencing may be applied to remove the seasonal trend so that $z'_t = \nabla_s^D z_t$ is obtained, where D is the number of times the seasonal differencing, ∇_s, must be applied to bring about stationarity of the time series at the s lag timescale. The corresponding model of the seasonal component can be written as $ARIMA(P,D,Q)_s$, which additionally indicates, via the subscript, that the model describes a seasonal component with period s. Combining the two models obtained for the two timescales of the process, the multiplicative seasonal model $ARIMA(p,d,q)\times ARIMA(P,D,Q)_s$ is arrived at. This may also be referred to as a SARIMA model. Letting $w_t = \nabla^d \nabla_s^D z_t$, the general SARIMA model takes the form:

$$\begin{aligned} \Phi*(B^s)\phi(B)w_t &= \Theta*(B^s)\theta(B)\varepsilon_t \\ \Phi*(B^s)(1-B^s)^D\phi(B)(1-B)^d z_t &= \Theta*(B^s)\theta(B)\varepsilon_t \\ \Phi*(B^s)\phi(B)\nabla_s^D\nabla^d z_t &= \Theta*(B^s)\theta(B)\varepsilon_t. \end{aligned} \tag{3.120}$$

3.17 Wavelet Analysis

In this section we will briefly summarise the basic intuition behind wavelet analysis. While wavelet analysis has many an application, our particular interest stems from our desire to firstly estimate the Hurst or fractal dimension of a given signal and secondly to denoise a signal where necessary. Of course due localisation in time of a wavelet decomposition, one may also identify "interesting" events in the analysed signal. This section is not intended to be a comprehensive text of wavelets, but rather aims to provide a lay intepretation of some deep topics and ideas in wavelet theory. Just enough to start following the ideas which are presented more with much more rigour and detail [80], [16], [85] and [59].

To begin understanding wavelet analysis, one has to go back to vector calculus - to the dot product and the vector product. In this setting, if we have a 2-dimensional vector a in the $x-y$ plane, we can fully describe this vector according to its x-coordinate and its y-coordinate. We can also partially describe this vector according to the amount of the vector which is captured/described by a single axis i.e. by projection of the vector onto either of the x or y axis, a_x and a_y respectively. This is the key idea behind the wavelet transform. Projecting a function onto an axis or basis in order to view the amount which such a projection is able to capture or describe the target function.

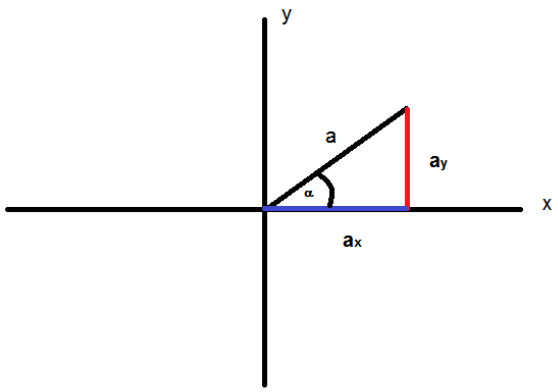

Fig. 3.16 Projections of a vector a onto orthogonal axes x and y as shown by the components a_x and a_y respectively.

To make this exercise more meaningful, the bases onto which the function is projected may be imbued with some desirable properties like orthogonality or normality. Furthermore the bases might be chosen such that they expose particular features of the target function. An example of this is the Fourier transform. When performing a Fourier analysis of some signal, the signal is projected onto a set of orthogonal bases composed of sines and cosines.

Since the frequency of these sinusoids is adjustable, it is possible to observe how much of the signal is present in, or described by, sinusoids at the desired frequencies. This is what gives us the frequncy spectrum of the signal. Yet another example of this is Principal Component Analysis. Here an eigen-decomposition of an n-dimensional signal is performed. The eigen-vectors produced are known to be linearly independent or orthogonal. The eigen-values produced quantify the amount of variance in the signal that is captured by the corresponding eigen-vector. In order to compress the signal or reduce its dimensionality, the signal is then projected onto its main (principal) components, which are the eigen-vectors that have the largest eigen-values. In so doing, one assumes the basis which they have chosen sufficiently describes the target signal.

One of the main motivations for the use of wavelet decomposition is to perform some sort of spectral analysis, much like the Fourier transform. A particularly interesting feature of wavelet analysis is that the basis chosen is finite in its extent (as opposed to the infinite extent of the sines and cosines of the Fourier transform). As a result, the wavelet transform provides a localised analysis of a given signal. This is to say that, where the Fourier transform exposes the frequency content of a signal "frozen in time", the wavelet transform exposes the spectrum as it evovles in time. This is a particularly nice feature to have when dealing with non-stationary signals. Now, let $\phi(x)$ define a function, which will be called the scaling function. This will form part of the basis onto which a signal will be projected. For the sake of simplicity, the example of the Haar wavelet will be used. The Haar wavelet scaling function is defined as

$$\phi(x) = \begin{cases} 1 \text{ for } 0 \leq x < 1 \\ 0 \text{ otherwise .} \end{cases} \tag{3.121}$$

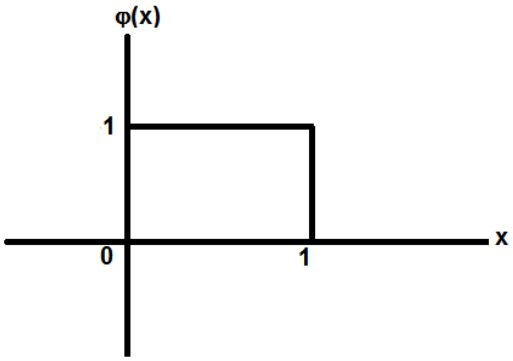

Fig. 3.17 Haar scaling function.

The corresponding wavelet function is then given by

$$\psi(x) = \begin{cases} 1 \text{ for } 0 \leq x < 1/2 \\ -1 \text{ for } 1/2 \leq x < 1 \\ 0 \text{ otherwise .} \end{cases} \tag{3.122}$$

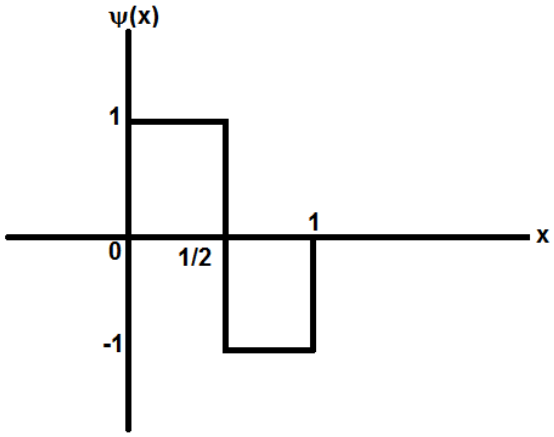

Fig. 3.18 Haar wavelet function.

The scaling function and the wavelet function can be considered to be complementary operations, where the scaling function performs a low-pass filtering of the data and the wavelet function performs a high-pass filtering. A basis for analysis can be constructed from these two functions by dilating and translating them as well as by imposing some restrictions that will make for a more meaningful or easier analysis. The action of dilation (or compression) of a function involves a scaling(or multiplication) of the independent variable. Take the example of the scaling function $\phi(x)$. If the independent variable x is scaled by a factor of two, the result is $\phi(2x)$. This is to say that for every unit increase in the variable x, our function $\phi(2x)$ will respond as if it had experienced a 2 unit increase. This can be seen mathematically by substituting $x = x+1$ into the scaled function $\phi(2x)$. This results in $\phi(2(x+1)) = \phi(2x+2)$, a two unit increase. Intuitively this means that the function $\phi(2x)$ will reach any value of $\phi(x)$ in half the time (or space) i.e. the function is compressed by a factor of two. Likewise, had the function been scaled by $1/2$, the result would be $\phi(1/2x)$. This function would take twice as long to reach the same value as $\phi(x)$, thus resulting in a dilation of $\phi(x)$ by a factor of two. The figures below demonstrate this graphically.

In order to construct a basis a set of wavelet functions which are dyadically compressed is compiled. That is, the wavelet function is compressed by a factor of two, then the result is

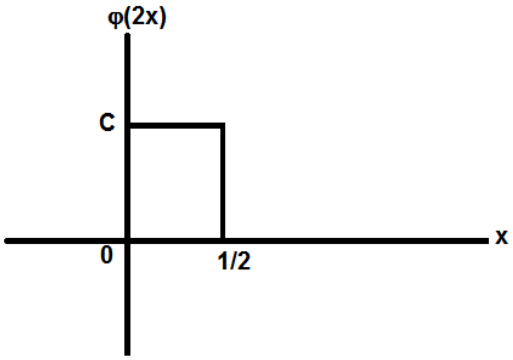

Fig. 3.19 Once dyadically compressed Haar scaling function

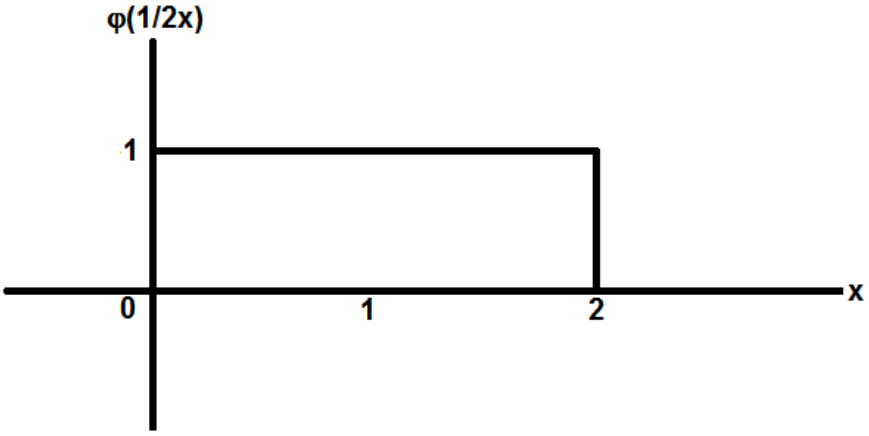

Fig. 3.20 Once dyadically dilated Haar scaling function

compressed in turn by a factor of two again. Continuing in this fashion of dyadically compressing the function j times, where j is an integer representing the number of compression levels achieved, the result obtained is $\psi(2^j x)$. At this point it should start to become apparent that by projecting some target signal/function onto these increasingly compressed functions, $\psi(2^j x)$, is in fact analysing the the signal at multiple $(j = 1, 2, ...)$ resolutions.

Because of the localisation of the wavelet function (as seen in the figures), the analysis of the signal would however be limited to within the domain $[0, 2^{-j}]$ at each level j. In order to span the entire length of the target signal, the set of dyadcally compressed wavelet functions must also be translated. Translation is the action of moving a function along the axis of the independent variable. This is achieved by simply adding or subtracting to the independent variable. For the example of the scaling function, this gives $\phi(x-k)$, where k is the amount to be added to the independent variable to move the function along the x-axis. The figure below shows this graphically.

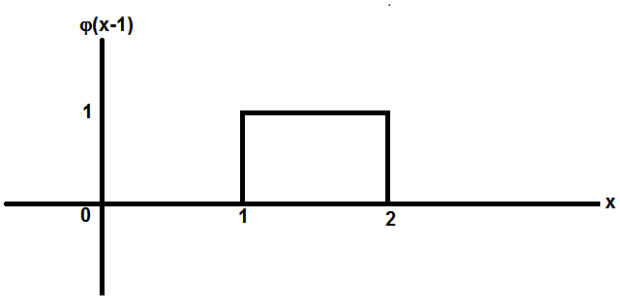

Fig. 3.21 Translated once dyadically compressed Haar scaling function.

Now, the basic machinery of a wavelet basis is available, which can be described quite generally and succinctly as $\psi(2^j x - k)$. This can also be written in the shorthand form $\psi_{j,k}$ where the sub-index j indicates the scale, resolution or compression(dilation) level of the wavelet function, and k indicates its translation.

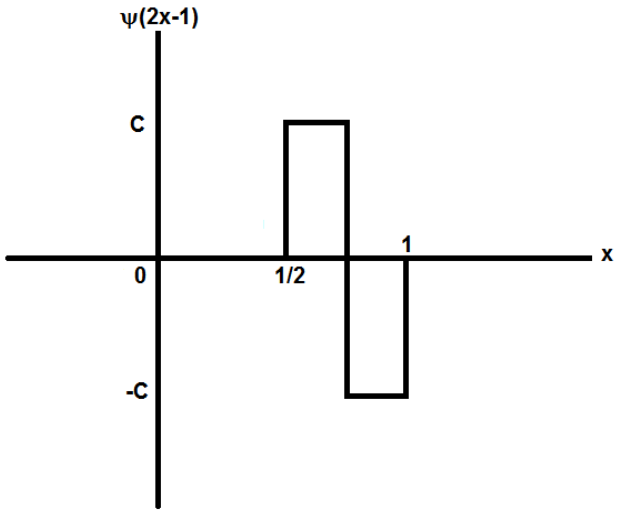

Fig. 3.22 Translated once dyadically compressed Haar wavelet function.

Orthogonality

As mentioned earlier, it may be desirable to impose certain restrictions on the wavelet basis for ease of analysis or computation. One such restriction, which is very common, is orthogonality. Orthogonality demands that the inner product of the wavelet function with integer translates of itself be equal to 0. Using bra-ket notation this can be stated as

$$< \psi_{j,k} | \psi_{j',k'} >= 0 \text{ when } j \neq j' \text{ or } k \neq k'. \tag{3.123}$$

This is akin to the requirement that the inner product of two vectors must be zero, if they are orthogonal. We see that for the Haar wavelet (and scaling) function the requirement in 3.123 is indeed met.

Normality

Another restriction encountered quite often, and is the case for the Haar wavelet, is that of normality. Simply stated, this restriction requires that the L^2 norm of the wavelet function be equal to one. What this implies is that

- the wavelet $\psi(x)$ is defined such that it has an area of one, and
- the energy in the wavelet function is preserved at all dilations(compressions) and translations.

The second point above requires that, as the dyadically compressed versions of $\psi(x)$ are formed, a constant factor C should be applied, which preserves the L^2 norm. This leads to the following general formulation:

$$\begin{aligned}
\int |C\dot{\psi}(2^j x-k)|^2\,dx &= 1,\\
|C|^2\int |\psi(2^j x-k)|^2\,dx &= 1,\\
C^2\int \psi^2(2^j x-k)\,dx &= 1,\\
\text{Let } t=2^j-k,\ \text{then } \frac{dt}{dx}=2^j \text{ and } dx=2^{-j}dt, \qquad & \qquad (3.124)\\
C^2\dot{2}^{-j}\int \psi^2(t)\,dt &= 1,\\
\text{Since } \int \psi^2(t)\,dt \text{ was defined to be 1,}& \\
C^2 &= 2^j,\\
C &= 2^{j/2}
\end{aligned}$$

And so, an orthonormal basis $\psi_{j,k}(x)=2^{j/2}\psi(2^j-k)$, is obtained.

3.17.1 Other constraints

One of the main features of wavelets is that they oscillate (being a wave), but are also compact (having finite support). Thus the following requirements are imposed on any choice of wavelet function

$$\int \psi(t)\,dt = 0, \tag{3.125}$$

and

$$\int_{-\infty}^{\infty} \psi(t)^2\,dt < \infty. \tag{3.126}$$

3.17.2 The Two-Scale or Dilation Equations

If the relationship between the basis functions at adjacent scales j and $j+1$ were to be established, it would in fact be looking for the dilation equation. In the context of dyadically compressed or dilated functions this amounts to the following two-scale relationship for the scaling function:

$$\phi(x) = \sqrt{2}\phi(2x) + \sqrt{2}\phi(2x-1). \tag{3.127}$$

What this equation communicates is that when dyadically compressing $\phi(x)$, two of the compressed functions (at scale $j+1$), $\sqrt{2}\phi(2x-k)$, are needed at translations $k=0$ and $k=1$ to represent the scaling function at the previous scale j. Furthermore, the reader will notice that each of the two scaling functions at scale $j+1$ have an L^2 norm equal to one. But, this now means that $\sqrt{2}\phi(2x) + \sqrt{2}\phi(2x-1)$ no longer has a unit L^2 norm. This can be seen graphically in the figure below.

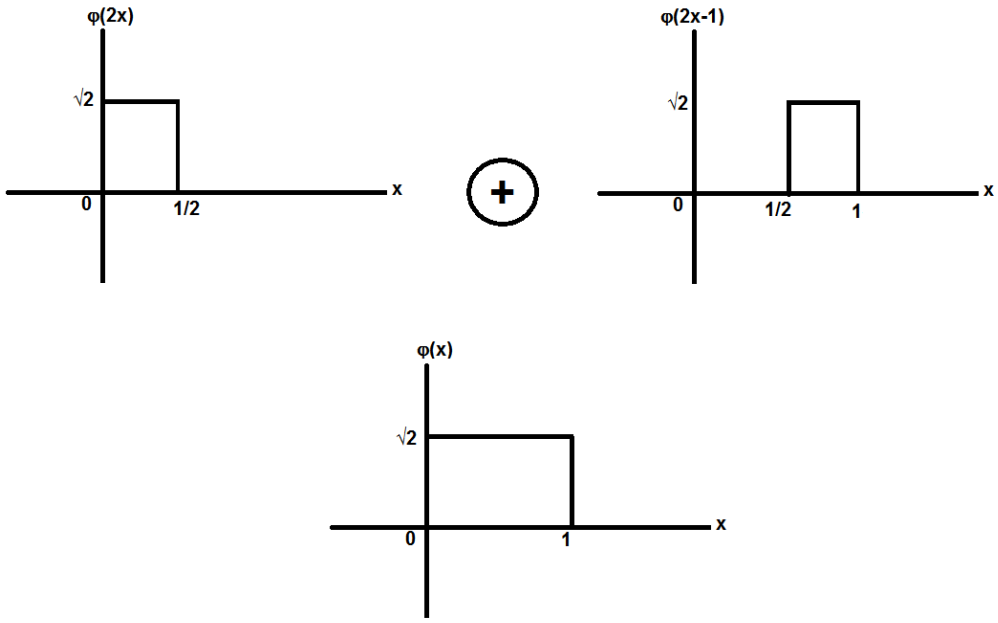

Fig. 3.23 Graphical depiction of the addition of the two translates of the normalised Haar scaling function at scale $j+1$ resulting in an unormalise scaling function at scale j

In order to remedy this, the basis functions at scale $j+1$ are linearly combined using a weighting function $h(k)$. From the above plot it is evident that the linear combination needed is

$$\phi(x) = \phi(2x) + \phi(2x-1). \tag{3.128}$$

In order to determine what the required weights of $h(k)$ are, the orthonormal bases $\sqrt{2}\phi(2x-k)$ are factored out from the RHS of equation 3.128 resulting in the following:

$$\phi(x) = \frac{1}{\sqrt{2}} \cdot \sqrt{2}\phi(2x) + \frac{1}{\sqrt{2}} \cdot \sqrt{2}\phi(2x-1). \tag{3.129}$$

Thus it is seen that $h(0) = \frac{1}{\sqrt{2}}$ and $h(1) = \frac{1}{\sqrt{2}}$. In fact, $h(k)$ is defined as $h(k) =< \phi(x), \sqrt{2}\phi(2x-k) >$. Having determined $h(k)$, the dilation equation for the scaling function can be written as

$$\phi(x) = \sum_k \sqrt{2}\phi(2x-k) \cdot h(k). \tag{3.130}$$

Following a similar line of reasoning, the dilation equation along with the weights $g(k)$ for the wavelet function can be obtained as

$$\begin{aligned} \psi(x) &= \psi(2x) - \psi(2x-1) \\ \psi(x) &= \frac{1}{\sqrt{2}} \cdot \sqrt{2}\psi(2x) - \frac{1}{\sqrt{2}} \cdot \sqrt{2}\psi(2x-1) \end{aligned} \tag{3.131}$$

Thus it is seen that $g(0) = \frac{1}{\sqrt{2}}$ and $g(1) = -\frac{1}{\sqrt{2}}$. Having determined $g(k)$, the dilation equation for the wavelet function can be written as

$$\psi(x) = \sum_k \sqrt{2}\psi(2x-k) \cdot g(k). \tag{3.132}$$

Quadrature Mirror Filters

Recalling the equations 3.130 and 3.132 the sequences $h(k)$ and $g(k)$ are Quadrature Mirror Filters (QMF) and they have the following relationship:

$$g(n) = (-1)^n h(1-n). \tag{3.133}$$

In addition, $h(k)$ is a low-pass filter whose coefficients sum to $\sqrt{2}$. On the other hand, $g(k)$ is a high-pass filter whose coefficients sum to zero.

Wavelet Analysis - Decomposition

Given a discrete function $f[n]$ on $[0, M-1]$, we would like to analyse this function according to our bases $\phi_{j,k}$ and $\psi_{j,k}$. This can be achieved by computing the dot product $< f[n]|\phi_{j,k}[n] >$ as well as $< f[n]|\psi_{j,k}[n] >$. The approximation coefficients, $c_{j,k}$, obtained by essentially low-pass filtering $f[n]$ are given by

$$c_{j,k} = \frac{1}{\sqrt{M}} \sum_n f[n] \cdot \phi_{j,k}[n]. \tag{3.134}$$

The detail coefficients, $d_{j,k}$, obtained by essentially high-pass filtering $f[n]$ are given by

$$d_{j,k} = \frac{1}{\sqrt{M}} \sum_n f[n] \cdot \psi_{j,k}[n] \tag{3.135}$$

This forms the core of analysis or decomposition of a function by a wavelet basis.

Wavelet Synthesis - Reconstruction

One of the desireable properties of using orthonormal wavelet bases is that perfect reconstruction of the analysed function follows quite easily. This is achieved by computing the product between the coefficients and the wavelet and scaling functions as follows:

$$f[n] = \frac{1}{\sqrt{M}} \sum_k c_{j_0,k} \phi_{j_0,k}[n] + \frac{1}{\sqrt{M}} \sum_{j=j_0}^{\infty} \sum_k d_{j,k} \psi_{j,k}[n]. \tag{3.136}$$

It is worth noting that in practise one need only retain the approximation coefficients at scale j_0 and the detail coefficients at all scales $j \geq j_0$. This is mainly due to the fact that the information contained in the approximation coefficients at higher scales is also present in the detail coefficients. To obtain a clearer view of this the reader is referred to the sections on subband coding in the following texts [80], [16], [85] and [59].

Chapter 4

Methodology

The task of modelling network traffic is a particularly challenging one as mentioned in the problem statement. The characteristics of this signal, some of which are non-stationarity, short-range dependence, long-range dependence, self-similarity and fractality, make it rather difficult for any one technique to successfully model it under general conditions. Many of the attempts that have been made, as seen in the literature, are at best able to address a small subset of particular aspects or instances of network traffic. These, in large part, rely on various assumptions that constrain the problem and are not always readily generalisable.

Time series analysis, at the most basic level, attempts to model an observed process as a transformation that has been applied to a random white noise process. It provides a fairly comprehensive suite of tools for modelling linear stationary processes with special regard to the dependence that exists between samples. These tools are particularly well suited to discrete data sampled at constant time intervals. While it is noted that the tools of time series analysis also rely on some assumptions to be made about the process being modelled, the reach of these tools can be readily extended to accommodate non-stationarity, fractality, and seasonality in the data amongst other complexities.

Box and Jenkins [6] prescribe a methodolgy for developing models of stochastic processes, using the tools of time series analysis. This methodology, since its initial publication, has come to be widely adopted in a plethora of time series applications. In this work, the very same methodology has inspired the formulation of a framework for modelling network traffic. In attempting to build models that describe the data that is observed from network traffic, the following basic steps, as advocated by Box and Jenkins, will be performed:

1. Prepare the data such that it is ammenable to analysis under the assumption of covariance stationarity (see [27]),

2. obtain a preliminary estimate of the parameters p and q, which are the orders of the autoregressive and moving-average polynomials respectively for an $ARMA(p,q)$ model (see [6],[27]),

3. estimate the parameters (that is the $p+q$ coefficients) of the autoregressive and moving-average polynomials $\phi(B)$ and $\theta(B)$ respectively that form part of a general $ARMA$ model,

4. perform diagnostic analysis to assess how well the model is able to reproduce the characteristics observed in the original data.

In addition to the traditional analysis leading to the formation of $ARMA(p,q)$ type models, the estimation of the real parameter d is undertaken thus extending our reach not only to $ARIMA(p,d,q)$ models where the value of d is still assumed to be an integer, but also to the $FARIMA(p,d,q)$ models which allow us to deal with fractality and in particular long range dependence.

4.1 Data Preparation

4.1.1 Data exploration

As a first step in the analysis process it is always useful to plot the observations in time. This can help in the visual identification of important features such as those mentioned above. In order to confirm our intuitions about the data, we may also look at the data through a number of different "lenses" in order to bring to light some of the characteristics of the data, which would inform how the analysis proceeds and modelling of the process. This in turn may lead to a re-evaluation of the scale or length of data to be used in the subsequent analysis or to apply a transformation to the data. This will referred to as the data exploration phase. For this particular work, the data exploration phase was comprised of plotting:

- the running mean,
- the running variance,
- the frequency spectrum,
- the cumulative periodogram.

The visual cues provided by the above plots were coupled with the following tests:

- Welch's t-test for constant mean [88, 53](or equal means between adjacent windows),
- Levene test (with trimmed means) for constant variance [9, 52] (or equal variance between adjacent windows),
- the K-S test applied to the cumulative periodogram [23, 5],
- significant Hurst, to test for fractility/LRD [82].

4.1.2 Data transformation

Based on the preceding data exploration, it may then be determined whether the data is suitable for time series analysis (under the assumption of stationarity) or whether a transformation of the data is needed to bring about stationarity. The network traffic data may then be treated in the following ways:

- If the mean varies with time, fit a polynomial or smooth the data (for example, using EWMA) to determine the trend. Then, subtract the mean trend from the observations.
- If the variance varies with time, apply the integer differencing operation to the data (possibly multiple times).
- If the variance and the mean vary in time and appear to be directly proportional, apply the log operator to the data.
- If there appears to be a cyclic component in the data estimate the period of the cycle or seasonal component and apply the seasonal differencing operation, if required.
- If the Hurst computed on the data is found to be significant (say greater than 0.7), then a fractional differencing operation is applied to the time series.

It is important to note that some transformations may also induce undesirable effects in the transformed data. Care should thus be taken to understand when these transformations are suitable and what the effects thereof are.

Dealing with trend

During the data preparation phase of this work, the Exponentially Weighted Moving Average (EWMA) is employed to perform the requisite low-pass filtering (smoothing) of the data, where applicable, in order to determine the trend in the data. This is a simple technique which is based on estimating the mean of the observed values of some generating process. With each new observation, x_t, the value of x_t is compared to an estimate of the mean, u, which is based on previous observations of the process [75]. Following the comparison of the latest observation, x_t, with the previously estimated mean, u, the mean is then re-estimated incorporating the current observation in the estimate, thus making the mean adaptive or sensitive to the latest trend in the data. The basic recursion equation of the procedure is given by:

$$u_t = \lambda * x_t + (1-\lambda) * u(t-1), \tag{4.1}$$

where λ is the EWMA factor. Expanding this formula over p iterations shows that the current observation at time t receives a weight of λ, while an observation from an earlier time $t-p$ receives a weight of $\lambda(1-\lambda)^p$. It is evident then that should one be interested in preserving the influence of observations for a longer period of time a small value for λ should be specified, while the converse is true should only the short term trend be of interest [94]. Formulating the EWMA smoothing operation as a filter, the following is obtained:

$$Sm(x_t) = \sum_{\tau=0}^{\infty} \lambda(1-\lambda)^{\tau} * x_{t-\tau}, \tag{4.2}$$

where $0 \leq \lambda \leq 1$, $a_\tau = \lambda(1-\lambda)^\tau$ decreases gemometrically as τ increases (i.e. the current mean value is influenced less and less by observations made earlier in the history of the process), and $\sum a_\tau = 1$.

From figure 4.1.2 it can be seen that the effects of an observation remain in the "memory" of the system for over 3000 time lags when the parameter $\lambda = 0.001$. The effect of such a long memory is depicted in 4.1.2 where the bursty and quite noisy input signal is almost completely smoothed. While the smoothed signal appears to converge quite consistently to a mean value of around 2700, meaning that it may be considered stationary, one must be cautious of oversmoothing to the extent that important features in the signal are lost. For this reason the choice of λ is a critical one as it may significantly influence the analysis that follows. In this instance it appears there is no significant trend, and a removal of the mean value suggested by the EWMA smoothing may be sufficient to proceed.

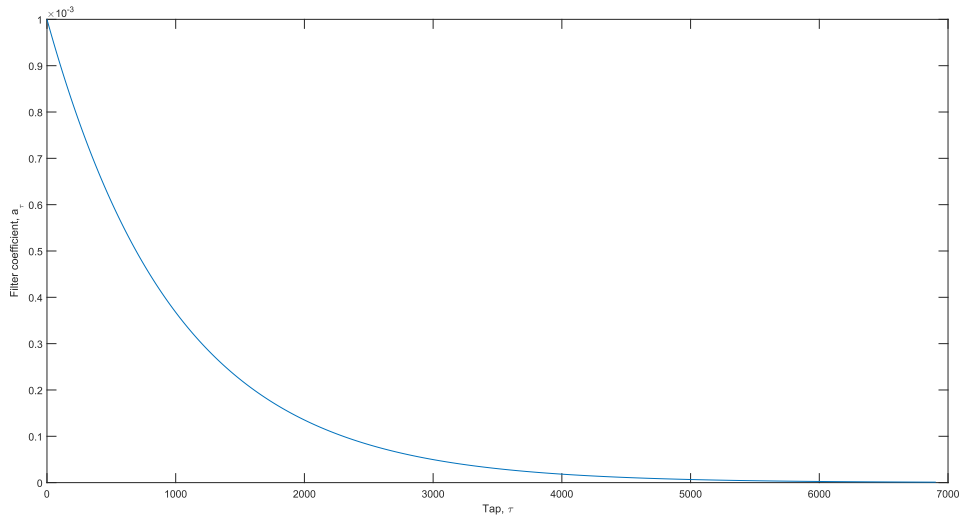

Fig. 4.1 Plot of the coefficients of an EWMA smoothing filter with $\lambda = 0.001$

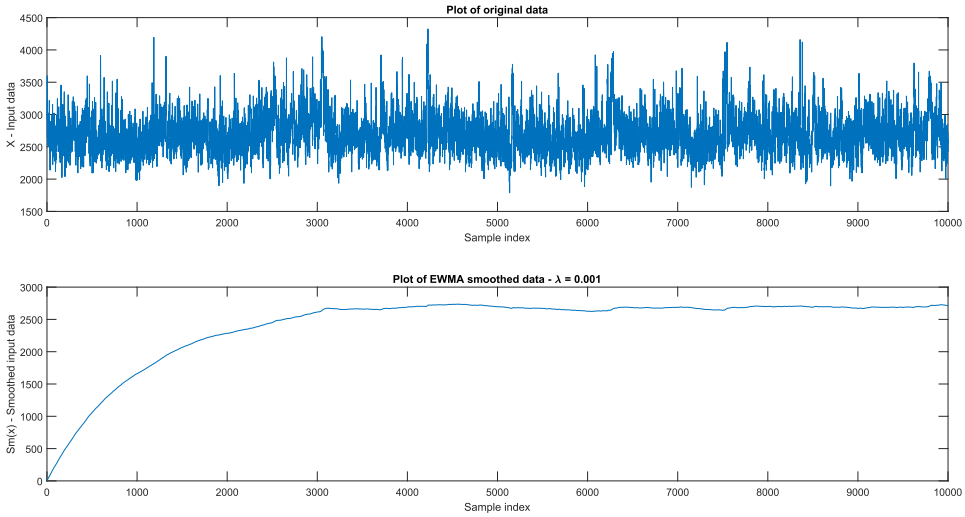

Fig. 4.2 Plot of 10000 observations of a network traffic process. The bottom plot shows the result of applying an EWMA smoothing filter with $\lambda = 0.001$ to the input data.

Estimation of Hurst

The Hurst exponent is a measure of the tendency of a time series to revert back or cluster to a long term equilibrium. Its value ranges from $[0,1]$. The effect of Hurst can be explained by considering its three key values:

- $H = 0$: This implies a mean reverting series; any positive movement is immediately followed by a negative movement in the series and vice versa. The result of this is that the process hovers around the mean over the long term, remaining fairly constant over time.
- $H = 0.5$: This implies a process with little to no correlation with its previous values. Such a characteristic is typical of Gaussian noise, which is also the increments process of Brownian Motion. If H is allowed to take on other values besides 0.5, we obtain fractional Gaussian noise and corresponding to that, fractional Brownian motion.
- $H = 1$: This implies a trending series; high positive or negative movements in the present are followed by high positive and negative movement in the future respectively.

The Hurst exponent arises in processes where power law decay of distribution is observed. This usually encompasses processes with Long Range Dependence (LRD) and, as mentioned in the literature, has been observed in network traffic. The fractal dimension d used to describe the amount of fractal differencing required to model a time series is related to Hurst as follows

$$d = H - 0.5. \tag{4.3}$$

In order to estimate the level of Hurst present in a signal the following estimators are employed:

- the wavelet based estimator of Abry and Veitch [82, 2, 83],
- the rescaled range statistic(R/S) based estimator [**Weron2002b**, 90], and
- the detrended fluctuation analysis(DFA) based estimator [48, 32].

The exploratory analysis will include the identification of long range dependence and the estimation of the fractal dimension of the data via the above listed techniques. These in turn might be used in an attempt to make the data stationary via fractional differencing.

4.2 Model Order Estimation

"Preliminary identification commits us to nothing except to tentatively entertaining a class of models which will later be efficiently fitted and checked. The task then, is to identify an appropriate subclass of models from the general ARIMA family." [6]. This quote refers to the situation the modeller finds themself in having performed the preceding data preparation, followed by initial model order estimation. Nonetheless model order estimation plays a vitally important role in determining to some extent the type (or subclass) of models that will be required to describe the given time series. For example, if the data required an integer differencing to bring about stationarity, then one is likely to end up with an ARIMA model. If a periodic component was identified in the data, then one is likely to end up with a SARIMA model. If fractality was detected in the time series, then a FARIMA model is likely. Having thus identified the type of model that will be required for the data being considered, the next step is to determine the *AR* and *MA* polynomials which form the core of any time series model. To this end, the orders of these polynomials are first estimated. The orders p and q of the *AR* and *MA* polynomials respectively further define the subclass of models to be entertained for fitting to the time series data.

In order to estimate the respective orders of the AR and MA polynomials, we study the the sample autocorrelation and partial-autocorrelation functions computed from the observed time series.

4.2.1 The order of the moving-average polynomial

In section 3.12 we saw that the theoretical autocorrelation function of a pure moving average process, given by equation 3.61 is finite in its extent; and more specifically, is composed of the q coefficients of the $MA(q)$ polynomial and the variance of the input random shocks, which remain constant if the process is stationary. Equation 3.61 further makes it explicit that the autocorrelation function beyond q lags is zero, where q is the order of the $MA(q)$ polynomial. We can therefore leverage this fact to estimate the order of an unknown moving average process by observing the lag at which the autocorrelation function of its time series becomes effectively zero. In practise, the sample autocorrelation of a real world time series cannot be expected to behave in the exact manner of its theoretical specification. We thus define the term "effectively zero" to mean that the autocorrelations fall below some accepted threshold. In this work, the threshold used is the standard error for estimated autocorrelations defined by [6] as

$$\hat{\sigma}[r_k] \approx \sqrt{\frac{1+2(r_1^2+r_2^2+...+r_q^2)}{N}}, \tag{4.4}$$

and is defined for lags $k > q$. N is the number samples used to compute the autocorrelations and r_k are the sample autocorrelations at lags $k = 1, 2, ...$

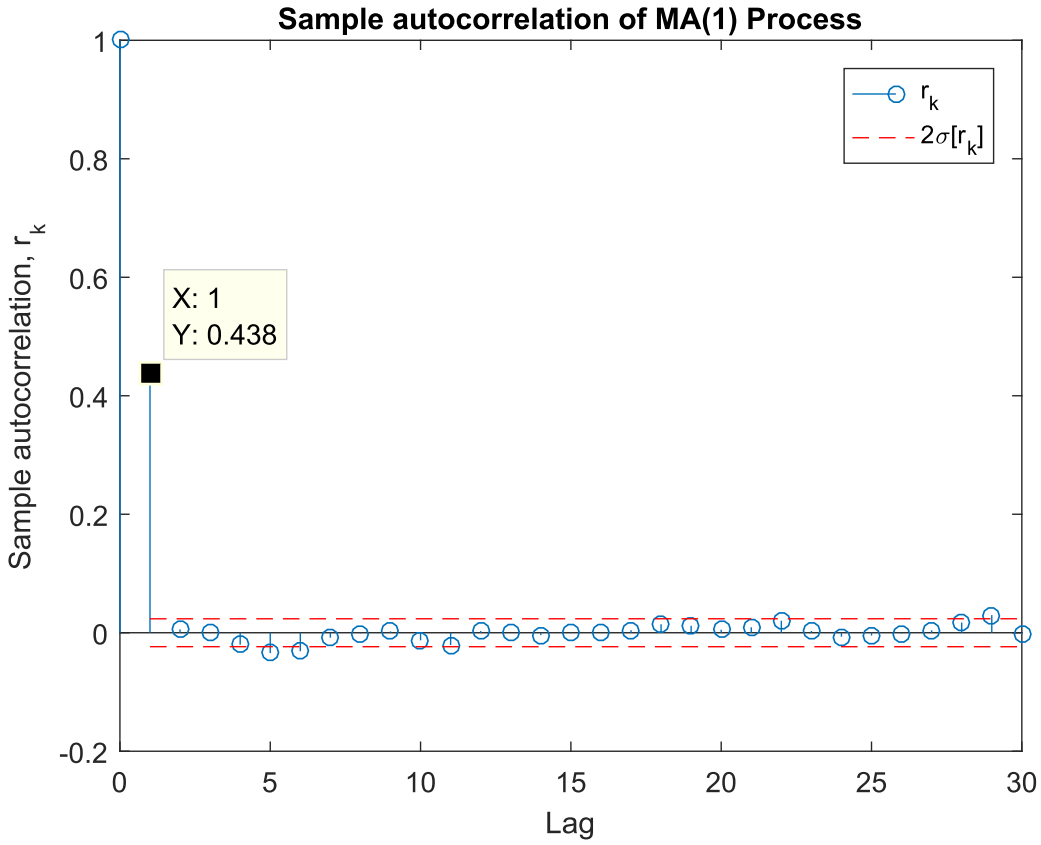

Fig. 4.3 Autocorrelation of an $MA(1)$ process, with $\theta_1 = -0.8$. The autocorrelations are computed from 10000 observations of the process. The autocorrelation cuts-off after the first lag.

For completeness, the partial-autocorrelation of a moving average process is shown below. It produces, by contrast, potentially infinitely many coefficients, as was discussed in section 3.13.

4.2.2 The order of the autoregressive polynomial

Recall from section 3.10 that the autocorrelation of an autoregressive process is composed of the p previous autocorrelations as seen in equation 3.52. Each of these previous autocorrela-

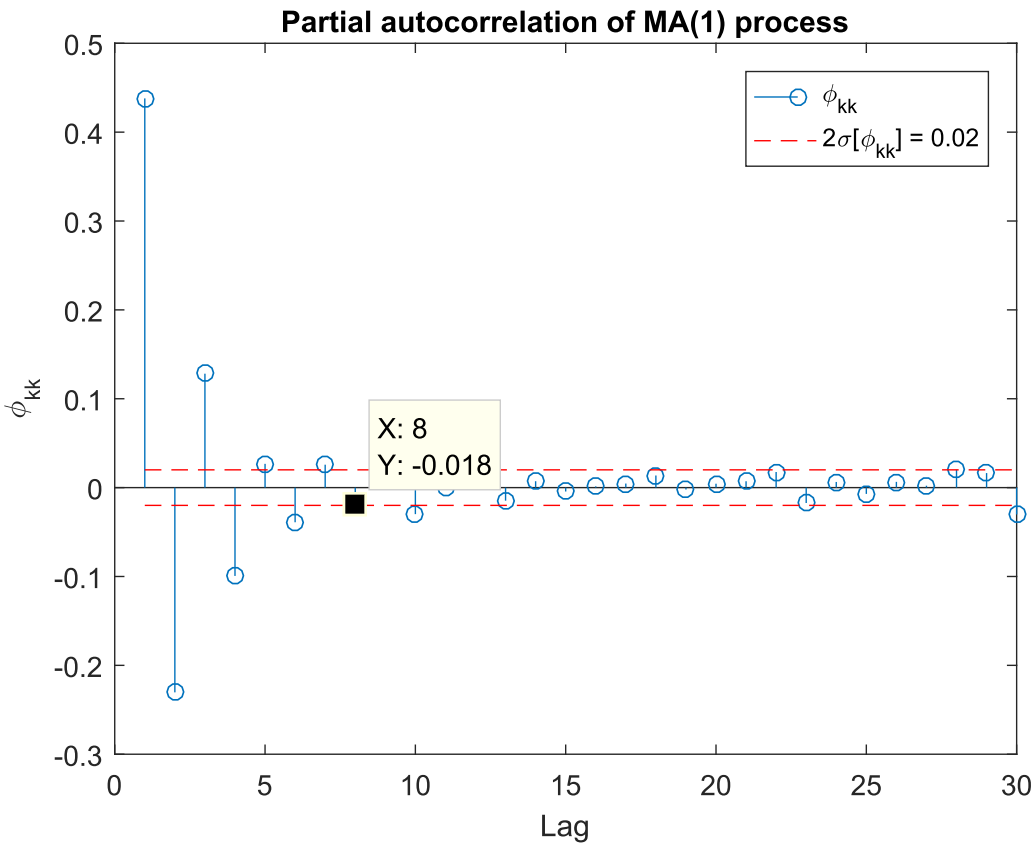

Fig. 4.4 Partial-autocorrelation of an $MA(1)$ process, with $\theta_1 = -0.8$. The partial-autocorrelations are computed from 30 lags of the sample autocorrelations. The partial-autocorrelations show an oscillatory decay towards zero.

tions are themselves dependent on the p autocorrelations preceding them. The computation of the autocorrelations of an autoregressive process proceeds in this recursive fashion resulting in autocorrelations that decay exponentially over potentially infinitely many lags.

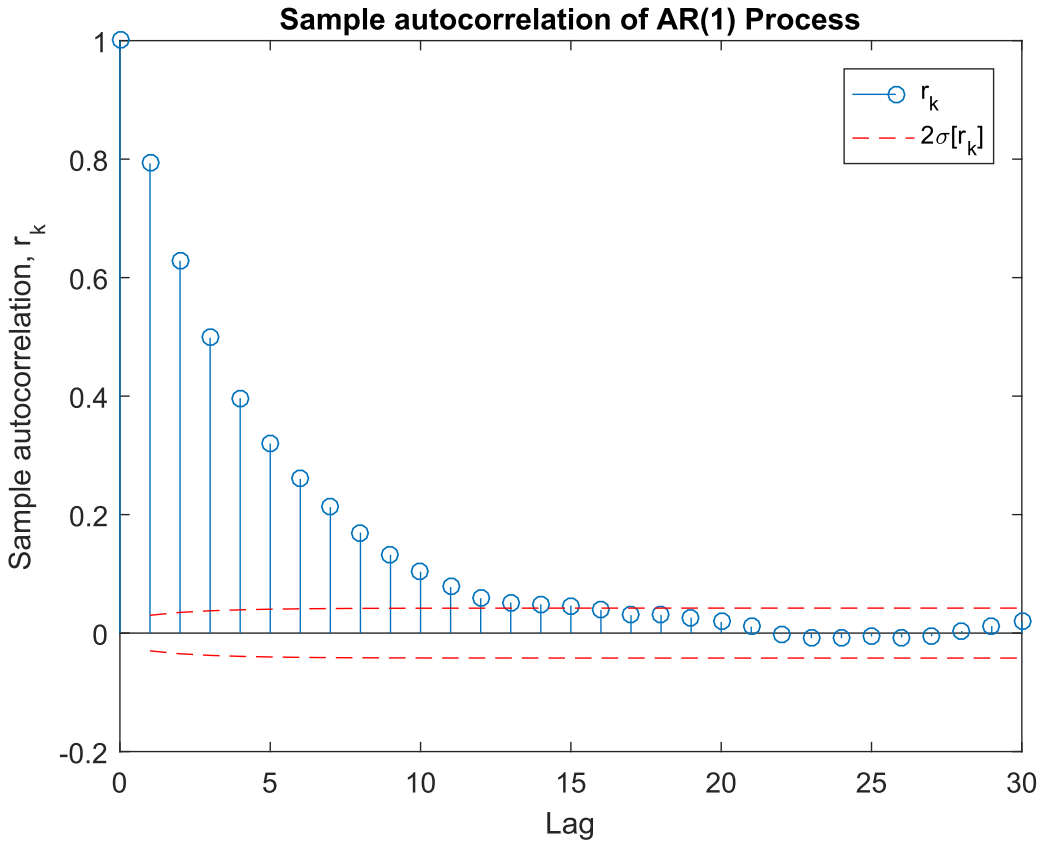

Fig. 4.5 Autocorrelation of an $AR(1)$ process, with $\phi_1 = 0.8$. The autocorrelations are computed from 10000 observations of the process. The autocorrelation coefficients show an exponential decay towards zero.

By contrast the partial-autocorrelations of an autoregressive process are obtained by solving the Yule-Walker equations for successively higher and higher orders $k = 1, ..., p$ and taking the last coefficient $\phi k, k$ at each iteration, which is the k^{th} coefficient of the k-order $AR(k)$ polynomial. Since the Yule-Walker equations for a given autoregressive process $AR(p)$ will always be composed of the same p coefficients $\{\phi_1, \phi_2, ...\phi_p\}$, they will give rise to p equations resulting in p partial-autocorrelations $\{\phi_{k,k}\}$, for $k = 1, ..., p$, which will likewise remain the same for all lags, if the process is stationary. It is this particular fact that we leverage in order to estimate the order of an unknown autoregressive process.

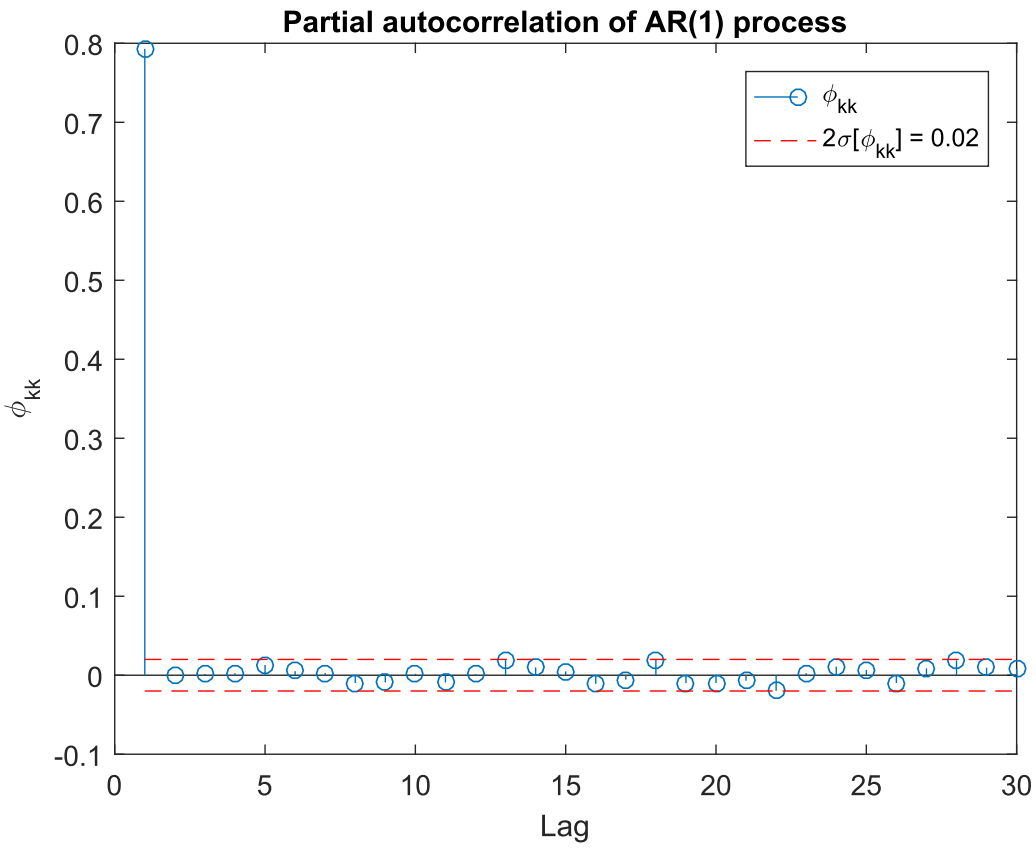

Fig. 4.6 Partial-autocorrelation of an $AR(1)$ process, with $\phi_1 = 0.8$. The partial-autocorrelations are computed from 30 lags of the sample autocorrelations. The partial autocorrelation cuts-off after the first lag.

By observing the point or lag at which the partial-autocorrelation function decays to effectively zero we can estimate the order p of that autoregressive process. In this instance the threshold used is the standard error for estimated partial-autocorrelations given by [6] as

$$\hat{\sigma}[\hat{\phi_{k,k}}] \approx \sqrt{\frac{1}{N}}, \quad \text{for } k > p \tag{4.5}$$

4.2.3 Order estimation procedure

The following procedure is adapted from the prescription in [6] and will be used to perform the tentative identification of the order parameters p and q that will be used to compose the candidate $ARMA(p,q)$ models that may describe the network traffic time series:

1. Compute the sample autocovariance function of the time series for 20 lags.
2. Compute the sample autocorrelation function of the time series for 20 lags by normalising the autocovariance from the previous step.
3. Using the sample autocorrelations obtained, repeatedly solve the Yule-Walker equations for $k = 1, ..., 20$, and extract the partial-autocorrelations.
4. Compute the standard errors for the estimated autocorrelation and the estimated partial-autocorrelation.
5. Apply the appropriate thresholding to the sample autocorrelations to determine the order q of the prospective moving average polynomial that may describe the data.
6. Apply the appropriate thresholding to the partial-autocorrelations to determine the order p of the prospective autoregressive polynomial that may describe the data.
7. From the above determinations, formulate the three potential models to be further investigated i.e. $ARMA(p,0)$, $ARMA(0,q)$, and $ARMA(p,q)$.

The fact that $MA(q)$ processes have autocorrelation functions, which cut-off after q lags, while $AR(p)$ processes have partial-autocorrelation functions, which cut-off after p lags, is exploited in the above procedure.

4.3 Model Parameter Estimation

Having transformed the observed time series of $\{z_t\}$ into a stationary time series $\{w_t\}$ via the techniques presented in section 4.1.2 and having subsequently determined the families

of $ARMA(p,q)$ models to be entertained as described in section 4.2.3, attention turns to the estimation of the parameters of each of these candidate models. Once the parameters have been estimated, each model will then be subjected to diagnostic checks, which should start to give an indication of which model is most suitable, given the observations at hand.

The parameter estimation problem is formulated by defining a vector $\bar{V}$, which will denote the population parameters of the desired $ARMA(p,q)$ process, as follows:

$$\bar{V} \equiv (c, \theta_1, \theta_2, ..., \theta_q, \phi_1, \phi_2, ..., \phi_p, \sigma^2), \tag{4.6}$$

where the stationary $ARMA(p,q)$ process is of the form

$$w_t = c + \phi_1 z_{t-1} + ... + \phi_p z_{t-p} - \theta_1 a_{t-1} + ... + \theta_q a_{t-q} + a_t, \tag{4.7}$$

and $a_t \sim i.i.d\ N(0, \sigma^2)$. Note that while the assumption of Gaussian white noise may appear to be rather restrictive, it turns out to be useful even for non-Gaussian processes [27]. Using this notation the parameter estimation problem may be stated as a maximum likelihood estimation (MLE), which can be broken down into two main steps:

1. Compute the likelihood function

$$f_{W_T, W_{T-1}, ..., W_1}(w_T, w_{T-1}, ..., w_1; \bar{V}), \tag{4.8}$$

   assuming T observations of the stationary process $\{w_t\}$ where the capital letter W denotes the random variable and the small letter w denotes the observed realisation of the random variable. The likelihood function in equation 4.8 may be interpreted as the probabilty density function (pdf), parameterised by $\bar{V}$, that gives the probability of having observed the particular sample set $(w_T, w_{T-1}, ..., w_1)$.

2. Find the values of $\bar{V}$ that maximise the likelihood function. Typically, to do this the log-likelihood function

$$\mathscr{L}(\bar{V}) = log f_{W_T, W_{T-1}, ..., W_1}(w_T, w_{T-1}, ..., w_1; \bar{V}) \tag{4.9}$$

   would be defined. The approach would then be to code a computer program that calculates $\mathscr{L}(\bar{V})$ for any given parameter vector $\bar{V}$ and sample set $(w_T, w_{T-1}, ..., w_1)$. One would then proffer different guesses for $\bar{V}$ against the observed sample set in order to try and attain the largest possible value of $\mathscr{L}(\bar{V})$ [27]. The parameter vector which brings about this maximum value in $\mathscr{L}(\bar{V})$ is then considered to be the one which is most likely to have resulted in the observed time series.

The topics of parameter estimation, numerical optimisation and maximum likelihood estimation are vast. A thourough presentation of all the considerations thereof would fill many volumes. Since these are not the primary focus of this work, and are employed here as ready-made tools, the reader is referred to [27, 6, 8, 21, 22] for more details regarding the parameter estimation of *ARMA* models.

4.4 Model Diagnostics

Having estimated the parameters of the various candidate models, an assessment of how well these models fit the data must be performed and subsequently the selection of the most appropriate model for the process being studied. In this work, the following steps are executed within the diagnostics phase of the model building for network traffic:

1. eliminate non-viable models,
2. compute the residuals produced when the model is fit to the observed time series,
3. compute the autocorrelation and partial-autocorrelation functions of the residuals,
4. apply the Ljung-Box test for lack of fit,
5. compute the cumulative periodogram of the residuals,
6. apply test for periodic nonrandomness,
7. compute Aikake information criterion (AIC), and
8. extract and rank qualifying models.

In what follows in this section, these steps will be elaborated upon providing the motivation and details of each. As recommended in [6] and many other texts, the first step in performing model diagnostics is to plot the residuals produced by the model and examine them visually along with their autocorrelations [49]. This is an indispensable step when looking to gain insight or debug the modelling process or model itself. This work is, however, particularly concerned with the automation of the modelling process, thus a great deal of emphasis will be placed on computable criteria which facillitate unassisted decision making by a computer. It will be noted that much of the diagnostic framework is based on the premise that "... autoregressive-moving average time series models, can be regarded as means of transforming the data to white noise, that is, to an uncorrelated sequence of errors[5]."

4.4.1 Elimination of nonviable models

Define nonviable models as those whose parameter estimates indicate that the model may be nonstationary or noninvertible due to the roots of either the autoregressive or the moving-average characteristic equations ($\phi(B) = 0$ or $\theta(B)$ respectively) lying inside the unit circle. In order to eliminate such models from consideration, the following steps are performed:

1. construct the polynomials $\hat{\phi}(B)$ and $\hat{\theta}(B)$ from the parameter estimates obtained from the previous step,
2. compute the roots of the polynomials $\hat{\phi}(B)$ and $\hat{\theta}(B)$,
3. count the number of roots whose absolute value is less than one, and
4. eliminate any model with a count greater 0.

In this way it is ensured that the models entertained going forward are both stationary and invertible.

4.4.2 Computation of residuals

The necessity of invertibility immediately comes into play when computing the residuals of the models. Recall that an *ARMA* model may be thought of as a filtering of an uncorrelated noise process $\{a_t\}$ by the transfer function $\psi(B) = \theta(B)/\phi(B)$ so that the observed process is

$$z_t = \frac{\theta(B)}{\phi(B)} a_t. \tag{4.10}$$

Now, having observed realisations of the process, and from them estimated the coefficients of the polynomials comprising the filter $\hat{\psi}(B)$, it must be determined whether the filter that has been estimated does indeed capture sufficiently the observed process. To make this assessment it should be that, if the inverse filter $\hat{\pi}(B) = \hat{\psi}(B)^{-1} = \hat{\phi}(B)/\hat{\theta}(B)$ is applied to the observations, the process $\{\tilde{a}_t\} \approx \{a_t\}$ is recovered, which is a close approximation to the input white noise process. The process $\{\tilde{a}_t\}$ that is recovered by undoing the effects of the *ARMA* model is referred to as the residual process; and it provides a measure of the error between the estimated *ARMA* model and the stationary time series time series $\{w_t\}$. Most goodness-of-fit tests for time series models are based on the assumption that if the $ARMA(p,q)$ model specified provides a sufficiently accurate description of the data, then the residuals expected should be $\sim i.i.d\ N(0,\sigma^2)$.

In the next chapter, the techniques presented in this chapter will be used to operate on various synthetic signals which posess characteristics that are similar to network traffic such as long-rang dependence, fractality, periodicity and trend. The efficacy in particular of detrending and fractal differencing in dealing with these characteristics is evaluated by means of experimentation.

Chapter 5

Experimentation

In this section the numerical simulations and experiments performed in this investigation are recorded. The experimentation performed involves synthetically generated data. In the section to follow firstly, a description of the synthetic data, used to evaluate the techniques presented in the previous chapter, is provided. Secondly, the experimental setup and procedure is stated.

5.1 Experimental Data

The experimental data used in this investigation was specifically chosen to try and uncover useful approaches of dealing with specific instances or contexts of network traffic. This is a somewhat different approach to most of the literature, where the traffic signal is usually in the form of highly aggregated backbone traffic. During the experimentation with synthetic data, the focus was to identify very specific known signals with the following properties:

1. produces a known or mathematically verifiable result when a transform of interest is applied to it. (example: a Fourier transform of a sinusoid produces a delta function), and

2. mimics a property or behaviour which network traffic signals tend to possess. (example: a periodic pulse train tends to behave in a similar fashion to packet bursts which emanate from automated network applications like dropbox-synching)

Having generated the idealised signals with the above criteria in mind, various corrupting sources were then added to gauge the effect of these noise sources on the techniques under scrutiny. The following is a list of the synthetic signals used in the experiments:

1. Set 1: Synthetically generated noise / sources of nonstationarity. We make use of synthetically generated signals which we will call our canonical noise signals. The signals considered here are meant to be representative of typical sources of corruption or non-stationarity that tend to affect the rest of the analysis. The presence of these noise components may even render our modelling techniques further down the line ineffective, if not adequately dealt with.

   Linear Trend

   Sinusoid

   White noise

   Fractional Gaussian noise

2. Set 2: Synthetically generated data signals. These are signals that embody certain characteristics of network traffic. These however are idealised, therefore it should become evident to us whether or not our tools are well suited enough to deal with the presence of such characteristics.

   Periodic pulse trains

   AR(1)

   MA(1)

   ARMA(1,1)

   fractional Brownian motion

   FARIMA(0,d,0)

   FARIMA(1,d,0)

   FARIMA(1,d,1)

5.2 Experimental Procedure

5.2.1 Data Exploration Experiments

The first set of experiments carried out is to determine and/or verify the capabilities of our initial data exploration techniques. In particular, we make use of the following tests for checking the stationarity of the input data:

1. Augmented Dickey Fuller test for unit roots

2. Welch's ttest for equal means
3. Levene test for equality of variance (using 10% percent trimmed means)

While the decisions made with respect to the modelling task are based on the above stationarity checks, the estimated skewness and kurtosis of signal as it progresses in time are also considered. This is expected to provide some insight as to what the appropriate distribution for the data might be.

During the exploratory phase, an understanding of the frequency content of the data under analysis is also sought. The following are the techniques used in this work for the said purpose:

1. periodogram and cumulative periodogram,
2. power spectrum (via FFT), and
3. wavelet spectrum.

The periodogram, and subsequently the cumulative periodogram of the data are used to quickly identify the presence of any periodicity in the data. Used in conjunction with the Kolmogorov-Smirnov upper and lower bounds, the cumulative periodogram can also be used to verify the Gaussianity of the data. The power spectrum generated via the FFT is also computed as a redundant sanity check. Finally, a wavelet analysis of the data is performed in order identify the frequency bands and locations of various activity in the signal.

The next stage in the exploratory analysis is the identification of long range dependence and the estimation of the fractal dimension of the data. In order to accomplish these tasks the following techniques are employed:

1. log-log plot of the periodogram, and
2. wavelet-based Hurst estimator of Abry and Veitch.

The wavelet-based Hurst estimator of Abry and Veitch is used to identify the presence of long range dependence and the differencing parameter $d\mathbb{R}$, which might be used to make the data stationary. Additionally, the Hurst and fractal dimension d are also estimated via the regression of the log-log plot of the periodogram.

The exploratory analysis continues by plotting the ACF and PACF of the input data. By studying the decay of these functions over a sufficient number of lags, we may gain some insight w.r.t. the following concerns:

1. stationarity,
2. which model is suitable AR, MA, or ARMA,
3. what is the order of the initially proposed model,
4. periodicity (repeating patterns in the ACF/PACF).

Finally a rough estimate of the probality density function of the data, in the form of a histogram, is generated. This serves to give an indication of how Gaussian the generated sample data is. This will further help to make a diagnosis as to whether or not non-Gaussianity has an effect on the efficacy of the employed techniques.

5.3 Experiments and Results

The following subsections document the actual experiments performed in this study and the results thereof.

5.3.1 Objective comparison of Hurst estimators

A particularly important question to answer during the exploration phase of model building, especially in uncertain circumstances, is whether or not there is any fractality that needs to be dealt with when we start treating a given time series. If the estimation of Hurst can be trusted, then this can be used to further make the decision as to whether or not ARMA parameter estimation can proceed, or whether (fractional) differencing of the time series is first required.

The objective of the first experiment conducted was to observe the efficacy of a number of competing Hurst estimators when estimating Hurst at different levels without any interference (i.e. no AR or MA components) in the signals. To achieve this, samples were generated from the fractional Gaussian noise process(fGn) according to the method in [43] and the fractional ARIMA (FARIMA(0,d,0)) process according to the method in [77]. The Hurst(H) levels used to generate the signals ranged from 0 to 1 in steps of 0.02, where $d = H - 0.5$. Each sample path generated consisted of $N = 360000$ observations. These sample paths were in turn fed as input to each of the estimators under review. The estimators evaluated were the following:

1. wavelet-based Hurst estimator of Abry and Veitch(AV) [2, 82, 83],
2. the adjusted rescaled range (R/S) analysis [90, 89], and
3. detrended fluctuation analysis (DFA) [48].

Each estimator was applied to the same input sample path, and the error in estimating the Hurst parameter used to generate the sample path was computed. For each value of Hurst, 8 sample paths were generated for each of the FARIMA(0,d,0) and fGn processes. The average Hurst estimation error at each H is tabulated in 5.1 and 5.2 for the fGn and FARIMA(0,d,0) processes respectively.

Table 5.1 Hurst estimation error for fGn process

Actual Hurst	Estim. Err. (AV)	Estim. Err. (R/S)	Estim. Err. (DFA)
0	0.50012	0.5015	0.51163
0.02	0.027875	0.0615	0.011875
0.04	0.01425	0.059	0.012
0.06	0.014125	0.05425	0.010875
0.08	0.009875	0.049375	0.011625
0.1	0.009375	0.044125	0.007625
0.12	0.012	0.041625	0.009375
0.14	0.007625	0.037125	0.01
0.16	0.009125	0.035375	0.01075
0.18	0.00725	0.029875	0.007875
0.2	0.00925	0.026	0.0095
0.22	0.006875	0.0245	0.008375
0.24	0.00625	0.018	0.006
0.26	0.0075	0.018625	0.0135
0.28	0.005375	0.017625	0.010875
0.3	0.0045	0.016875	0.008875
0.32	0.00675	0.011	0.005125
0.34	0.00625	0.011875	0.010875
0.36	0.005125	0.008	0.007875
0.38	0.00525	0.00525	0.005
0.4	0.004375	0.005125	0.007
0.42	0.0035	0.005	0.010875
0.44	0.004	0.00475	0.010625

0.46	0.003375	0.00575	0.006125
0.48	0.002	0.004	0.009
0.5	0.000625	0.004625	0.012875
0.52	0.002	0.00475	0.012375
0.54	0.0035	0.004375	0.014
0.56	0.003375	0.007125	0.010625
0.58	0.002375	0.00575	0.0065
0.6	0.003625	0.00825	0.01
0.62	0.0025	0.004625	0.01225
0.64	0.002625	0.00625	0.01375
0.66	0.003875	0.004375	0.01125
0.68	0.0035	0.005875	0.010875
0.7	0.00325	0.00825	0.01675
0.72	0.002625	0.014	0.0275
0.74	0.0025	0.01075	0.02575
0.76	0.005375	0.00675	0.018375
0.78	0.00175	0.00975	0.02525
0.8	0.004625	0.015875	0.03475
0.82	0.003625	0.0215	0.063375
0.84	0.004375	0.020625	0.069375
0.86	0.0035	0.021	0.07
0.88	0.003625	0.032125	0.077625
0.9	0.004625	0.03125	0.0755
0.92	0.004375	0.037875	0.095625
0.94	0.0035	0.042125	0.116
0.96	0.00425	0.054875	0.14525
0.98	0.005125	0.06275	0.20163
1	0	NaN	2.6084

Table 5.2 Hurst estimation error for FARIMA(0,d,0) process

Actual Hurst	Estim. Err. (AV)	Estim. Err. (R/S)	Estim. Err. (DFA)
0	0.00525	0.13338	0.10275
0.02	0.002875	0.12313	0.095125
0.04	0.00375	0.112	0.08675
0.06	0.004125	0.102	0.077375

0.08	0.003	0.091625	0.0675
0.1	0.005875	0.083375	0.064625
0.12	0.0035	0.07425	0.057125
0.14	0.003125	0.067125	0.0525
0.16	0.004125	0.061125	0.04675
0.18	0.00525	0.051375	0.036625
0.2	0.0025	0.046875	0.03525
0.22	0.00575	0.038375	0.029
0.24	0.0055	0.033625	0.025625
0.26	0.004125	0.031625	0.02825
0.28	0.00525	0.021125	0.018625
0.3	0.00425	0.022	0.02475
0.32	0.004625	0.019875	0.0205
0.34	0.004	0.014375	0.014875
0.36	0.003	0.013875	0.01575
0.38	0.00375	0.00975	0.01475
0.4	0.002875	0.0115	0.017625
0.42	0.002125	0.008375	0.013375
0.44	0.001625	0.00575	0.00725
0.46	0.002125	0.005625	0.012
0.48	0.001	0.00575	0.012375
0.5	0.001	0.004875	0.01225
0.52	0.0015	0.00675	0.015375
0.54	0.001375	0.00775	0.00975
0.56	0.002625	0.006125	0.0095
0.58	0.002375	0.0025	0.010625
0.6	0.00375	0.00825	0.007375
0.62	0.003375	0.00525	0.012625
0.64	0.003125	0.012625	0.0135
0.66	0.003875	0.012125	0.015125
0.68	0.00325	0.0095	0.008125
0.7	0.004875	0.013625	0.01125
0.72	0.004375	0.00725	0.01575
0.74	0.002375	0.01325	0.016125
0.76	0.003875	0.022625	0.008875
0.78	0.00625	0.015125	0.01425

0.8	0.006875	0.01125	0.01475
0.82	0.009	0.02225	0.014
0.84	0.005875	0.019875	0.022125
0.86	0.00425	0.021625	0.020875
0.88	0.002875	0.034	0.023875
0.9	0.004875	0.02325	0.024875
0.92	0.003125	0.041625	0.02975
0.94	0.003125	0.053625	0.04325
0.96	0.004875	0.04875	0.047125
0.98	0.004125	0.061375	0.044375
1	0.003375	0.0645	0.06175

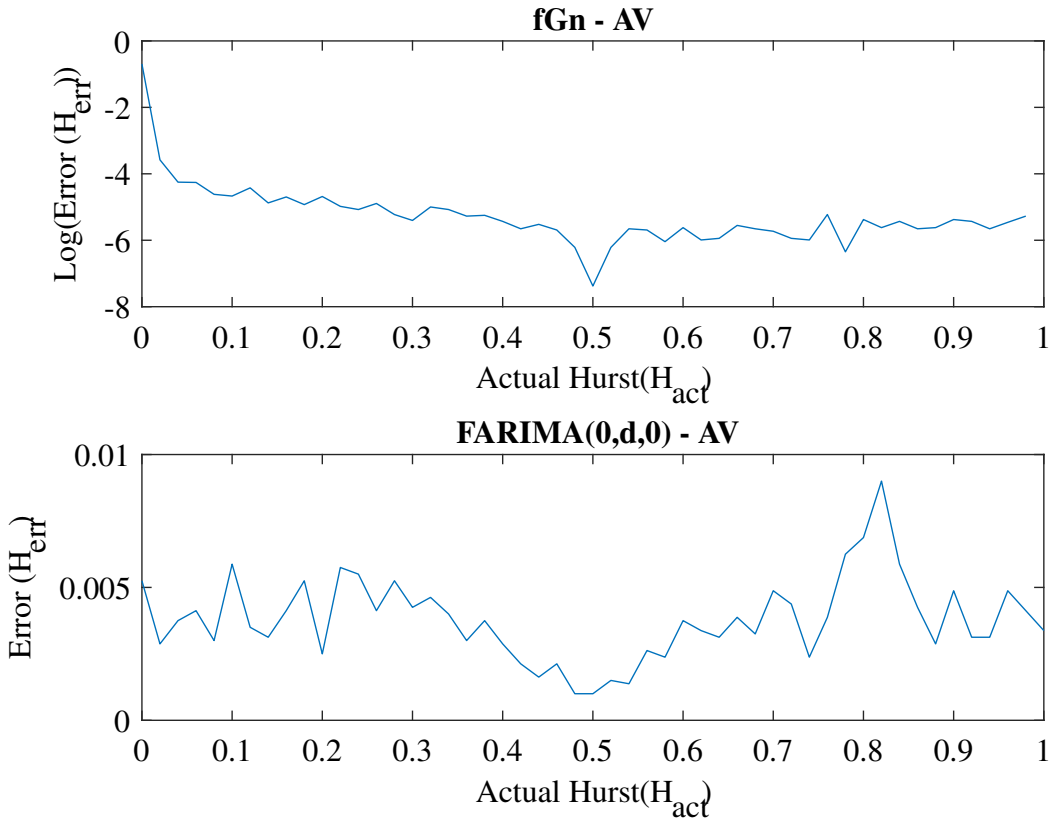

Fig. 5.1 Hurst estimation error as produced by the Abry-Veitch wavelet (AV) based estimator. The Hurst was estimated from synthetically generated samples from the fGn and FARIMA(0,d,0) processes.

Runtime observations from Hurst estimation tests:

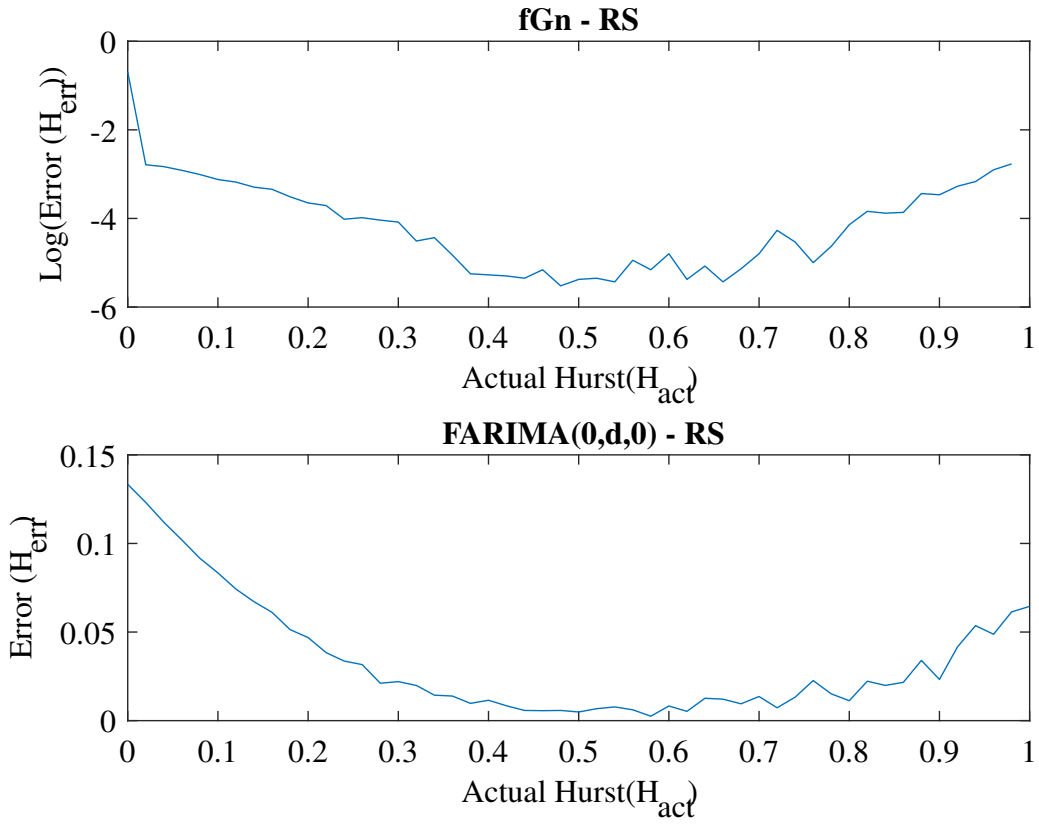

Fig. 5.2 Hurst estimation error as produced by the Rescaled-range statistic (R/S) based estimator. The Hurst was estimated from synthetically generated samples from the fGn and FARIMA(0,d,0) processes.

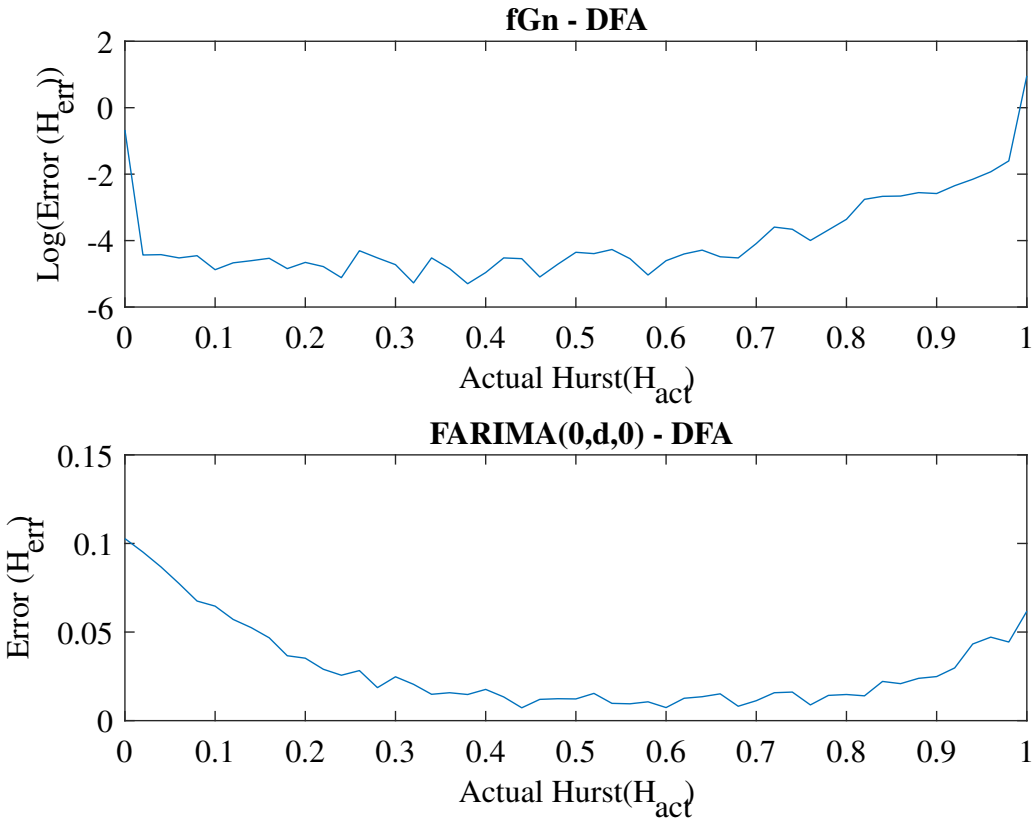

Fig. 5.3 Hurst estimation error as produced by the Detrended Fluctuation Analysis (DFA) based estimator. The Hurst was estimated from synthetically generated samples from the fGn and FARIMA(0,d,0) processes.

1. The R/S Hurst estimator has the longest runtime of the three estimators, typically by up to 2 orders of magnitude compared to AV and DFA estimators.

2. The AV estimator has the smallest average error compared to the other estimators over the majority of H values.

3. For the fractal Gaussian noise process, it appears that the Hurst is difficult to estimate when close to zero. This is seen for all three estimators. The rate at which the error declines as H increases appears to be much quicker for the AV estimator than for the others.

4. For the fractal Gaussian noise process, the error in estimating H appears to rise again as H approaches 1. This is however not the case for the AV estimator, which appears fairly constant.

5. Similar traits to the fGn process, in terms of H estimation error, are observed for the FARIMA(0,d,0) process. Once again the Av estimator emerges as the most consistent in terms of error, which is maintained around a value of 0.005.

5.3.2 Effect of AR and MA components on the estimation of Hurst

In line with trying to perform a thorough exploration of a given time series, the next experiment aims to observe the effect of autoregressive(AR) and moving-average(MA) components on the estimation of Hurst. The premise here is that if the Hurst parameter can be reasonably well estimated, even in the presence of AR and MA components, then a reasonable means of identifying and possibly dealing with fractality in a given time series is available, and in particular network traffic measurements observed over time.

For this experiment the same FARIMA process generator from [77] was used to generate numerous permutations of FARIMA(1,d,1) processes, where ϕ, and θ of the AR and MA components respectively were allowed to vary in range $(-1,1)$ in steps of 0.1. The Hurst parameter H was varied in the range $[0,1]$. Each sample path generated consisted of $N = 360000$ observations. The average estimation error for each value of ϕ, calculated over the entire range of H values, is tabulated in 5.3. The left most column is the coefficient ϕ of the AR(1) polynomial. For each value ϕ the average estimation error produced by each of the three estimators is recorded.

Table 5.3 Hurst estimation error as a function of the AR component ϕ

ϕ	Estim. Err.(AV)	Estim. Err.(R/S)	Estim. Err.(DFA)
-1	0.82015	0.47215	0.52554
-0.9	0.12506	0.35835	0.3905
-0.8	0.055377	0.11067	0.13074
-0.7	0.056208	0.097385	0.11359
-0.6	0.055212	0.091022	0.10458
-0.5	0.055095	0.086152	0.096684
-0.4	0.055506	0.082212	0.091706
-0.3	0.055987	0.080022	0.088346
-0.2	0.055991	0.077805	0.085532
-0.1	0.056225	0.077234	0.084459
1.00E-06	0.056303	0.07629	0.08229
0.1	0.056801	0.075043	0.081126
0.2	0.055939	0.073978	0.082636
0.3	0.057784	0.074446	0.080835
0.4	0.05829	0.074818	0.081563
0.5	0.057316	0.077329	0.086684
0.6	0.057823	0.078078	0.088589
0.7	0.059987	0.081212	0.095879
0.8	0.059498	0.086312	0.10823
0.9	0.064165	0.096268	0.12905
1	0.069645	0.12047	0.1688

The average estimation error for each value of θ, calculated over the entire range of H values, is tabulated in 5.4. The left most column is the coefficient θ of the MA(1) polynomial. For each value θ the average estimation error produced by each of the three estimators is recorded.

Table 5.4 Hurst estimation error as a function of the MA component θ

θ	Estim. Err.(AV)	Estim. Err.(R/S)	Estim. Err.(DFA)
-1	0.9158	0.41076	0.4397
-0.9	0.073255	0.20179	0.21495
-0.8	0.065996	0.15114	0.17105
-0.7	0.060857	0.12711	0.14526

-0.6	0.061673	0.11335	0.13086
-0.5	0.058641	0.10562	0.12239
-0.4	0.057578	0.10019	0.11703
-0.3	0.058045	0.096139	0.1128
-0.2	0.057266	0.092355	0.10467
-0.1	0.055628	0.089716	0.10569
1.00E-06	0.055065	0.090294	0.10445
0.1	0.055221	0.088446	0.10372
0.2	0.054472	0.088532	0.10317
0.3	0.054006	0.088617	0.10288
0.4	0.053775	0.089426	0.10279
0.5	0.053097	0.087385	0.10219
0.6	0.052569	0.090747	0.10661
0.7	0.051905	0.087333	0.10647
0.8	0.050842	0.085978	0.10366
0.9	0.050009	0.084736	0.10323
1	0.048656	0.077587	0.093775

Runtime observations from Hurst estimation in the presence of AR and MA components experiments:

- The AV estimator of Hurst has the lowest and most consistent average error among the estimators evaluated.
- There appears to be little to no correlation between the AR and MA components and the Hurst estimation error of the AV estimator, with the exception of AR and MA components close to -1.
- The R/S and DFA estimators display an increase in estimation error as the AR component tends towards either $+/-1$.
- The R/S and DFA estimators display an increase in estimation error as the MA component tends towards -1, but the error seems to decrease even further as the MA compnent approaches $+1$.
- Overall, it seems that the Hurst estimation error falls within reasonable bounds, with the exception of when ϕ or θ is close to -1. Thus the fractal dimension, computed as $d = H - 0.5$, can be used for the purpose of removing some of the fractality that may be present in a given time series, even in the presence of AR and MA components.

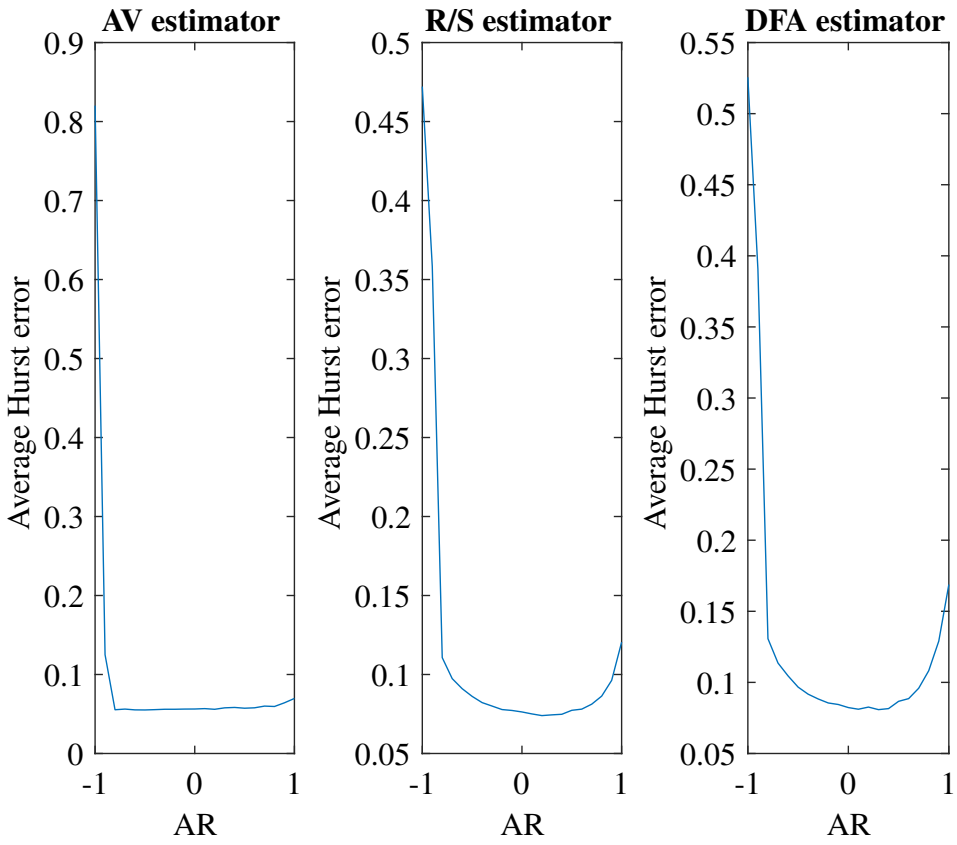

Fig. 5.4 Plot of the average Hurst estimation error as a function of the AR component ϕ of a FARIMA(1,d,1) process

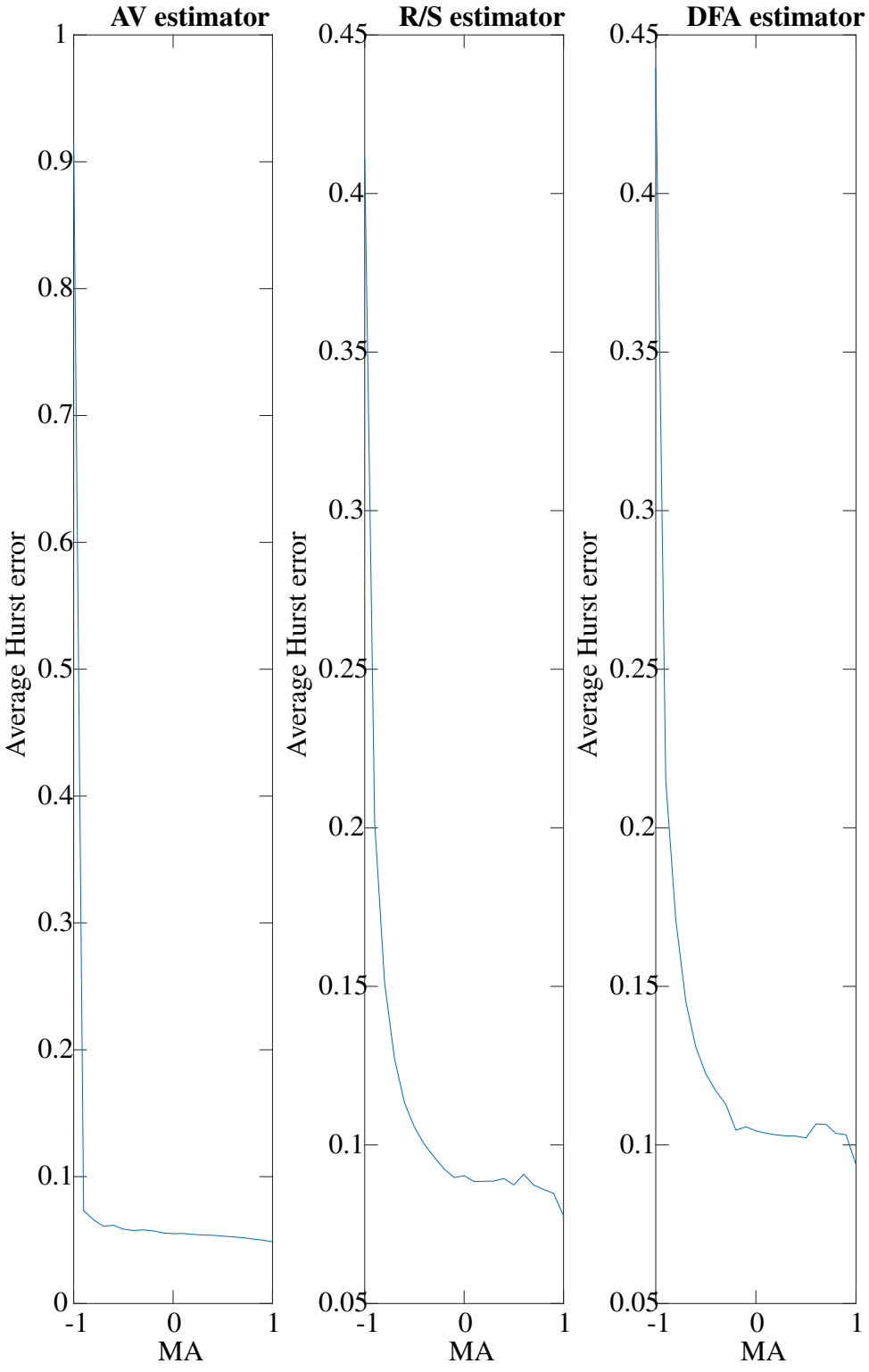

Fig. 5.5 Plot of the average Hurst estimation error as a function of the MA component θ of a FARIMA(1,d,1) process

5.3.3 Effect of Hurst on AR and MA component estimation

The objective of this experiment is, firstly, to determine if the presence of Hurst in a given time series has any effect on the estimation of the AR and MA components of that process. Secondly, if the time series is determined to be fractal, can the subsequent error in the estimation of ϕ an θ be reduced via fractional differencing?

The experiment is carried out by simulating FARIMA(1,d,1) sample paths as above. For each sample path corresponding to a particular combination of ϕ, θ and H as described above, ϕ and θ are estimated using the ARMA estimation procedure found in Matlab. A fractional differencing according to the work of [38] is then applied to the sample path and the estimation of the AR and MA components repeated. The results obtain for estimating the AR and MA components before and after differencing are tabulated are tabulated in 5.5 and 5.6 respectively.

Table 5.5 Error in estimating the AR and MA components of a FARIMA(1,d,1) process as a function of Hurst before any differencing.

H	ϕ_{error}	θ_{error}
0	0.35894	0.47216
0.1	0.33162	0.40388
0.2	0.27762	0.32361
0.3	0.21371	0.23959
0.4	0.13042	0.14425
0.5	0.034571	0.048154
0.6	0.15959	0.16412
0.7	0.27612	0.2711
0.8	0.39153	0.36591
0.9	0.52324	0.45095
1	0.67136	0.49216

Table 5.6 Error in estimating the AR and MA components of a FARIMA(1,d,1) process as a function of Hurst after fractional differencing.

H	ϕ_{error}	θ_{error}
0	0.030972	0.089283

0.1	0.031041	0.055814
0.2	0.029084	0.047016
0.3	0.033095	0.047052
0.4	0.031011	0.042451
0.5	0.03598	0.046556
0.6	0.03534	0.051762
0.7	0.032345	0.056977
0.8	0.032288	0.061946
0.9	0.026814	0.057794
1	0.035669	0.066512

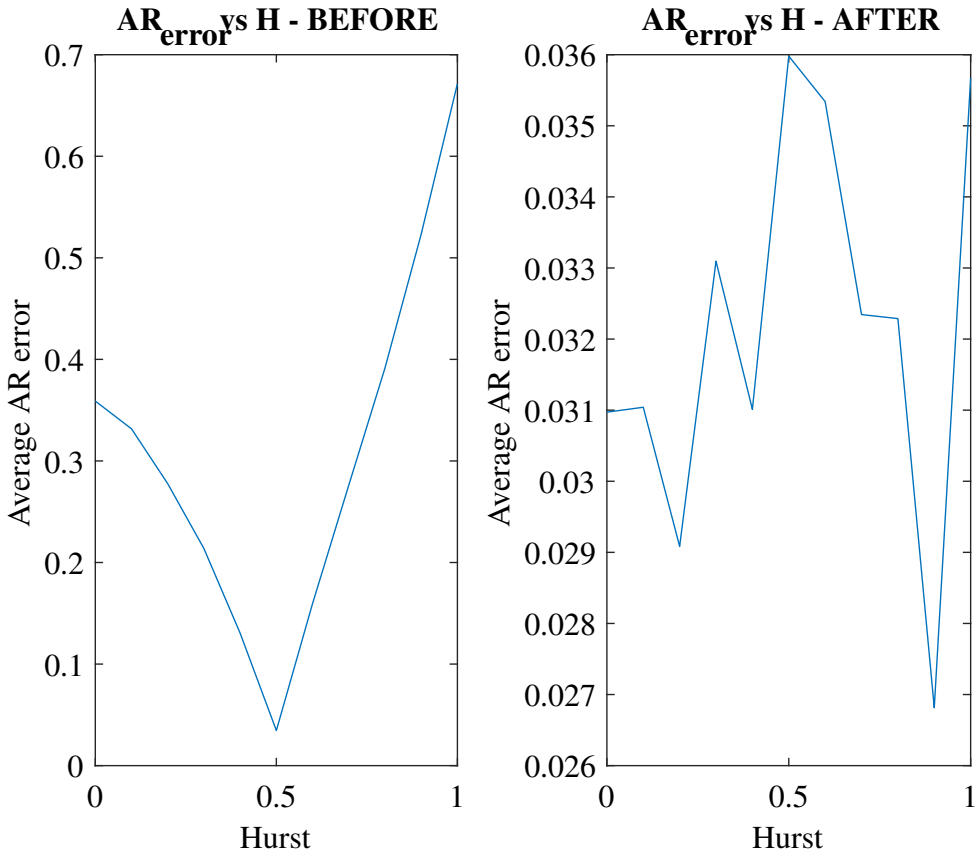

Fig. 5.6 Plot of the average estimation error for the AR component, ϕ, of a FARIMA(1,d,1) as a function of Hurst, H. The left plot shows the error with the Hurst component present. The right plot shows the error after applying a fractional differencing, using $d = H - 0.5$, to the time series.

Runtime observations from ARMA parameter estimation experiment:

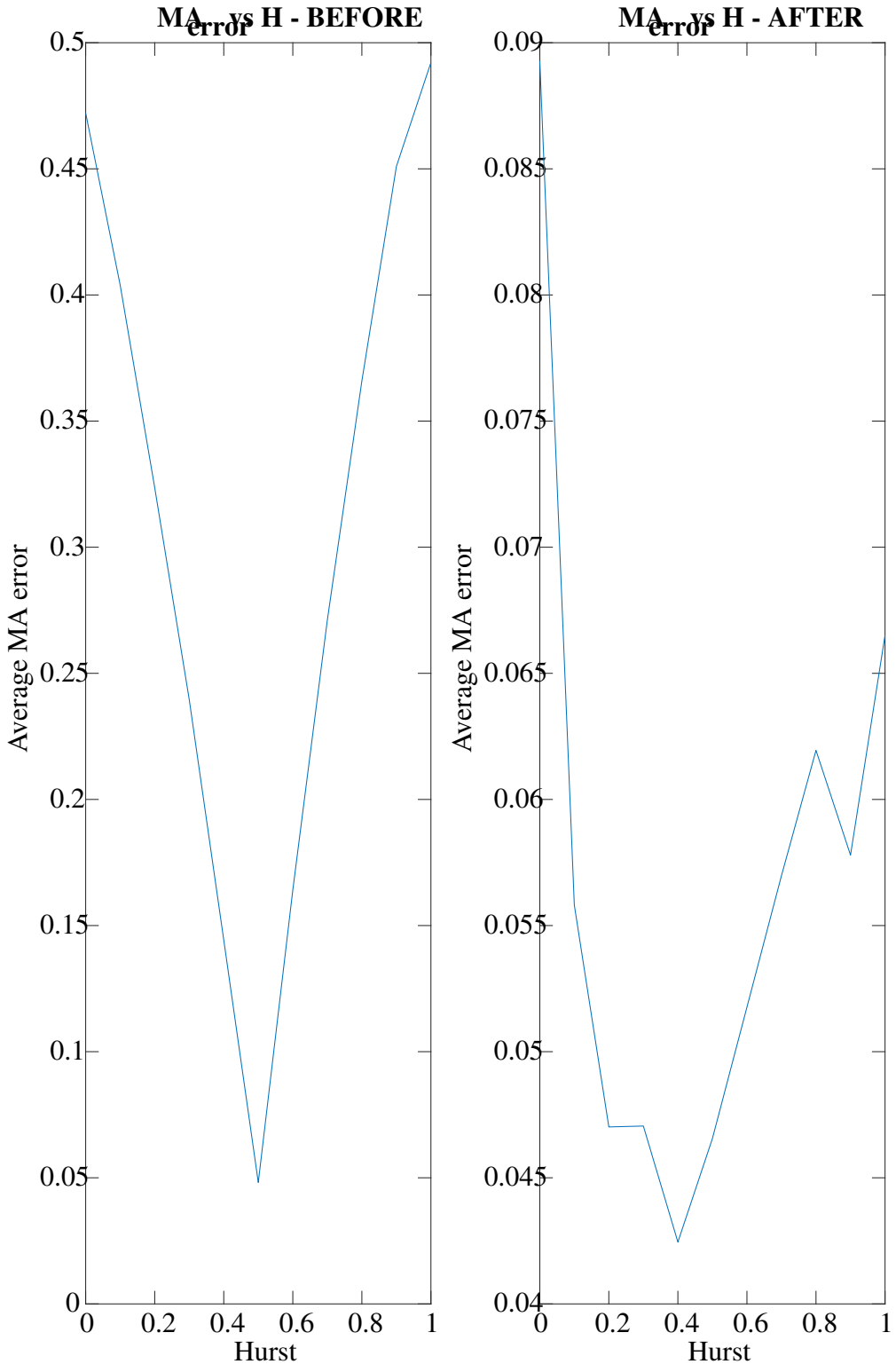

Fig. 5.7 Plot of the average estimation error for the MA component, θ, of a FARIMA(1,d,1) as a function of Hurst, H. The left plot shows the error with the Hurst component present. The right plot shows the error after applying a fractional differencing, using $d = H - 0.5$, to the time series.

- There appears to be a significant amount of correlation between Hurst and the estimation error of the AR and MA components in a time series. This might be explained in part by the fact that processes with H tending towards 1 (d tending towards 0.5) exhibit behaviour such that high positive or negative movements in the present tend to be followed by high positive or negative movements in the future respectively. This behaviour is clearly visible when one looks at the ACF of processes such as fGn or FARIMA(0,d,0) where H is set close to 1. Since the estimation of the AR and MA components tends to be based on the ACF and PACF, it is likely that the aforementioned behaviour may influence the AR and MA component estimators.

- Comparing tables 5.5 and 5.6, the estimation error of the AR and MA components ϕ_{error} and θ_{error} respectively, is significantly reduced after fractional differencing is applied to the input sample paths.

- The increase in estimation error of the AR and MA components that is observed as H approaches 0 or 1 appears to be more constant after the differencing is applied as depicted in figures 5.6 and 5.7. This suggests that the correlation between the presence of Hurst and AR or MA component estimation error has been significantly neutralised.

- The effect of the fractional differencing also appears to be more pronounced for the AR component estimation than the MA component estimation.

A look at the frequency domain characteristics of the AR(1), MA(1), and FARIMA(0,d,0) processes sheds some light on the interaction between theses process even more clearly.

The FARIMA(0,d,0) and fGn processes with H close to one have a very pronounced peak in their frequency spectrum around $f = 0$ as seen in figure 5.8. This is confirmed again by the cumulative periodogram in the same figure, which quickly accumulates around the zero cycles area and then plateaus over the rest of the frequencies indicating that not much energy is added beyond the 0.05 cycles frequency. This process is also characterised by an ACF which decays very slowly as shown in figure 5.9. This is in fact very similar to what is observed for the AR(1) process, where ϕ is close to 1. It can be seen in 5.11 that the frequency content of the AR(1) process also tends to be concentrated around $f = 0$, which may mean that a substantial proportion of the signal content is affected by the presence of Hurst. In general, the activity or content of these signals is distributed over long timescales, a fact which is further confirmed by a plot of the wavelet scale-translation space for both the FARIMA(0,0.5,0) and AR(1) processes depicted in 5.10 and 5.13 respectively.

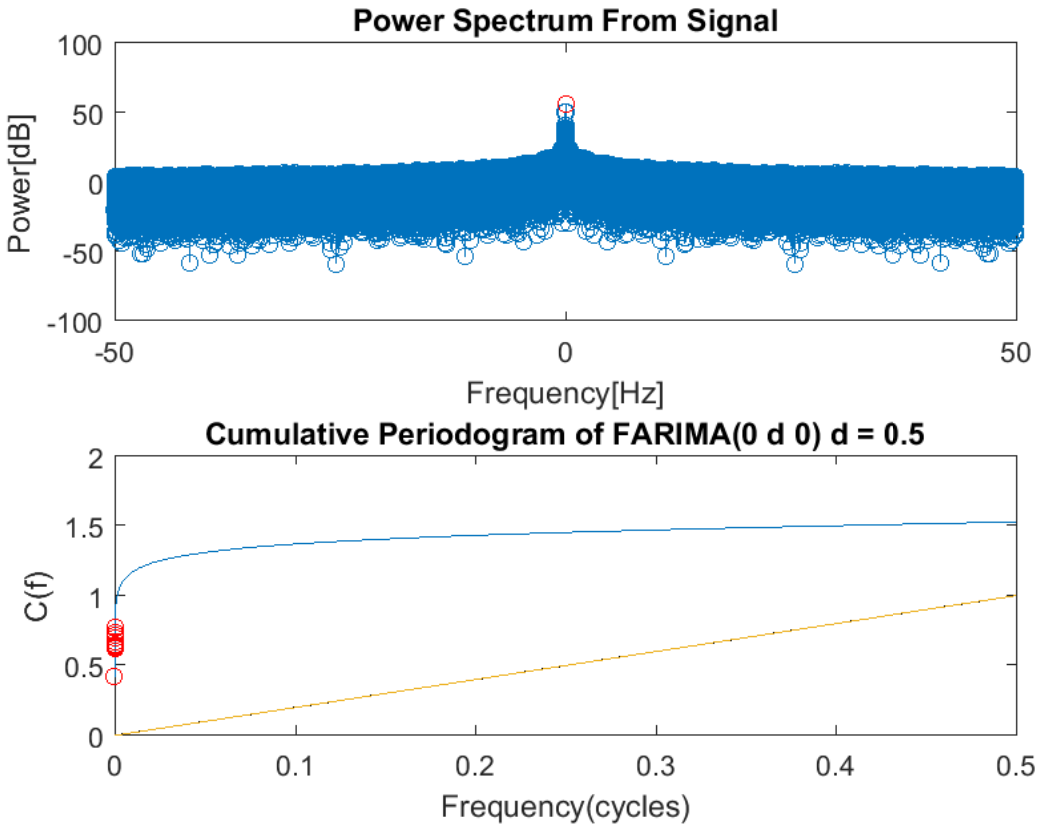

Fig. 5.8 Above is the plot of the frequency spectrum of a FARIMA(0,d,0) sample path when $H = 1$. Below is the corresponding cumulative periodogram.

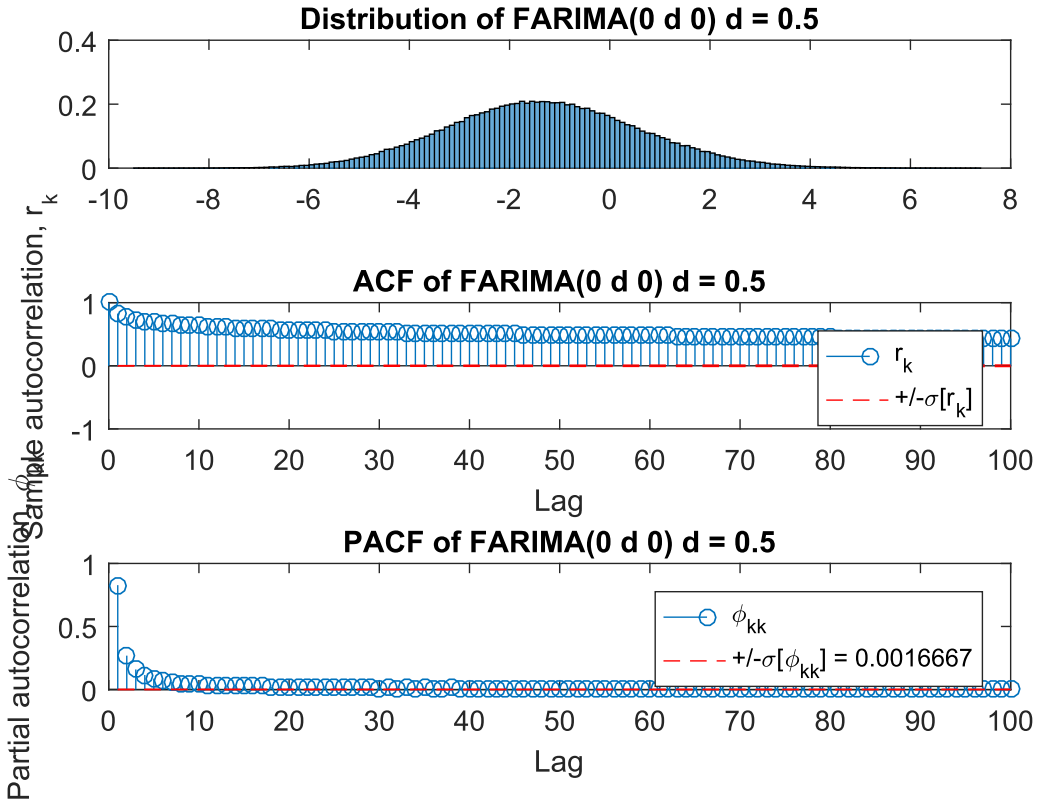

Fig. 5.9 The top plot shows the distribution of the samples using a histogram. The middle plot show the autocorrelation function of the samples generated from a FARIMA(0,d,0) process when $H = 1$. The bottom plot is the corresponding partial autocorrelation function.

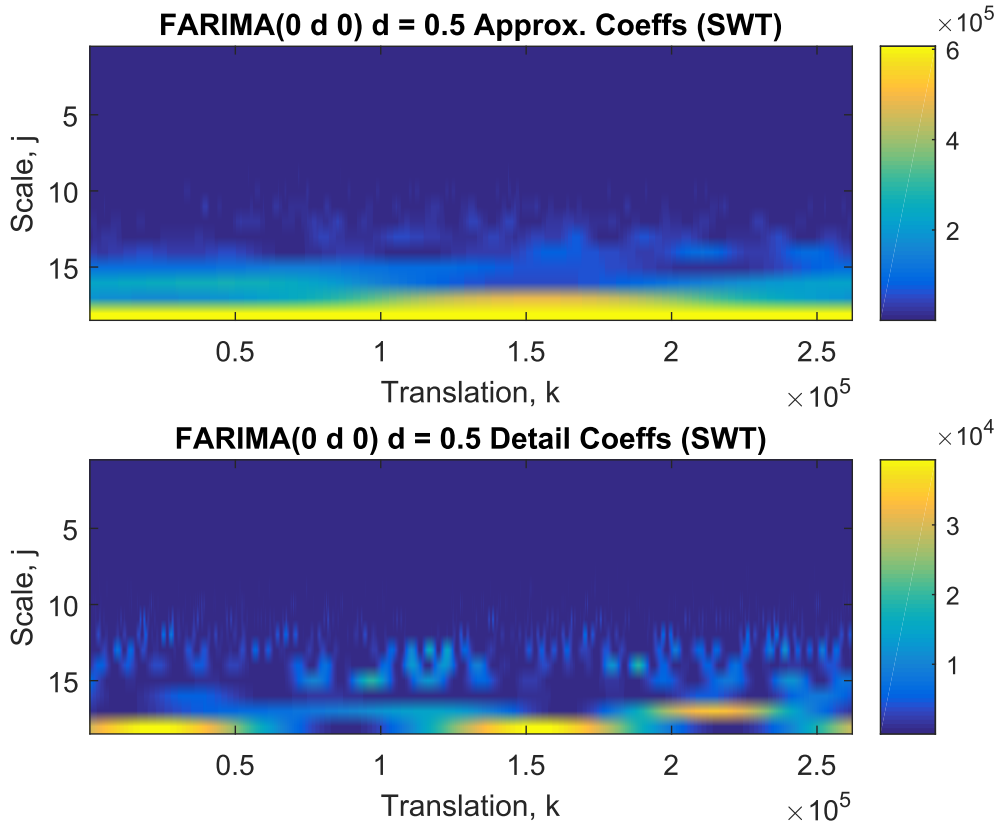

Fig. 5.10 The top plot shows the approximation coefficients from a stationary wavelet transform of a FARIMA(0,d,0) process when $H = 1$. The bottom plot shows the corresponding detail coefficients.

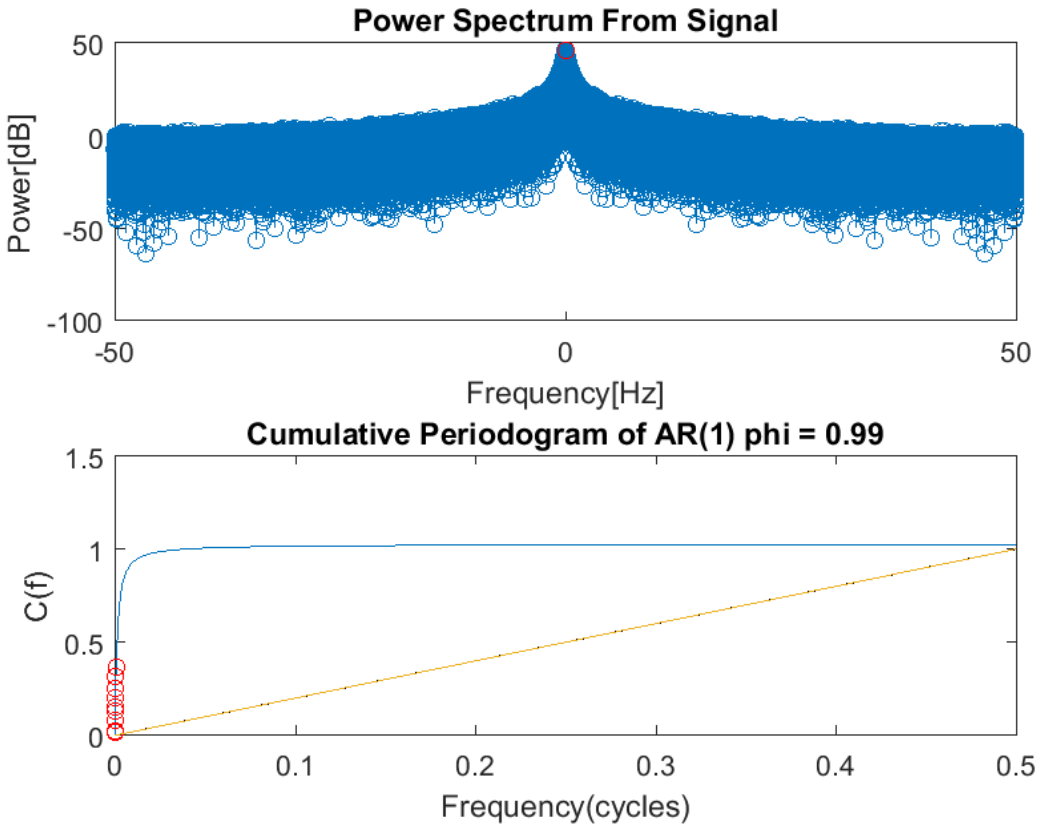

Fig. 5.11 Above is the plot of the frequency spectrum of an AR(1) process when ϕ is close to 1. Below is the corresponding cumulative periodogram.

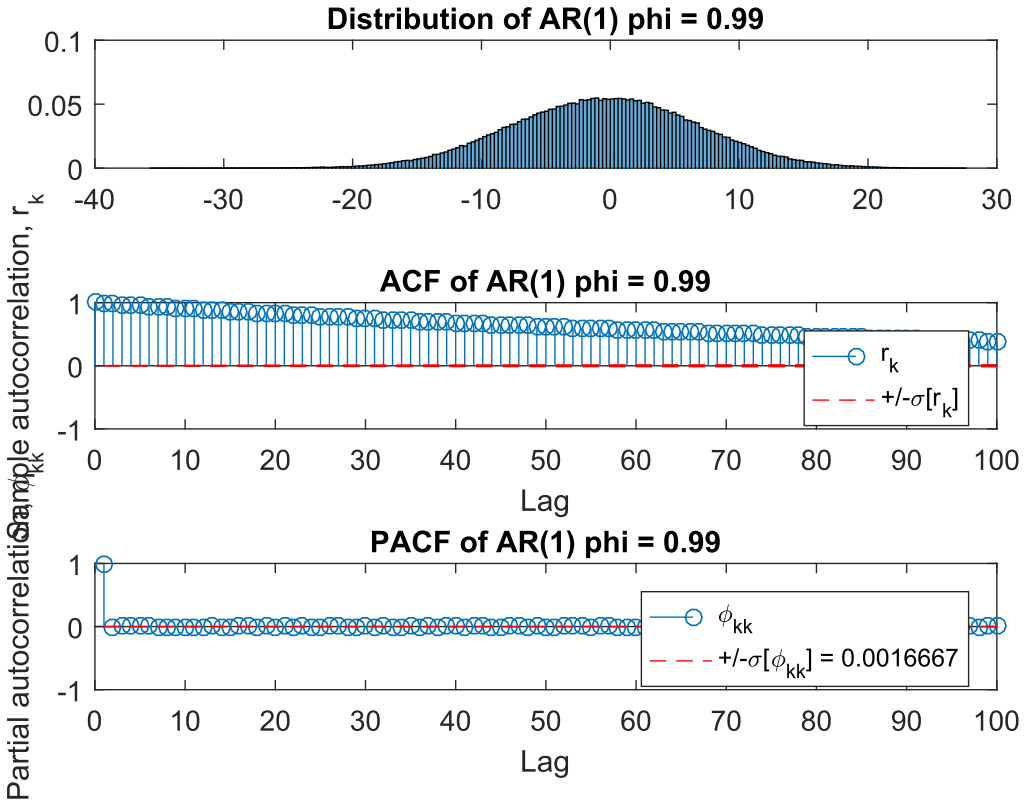

Fig. 5.12 The top plot shows the distribution of the samples using a histogram. The middle plot show the autocorrelation function of the samples generated from an AR(1) process when ϕ is close to 1. The bottom plot is the corresponding partial autocorrelation function.

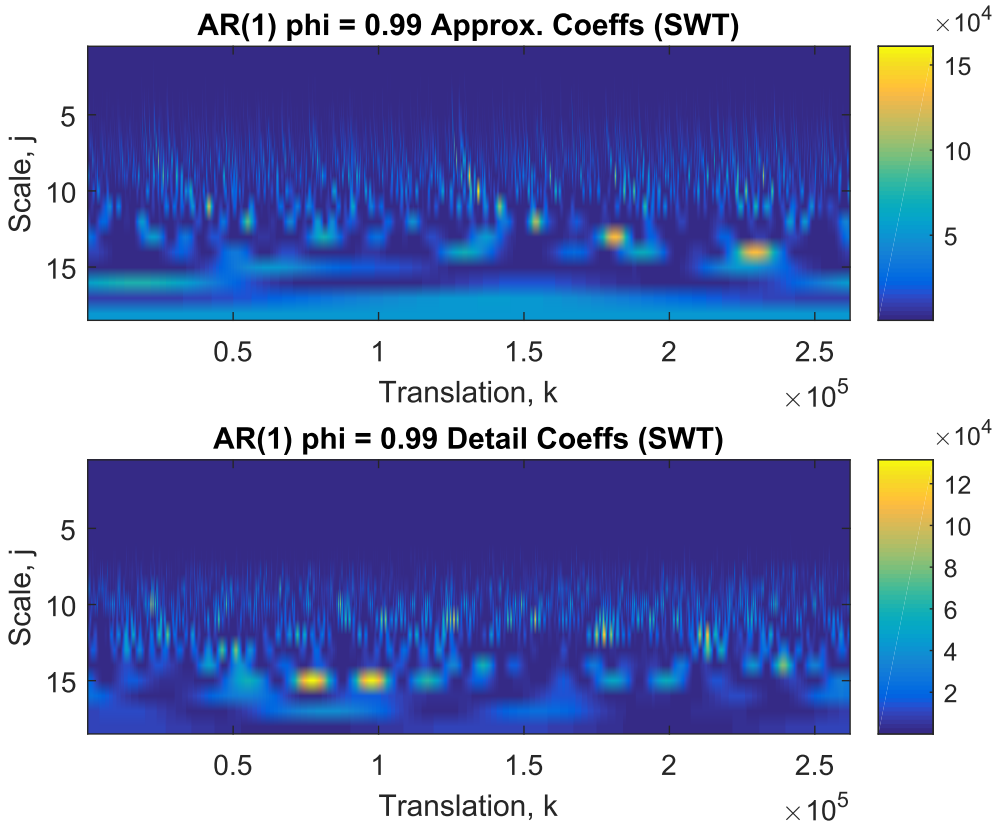

Fig. 5.13 The top plot shows the approximation coefficients from a stationary wavelet transform of an AR(1) process when ϕ is close to 1. The bottom plot shows the corresponding detail coefficients.

On the other hand, because the frequency content of the MA(1) process is less concentrated around $f = 0$(see figure 5.14, the effect of the Hurst is somewhat muted, albeit still present i.e. the band of frequencies which characterise a MA(1) process is much wider than the band of frequencies typically affected by the presence of Hurst. Consequently, the signal content is also spread out over a wider range of timescales in the wavelet scale-translation space as seen in 5.16.

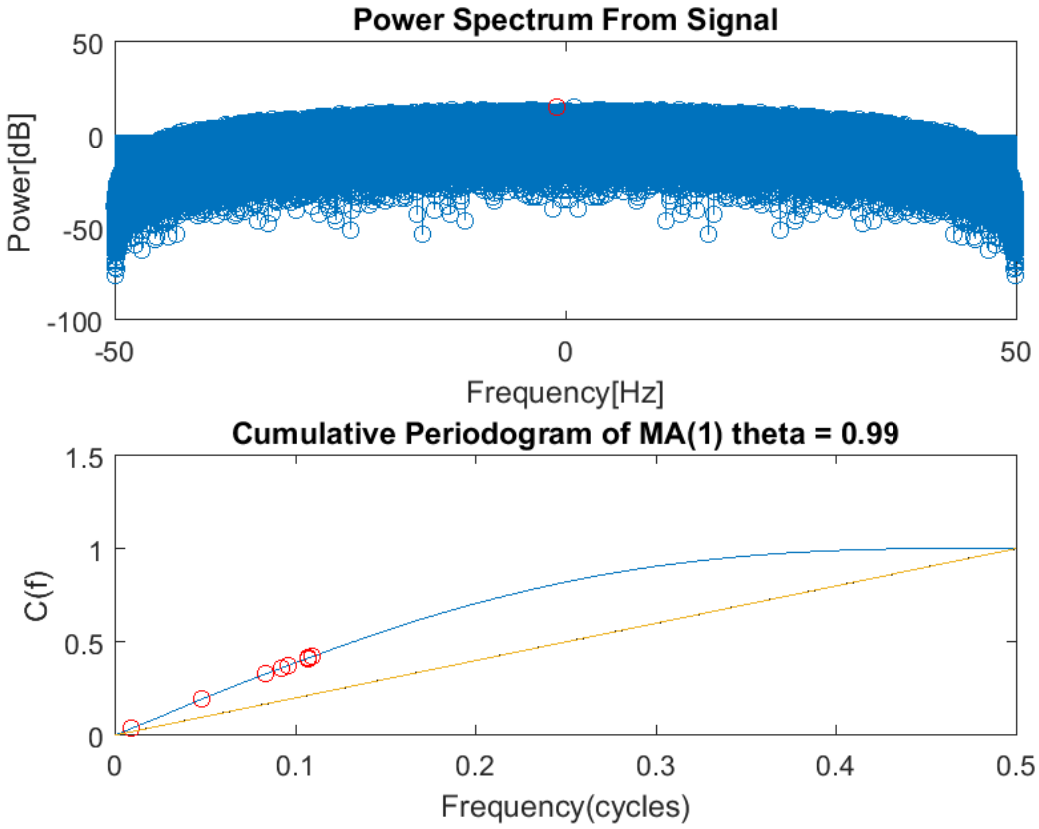

Fig. 5.14 Above is the plot of the frequency spectrum of an MA(1) process when θ is close to 1. Below is the corresponding cumulative periodogram.

Looking at the frequency domain further explains the decline in the estimation error of the AR and MA components, seen in figures 5.6 and 5.7, as H approaches 0.5. Since the frequency spectrum of processes such as fGn or FARIMA(0,d,0) tends to be flat when H is close to 0.5, the presence of such Hurst provides little to no interference in the analysis of a signal. Furthermore, the correlation structure of a FARIMA(0,0,0) process, depicted in 5.17, resembles that of white noise.

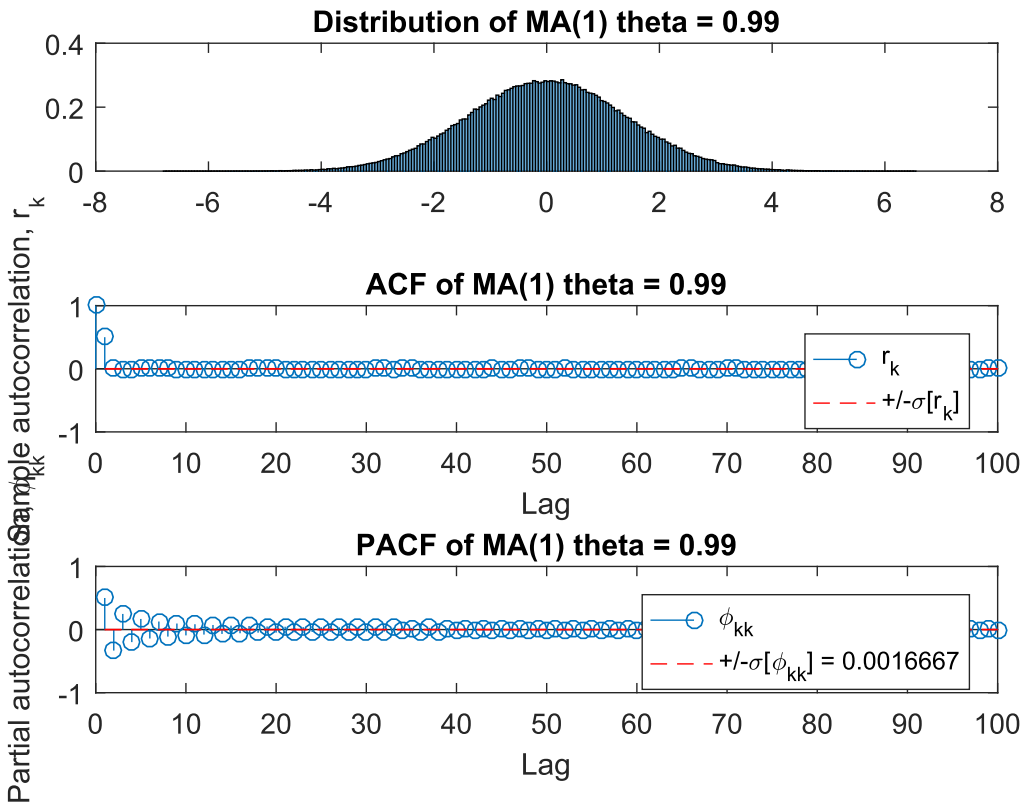

Fig. 5.15 The top plot shows the distribution of the samples using a histogram. The middle plot show the autocorrelation function of the samples generated from an MA(1) process when θ is close to 1. The bottom plot is the corresponding partial autocorrelation function.

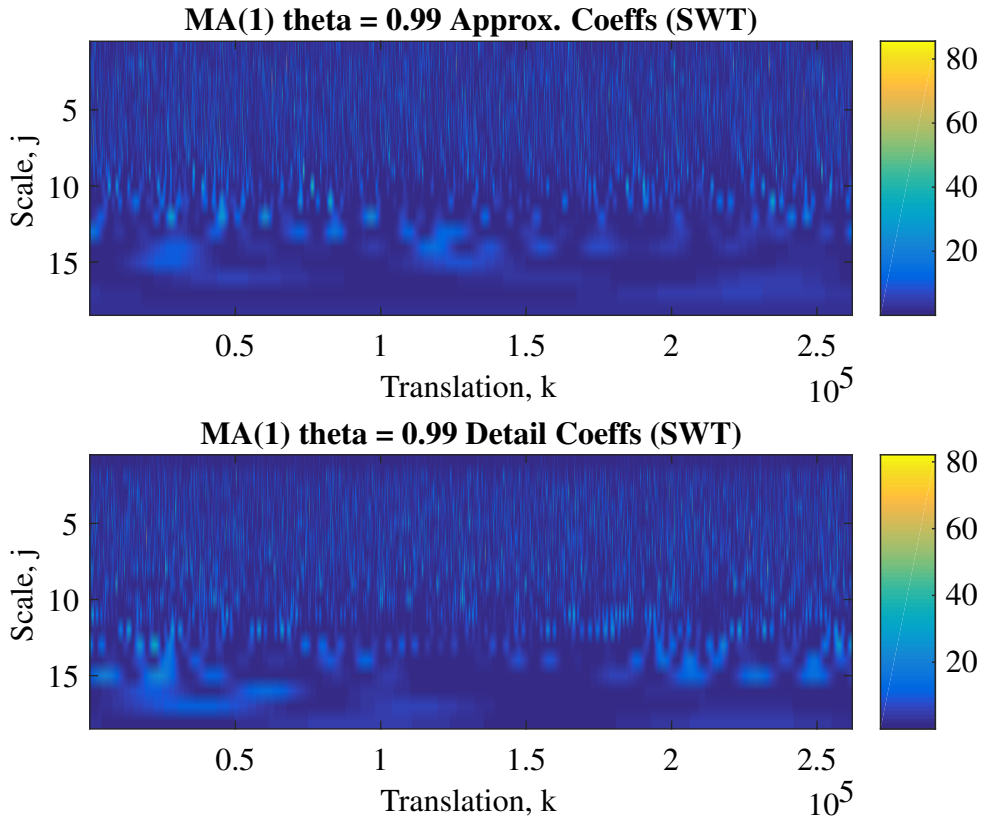

Fig. 5.16 The top plot shows the approximation coefficients from a stationary wavelet transform of a MA(1) process when θ is close to 1. The bottom plot shows the corresponding detail coefficients.

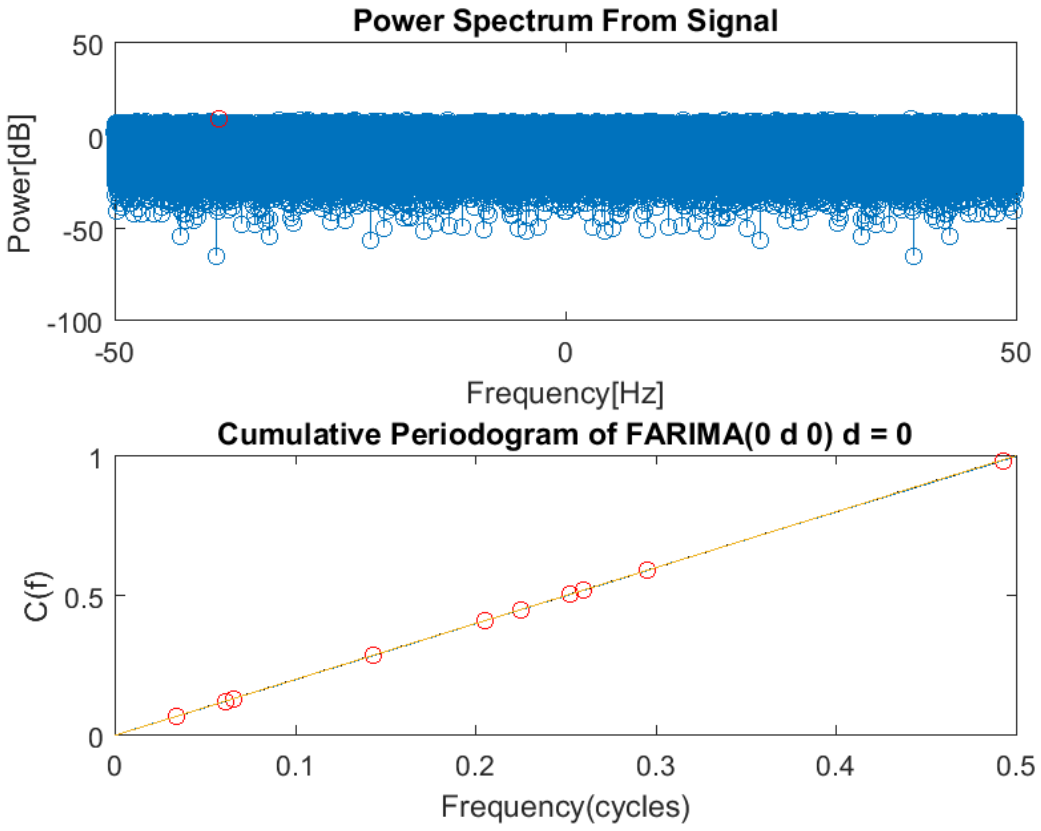

Fig. 5.17 Above is the plot of the frequency spectrum of a FARIMA(0,d,0) sample path when $H = 0.5$. Below is the corresponding cumulative periodogram.

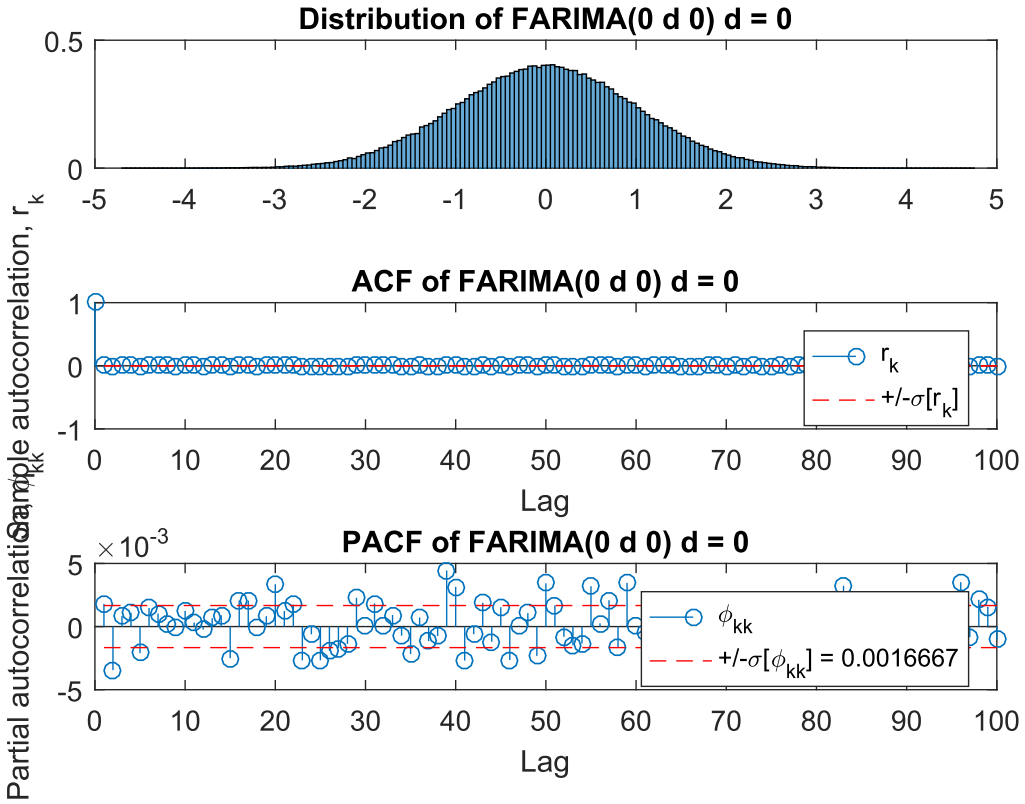

Fig. 5.18 The top plot shows the distribution of the samples using a histogram. The middle plot show the autocorrelation function of the samples generated from a FARIMA(0,d,0) process when $H = 0.5$. The bottom plot is the corresponding partial autocorrelation function.

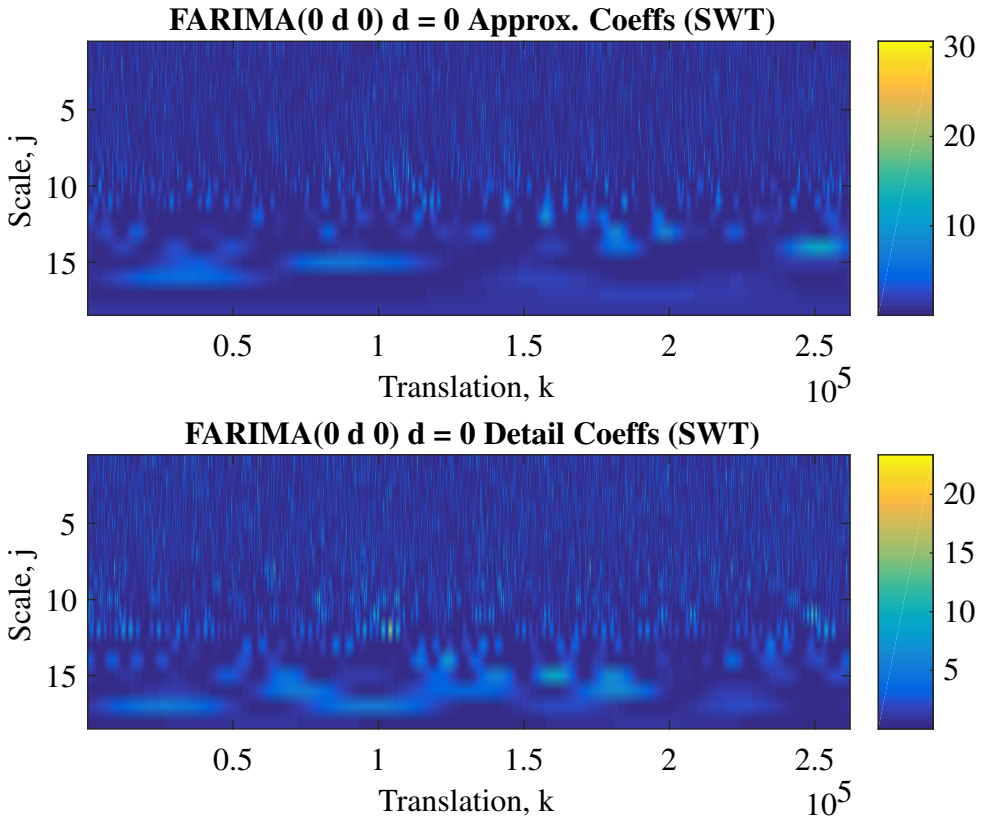

Fig. 5.19 The top plot shows the approximation coefficients from a stationary wavelet transform a FARIMA(0,d,0) process when $H = 0.5$. The bottom plot shows the corresponding detail coefficients.

Finally, when the AR or MA component is negative, the frequency content of the generated sample path is similar to that of fGn and FARIMA(0,d,0) where the Hurst now approaches 0 or equivalently d approaches -0.5.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Network traffic, from the literature studied, has been found to be fractal, displaying high variability at certain scales while also exhibiting correlations over long time scales. Time series analysis specifically tries to describe a process as a linear combination of its previous realisations and random innovations. This has the benefit of capturing explicitly the covariation between sample observations. For network traffic modelling, this means that we are no longer beholden to the assumption of identical and independently distributed samples. Both the short and long range dependence of network traffic data can be captured using time series models, potentially avoiding the need to deal with the intricacies of multifractality. Time series analysis also provides the tools to deal with some of the difficulties encountered in network traffic data such as non-stationarity, fractality and seasonality.

From the experimentation conducted it was found that the estimation of Hurst (or fractal dimension) can be successfully performed for FARIMA processes with single AR and MA components. The Abry-Veitch wavelet-based estimator emerged as the most accurate and robust of the estimators considered. Furthermore, having automatically estimated the fractal dimension, one is able to automatically apply a fractional differencing to a given time series in order to make it ammenable to parameter estimation which is to say making the time series stationary. The benefit of this approach is exhibited by the decrease in estimation error of the AR and MA components of FARIMA(1,d,1) processes after fractal differencing as shown in figures 5.6 and 5.7. Using a combination of time series, frequency and wavelet analysis, the level of fractality present in the signal was determined. Furthermore the determination of the major seasonal components in the signal and removing these effects through various forms of differencing was also studied and found to be feasible, however this work is incomplete

and remains to be studied more closely.

A survey of the literature in a wide array of fields has found that there are numerous techniques which hold promise for addressing the particular needs of traffic modelling, especially in the IoT domain, where the more regular fashion of the traffic generated by automated processes may make it more ammenable to being captured in rather simple time series models. Given that the idiosyncracies of network traffic, viz. fractality, seasonality and non-stationarity, have been demonstrated to be sufficiently neutralisable, albeit under synthetic and somewhat idealised circumstances, these is sufficient evidence that time series modelling does indeed hold promise for capturing useful models of netowrk traffic. Consequently an effort to consolidate and improve these techniques may provide powerful tools for the management and securing IoT networks and computer networks in general.

Over and above the development of these tools, in the course of conducting this study, a number of interesting avenues for future investigation have been identified. These are listed in the final section.

6.2 Future Work

1. Use Maximum Autocorrelation Factors (MAF) to linearly combine the time series generated by monitoring each feature/component of network traffic. This is essentially similar to PCA, except instead of maximising the variance that is captured, the autocorrelation is maximised. See [28].

2. Handle non-linearly/irregularly sampled data or data with missing samples using non-decimated wavelet transform (NDFT). See [72].

3. Perform statistical analysis of wavelet coefficients up to 4th moment (kurtosis) to see if we can get a consistent description/characterisation of network traffic or a means by which to identify network anomalies. See [62]. Look at the distributions of network traffic on idle state, steady peak state, and anomaly state w.r.t. Kurtosis at each wavelet decomposition level. This might enable us to identify various anomalies according to the way in which they modulate the distribution(or its moments) of wavelet coefficients compared to some reference distribution(representing the normal state). i.e. Estimate of the four Pearson moments including (A) mean, (B) standard deviation, (C) skewness and (D) kurtosis of the raw signal and the calculated Db10

detail and approximation coefficient sequences at each decomposition level for both the reference signal (noise,idle) and the target signal(normal and anomalous activity).

4. Look at modelling network traffic as a non-linear dynamic/chaotic system [41].

5. Model IOT related protocols [40].

6. Adaptive selection of Hurst estimator - Perhaps upon noticing any one of these corrupting influences, we can choose the appropriate estimator which is immune to the type of corruption identified in the signal [39].

7. Apply the Box-Cox power transformation to data which violates assumptions such as additivity, gaussianity, and homoscedasticity [68, 7]. "Reality is that almost all analyses (even nonparametric tests) benefit from improved normality of variables, particularly where substantial non-normality is present. While many are familiar with select traditional transformations (e.g., square root, log, inverse) for improving normality, the Box-Cox transformation [7] represents a family of power transformations that incorporates and extends the traditional options to help researchers easily find the optimal normalizing transformation for each variable. As such, Box-Cox represents a potential best practice where normalizing data or equalizing variance is desired " [56].

8. Look at encoding models into agent based systems and studying the emergent phenomena thereof in order to gain insight to the potential characteristics of IoT networks and their behaviour.

References

[1] A.Anand and B.Patel. “An Overview on Intrusion Detection Systems and Types of Attacks It Can Detect Considering different Protocols”. In: *International Journal of Advanced Research in Computer Science and Software Engineering* 2.8 (2012), pp. 987–992, 94–98.

[2] Patrice Abry and Darryl Veitch. “Wavelet analysis of long-range-dependent traffic”. In: *IEEE transactions on information theory* 44.1 (1998), pp. 2–15.

[3] Michèle Basseville, Igor V Nikiforov, et al. *Detection of abrupt changes: theory and application*. Vol. 104. Prentice Hall Englewood Cliffs, 1993.

[4] Hakem Beitollahi and Geert Deconinck. “Analyzing well-known countermeasures against distributed denial of service attacks”. In: *Computer Communications* 35.11 (2012), pp. 1312–1332.

[5] G. E. P. Box and David A. Pierce. “Distribution of Residual Autocorrelations in Autoregressive-Integrated Moving Average Time Series Models”. In: *Journal of the American Statistical Association* 65.332 (1970), pp. 1509–1526. ISSN: 01621459. URL: http://www.jstor.org/stable/2284333.

[6] George E. P. Box, Gwilym M. Jenkins, and Gregory C. Reinsel. *Time Series Analysis*. Wiley-Blackwell, June 2008. DOI: 10.1002/9781118619193.

[7] George EP Box and David R Cox. “An analysis of transformations”. In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1964), pp. 211–252.

[8] Peter J. Brockwell and Richard A. Davis. *Introduction to Time Series and Forecasting*. Springer New York, 2002. DOI: 10.1007/b97391.

[9] Morton B Brown and Alan B Forsythe. “Robust tests for the equality of variances”. In: *Journal of the American Statistical Association* 69.346 (1974), pp. 364–367.

[10] C.Grice. *How a basic attack crippled Yahoo @ONLINE*. Accessed June 19, 2014. 1998. URL: http://news.cnet.com/2100-1023-236621.html.

[11] C.Janssen. *Computer Networks@ONLINE*. Accessed June 19, 2014. 2014. URL: http://www.techopedia.com/definition/25597/computer-network.

[12] C.Preimesberger. *DDoS Attack Volume Escalates as New Methods Emerge @ONLINE*. Accessed July 23, 2014. May 2014. URL: http://www.eweek.com/security/slideshows/ddos-attack-volume-escalates-as-new-methods-emerge.html.

[13] Christian Callegari et al. “Improving PCA-based anomaly detection by using multiple time scale analysis and Kullback–Leibler divergence”. In: *International Journal of Communication Systems* 27.10 (2014), pp. 1731–1751.

[14] Balakrishnan Chandrasekaran. "Survey of network traffic models". In: *Waschington University in St. Louis CSE* 567 (2009).

[15] Christopher Chatfield. *The Analysis of Time Series: An Introduction*. Chapman and Hall/CRC, 2003.

[16] Liu Chun-Lin. "A tutorial of the wavelet transform". In: *NTUEE, Taiwan* (2010).

[17] D.Goldman. *Major banks hit with biggest cyberattacks in history @ONLINE*. Accessed June 19, 2014. 2012. URL: http://money.cnn.com/2012/09/27/technology/bank-cyberattacks/.

[18] D.Goodin. *New DoS attacks Taking Down Game Sites Deliver Crippling 100 Gbps Floods @ONLINE*. Accessed June 09, 2014. 2014. URL: http://arstechnica.com/security/2014/01/new-dos-attacks-taking-down-game-sites-deliver-crippling-100-gbps-floods.

[19] Christos Douligeris and Aikaterini Mitrokotsa. "{DDoS} attacks and defense mechanisms: classification and state-of-the-art". In: *Computer Networks* 44.5 (2004), pp. 643–666.

[20] Dr.F.Gong. *Deciphering Detection Techniques: Part II Anomaly-Based Intrusion Detection*. Tech. rep. McAfee, 2003.

[21] J. Durbin. "Estimation of Parameters in Time-Series Regression Models". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 22.1 (1960), pp. 139–153. ISSN: 00359246. URL: http://www.jstor.org/stable/2983884.

[22] J. Durbin. "The Fitting of Time-Series Models". In: *Revue de l'Institut International de Statistique / Review of the International Statistical Institute* 28.3 (1960), pp. 233–244. ISSN: 03731138. URL: http://www.jstor.org/stable/1401322.

[23] James Durbin. "Tests for serial correlation in regression analysis based on the periodogram of least-squares residuals". In: *Biometrika* 56.1 (1969), pp. 1–15.

[24] Thomer Gil and Massimiliano Poletto. "MULTOPS: A Data-structure for Bandwidth Attack Detection". In: *Proceedings of the 10th Conference on USENIX Security Symposium - Volume 10*. SSYM'01. Washington, D.C.: USENIX Association, 2001, pp. 3–3.

[25] ICT Global. *Network Basics: The Importance of Computer Networks @ONLINE*. Accessed June 19, 2014. 2001. URL: http://www.ictglobal.com/imp_networks.html.

[26] Louis Gordon and Moshe Pollak. "An Efficient Sequential Nonparametric Scheme for Detecting a Change of Distribution". In: *The Annals of Statistics* 22.2 (1994), pp. 763–804.

[27] James D. Hamilton. *Time Series Analysis*. Princeton University Press, 1994.

[28] Matz A Haugen, Bala Rajaratnam, and Paul Switzer. "Extracting Common Time Trends from Concurrent Time Series: Maximum Autocorrelation Factors with Application to Tree Ring Time Series Data". In: *arXiv preprint arXiv:1502.01073* (2015).

[29] Jonathan RM Hosking. "Fractional differencing". In: *Biometrika* 68.1 (1981), pp. 165–176.

[30] Shyh-Jier Huang and Kuang-Rong Shih. "Short-term load forecasting via ARMA model identification including non-Gaussian process considerations". In: *IEEE Transactions on power systems* 18.2 (2003), pp. 673–679.

[31] Alefiya Hussain, John Heidemann, and Christos Papadopoulos. "A Framework for Classifying Denial of Service Attacks". In: *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. SIGCOMM '03. Karlsruhe, Germany: ACM, 2003, pp. 99–110.

[32] Espen Alexander Fürst EAFI Ihlen. "Introduction to multifractal detrended fluctuation analysis in Matlab". In: *Frontiers in physiology* 3 (2012), p. 141.

[33] Sans Institute. *Host- vs. Network-Based Intrusion Detection Systems*. Tech. rep. Sans Institute, 2005.

[34] *Internet @ONLINE*. Accessed June 19, 2014. June 2014. URL: 'https://www.google.com/search?client=ubuntu&channel=fs&q=what+is+the+internet&ie=utf-8&oe=utf-8&gl=za'.

[35] Muhammad Faisal Iqbal and Lizy K John. "Power and performance analysis of network traffic prediction techniques". In: *Performance Analysis of Systems and Software (ISPASS), 2012 IEEE International Symposium on*. IEEE. 2012, pp. 112–113.

[36] J.Mirkovic, G.Prior, and P.Reiher. *Attacking DDoS at the Source*. Tech. rep. Los Angeles: Computer Science Department, University of California, 2002.

[37] J.Pierce. *Egress Filtering For a Better Internet*. Tech. rep. InfoSec Reading Room, Sept. 2004.

[38] Andreas Noack Jensen and Morten Ørregaard Nielsen. "A fast fractional difference algorithm". In: *Journal of Time Series Analysis* 35.5 (2014), pp. 428–436.

[39] Thomas Karagiannis, Mart Molle, and Michalis Faloutsos. "Understanding the limitations of estimation methods for long-range dependence". In: *University of California* (2006).

[40] Vasileios Karagiannis et al. "A survey on application layer protocols for the internet of things". In: *Transaction on IoT and Cloud Computing* 3.1 (2015), pp. 11–17.

[41] Alexand Karpukhin, Alex Kav, and Evgeny V. Nikulchev. "SIMULATION OF CHAOTIC PHENOMENA IN INFOCOMMUNICATION SYSTEMS WITH THE TCP PROTOCOL". In: *Journal of Theoretical and Applied Information Technology* (2018).

[42] Yoshinobu Kawahara and Masashi Sugiyama. "Change-Point Detection in Time-Series Data by Direct Density-Ratio Estimation". In: *Proceedings of the SIAM International Conference on Data Mining, SDM 2009, April 30 - May 2, 2009, Sparks, Nevada, USA*. SIAM, 2009, pp. 389–400.

[43] Dirk P Kroese and Zdravko I Botev. "Spatial process simulation". In: *Stochastic geometry, spatial statistics and random fields*. Springer, 2015, pp. 369–404.

[44] Kamlesh Lahre et al. "Analyze Different approaches for IDS using KDD 99 Data Set". In: *'International Journal on Recent and Innovation Trends in Computing and Communication, ISSN'* (), pp. 2321–8169.

[45] Markus Laner, Philipp Svoboda, and Markus Rupp. "Modeling randomness in network traffic". In: *ACM SIGMETRICS Performance Evaluation Review*. Vol. 40. 1. ACM. 2012, pp. 393–394.

[46] Markus Laner, Philipp Svoboda, and Markus Rupp. "Parsimonious network traffic modeling by transformed ARMA models". In: *IEEE Access* 2 (2014), pp. 40–55.

[47] W.E. Leland et al. “On the self-similar nature of Ethernet traffic (extended version)”. In: 2.1 (Feb. 1994), pp. 1–15. DOI: 10.1109/90.282603.

[48] Max Little et al. “Nonlinear, biophysically-informed speech pathology detection”. In: *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*. Vol. 2. IEEE. 2006, pp. II–II.

[49] G. M. Ljung and G. E. P. Box. “On a Measure of Lack of Fit in Time Series Models”. In: *Biometrika* 65.2 (1978), pp. 297–303. ISSN: 00063444. URL: http://www.jstor.org/stable/2335207.

[50] Jerry M Mendel. “Tutorial on higher-order statistics (spectra) in signal processing and system theory: Theoretical results and some applications”. In: *Proceedings of the IEEE* 79.3 (1991), pp. 278–305.

[51] Douglas C Montgomery, Cheryl L Jennings, and Murat Kulahci. *Introduction to time series analysis and forecasting*. John Wiley & Sons, 2015.

[52] NIST/SEMATECH. *Levene Test for Equality of Variances*. 2013. URL: http://www.itl.nist.gov/div898/handbook/eda/section3/eda35a.htm (visited on 09/21/2017).

[53] NIST/SEMATECH. *Two-Sample t-Test for Equal Means*. 2013. URL: http://www.itl.nist.gov/div898/handbook/eda/section3/eda353.htm (visited on 09/21/2017).

[54] Ilkka Norros. “On the use of fractional Brownian motion in the theory of connectionless networks”. In: *IEEE Journal on selected areas in communications* 13.6 (1995), pp. 953–962.

[55] O.Hyde. *Machine Learning for Cyber-Security at Network Speed and Scale*. Tech. rep. AI-ONE, Oct. 2011.

[56] Jason W Osborne. “Improving your data transformations: Applying the Box-Cox transformation”. In: *Practical Assessment, Research & Evaluation* 15.12 (2010), p. 2.

[57] E.S. Page. “Continuous Inspection Schemes”. English. In: *Biometrika* 41.1/2 (1954), pp. 100–115.

[58] Vern Paxson and Sally Floyd. “Wide area traffic: the failure of Poisson modeling”. In: *IEEE/ACM Transactions on Networking (ToN)* 3.3 (1995), pp. 226–244.

[59] Robi Polikar. *The wavelet tutorial*. 1996.

[60] Maurice Bertram Priestley. *Spectral analysis and time series*. Elsevier, 2004.

[61] Prolexic. *Global DoS and DDoS Attack Reports, Trends and Statistics @ONLINE*. Accessed June 19, 2014. 2014. URL: http://www.prolexic.com/knowledge-center-dos-and-ddos-attack-reports.html.

[62] Shaoyu Qiao et al. “Stationary wavelet transform and higher order statistical analyses of intrafascicular nerve recordings”. In: *Journal of neural engineering* 9.5 (2012), p. 056014.

[63] R.Bace. *An Introduction to Intrusion Detection and Assessment*. Tech. rep. ICSA Inc., 1999.

[64] Rajkumar and M.J.Nene. “A Survey on Latest DoS Attacks:Classification and Defense Mechanisms”. In: *International Journal of Innovative Research in Computer and Communication Engineering* 1.8 (2013), pp. 1847–1860.

[65] Rudolf H Riedi and Walter Willinger. “Toward an improved understanding of network traffic dynamics”. In: *Self-similar Network Traffic and Performance Evaluation, eds Park and Willinger, Wiley* (2000), pp. 507–530.

[66] Rudolf H Riedi et al. “A multifractal wavelet model with application to network traffic”. In: *IEEE Transactions on Information Theory* 45.3 (1999), pp. 992–1018.

[67] Rudolf Riedi and Jacques Lévy Véhel. “Multifractal properties of TCP traffic: a numerical study”. PhD thesis. INRIA, 1997.

[68] RM Sakia. “The Box-Cox transformation technique: a review”. In: *The statistician* (1992), pp. 169–178.

[69] A. Scherrer et al. “Non-Gaussian and Long Memory Statistical Characterizations for Internet Traffic with Anomalies”. In: 4.1 (Jan. 2007), pp. 56–70. DOI: 10.1109/TDSC.2007.12.

[70] New York University Information Technology Services. *Data and System Security Measures @ONLINE*. Accessed July 23, 2014. Nov. 2010. URL: http://www.nyu.edu/its/policies/sec_datasys.html.

[71] Yantai Shu et al. “Wireless traffic modeling and prediction using seasonal ARIMA models”. In: *Communications, 2003. ICC’03. IEEE International Conference on*. Vol. 3. IEEE. 2003, pp. 1675–1679.

[72] Bernard W Silverman. “Wavelets in statistics: beyond the standard assumptions”. In: *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 357.1760 (1999), pp. 2459–2473.

[73] Federico Simmross-Wattenberg et al. “Anomaly Detection in Network Traffic Based on Statistical Inference and α-Stable Modeling”. In: 8.4 (July 2011), pp. 494–509. DOI: 10.1109/tdsc.2011.14.

[74] Federico Simmross-Wattenberg et al. “Modelling Network Traffic as α–Stable Stochastic Processes. An Approach Towards Anomaly Detection”. In: *Proc. VII Jornadas de Ingeniería Telematica (JITEL)* (2008), pp. 25–32.

[75] Vasilios A Siris and Fotini Papagalou. “Application of anomaly detection algorithms for detecting SYN flooding attacks”. In: *Computer communications* 29.9 (2006), pp. 1433–1442.

[76] Tetiana Stadnytska, Simone Braun, and Joachim Werner. “Comparison of automated procedures for ARMA model identification”. In: *Behavior research methods* 40.1 (2008), pp. 250–262.

[77] Stilian Stoev and Murad S Taqqu. “Simulation methods for linear fractional stable motion and FARIMA using the Fast Fourier Transform”. In: *Fractals* 12.01 (2004), pp. 95–121.

[78] A.G. Tartakovsky et al. “A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point detection methods”. In: *Signal Processing, IEEE Transactions on* 54.9 (Sept. 2006), pp. 3372–3382.

[79] Nancy Tran and Daniel A Reed. “Automatic ARIMA time series modeling for adaptive I/O prefetching”. In: *IEEE Transactions on parallel and distributed systems* 15.4 (2004), pp. 362–377.

[80] Clemens Valens. “A really friendly guide to wavelets”. In: *ed. Clemens Valens* (1999).

[81] Jacques Vehel and Rudolf H Riedi. “Fractional Brownian motion and data traffic modeling: The other end of the spectrum”. In: *Fractals in Engineering* (1997).

[82] Darryl Veitch and Patrice Abry. “A wavelet-based joint estimator of the parameters of long-range dependence”. In: *IEEE Transactions on Information Theory* 45.3 (1999), pp. 878–897.

[83] Darryl Veitch, Patrice Abry, and Murad S Taqqu. “On the automatic selection of the onset of scaling”. In: *Fractals* 11.04 (2003), pp. 377–390.

[84] Darryl Veitch, Nicolas Hohn, and Patrice Abry. “Multifractality in TCP/IP traffic: the case against”. In: *Computer Networks* 48.3 (2005), pp. 293–313.

[85] Brani Vidakovic and Peter Mueller. “Wavelets for kids”. In: *Instituto de Estadística, Universidad de Duke* (1994).

[86] Flávio Henrique Teles Vieira, Gabriel Rocon Bianchi, and Luan Ling Lee. “A network traffic prediction approach based on multifractal modeling”. In: *Journal of High Speed Networks* 17.2 (2010), pp. 83–96.

[87] W.Lee et al. *Performance Adaptation in Real-Time Intrusion Detection Systems*. Tech. rep. Atlanta, GA, USA: College of Computing, Georgia Institute of Technology, Oct. 2002.

[88] BL Welch. “On the comparison of several mean values: an alternative approach”. In: *Biometrika* 38.3/4 (1951), pp. 330–336.

[89] Rafał Weron. “Estimating long-range dependence: finite sample properties and confidence intervals”. In: *Physica A: Statistical Mechanics and its Applications* 312.1-2 (2002), pp. 285–299.

[90] Rafał Weron. “Measuring long-range dependence in electricity prices”. In: *Empirical science of financial fluctuations*. Springer, 2002, pp. 110–119.

[91] Walter Willinger, Murad S Taqqu, and Ashok Erramilli. “A bibliographical guide to self-similar traffic and performance modeling for modern high-speed networks”. In: *Stochastic networks: Theory and applications* (1996), pp. 339–366.

[92] Walter Willinger et al. “Self-Similarity in High-Speed Packet Traffic: Analysis and Modeling of Ethernet Traffic Measurements”. In: *Stat. Sci.* 10.1 (Feb. 1995), pp. 67–85. DOI: 10.1214/ss/1177010131.

[93] Asrul H Yaacob et al. “Arima based network anomaly detection”. In: *Communication Software and Networks, 2010. ICCSN’10. Second International Conference on*. IEEE. 2010, pp. 205–209.

[94] Nong Ye, Connie Borror, and Yebin Zhang. “EWMA techniques for computer intrusion detection through anomalous changes in event intensity”. In: *Quality and Reliability Engineering International* 18.6 (2002), pp. 443–451.

[95] Hui Zou and Yuhong Yang. “Combining time series models for forecasting”. In: *International journal of Forecasting* 20.1 (2004), pp. 69–84.