Université d'Ottawa · University of Ottawa

Université d'Ottawa · University of Ottawa

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

Yingzi Li LI

AUTEUR DE LA THÈSE - AUTHOR OF THESIS

M. Sc. (Mathematics)

GRADE - DEGREE

Department of Mathematics

FACULTÉ, ÉCOLE, DÉPARTEMENT - FACULTY, SCHOOL, DEPARTMENT

TITRE DE LA THÈSE - TITLE OF THE THESIS

Varma Modelling for Window Size and RTT in TCP/IP Networks

D. McDonald

DIRECTEUR DE LA THÈSE - THESIS SUPERVISOR

A. Dabrowski

CO-DIRECTEUR DE LA THÈSE - THESIS CO-SUPERVISOR

EXAMINATEURS DE LA THÈSE - THESIS EXAMINERS

R. Balan

M. Ould Haye

J-M. De Koninck, Ph D

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

DEAN OF THE FACULTY OF GRADUATE AND POSTODORAL STUDIES

VARMA MODELLING FOR WINDOW SIZE AND RTT IN TCP/IP NETWORKS

By
Yingzi Li, B.Sc.
September 2004

A Thesis
submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements
for the degree of
Master of Science in Mathematics[1]

[1]The M.Sc. Program is a joint program with Carleton University, administered by the Ottawa-Carleton Institute of Mathematics and Statistics

Your file Votre référence
ISBN: 0-494-01525-X
Our file Notre référence
ISBN: 0-494-01525-X

Abstract

TCP (Transmission Control Protocol) and IP (Internet Protocol) are the most important protocols used for most of the data transmission across the Internet. Modern implementations of TCP mostly involve the concepts of Window Size and Round Trip Time (RTT) in controlling network congestion .

We model simulated time series of Window Size and RTT to evaluate the characteristics and the relationship of the data over time using autoregressive moving average methods. These methods help us to forecast network performance and detect server congestion.

Acknowledgements

I would like to thank my supervisors Prof. Andre Dabrowski, and Prof. David McDonald for all their help and advice. I would also like to thank the Ottawa-Carleton Institute of Mathematics and Statistics for funding throughout my research.

Dedication

My parents, my husband and my daughter.

Contents

List of Tables

List of Figures

Chapter 1

Introduction on TCP/IP

1.1 Definition of TCP/IP

A network is a system for resource sharing and message transmission, using the communication medium and network software to connect many computer systems that have different locations and independent functions. The Internet is a network of networks, in which the two most important protocols, Transmission Control Protocol (TCP) and Internet Protocol (IP), are used for most of the data transmission across the Internet. TCP/IP networks use four layers to map network interaction. The four layers from the top are Application Layer, Transport Layer, Network Layer and Data Link Layer (see Table 1). TCP/IP is based on the data link layer and interacts with the user application at the application layer. (see [4])

TCP provides a reliable data transfer service between two systems or devices connected on the internet at the transport layer. Sometimes error transmission disturbs the data. Sometimes the fundamental network has a problem. Sometimes the network is overloaded and causes the problem that the inter-media network transmission system did not work as normal. We need a protocol to provide a destination node with delivery information about transmitted packets such as sequence numbers and acknowledgement messages. This protocol can not only detect and discard duplicate messages, but can also detect errors or lost data and retransmit them until timeout

Application Layer (End-user applications)
Transport Layer (Communication handled among programs)
Network Layer (Addressing and routing)
Data Link Layer (Hardware and device drivers)

Table 1: TCP/IP networks use four layers to map network interaction

or the delivery is correctly completed. TCP is this kind of protocol that ensures the reliability of the transmission. When data has been lost, TCP can retransmit the data until either a timeout condition is reached or until successful delivery has been achieved. If the sending computer is transmitting too fast for the receiving computer, TCP can employ flow control mechanisms to slow data transfer. Furthermore, TCP can communicate delivery information to the upper-layer protocols and applications it supports.

IP is the most important protocol at the network layer. All other protocols in the TCP/IP suite depend on IP to move packets across the internet. It forwards each packet of the data around the network independently of network medium, based on the destination address i.e. the IP number. The internet authorities assign ranges of numbers to different organizations. The organizations assign groups of their numbers to departments. IP operators on gateway machines move data from department to organization and then around the world. In addition, it provides error reporting, fragmentation and the reassembly of the information units for the transmission over the networks. Full details of TCP/IP can be found in many engineering texts, here we mention only those features of interest to us here. Reference to http://www.samspublishing.com/articles/article.asp?p=28782&seqNum=3.

TCP depends on IP to move packets around the network. IP is inherently unreliable,

protection to the transport protocol. TCP has mechanisms that guarantee that the data it delivers to an application are correct. These mechanisms protect against data loss, data corruption, packet reordering and data duplication by adding 16 bit checksums and 32 bit sequence numbers to transmitted data and, on the receiving side, sending back tickets that acknowledge the receipt of data. Before sending data across the network, TCP establishes a connection between the source and the destination. The connection is destroyed, when the application that was using TCP indicates that no more data will be transferred.
Reference to http://www.smarthomeforum.com/start/faq.asp?ID=21.

1.2 TCP Error Correction

TCP has mechanisms that guarantee the data is correctly delivered to an application. Before sending data across the network, TCP establishes a connection between the source and the destination. If the receiver gets a packet from the sender, it sends back Acknowledgement of the receipt of data. If the timer expires before the acknowledgement is sent back, TCP assures that the packet is lost or corrupted and the source will retransmit the packet. In order to improve the network performance by avoiding both congestion and packet losses in the network, TCP continuously adjusts the transmission rate of the data at the source computer(see [16])

Because of crucial importance of TCP error correction, TCP includes several mechanisms such as 'Slow Start', 'Congestion Avoidance', 'Fast Retransmission' and 'Fast Recovery' algorithms that attempt to sustain good data transfer rates while avoiding placing excessive load on the network. Meanwhile, a lot of research and development have also been put into the improvement of the network performance.

1.3 Goals of This Thesis

The two concepts of Window Size and Round Trip Time (RTT) are frequently involved in modern implementations of TCP. We will study the statistical properties of

Window Size and RTT as stochastic processes in the following chapters in this thesis to help forecast Window Size and RTT so as to improve TCP performance.

Chapter 2 provides additional background and information on the Window Size and RTT variables. The basic tools for fitting autoregressive moving average univariate models are given in Chapter 3, and for bivariate time series in Chapter 4. We will fit an ARMA model to Window Size and RTT, singly and jointly, in Chapter 5. In summary, we simulate Win and RTT data using a commercial package and record Win and RTT at a single node, and apply the methods of Chapters 3 and 4.

The novelty of this thesis is to fit a bivariate ARMA model to the bivariate time series of Window Size and RTT to evaluate the relationship between them.

Chapter 2

Introduction on Window Size and RTT

2.1 The Concepts of Window Size and RTT

2.1.1 Round Trip Time-out(RTO)

One of the most important and complex ideas in TCP is embedded in the way it handles timeout and retransmission. The value of the retransmission time-out is critical. Since it is difficult to know the delay characteristics of networks in advance, and they can vary during the life of a connection, depending on route and loading, this makes it almost impossible to decide on one time-out value for this retransmit timer. If the time-out is set too low, it will expire before the acknowledge returns, causing the segment to be sent again unnecessarily, and increasing the traffic. If the value is set too high, it causes long delays waiting for the time-out to mature before a retransmission can be sent, so the loss of one segment will significantly decrease throughput(see [16]).

Modern implementations of TCP are now supposed to use a mechanism that works out the average delay of a connection and adds a threshold above that delay to deal with the variations. This dynamic timer depends on the Round Trip Time(RTT) on the path in use, which is a measure of delay between two hosts.

2.1.2 Round Trip Time (RTT)

At the time when the source sends a packet, it starts a timer and waits for an acknowledgement from the destination. The RTT is the total time taken for a packet to be sent from the source computer to the destination computer and an acknowledgement sent back to the source. If the timer expires before the acknowledgement is received, TCP assumes that the packet was lost or corrupted and retransmits it.

2.1.3 Window Size

The TCP window size is the amount of packets to be sent at the start of the time of RTT. For example, in a sliding window protocol with window size 8, the sender is permitted to transmit 8 packets before it receives an acknowledgement. This is the most important parameter for controlling congestion and to achieve maximum TCP bandwidth across the network.

2.2 The Measurement of Window Size and RTT

2.2.1 Window Size

According to the description by Von Welch (see [30]), the amount of data that can be in the network between two endpoints of the connection is the bandwidth delay product:

$$bandwidth\ delay\ product = bottleneck\ bandwidth * round\ trip\ time$$

To use the full bandwidth, the TCP window size should be at least as large as the bandwidth delay product so as to ensure the sender is continuously sending packets to fill the receiver's window. That means the sender can never send more than the maximum TCP window size of the receiver per RTT. The window size is equal to the amount of data that the sender can send in time RTT. If the sender has sent the entire window, it has to wait for an acknowledgement (ACK) of the oldest outstanding packets.

2.2.2 RTT

TCP uses the history of the measurements of Mean RTT to estimate the current value of Mean RTT and weights recent measurements more heavily than older ones. The individual measurements of RTT vary over time based on the congestion occurrence and may have high standard deviation. TCP performance depends on its ability to estimate the mean RTT on a connection. Whenever it obtains a new elapsed current RTT from two times, it adjusts the estimate mean RTT for the connection by the following formula:

$$EstimateMeanRTT(n) = \alpha \cdot EstimateMeanRTT(n-1) + (1-\alpha) \cdot CurrentRTT(n)$$

This is an exponential weighted moving average, with the typical value of α as $\frac{7}{8}$ by current researchers (see [4]). We will further analyze RTT, as well as Window Size, in the following chapters of this thesis and obtain alternate estimators.

2.3 The Importance of Estimating Window Size and RTT

Part of the TCP specification declares audition for a round-trip timeout(RTO) which means the decision to detect and then retransmit lost segments. It is based on the estimates of the round-trip time and is critical to maximizing TCP's performance. When a packet is sent and an acknowledgement never arrives, at some point the sender must declare a time-out, that is the sent packet is delayed and an acknowledgement will never arrive. The formula: $RTO = RTT \cdot \beta$ suggests that constant β is usually 2, reference to http://cs.baylor.edu/ donahoo/NIUNet/tcptour/javis/tcp-rttvar.html. If the timeouts are too small, it is not safe to retransmit packets since they have actually not been dropped, instead, they are just taking too long to arrive. Large timeout makes inefficient use of the bandwidth and increases congestion in the network. Therefore, the better the timeouts are adjusted, the better the protocol will perform. By accurately modelling and estimating the round-trip time, we can more

efficiently measure network performance, such as the identification of congestion unresponsive traffic.

Many congestion avoidance control schemes stated in the following section such as TCP Reno(see [5]) and TCP Vegas (see [19]) are based on an analytic derivation of an optimum window size for a deterministic network. Every connection in the network controls the window size by the relative algorithms that limit the number of packets from this connection allowed into the network during one RTT. Meanwhile, the RTT in the connection varies due to queueing delay variations or routing changes. The more accurately we expect to evaluate the window size and RTT, the more complicated the algorithm will be. By accurately modelling and estimating the window size and RTT, we can more efficiently measure network performance including the identification of congestion traffic.

2.4 Relationship between Window Size and RTT

Recent research exploits the relationship between Window Size and RTT to control network congestion.

TCP Reno(see [5]) is a congestion avoidance control schemes based on reducing the window size when congestion causes packet losses. An analytic derivation of the evolution of the window size in the network is given in [5]. Since the source writes its current window size and the current RTT in each packet it will send, we define $W_n(t)$ and $R_n(t)$ be the window size and RTT written in a packet from connection n arriving at the router at time t. When no loss occurs, the window size increases linearly with time at rate $\frac{1}{R_n(t)}$. When a source sees that a packet was lost at time $t - R_n(t)$, the source cuts the current window $W_n(t^-)$ by half, the slow-start threshold is set to $\frac{W_n(t^-)}{2}$ and the lost packet is retransmitted. Then the source enters a fast recovery phase which lasts one RTT. During the rest of the RTT, these $\frac{W_n(t^-)}{2}$ packets are transmitted and there is no linear increase in this RTT. McDonald (see [5]) describes

the above evolution of the window size by the stochastic differential equation

$$dW_n(t) = \frac{1}{R_n(t)} \cdot (1 - \chi_{S_n(t)})dt - \frac{W_n(t^-)}{2} dN_n(\Lambda_n(t))$$

where $s_n(t)$ denotes the event that in the RTT before time t the window for connection n entered fast retransmit due to losses in the preceding RTT. $N_n(\Lambda_n(t))$ is the time changed Possion process, indicating the occurrence of the losses of connection n.

TCP Vegas (see [25]), another kind of congestion scheme, adjusts the window size once every two round-trip times based on the product of $(WindowSize_{current} - WindowSize_{old}) \times (RTT_{current} - RTT_{old})$. If the result is positive, decrease the window size by one-eighth, or else increase the window size by one maximum segment size.

The adjust scheme for Vegas is not a loss based algorithm but rather is based on reducing the window size when congestion causes the estimate RTT to increase.

The relationship between window size and RTT is also described by Baccelli et al in the research of the active queue management modules in particular Random Early Detection (RED) schemes (see [17]). According to the nature of the window flow control mechanism of TCP, the transmission rate of each source can be defined by the following formula(in [17]):

$$TransmissionRate = \frac{WindowSize}{RTT}$$

2.5 Approaches to Modelling and Analysis of Window Size and RTT

In our simulated network system, RTT affects Window Size a great deal, while Window Size also has an effect on RTT although less directly. Some approaches can be proposed to model this behavior and this relationship.

- System Analysis

According to the method of system analysis (see [25]), the network can be treated as a black-box, a SISO (single-input and single-output) system. However, many system identification techniques assume an independence between the input and the output, where the independence assumption cannot be satisfied with TCP since a lot of factors we can use in the system are dependent on the past round-trip times. We will not use this approach here.

Since the window flow control mechanisms of TCP protocol adjust the values of window size and RTT over time based on their historical characteristics, we will choose a time series approach instead of system analysis to model their dynamic behaviors.

- Spectral Time Series Method

With the spectral(Fourier) analysis, we can exploit the cyclical patterns of the source data, by separating a complex time series with cyclical components into some sine and cosine functions. We can tell the existence of a strong periodicity of the relative period in the data if a large correlation between sine or cosine coefficient is identified. We will not pursue this approach in the thesis here.

- Box-Jenkins Methods

The Box-Jenkins method requires the assumption of a stationary time series. The most general Box-Jenkins model includes (seasonal) difference operators, autoregressive terms and moving average terms. These methods have been programmed. The ARIMA procedure in SAS (see [26]) provides the detailed analysis of one series at a time and outputs the detailed series including original series, forecast series, residual series and confidence limit series. It is simple and convenient for us to fit the ARMA model to the series of Window Size and RTT, respectively using this program.

Just as the ARIMA procedure, the VARMAX procedure in SAS is also based on Box-Jenkins methodology, giving us detailed tools for bivariate time series analysis and modelling. We will use the ARIMA and VARMAX procedures for our univariate time series modelling and bivariate time series modelling based on Box-Jenkins methodologies.

We will use all the criterion to select which ARMA(p,q) among computing models is preferred. We also wish to consider transforms of the original series and, for the purpose of selecting the transform, we use least prediction error as a selection criterion. (see Section 5.2 and Section 5.3). Finally, we get ARMA and VARMA models for window size and RTT to evaluate the behavior of the system as well as the input-output relationship.

Chapter 3

ARMA Methods

A univariate time series is a set of observations $\{X_t\}$, each one being recorded at a specific time t. An important part of the analysis of a time series is the selection of a suitable probability model for the data. In this paper, we use a Box and Jenkins approach that includes the examination of empirical correlations and partial correlations. General model selection criteria such as AIC, the Akaike information criterion and SBC, Schwartz's asymptotic Bayesian modification of the AIC, have also been used in model selection. We will employ AIC. Most of this chapter is based on materials found in [6].

3.1 Data Preparation

Box-Jenkins Approach is only meaningful if the time series is covariance-stationary and any trend should be removed before calculating the correlations. The definition on stationarity is given in Section 3.2.1B. We use a common paradigm for preparing time series data for model fitting, which is called the Box-Cox transformation. If there are gaps in the Box-Cox transformed series, we fill them by interpolation to produce equally spaced observations. The key idea is that after fitting the model, the residuals should be like independent normal variables, and we want the transform that produces the most normal residuals.

The Box-Cox transformation f_λ converts the original observations to the transformed data by the following formula:

$$f_\lambda(y) = \begin{cases} \frac{y^\lambda - 1}{\lambda}, & \lambda \neq 0 \\ log(y), & \lambda = 0 \end{cases}$$

We dump the simulated time series into SAS data sets. Then we choose the different values of λ for Box-Cox transformation to get the data set for modelling.

3.2 Single Model

3.2.1 ARMA(p,q) Model

A. The Definition of ARMA(p,q) Model

The Univariate Auto Regressive Moving Average Model ARMA(p,q) is defined as

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + a_t - \theta_1 a_{t-1} - \cdots - \theta_q a_{t-q} \tag{3.1}$$

i.e.

$$(1 - \phi_1 B - \cdots - \phi_p B^p) X_t = (1 - \theta_1 B - \cdots - \theta_q B^q) a_t$$

where $a_t \sim NID(0, \sigma_a^2)$(Normal Independent Distribution). Here p is the order of autoregressive part and q is the order of moving average part in the model, which is usually less than p (see [28]). Also, $\phi_1, \cdots, \phi_p$ are the autoregressive parameters and $\theta_1, \cdots, \theta_q$ are the moving average parameters. B represents the backward operator.

For fixed p and q, we can choose a suitable model by estimating 95% C.I.'s for $\phi_i's$ and θ_i's and checking if they include zero. For example, in an ARMA(2,1) model $X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} - \theta_1 a_{t-1} + a_t$. If θ_1 is small, AR(2) fits; if ϕ_2 is small, ARMA(1,1) fits; if both ϕ_1 and ϕ_2 are small, MA(1) fits. Therefore, the main question that remains in estimation is how to obtain the best guess values of the parameters $\phi_i's$ and θ_i's when q is nonzero. The Box-Jenkins approach is a general method for obtaining estimates of the parameters of ARMA(p,q)(as equation (3.1)), which is based on covariance analysis(see [15],[24]),[7],[20]).

B. Auto-Correlation ρ and Auto-Covariance γ

The mean function and variance function for the process X_t are defined separately as following:

$$\mu(t) = E(X_t) \quad \sigma^2(t) = Var(X_t) \quad for\ all\ t$$

In general, $\mu(t)$, as well as $\sigma^2(t)$, can be different at each time t.

The stationary property is of importance for the time series we are studying. A time series is said to be strictly stationary if the joint distribution of $X_{t_1}, \cdots, X_{t_k}$ is the same as the joint distribution of $X_{t_1+\tau}, \cdots, X_{t_k+\tau}$ for all $t_t, \cdots, t_k, \tau$.

If k=2, the joint distribution of $X_{t_1+\tau}, \cdots, X_{t_k+\tau}$ depends only on the time difference $(t_2 - t_1 = \tau$, which is called the lag. Series satisfying $E(X_t) = \mu$, $\gamma_h = E[X_{t+h} - \mu][X_t - \mu]$ is referred to as weakly stationary or just as stationary, for short.(see[24],P20-21)

The auto-correlation function(acf) at lag k is defined as: $\rho(k) = \frac{\gamma(k)}{\gamma(0)}$. Here $\rho(k)$ measures the correlation between X_t and X_{t+k}.

C. Sample Auto-Covariance Coefficient and Sample Auto-Correlation Coefficient

Suppose we have N observations on a stationary process, say $x_1, x_2, \cdots, x_N$. The sample auto-covariance coefficient at lag k (see [3], P.23) is:

$$\hat{\gamma}_k = \frac{\Sigma_{t=1}^{N-k}(x_t - \overline{x})(x_{t+k} - \overline{x})}{N}$$

The sample ACF (auto-correlation coefficient) at lag k is:

$$\hat{\rho}_k = \frac{\hat{\gamma}_k}{\hat{\gamma}_0}$$

Generally, the plot of sample auto-correlation function(ACF), called correlogram, is helpful for us to identify the possible models for a given time series and to choose the

appropriate orders of AR or MA model. Since the sample ACF (Auto-Correlation Function)is only meaningful for data from a stationary time series model, we should check the stationary and seasonal trends by the correlogram. For a stationary series, the plot of ACF cuts off or dies down fairly quickly. For a seasonal series, the plot of ACF oscillates at the same frequency. Removing the trend in a series will be described in Section 3.6.

3.2.2 Estimation of ARMA(p,q) Parameters

For a given ARMA model, there is a unique covariance structure while the converse is not true. Box-Jenkins provided the method of parameter estimation, which is firstly obtaining initial estimates by the use of the estimated auto-covariance, and then getting the only one ARMA model exclusively in terms of previous history and in stationary invertible form. See [[6] P.187-203] for the detailed procedure.

A. Initial Parameter Estimation

For ARMA(p,q),

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + a_t - \theta_1 a_{t-1} - \cdots - \theta_q a_{t-q}$$

multiplying by X_{t-k} and taking the expectation at the both sides, we have:

$$\gamma_k = \phi_1 \gamma_{k-1} + \cdots + \phi_p \gamma_{k-p} + \gamma_{xa}(k) - \theta_1 \gamma_{xa}(k-1) - \cdots - \theta_q \gamma_{xa}(k-q)$$

$$\Rightarrow \quad \gamma_k = \phi_1 \gamma_{k-1} + \cdots + \phi_p \gamma_{k-p} \qquad k \geq q+1$$

$$where \quad \gamma_{xa}(k) = E[X_{t-k} a_t] \begin{cases} = 0, & k > 0 \\ \neq 0, & k \leq 0 \end{cases}$$

1. Calculate $\hat{\phi}_1, \hat{\phi}_2, \cdots, \hat{\phi}_p$. The estimates of $\phi_1, \phi_2, \cdots, \phi_p$ of $AR(p)$ are from the estimated auto-covariances $\hat{\gamma}_{q-p+1}, \cdots, \hat{\gamma}_{q+1}, \hat{\gamma}_{q+2}, \cdots, \hat{\gamma}_{q+p}$ by the equations:

$$\begin{cases} \hat{\gamma}_{q+1} = \hat{\phi}_1 \hat{\gamma}_q + \hat{\phi}_2 \hat{\gamma}_{q-1} + \cdots + \hat{\phi}_p \hat{\gamma}_{q-p+1} \\ \hat{\gamma}_{q+2} = \hat{\phi}_1 \hat{\gamma}_{q+1} + \hat{\phi}_2 \hat{\gamma}_q + \cdots + \hat{\phi}_p \hat{\gamma}_{q-p+2} \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \hat{\gamma}_{q+p} = \hat{\phi}_1 \hat{\gamma}_{q+p-1} + \hat{\phi}_2 \hat{\gamma}_{q+p-2} + \cdots + \hat{\phi}_p \hat{\gamma}_q \end{cases}$$

2. Calculate the first $q+1$ auto-covariance $\hat{\gamma}_0^{'}, \cdots, \hat{\gamma}_q^{'}$ of the derived series $x_t' = x_t - \hat{\phi}_1 x_{t-1} - \cdots - \hat{\phi}_p x_{t-p} = \phi(B)x_t$ by using $\phi_1, \cdots, \phi_p$ obtained at step 1.

$$\hat{\gamma}_j' = \sum_{i=0}^{p} \hat{\phi}_i^2 \hat{\gamma}_j + \sum_{i=1}^{p} (\hat{\phi}_0\hat{\phi}_i + \hat{\phi}_1\hat{\phi}_{i+1} + \cdots + \hat{\phi}_{p-i}\hat{\phi}_p)(\hat{\gamma}_{j+i} + \hat{\gamma}_{j-i})$$

$$where \quad j = 0, 1, \cdots, q, \quad \hat{\phi}_0 = -1$$

3. Get $\hat{\theta}_1, \hat{\theta}_2, \cdots, \hat{\theta}_q, \hat{\sigma_a}^2$ by using $\hat{\gamma}_0', \hat{\gamma}_1', \cdots, \hat{\gamma}_q'$ obtained in step 2.

$$By \quad x_t' = \phi(B)x_t = \theta(B)a_t$$

We have $\gamma_0' = (1 + \hat{\theta}_1^2 + \cdots + \hat{\theta}_q^2)\hat{\sigma}_a^2$ by linearly convergent process.

$$\hat{\gamma}_k' = (-\hat{\theta}_k + \hat{\theta}_1\hat{\theta}_{k+1} + \cdots + \hat{\theta}_{q-k}\hat{\theta}_q)\hat{\sigma}_a^2 \quad for \quad k \geq 1$$

$$\Rightarrow \begin{cases} \hat{\sigma}_a^2 = \frac{\hat{\gamma}_0'}{1+\hat{\theta}_1^2+\cdots+\hat{\theta}_q^2} \\ \hat{\theta}_j = -(\frac{\hat{\gamma}_j'}{\hat{\sigma}_a^2} - \hat{\theta}_1\hat{\theta}_{j+1} - \hat{\theta}_2\hat{\theta}_{j+2} - \cdots - \theta_{q-j}\theta_q) \end{cases} \quad j = 1, \cdots, q \quad \theta_0 = 0$$

B. Confidence Intervals for the Parameters

For the ARMA models, we denote β as the set of unknown parameters,i.e.$\beta = (\mu, \phi_1, \cdots, \phi_p, \theta_1, \cdots, \theta_q)$ and $Q(\beta)$ is the sum of squares, i.e.$Q(\beta) = \Sigma_t \varepsilon_t^2$, where $\varepsilon_t = X_t - (\hat{\phi}_1 X_{t-1} + \cdots + \hat{\phi}_p X_{t-p} + a_t - \hat{\theta}1 a_{t-1} - \cdots - \hat{\theta}q a_{t-q})$.
Let $\hat{\beta}$ be the maximum likelihood estimate of β. Based on the white noise ε_t, a $(100-\alpha)\%$ confidence interval for β can be derived from(see [20],P.369):

$$Q(\beta) \leqslant Q(\hat{\beta})\{1 + \frac{\chi_q^2(\alpha)}{N}\}$$

3.2.3 Model Diagnostics

Model diagnostics are used to check the fit of a specific ARMA(p,q) model to the source data.

A. Residuals

When regression is performed on time series data, the errors may not be independent. Often errors are autocorrelated i.e. each error is correlated with the error immediately before it. However, the residual series for an ARMA model is defined by

$$Residual = OriginalObservationData - FittedForecastData$$

The residual series must be a white noise, which is defined to be a sequence of independent and identically distributed(i.i.d.) random variables. If the residuals are white noises, then the estimation is correct. If a pattern of significant spikes persists in the residuals, alternative identification and estimation may be in order. When the characteristic patterns of spikes have been properly identified and estimated in the model, the residuals of the estimation process will resemble random insignificant white noise and that stage of the modelling will be completed.

B. Prediction Error

In order to get source series stationary, we run specific ARMA(10,10) on different transformed series and calculate the prediction error by the following formula:

$$Prediction \;\; Error = \frac{\sum (OriginalData - ForecastData)^2}{TotalNumber}$$

The transformation with the smallest value of the prediction error will be judged to be the best.

C. ACF and PACF

As stated above, the autocorrelation function (ACF) of X_t is a graph of autocorrelation coefficients. We use r_k to denote the sample autocorrelation coefficient of a stationary series, where k denotes the difference between two times. (see [32])

The partial autocorrelation (PACF) is the plot of ϕ_{kk} for different values of k, against k. ϕ_{kk} is the correlation between x_t and x_{t+k} after removing their mutual linear dependence on the intervening variables $x_{t+1}, x_{t+2}, \cdots, x_{t+k-1}$. $P_k = \phi_{kk} =$

$Corr(Z_t, Z_{t+k}|Z_{t+1}, \cdots, Z_{t+k-1}) = \frac{A}{B}$, where A is the determinant of

$$\begin{bmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-2} & \rho_1 \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-3} & \rho_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & \rho_1 & \rho_k \end{bmatrix}$$

B is the determinant of

$$\begin{bmatrix} 1 & \rho_1 & \cdots & \rho_{k-1} \\ \vdots & \vdots & \vdots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \cdots & 1 \end{bmatrix}$$

The standard way to check the model is to plot the time series, residual series, the sample ACF and sample PACF, visually examine the graphs of the series over time to see if they match. If the data are from a white noise, then the approximate standard error of the ACF and PACF is $\frac{1}{\sqrt{n}}$.

Model	*ACF*	*PACF*
White	*Clean*	*Clean*
AR(p)	*Tail − off*	*Cut − off at lag p*
MA(q)	*Cut − off at lag q*	*Tail − off*

For ARMA model, looking for a cutoff in either the ACF or PACF will be very hard and neither the function of ACF nor the function of PACF is especially dramatic. Therefore, considerable experience and skills are needed to specify a mixed model. In our paper and our current research, we use the predicted error of the relative models, combined with some criteria, to find fits to ARMA model for the source data.

3.3 Choosing the Value of λ

To get a Box-Cox transformation, we need to find λ. Fixing an ARMA model with a pair of p_0 and q_0, we test all ARMA models with different λ. For different λ under the Box-Cox transformed series, we compute Prediction Error according to the

respective minimum AICs for each λ. Among a list of results generated from different transformations, we choose λ according to the minimum prediction error instead of minimum AIC. This is because AICs could be used in the comparison among different pair of p and q for one data set, not to be used in the comparison among different pair of p and q across different data sets. After inspecting the graphs generated from a group of λs, we can choose a specific λ_0 and then transform our series to that used for subsequent modelling. (see [3])

3.4 Choosing the Orders of p and q

We decide ARMA(p,q) among competing models by using the most popular information criteria.

The approach to model selection consists in evaluating the 'distance' between the selected model and the true (unknown) model. $f(x)$ and $f_0(x)$ are the density functions associated with these respective models, where $x = (x_1, x_2, \cdots, x_T)'$ and T is sample size. One kind of distance, Kullback-Leibler distance, is as following:

$$I(f, f_0) = \int ln[\frac{f_0(x)}{f(x)}]f_0(x)dx$$

$$= E_{f_0} ln\frac{f_0(x)}{f(x)} = E_{f_0} ln[f_0(x)] - E_{f_0}(ln[f(x)]$$

Minimizing $I(f, f_0)$ with respect to f is equivalent to minimizing $-E_{f_0} ln[f(x)]$. We obtain an information criterion by selecting an 'estimator' of $-E_{f_0} ln(f)$. These different criteria take the following general form: $IC^* = -\frac{1}{T} ln(f) + \alpha(T)(p+q)$, where $\alpha(T)$ is a decreasing function of T. Since f is a normal density, IC^* is equivalent to $IC = ln(\hat{\sigma}_T^2) + \alpha(T)(p+q)$.

Different criteria are obtained by selecting different functions $\alpha(T)$(see [20]). The most useful are as following:

- AIC (Akaike's Information Criterion) [Akaike, 1969]

$$\alpha(T) = \frac{2}{T}$$

$$\Rightarrow AIC(p,q) = ln(\hat{\sigma}_T^2) + \frac{2(p+q)}{T}$$

For T observation, $AIC = Tln(\hat{\sigma}_T^2) + 2(p+q)$.

- BIC (Schwarz's Information Criterion) [Schwarz, 1978]

$$\alpha(T) = \frac{log(T)}{T}[Schwarz, 1978]$$

$$\Rightarrow BIC(p,q) = log(\hat{\sigma}_T^2) + (p+q)\frac{log(T)}{T}$$

For T observation, $BIC = Tln(\hat{\sigma}_T^2) + (p+q)lnT$.

By using the minimum information criteria such as AIC and BIC, we can specify the best model for AR, MA and ARMA time series. Here we will use the AIC because of large data. By fixing λ as a chosen value λ_0, among all ARMA models with different pairs of p and q, we select p_0 and q_0 to minimize AIC.

3.5 Summary of Fitting the ARMA Model to The Data

3.5.1 General Procedure

Figure 1 shows the general procedure of ARMA modelling. Since the ACF and PACF can not help for the mixed ARMA series modelling, we use various transformations with specific λ on the data, together with AIC and Predicted Error, for our analysis and modelling. See details in Section 5.2.2.

At model selection step, we try different values of p, q and λ to fit the ARMA(p,q) models to the data. After comparing the values of predicted error under different

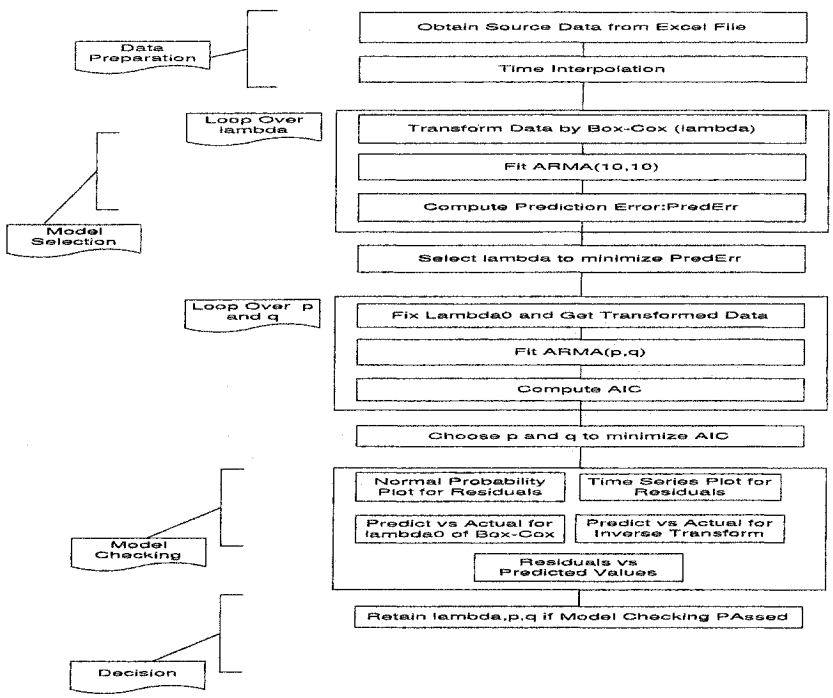

Figure 1: General Procedure of Fitting a Time Series Model to the Source Data.

chosen transform, we compare the values of AIC to select a pair of p and q to minimize AIC.

At model checking step, we check the overall prediction error and the residual series. If the residuals are independent from forecast time series, and if the residuals are random normal distributed, the chosen model is considered to be well fitted to the data.

3.5.2 Computing Tools

A. OPNET Tools

Online document of OPNET software tells us that OPNET software embeds expert knowledge about how network devices, network protocols, applications, and servers operate. OPNET Modeler is the leading network modelling and simulation environment, allowing us to design and study communication networks, devices, protocols, and applications with unmatched flexibility. We will use it to generate the raw data to be modelled here.

B. SAS Tools

SAS/ETS software is a data analysis component of the SAS System that focuses on processes that take place over time. 'ETS' stands for Econometrics and Time Series. SAS/ETS offers the ARIMA procedure, which analyzes and forecasts equally spaced univariate time series data, transfer function data, and intervention data using the ARIMA or ARMA model. Three statements are useful in the ARIMA procedure(see [26]).

- The IDENTIFY statement that specifies the time series to be modelled, differences the series if desired, and computes statistics to help identify models to fit.

- The ESTIMATE statement that specifies an ARMA model or transfer function model for the response variable specified in the previous IDENTIFY statement, and

produces estimates of its parameters. The ESTIMATE statement also prints diagnostic information by which to check the model.

- The FORECAST statement that generates forecast values for a time series using the parameter estimates produced by the previous ESTIMATE statement.

Chapter 4

VARMA Methods for Bi-Variate Series

From our original intention, we wish to study the relationship between the window size and RTT. Since a completely different situation arises when the 'outputs' affect the 'inputs', there is a closed-loop system here. Rather, a model, called a multivariate time-series model or Vector ARMA model, with more than one dimension will be needed to satisfactorily model the system: $\tilde{X}_t = \tilde{\phi}_1\tilde{X}_{t-1} + \cdots + \tilde{\phi}_p\tilde{X}_{t-p} + \tilde{a}_t - \tilde{\theta}_1\tilde{a}_{t-1} - \cdots - \tilde{\theta}_q\tilde{a}_{t-q}$ (see [28] and [2]). The overall model-building process is more difficult for multivariate time series than for univariate time series.

4.1 VARMA(p,q) Model

A. The Definition of Bi-Variate ARMA(p,q) Model

Suppose we have N observations on two variables at unit time intervals over the same period and denote W_t as an input stochastic process or time series for Window Size, Y_t as the relative output time series for RTT, where W_t not only depends on its past values but also on Y_t's. Letting $\tilde{Z}_t = \begin{bmatrix} W_t \\ Y_t \end{bmatrix}, \tilde{a}_t = \begin{bmatrix} a_{t1} \\ a_{t2} \end{bmatrix}$ where $\tilde{a}_t \sim NID(\tilde{0}, \Sigma_a)$. We say that $\{\tilde{Z}_t\}$ is stationary if $\{W_t\}$ and $\{Y_t\}$ are each stationary and $COV\{W_t, Y_s\}$ is a function of s-t only.(see [20],P655)

The model is:
$\phi(B)\tilde{Z}_t = \theta(B)\tilde{a}_t$ i.e. $\tilde{Z}_t = \tilde{\phi}_1\tilde{Z}_{t-1} + \cdots + \tilde{\phi}_p\tilde{Z}_{t-p} + \tilde{a}_t - \tilde{\theta}_1\tilde{a}_{t-1} + \cdots + \tilde{\theta}_p\tilde{a}_{t-q}$

$$\begin{bmatrix} Z_{t1} \\ Z_{t2} \end{bmatrix} = \begin{bmatrix} \phi_{111} & \phi_{121} \\ \phi_{211} & \phi_{221} \end{bmatrix} \begin{bmatrix} Z_{t-1,1} \\ Z_{t-1,2} \end{bmatrix} + \cdots + \begin{bmatrix} \phi_{11p} & \phi_{12p} \\ \phi_{21p} & \phi_{22p} \end{bmatrix} \begin{bmatrix} Z_{t-p,1} \\ Z_{t-p,2} \end{bmatrix}$$

$$+ \begin{bmatrix} a_{t1} \\ a_{t2} \end{bmatrix} - \begin{bmatrix} \theta_{111} & \theta_{121} \\ \theta_{211} & \theta_{221} \end{bmatrix} \begin{bmatrix} a_{t-1,1} \\ a_{t-1,2} \end{bmatrix} - \cdots - \begin{bmatrix} \theta_{11q} & \theta_{12q} \\ \theta_{21q} & \theta_{22q} \end{bmatrix} \begin{bmatrix} a_{t-q,1} \\ a_{t-q,2} \end{bmatrix} \tag{4.2}$$

The model is stationary if the eigenvalues of $\tilde{\phi}_1, \cdots, \tilde{\phi}_p$ are found inside the unit circle. It is invertible if the eigenvalues of $\tilde{\theta}_1, \cdots, \tilde{\theta}_p$ lie inside the unit circle. By stationarity, $\tilde{Z}_t = [\tilde{\phi}_p(B)]^{-1}\tilde{\theta}_q(B)\tilde{a}_t$. By invertibility, $\tilde{a}_t = [\tilde{\theta}_q(B)]^{-1}\tilde{\phi}_p(B)\tilde{Z}_t$. (see [31], p.339)

$$\textit{Letting} \quad \tilde{Z} = [Z_1, Z_2] = \begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \\ \vdots & \vdots \\ z_{n1} & z_{n2} \end{bmatrix} \quad \tilde{\phi} = \begin{bmatrix} \phi_{111} & \phi_{211} \\ \phi_{121} & \phi_{221} \\ \phi_{112} & \phi_{212} \\ \phi_{122} & \phi_{222} \\ \vdots & \vdots \\ \phi_{11p} & \phi_{21p} \\ \phi_{12p} & \phi_{22p} \end{bmatrix}$$

$$\tilde{\theta} = \begin{bmatrix} \theta_{111} & \theta_{211} \\ \theta_{121} & \theta_{221} \\ \theta_{112} & \theta_{212} \\ \theta_{122} & \theta_{222} \\ \vdots & \vdots \\ \theta_{11q} & \theta_{21q} \\ \theta_{12q} & \theta_{22q} \end{bmatrix} \qquad \tilde{a} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ \vdots & \vdots \\ a_{n1} & a_{n2} \end{bmatrix}$$

$$X = \begin{bmatrix} z_{01} & z_{02} & z_{-1,1} & z_{-1,2} & \cdots & z_{-p+1,1} & z_{-p+1,2} \\ z_{11} & z_{12} & z_{01} & z_{02} & \cdots & z_{-p+2,1} & z_{-p+2,2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ z_{n-1,1} & z_{n-1,2} & z_{n-2,1} & z_{n-2,2} & \cdots & z_{-p+n,1} & z_{-p+n,2} \end{bmatrix}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{01} & a_{02} & a_{-1,1} & a_{-1,2} & \cdots & a_{-q+1,1} & a_{-q+1,2} \\ a_{21} & a_{22} & a_{11} & a_{12} & a_{01} & a_{02} & \cdots & a_{-q+2,1} & a_{-q+2,2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \\ a_{n1} & a_{n2} & a_{n-1,1} & a_{n-1,2} & a_{n-2,1} & a_{n-2,2} & \cdots & a_{-p+n,1} & a_{-p+n,2} \end{bmatrix}$$

$$\Rightarrow Z = X\tilde{\phi} + A\tilde{\theta}$$

(see [23], [31])

B. Cross-Covariance and Cross-Correlation Matrix Functions

A key additional tool in modelling multivariate time series data is the cross-correlation function. The moments up to second order now consist of the mean and auto covariance functions for each of the two components plus a new function, called the cross-covariance function. Based on the assumption of stationary moments, the relative notations are defined as following. (see [20],p.655,p.656)

Means:

$$E(W_t) = \mu_w; \quad E(Y_t) = \mu_y$$

Auto-covariances:

$$\gamma_W(k) = Cov(W_t, W_{t+k}) = E[(W_t - \mu_w)(W_{t+k} - \mu_w)]$$

$$\gamma_Y(k) = Cov(Y_t, Y_{t+k}) = E[(Y_t - \mu_y)(Y_{t+k} - \mu_y)]$$

$$\gamma_W(0) = E[W_t W_t] \quad \gamma_Y(0) = E[Y_t Y_t]$$

These are estimated by: $\hat{\gamma}_W(k), \hat{\gamma}_Y(k)$.

$$\hat{\gamma}_W(k) = \frac{1}{n}\sum_{t=1}^{n-k}(w_t - u_w)(y_{t+k} - u_y)$$

$$\hat{\gamma}_Y(k) = \frac{1}{n}\sum_{t=1}^{n-k}(y_t - u_y)(y_{t+k} - u_y)$$

$$\hat{\gamma}_W(0) = \frac{1}{n}\sum_{t=1}^{n}(w_t - u_w)(w_t - u_w)$$

$$\hat{\gamma}_Y(0) = \frac{1}{n}\sum_{t=1}^{n}(y_t - u_y)(y_t - u_y)$$

The Cross-covariance:

$$\gamma_{WY}(k) = Cov(W_t, Y_{t+k}) = E[(W_t - \mu_w)(Y_{t+k} - \mu_y)]$$

$$\gamma_{WY}(0) = E[W_t Y_t]$$

These are estimated by: $\hat{\gamma}_{WY}(k) = \frac{1}{n}\sum_{t=1}^{n-k}(w_t - u_w)(t_{t+k} - u_y)$, $\hat{\gamma}_{WY}(0) = \frac{1}{n}\sum_{t=1}^{n}(w_t - u_w)(y_t - u_y)$

The covariance matrices of lag k can be written as:

$$\Gamma(k) = Cov[\begin{bmatrix} W_t \\ Y_t \end{bmatrix}, \begin{bmatrix} W_{t+k} \\ Y_{t+k} \end{bmatrix}] = \begin{bmatrix} \gamma_W(k) & \gamma_{WY}(k) \\ \gamma_{WY}(k) & \gamma_Y(k) \end{bmatrix}$$

The auto-correlation:

$$\rho_W(k) = \frac{\gamma_W(k)}{\gamma_W(0)}$$

$$\rho_Y(k) = \frac{\gamma_Y(k)}{\gamma_Y(0)}$$

The Cross-correlation:

$$\rho_{WY}(k) = \rho_{WY}(k) = \frac{\gamma_{WY}(k)}{\sqrt{\gamma_W(0)\gamma_Y(0)}} = \frac{\gamma_{WY}(k)}{\sigma_W \sigma_Y}$$

where $\sigma_W = \sqrt{\gamma_W(0)}$ denotes the standard deviation of the W-process, and similarly for σ_Y.

$$\hat{\rho}_{WY}(k) = \frac{\sum_{t=1}^{n-k}(w_t - \mu_w)(y_{t+k} - \mu_y)}{\sqrt{\sum_{t=1}^{n}(w_t - \mu_w)^2}\sqrt{\sum_{t=1}^{n}(y_t - \mu_y)^2}}$$

(see [3], p.244)

The correlation matrices can be defined as:

$$R(k) = D^{-1}\Gamma(k)D^{-1}, \quad D = \begin{bmatrix} \sqrt{\gamma_W(0)} & 0 \\ 0 & \sqrt{\gamma_Y(0)} \end{bmatrix}$$

(see [31], p.333)

4.1.1 VARMA Model Estimation

A. Parameter Estimation

We provide the approach to VARMA(p,q), which is an extension of the univariate ARMA(p,q) modelling. We use the following two methods to obtain the initial guess values of the parameters for the input and output series models.

The extensive Box-Jenkins approach is to obtain the parameter matrices, given covariance matrix function $\Gamma(k)$ (see [10]).

For bi-variate ARMA(p,q) model $\phi(B)Z_t = \theta(B)a_t$, multiplying by $\tilde{Z}_{t-k}$ on $\tilde{Z}_t'$, we have:

$$\Gamma(k) = E[\tilde{Z}_{t-k} \cdot \tilde{Z}_t'] = E[\tilde{Z}_{t-k} \cdot (\phi_1\tilde{Z}_{t-1} + \cdots + \phi_p\tilde{Z}_{t-p} + \tilde{a}_t - \theta_1\tilde{a}_{t-1} - \cdots - \theta_q\tilde{a}_{t-q})']$$

$$= E[\tilde{Z}_{t-k}(\tilde{Z}_{t-1}'\phi_1' + \cdots + \tilde{Z}_{t-p}'\phi_p' + \tilde{a}_t' - \tilde{a}_{t-1}'\theta_1' - \cdots - \tilde{a}_{t-q}'\theta_q')]$$

$$= \begin{cases} \Gamma(-1)\phi_1' + \Gamma(-2)\phi_2' + \cdots + \Gamma(-p)\phi_p' + \Sigma, & k = 0 \\ \Gamma(k-1)\phi_1' + \cdots + \Gamma(k-p)\phi_p' - \Sigma\dot{\theta}_k', & k \geq 1 \end{cases}$$

$E[\tilde{Z}_t \cdot \tilde{Z}_t'] = \Gamma(0)$

$E[\tilde{Z}_t \cdot \tilde{a}_t'] = E[(\phi_1\tilde{Z}_{t-1} + \cdots + \phi_p\tilde{Z}_{t-p} + \tilde{a}_t - \theta_1\tilde{a}_{t-1} - \cdots - \theta_q\tilde{a}_{t-q})\tilde{a}_t'] = E[\tilde{a}_t \cdot \tilde{a}_t'] = \Sigma$

$E[\tilde{Z}_{t-k} \cdot \tilde{a}_t'] = E[(\phi_1\tilde{Z}_{t-k-1} + \cdots + \phi_p\tilde{Z}_{t-k-p} + \tilde{a}_{t-k} - \theta_1\tilde{a}_{t-k-1} - \cdots - \theta_q\tilde{a}_{t-q-k}) \cdot \tilde{a}_t'] = 0 \; (for \; k \geq 1)$

$$E[\tilde{Z}_t \cdot \tilde{a}'_{t-1}] = E[(\phi_1 \tilde{Z}_{t-1} + \cdots + \phi_p \tilde{Z}_{t-p} + \tilde{a}_t - \theta_1 \tilde{a}_{t-1} - \cdots - \theta_q \tilde{a}_{t-q}) \cdot \tilde{a}'_{t-1}] = (\phi_1 - \theta_1)\Sigma$$

Letting $\tilde{X}_t = \theta_1 \tilde{a}_{t-1} + \cdots + \theta_q \tilde{a}_{t-q}$,
we have equations: $E[\tilde{Z}_{t-k}(\tilde{a}_t - \tilde{X}_t)] = E[\tilde{Z}_{t-k}(\tilde{Z}_t - \phi_1 \tilde{Z}_{t-1} - \cdots - \phi_p \tilde{Z}_{t-p})']$

$$= \begin{cases} \Gamma(0) - \Gamma(-1)\phi'_1 - \cdots - \Gamma(-p)\phi'_p & k = 0 \\ \Gamma(k) - \Gamma(k-1)\phi'_1 - \cdots - \Gamma(k-p)\phi'_p & k \geq 1 \end{cases} \tag{4.3}$$

$$E[\tilde{Z}_t \cdot \tilde{X}'_t] = E[\tilde{Z}_t(\theta_1 \tilde{a}_{t-1} + \cdots + \theta_q \tilde{a}_{t-q})'] = \phi_1 \Sigma \theta'_1 + \cdots + \phi_\iota \Sigma \theta'_\iota - \theta_1 \Sigma \theta'_1 - \cdots - \theta_q \Sigma \theta'_q \tag{4.4}$$

where $\iota = min(p, q)$.

By equations (4.3) and (4.4), we have following equations:
When k=0:

$$E[\tilde{Z}_t \cdot (\tilde{a}_t - \tilde{X}_t)'] = E[\tilde{Z}_t \cdot (\theta_1 \tilde{a}_{t-1} + \cdots + \theta_q \tilde{a}_{t-q})'] = \Gamma(0) - \Gamma(-1)\phi'_1 - \cdots - \Gamma(-p)\phi'_p$$

$$E[\tilde{Z}_t \cdot \tilde{X}'_t] = \phi_1 \Sigma \theta'_1 + \cdots + \phi_\iota \Sigma \theta'_\iota - \theta_1 \Sigma \theta'_1 - \cdots - \theta_q \Sigma \theta'_q$$

$$\Rightarrow E[\tilde{Z}_t \cdot \tilde{a}'_t] = \Gamma(0) - \cdots - \Gamma(-p)\phi'_p + \phi_1 \Sigma \theta'_1 + \cdots + \phi_q \Sigma \theta'_q - \theta_1 \Sigma \theta'_1 - \cdots - \theta_q \Sigma \theta'_q \tag{4.5}$$

When k=1:
$E[\tilde{Z}_{t-1}(\tilde{a}_t - \tilde{X}_t)'] = 0 - E[\tilde{Z}_{t-1}\tilde{X}'_t] = -E[\tilde{Z}_{t-1}(\tilde{a}'_{t-1}\theta'_1 + \cdots + \tilde{a}'_{t-q}\theta'_q)] = \Sigma\theta'_1$
$E[\tilde{Z}_t \cdot \tilde{X}'_t] = \Sigma(\phi_i - \theta_i)\Sigma\theta'_i$
$E[\tilde{Z}_{t-1} \cdot \tilde{X}'_t] = E[\tilde{Z}_{t-1} \cdot (\tilde{a}_{t-1}\theta'_1 + \cdots + \tilde{a}_{t-q} \cdot \theta'_q)] = \Sigma\theta'_1$

$$\Rightarrow \Gamma(1) - \Gamma(0)\phi'_1 - \cdots - \Gamma(1-p)\phi'_p + \Sigma\theta'_1 = 0 \tag{4.6}$$

When $k \geq 2$:
$E[\tilde{Z}_{t-k}(\tilde{a}_t - \tilde{X}_t)'] = \Gamma(k) - \Gamma(k-1)\phi'_1 - \cdots - \Gamma(k-p)\phi'_p$
$E[\tilde{Z}_{t-k}\tilde{X}'_t] = \Sigma\theta'_k$

$$\Rightarrow \Gamma(k) - \Gamma(k-1)\phi'_1 - \cdots - \Gamma(k-p)\phi'_p = \Sigma\theta'_k \tag{4.7}$$

The extensive Box-Jenkins approach obtains estimates of the parameters of VARMA(p,q), which based on the auto-covariance function. i.e. Given the covariance

we can calculate Σ, $\tilde{\phi}$ and $\tilde{\theta}$ through Equation [4.5], [4.6] and [4.7].

We can also use a general least square method to approach the VARMA model.

$$\tilde{Z} = [Z_1, Z_2] = \begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \\ \vdots & \vdots \\ z_{n1} & z_{n2} \end{bmatrix} \quad \tilde{\varepsilon} = [\varepsilon_1, \varepsilon_2] = \begin{bmatrix} \varepsilon_{11} & \varepsilon_{12} \\ \varepsilon_{21} & \varepsilon_{22} \\ \vdots & \vdots \\ \varepsilon_{n1} & \varepsilon_{n2} \end{bmatrix} \sim N(\tilde{0}, \Sigma_\varepsilon)$$

$$\tilde{X} = \begin{bmatrix} z_{01} & z_{02} & z_{-1,1} & z_{-1,2} & \cdots & z_{-p+1,2} & a_{01} & a_{02} & \cdots & a_{-q+1,1} & a_{-q+1,2} \\ z_{11} & z_{12} & z_{01} & z_{02} & \cdots & z_{-p+2,2} & a_{11} & a_{12} & \cdots & a_{-q+2,1} & a_{-q+2,2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ z_{n-1,1} & z_{n-1,2} & z_{n-2,1} & z_{n-2,2} & \cdots & z_{-p+n,2} & a_{n-1,1} & a_{n-1,2} & \cdots & a_{-q+n,1} & a_{-q+n,2} \end{bmatrix}$$

$$\tilde{\beta} = \begin{bmatrix} \phi_{111} & \phi_{211} \\ \phi_{121} & \phi_{221} \\ \phi_{112} & \phi_{212} \\ \phi_{122} & \phi_{222} \\ \vdots & \vdots \\ \phi_{11p} & \phi_{21p} \\ \phi_{12p} & \phi_{22p} \\ -\theta_{111} & -\theta_{211} \\ -\theta_{121} & -\theta_{221} \\ -\theta_{112} & -\theta_{212} \\ -\theta_{122} & -\theta_{222} \\ \vdots & \vdots \\ -\theta_{11q} & -\theta_{21q} \\ -\theta_{12q} & -\theta_{22q} \end{bmatrix}$$

For this system $\tilde{Z} = \tilde{X}\tilde{\beta} + \tilde{\varepsilon}$,i.e. $Z_{(i)} = X\beta_{(i)} + \varepsilon_{(i)}$ By least square method, $\hat{\beta} = (X'X)^{-1}X'Z$, i.e. $\hat{\beta}_{(i)} = (X'X)^{-1}X'Z_{(i)}$, $i = 1, 2$.

B. Confidence Intervals

- By Least Square Approach

We may separately put this basic result, derived by Box-Jenkins(1970) from Linear Hypothesis Theory(see [20],P.364), into the use of VARMA model checking.

$$H: \quad \tilde{Z} = X\tilde{\beta} + \tilde{\varepsilon}, \quad \tilde{\varepsilon} \sim N(0, \sigma),$$

$\tilde{\beta}$ is a vector of 4(p+q) unknown parameters in parameter space $\Omega = R^{4(p+q)}$.

We have the null hypothesis $H_0 : \tilde{\beta} \epsilon \omega$. Let ω be that single point in Ω which corresponds to the "true" value of $\tilde{\beta}$.

$$\beta_1 = min_{\tilde{\beta}\epsilon\Omega} \sum_{i=1}^{n} [(z_{i1} - E[z_1])^2 + (z_{i2} - E[z_2])^2] = ||\tilde{Z} - X\hat{\tilde{\beta}}||^2 \sim \Sigma_\varepsilon \chi^2_{n-4(p+q)}$$

which involves 4(p+q) unknown parameters.

$$\beta_0 = min_{\tilde{\beta}\epsilon\omega} \sum_{i=1}^{n} [(z_{i1} - E[z_1])^2 + (z_{i2} - E[z_2])^2] = ||\tilde{Z} - X\tilde{\beta}||^2$$

which involves 0 unknown parameters.

Under H_0, β_1 and $\beta_0 - \beta_1$ are independent,

$$\Rightarrow \frac{\frac{\beta_0 - \beta_1}{q}}{\frac{\beta_1}{n-4(p+q)}} \sim^D F_{(4(p+q), n-4(p+q))}$$

For each $\tilde{\beta}$,

$$Q(\tilde{\beta}) = \sum_{i=1}^{n} [(z_{i1} - \sum_{j=1}^{2p+2q} x_{ij}\beta_{j1})^2 + (z_{i2} - \sum_{j=1}^{2p+2q} x_{ij}\beta_{j2})^2] = ||\tilde{Z} - X\tilde{\beta}||^2,$$

which is the sum of squares of the derivation of the Z_i from their theoretical mean.

$$\Rightarrow Q_1 = Q(\hat{\tilde{\beta}}), \quad Q_0 = Q(\tilde{\beta}), \quad \Rightarrow \frac{Q(\tilde{\beta}) - Q(\hat{\tilde{\beta}})}{Q(\hat{\tilde{\beta}})} \sim \frac{4(p+q)}{n-4(p+q)} F_{(4(p+q), n-4(p+q))}$$

Let $F_{4(p+q),n-4(p+q)}(\alpha)$ denote the upper $\alpha\%$ point of the $F_{4(p+q),n-4(p+q)}(\alpha)$ distribution, $\Rightarrow (100-\alpha)\%$ C.I. for $\hat{\beta}$ can be obtained from inequality:

$$Q(\hat{\beta}) \leq Q(\tilde{\hat{\beta}})\{1+\frac{4(p+q)}{n-4(p+q)}F_{4(p+q),n-4(p+q)}\} \tag{4.8}$$

The alternative expression for C.I. of $\hat{\tilde{\beta}}$ is:

$$(\hat{\tilde{\beta}}-\tilde{\beta})'X'X(\hat{\tilde{\beta}}-\tilde{\beta}) \leq 4(p+q)\cdot\Sigma_\varepsilon\cdot F_{4(p+q),n-4(p+q)}$$

since $Q(\tilde{\beta})-Q(\hat{\tilde{\beta}}) = (\hat{\tilde{\beta}}-\tilde{\beta})'X'X(\hat{\tilde{\beta}}-\tilde{\beta})$.

Box-Jenkins suggest this method since all one has to do is to plot the contours of $Q(\beta)$, which satisfies the equation (4.5), and this immediately gives the region of points $\tilde{\beta}$, which constitutes the $100(1-\alpha)\%$ C.I.(see [20],p.367).

- By Likelihood Approach

This method can be used in general models, both linear and non-linear. For all standard Gaussian time series model, the log likelihood function:

$$L(\tilde{\beta}) = const - \frac{1}{2}\Sigma_\varepsilon^{-1}Q(\tilde{\beta}) \tag{4.9}$$

$\tilde{\beta}$ and $Q(\tilde{\beta})$ are the same forms as those in Page 29 and Page 30.

Let $\hat{\tilde{\beta}}$ denote the least square estimate of $\tilde{\beta}$.

$$L(\tilde{\beta}) = L(\hat{\tilde{\beta}}+(\tilde{\beta}-\hat{\tilde{\beta}}))$$

$$= L(\hat{\tilde{\beta}})+\sum_{k=1}^{2}\sum_{i=1}^{2(p+q)}(\beta_{ik}-\hat{\beta}_{ik})\frac{\partial L(\hat{\tilde{\beta}})}{\partial\beta_{ik}}+\frac{1}{2}\sum_{k=1}^{2}\sum_{i=1}^{2(p+q)}\sum_{j=1}^{2(p+q)}(\beta_{ik}-\hat{\beta}_{ik})(\beta_{jk}-\hat{\beta}_{jk})\frac{\partial^2 L(\hat{\tilde{\beta}})}{\partial\beta_{ik}\partial\beta_{jk}}+\cdots$$

Assuming $L(\tilde{\beta})$ be locally quadratic,

$$\frac{\partial L(\hat{\tilde{\beta}})}{\partial\beta_{ij}} = 0, j = 1,2 \quad L_{ij} = \frac{\partial^2 L(\hat{\tilde{\beta}})}{\partial\beta_i\partial\beta_j}$$

$$\Rightarrow L(\tilde{\beta}) = L(\hat{\tilde{\beta}}) + \frac{1}{2}\sum_{k=1}^{2}\sum_{i=1}^{2(p+q)}\sum_{j=1}^{2(p+q)}(\beta_{ik}-\hat{\beta}_{ik})(\beta_{ji}-\hat{\beta}_{jk})L_{ij}$$

Since $\sum_{k=1}^{2}\sum_{i=1}^{2(p+q)}\sum_{j=1}^{2(p+q)}(\beta_{ik}-\hat{\beta}_{ik})(\beta_{ji}-\hat{\beta}_{jk})L_{ij} \sim \chi^2_{4(p+q)}$, by Equation 4.6, we have:

$$Q(\tilde{\beta}) - Q(\hat{\tilde{\beta}}) \sim 2|\Sigma_\varepsilon| \cdot \frac{1}{2}\chi^2_{4(p+q)} = |\Sigma_\varepsilon|\chi^2_{4(p+q)}$$

If the model is linear, $(n-4(p+q))\hat{\Sigma}_\varepsilon = |\Sigma_\varepsilon| \cdot \chi^2_{n-4(p+q)}$ independent of $Q(\hat{\beta}) - Q(\tilde{\hat{\beta}})$ where the unknown Σ_ε can be estimated by $\hat{\Sigma}_\varepsilon = \frac{1}{n-4(p+q)}Q(\tilde{\hat{\beta}})$.

$$\Rightarrow \frac{Q(\hat{\beta}) - Q(\hat{\tilde{\beta}})}{Q(\hat{\tilde{\beta}})} = \frac{4(p+q)}{n-4(p+q)}F_{(4(p+q),n-4(p+q))}$$

$\Rightarrow$ A $100(1-\alpha)\%$ C.I. for $\hat{\beta}$ is:

$$\frac{Q(\hat{\beta}) - Q(\hat{\tilde{\beta}})}{Q(\hat{\tilde{\beta}})} \leq \frac{4(p+q)}{n-4(p+q)}F_{(4(p+q),n-4(p+q))}$$

4.1.2 Model Diagnostics

The parameter estimation above is on the assumption that the order is a priori known. In practice, the orders are almost invariably unknown. One of the methods of identification is testing ACF.

For bi-variate ARMA(p,q) model, the auto-covariance matrix is:

$$\Gamma_z(0) = \begin{bmatrix}
\Gamma_z(0) & \Gamma_z(1) & \cdots & \Gamma_z(p-1) & E[\tilde{Z}_t\tilde{a}'_t] & E[\tilde{Z}_t\tilde{a}'_{t-1}] & \cdots & E[\tilde{Z}_t\tilde{a}'_{t-q+1}] \\
\Gamma_z(-1) & \Gamma_z(0) & \cdots & \Gamma_z(p-2) & 0 & E[\tilde{Z}_{t-1}\tilde{a}'_{t-1}] & \cdots & E[\tilde{Z}_{t-1}\tilde{a}'_{t-q+1}] \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\Gamma_z(-p+1) & \Gamma_z(-p+2) & \cdots & \Gamma_z(0) & 0 & 0 & \cdots & E[\tilde{Z}_{t-p+1}\tilde{a}'_{t-q+1}] \\
\cdot & \cdot & \cdot & \cdot & \Sigma_a & 0 & \cdots & 0 \\
\cdot & \cdot & \cdot & \cdot & 0 & \Sigma_a & \cdots & 0 \\
\cdot & \cdot & \cdot & \cdot & 0 & 0 & \cdots & \Sigma_a
\end{bmatrix}$$

Under the requirement of p¿q, we have $\Gamma_z(h) = \phi_1\Gamma_z(h-1) + \cdots + \phi_p\Gamma_z(h-p)$. The auto-correlations are: $R_z(h) = D^{-1}\Gamma_z(h)D^{-1}$, where D is a diagonal matrix with the square roots of the diagonal elements of $\Gamma_z(0)$ on the diagonal(see [10],P.226-227).

4.2 Choosing the Orders of p and q

4.2.1 Akaike's AIC Criterion

AIC is a very general criterion, viewed as a "multiple decision procedure" rather than as a "hypothesis testing" and used in wide range of applications.
AIC=-2ln[maximized likelihood L]+2[number of parameters in the model].

For bi-variate ARMA(p,q) model defined as equation(4.2), a 2-dimensional (k=2) normal density for the innovation vector $\vec{a}_t = [a_{t1}, a_{t2}]'$ has the form:

$$f(\vec{a}_t) = \frac{1}{(2\pi)^{k/2}|\Sigma|^{1/2}} e^{-(\vec{a}-\vec{u})'\Sigma^{-1}(\vec{a}-\vec{u})/2}$$

$$where \quad \vec{a}_t = [a_{t1}, a_{t2}]' \quad \Sigma = cov(\vec{a}_1, \vec{a}_2')$$

The likelihood function for n observations is:

$$\iota = \frac{1}{(2\pi\sqrt{|\Sigma|})^{\frac{n}{2}}} e^{-\frac{1}{2}\sum_{t=1}^{n}(\vec{a}_t\Sigma^{-1}\vec{a}_t')}$$

The maximum likelihood estimate of log likelihood function is:

$$\hat{L} = -\frac{n}{2} ln|\hat{\Sigma}| - constant$$

Ignoring the constant, we have:

$$\Rightarrow \; AIC(p,q) = nln|\hat{\Sigma}| + 2(p+q)k^2$$

Reference to [20],p.373).

4.2.2 Bayesian Information Criterion BIC

BIC was derived from a Bayesian modification of the AIC criterion for a model involving q parameters fitted to n observations. BIC=-2ln[maximized likelihood]+(lnn)(number of parameters in the model) which is better than AIC in identifying the uni-variate ARMA models. It still needs to be discussed for VARMA model. e.g. (see [31] and [26])

$$BIC(p,q) = nln|\hat{\Sigma}| + k^2(p+q)ln(n)$$

Here we will use AIC as our selection criterion.

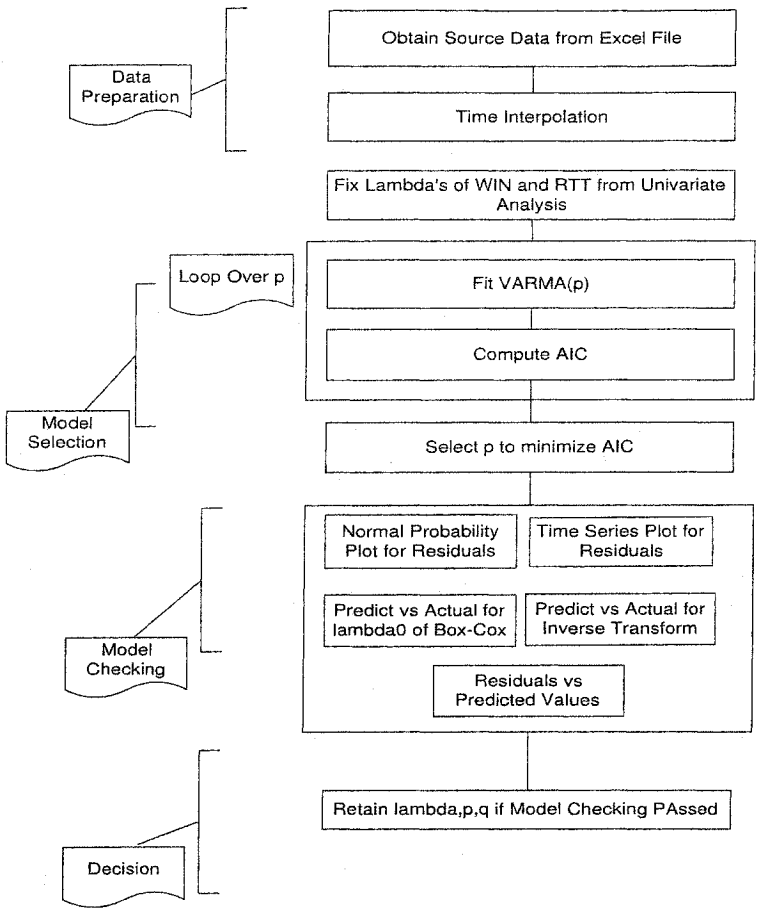

Figure 2: General Procedure of Fitting a Time Series Model to the Bivariate Data.

4.3 Summary of Fitting the Data the VARMA Model to the Bivariate Data

4.3.1 General Procedure

Figure 2 shows the procedure of VARMA modelling we use in the thesis. Since the ACF and PACF could not help for the VARMA modelling, we use Box-Cox transformations and AIC for our analysis and modelling. See the details at Section 5.4.

We begin with choosing the transformations obtained by univariate modelling. Then

we use VARMAX procedure to get a list of AIC with relative pairs of p and q. Comparing among the values of AIC, we make a decision of the best model with the pair of p and q, according to the minimum AIC. Then we test the properties of Residuals. If there is less cross-correlation between the residuals of two series, and if the both residuals are random normally distributed, the chosen model is fitted to the data.

4.3.2 Computing Tools

SAS/ETS also provides us the VARMAX procedure that can estimate the model parameters and generate forecasts associated with Vector Auto Regressive and Moving-Average process with eXogenous regressors (VARMAX) models. The VARMAX procedure enables us to model both the dynamic relationship between the dependent variables and between the dependent and independent variables.

When we use the VARMAX procedure, the orders of the autoregressive or moving-average process or both can be specified by options by the aids of partial cross-correlations, Yule-Walker estimates, partial autoregressive coefficients and partial canonical correlations. Those orders can also be automatically determined. Criteria for automatically determining these orders include AIC, Corrected AIC(AICC), Hannan-Quinn(HQ) Criterion, Final Prediction Error(FPE) and Schwarz Bayesian Criterion(SBC). The difference among these criterion was shown at [[3]],p.256-257, and suggests that we choose AIC because of the large sample size of our data.

VARMAX procedure only clearly offers a way to identify VAR or VMA models. We have found that there is a relation between the selected VARMA(p,q) and minimum AIC. Therefore, we use this way to identify a VARMA model for our simulated time series of window size and RTT.

Using the MODEL statement within VARMAX procedure, we can use the following methods to identify a VARMA model.

- METHOD options: requires the type of estimates to be computed. The possible values are: LS specifying least-squares estimates and ML, which specifies maximum likelihood estimates.

- Tentative Order Selection Options

MINIC=(TYPE=ValueOfInformationCriteria p=number q=number PERROR=number) prints the information criterion for the appropriate AR and MA tentative order selection and for the diagnostic checks of the fitted model. Here TYPE indicates the criteria of AIC, AICC, FPE, HQC and SBC.

We can also try to compute a VARMA prediction error for each choice of the pair of λs for the two series and choosing the best one as our VARMA model.

Chapter 5

Fitting Models to Univariate and Bivariate Data

Our target is to look into the behavior of the network through finding the best model for the simulated data sets of Window Size and Round Trip Time. In this chapter we will apply the methods of Chapter 3 and 4 to data modelling.

5.1 Preparation for Modelling

OPNET, which is the Optimized Network Engineering Tool, is a commercially available network simulation package. Under OPNET, we focus on the specific source, #21 in the source-subnet 0, see Figure 3, catching its window sizes $\{w(t)\}$'s and its RTTs $\{y(t)\}$'s to obtain the simulated data, see Figure 4. The Window Size data, for example, as dumped into the SAS system through a spreadsheet is shown in Figure 5. We discretize time to a constant increment of 0.5 seconds (see Figure 6), average the records within a single increment period and remove both burn-in phase and extremely explosive tail (see Figure 8). These still may be gaps in the data stream. We use "piecewise constant" interpolation for this missing data (see Figure 9) to get the final data set for ARMA and VARMA Modelling, where equally time-spaced data are required by time series modelling.

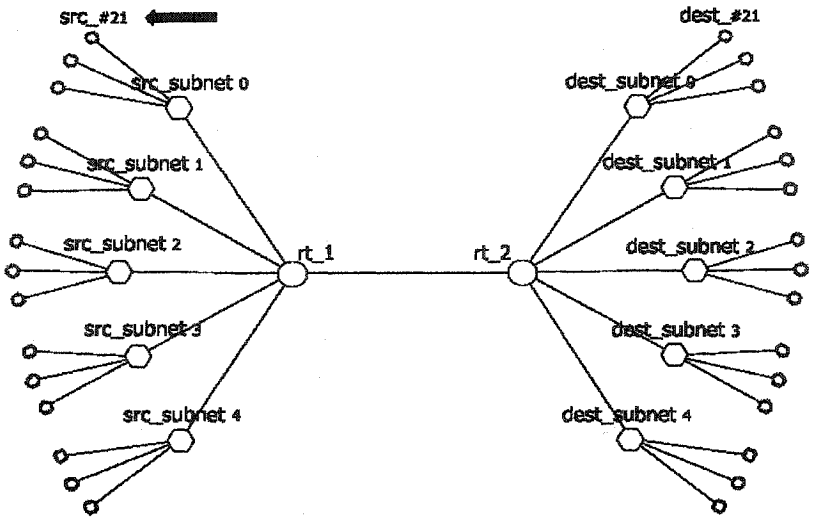

Figure 3: The network simulated under Opnet. The WIN and RTT data are recorded for source #21, indicated by the arrow above.

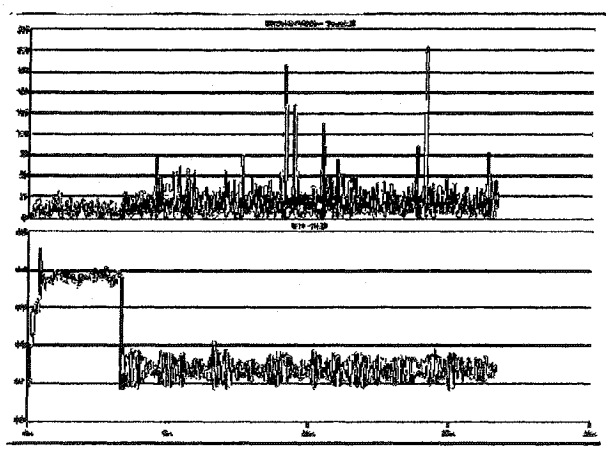

Figure 4: The Output of Simulated Window Size(top graph) and RTT(bottom graph) generated by Opnet Program, which lasts 17 minutes.

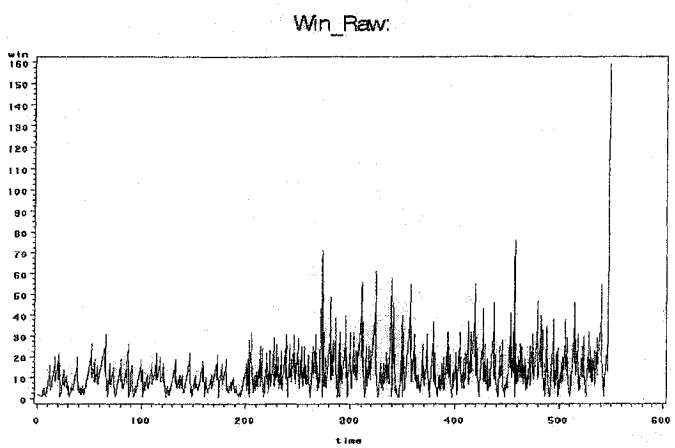

Figure 5: The original raw Window Size Data input from spreadsheet, covering 0 to 9 minutes (548 seconds).

There are approximately 64,000 records, with a burn-in phase and an extremely explosive right tail that needs to be improved. The figure resemble that of Figure 5, indicating that the time discretization and averaging have not qualitatively changed the data.

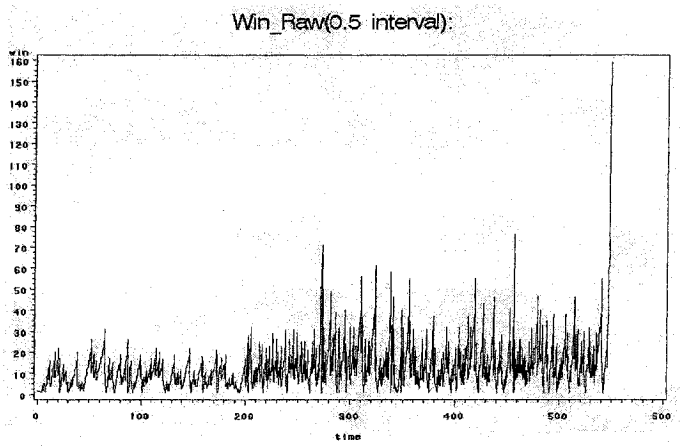

Figure 6: The original raw Window Size Data input from spreadsheet, covering 0 to 9 minutes (548 seconds) with the time interval of 0.5 seconds.

Through our network knowledge, we know that RTT affects window size a lot but the window size of a single source has only a very small effect on the RTT. This relationship will be verified by our final modelling and analysis(see Section 5.6). The correlation between window size and RTT is also hard to see in the raw data since these two series are sampled at different times. The window size is updated every time the source sends a packet, but the RTT is updated whenever a packet arrives to the router, so there is a slight difference in the times. The process of discretizing time to 0.5s intervals, averaging observations over intervals, and interpolating over gaps puts these series on a comparable footing. Now, we will model the behaviors of Win data and RTT data, and capture the relationship between them by the use of ARMA and VARMA methods.

5.2 Fitting ARMA Model to The Data of Window Size

5.2.1 General Procedures

A. Data Preparation

We run an Opnet simulation program to get these two simulation data sets(see Figure 4). In order to find out the complicate relationships between window size and RTT, we dump the graphs into a spreadsheet and then we transform the simulated data into SAS data sets. More than 64k records were generated during simulation. We discard the first three minutes and pick up the data between the fourth minute and the ninth minute. This is because the network is not in stationary operation and we must discard that initial period where it is not stabilizing. We choose the unit of time as 0.5 seconds. After averaging the multiple data records and filling gaps in the series by 'piecewise constant' interpolation, we get the data sets win_raw and rtt_raw, which are the source data sets for the coming analysis under the SAS system.

We have already dropped the burn-in segment (first four minutes) of the data, and a

visual inspection of the observation plotted against time shows stationary behaviors over the remainder of the period, see Figure 9.

B. Find the Best Transformed Data for Modelling

The Box-Cox transform formula we will use is:

$$tran(y) = \begin{cases} \frac{y^{\lambda}-1}{\lambda}, & \lambda \neq 0 \\ log(y), & \lambda = 0 \end{cases}$$

By the use of this transformation, we get the best transformed data set with sample size of 613 for win_sas and rtt_sas for our further SAS modelling.

For every different Box_Cox transformation with the different λ on win_sas or rtt_sas, we set the same values for p and q, called p0 and q0, of an ARMA model. After running ARIMA procedure in a loop with λ from 0.3 to 1.8 step 0.25, we obtained information of the average prediction errors with relative λs. The formula of the average prediction error is: $PrdErr(\lambda) = \frac{\sum_{i=1}^{N}(y_i - y_{i_0})^2}{N}$, where y_i is original data, y_{i_0} is inverse data (see the formula at page 45) of the forecast value under the model. Through the plot of Average PredErr versus λ, we determine the λ with the PrdErr as low as possible. The transformation of λ_0 with the minimum average prediction error is our preferred.

C. Model Selection

Fixing the transform data (see Figure 10) with above chosen λ_0, we run ARIMA procedure in a loop with p from 1 to 10 and q from 1 to 10 and get information of orders of p and q, and AICs. We are interested in the model with the minimum AIC. Therefore, we make the decision with the relative p_0 and q_0 achieving the least AIC value for that λ_0.

D. Model Checking and Decision

In order to check the obtained model, we rerun the procedure of ARIMA with obtained p_0 and q_0 under the chosen λ_0, we get the data sets of testest, testmod, teststat and predict. We make the decision that the model is the best fitting to the data, if the plot of standardized residual is a white noise and the plot of residuals versus the forecasted series is scattered without any pattern.

5.2.2 Algorithms Implemented

In this subsection, we apply the general methods of the previous subsection to the Window Size data.

A. Data preparation

1. Run Opnet program to get the simulation data. Figure 4 shows the output of Window Size and RTT, which lasts 17 minutes and includes burn-in phase.

2. Dump the output figure into a spreadsheet, remove burn-in phase and import the Excel data into SAS data set: win_raw. The win data, shown in Figure 7, corresponds to the period from 4 minutes to 9 minutes and provides 52675 records.

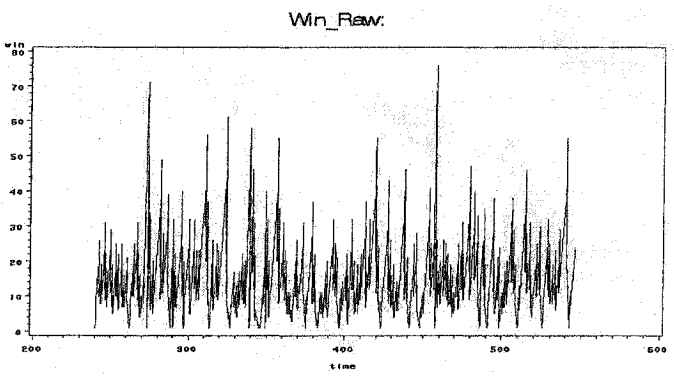

Figure 7: The dumped raw Window Size Data has been imported into SAS data set, and the burn-in phase and tail are removed.

3. Get the improved data set by discretizing time and averaging data. Since the data set win_raw contains 52675 records at unequally time points, we now discretize time to a constant increment of 0.5 seconds and average the multiple observations in each time bin. The increment of 0.5 seconds generates the least gaps based on the source series. Then we obtain 613 records in data set of win_impr. There are gaps in this series corresponding to longer stretches where no WIN data were recorded. The data are shown in above Figure 8.

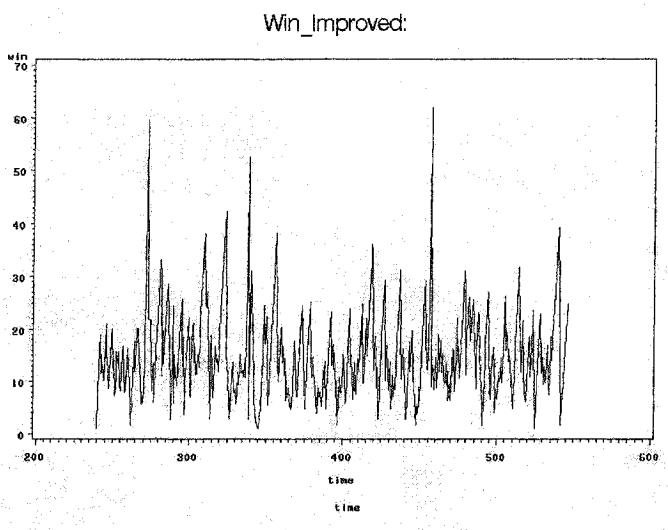

Figure 8: Improved Win data with averaging multiple data of same time.

4. Fill the gaps to get the data set of win_interp. The win_impr data set contains time series data at equally-spaced times but with gaps. We piecewise-constantly interpolate this data to produce a time series with equally-spaced observations. There are 613 observations with interval of 0.5 seconds in the data set of win_interp, see below Figure 9.

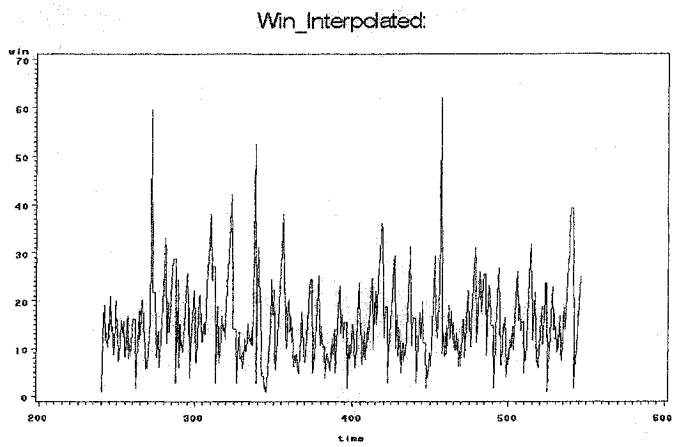

Figure 9: The Win data after linear-interpolation, with size of 613

After time interpolation, we can see that there is still a large variance in the data set. The values vary from zero to more than 40. So we will use the Box-Cox transformation to remove this feature at the next step.

B. Model Selection

1. Get transformed data set, win_interp, by transforming data set of win_interp into data set of win_sas by the Box-Cox Transformation. We provide an example when Win series is transformed using λ=0.75, see Figure 10.

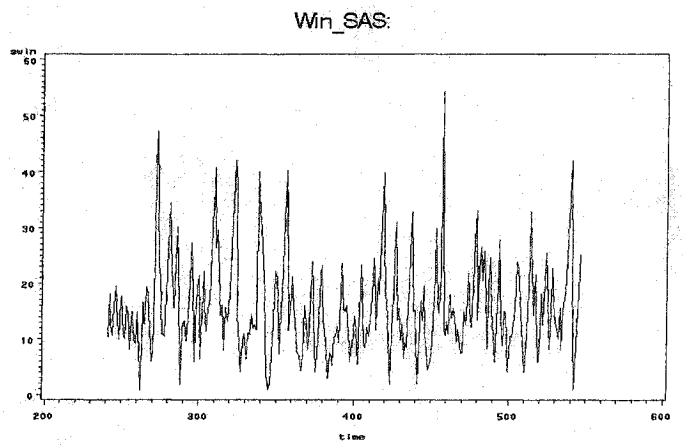

Figure 10: Transformed Data of Window Size for λ=0.75.

We can see that the wild variation has been removed and the appearance of the graph is more symmetric about its mean. The values of data are between zero and fifty-two. It looks stationary and is ready for further ARMA modelling.

The method we use is fitting an ARMA(10,10) model for different λ and plot the PredErr against λ. Loop: λ from 0.3 to 1.8 step 0.25
(a) Run PROC ARIMA(10,10) on different transformed data set:win_sas.
(b) Get inverse value of y, by

$$y^{-1} = InverseY = \begin{cases} (1+\lambda ForecastY)^{\frac{1}{\lambda}}, & \lambda \neq 0 \\ e^{ForecastY}, & \lambda = 0 \end{cases}$$

$$i.e. InverseWin\_win = \begin{cases} (1+\lambda Win)^{\frac{1}{\lambda}}, & \lambda \neq 0 \\ e^{WIn}, & \lambda = 0 \end{cases}$$

(c) Add predict error under each p,q and transformation. The result is in Table 2.
$prediction\ error = \frac{\sum (inverse-original)^2}{N}$

Obs	Lambda	PrdErr
13	0.600	36.8330
14	0.625	36.8105
15	0.650	36.7925
16	0.675	36.7787
17	0.700	36.7688
18	0.725	36.7625
19	0.750	36.7596
20	0.775	36.7598
21	0.800	36.7630
22	0.825	36.7689
23	0.850	36.7775
24	0.875	36.7886
25	0.900	36.8020

Table 2: Part of The Table of λ and PrdErr for Transformed Window Size, sorted by PrdErr and including minimum PrdErr. We will choose λ=0.75 since its PrdErr is minimum.

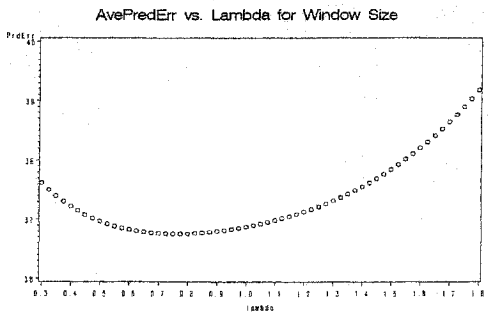

Figure 11: The Plot of Average Predicted Error against λ for ARMA(10,10). The bottom of the curve is approximately found at $\lambda = 0.75$, which we will choose.

2. For $\lambda_0 = 0.75$, we now try to find a proper model by AIC and rank of p+q.

Autocorrelation Plot of Residuals

Lag	-1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
0	|**********|
1	. |********** |
2	. |********* |
3	. |***** |
4	. |*** |
5	. |* . |
6	. | . |
7	. | . |
8	.*| . |
9	.*| . |
10	**| . |

Table 3: ACF shows the stationarity of win_sas with lambda=0.75 that Box_Jenkins approach can be used.

```
                Partial Autocorrelations
Lag      -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
  1       |                   . |**********         |
  2       |                   . |  .                |
  3       |                   .*|  .                |
  4       |                   .*|  .                |
  5       |                   .*|  .                |
  6       |                   .  |  .                |
  7       |                   .  |  .                |
  8       |                   .  |  .                |
  9       |                   .  |  .                |
```

Table 4: PACF shows the stationarity of win_sas with lambda=0.75 and the model will be AR(1) model, according Section 3.2.3 C.

loop: i from 0 to max_N

loop: j from 0 to max_N

(a) Run ARIMA(p=i,q=j) and get its AIC

(b) For λ_0, we obtain a list of pairs of p, q and AIC. This is shown in table 9.

Obs	p	q	AIC
1	1	0	3087.46
2	2	1	3088.54
3	4	3	3089.09
4	2	0	3089.30
5	1	1	3089.32
6	3	0	3090.32
7	3	1	3090.37

Table 5: Part of The Table of p, q and AIC sorted by AIC, for $\lambda = 0.75$. Here we choose p=1 and q=0.

3. Rerun ARIMA(1, 0) for $\lambda = 0.75$ with the smallest AIC and obtain the data sets testest, testmod, teststat and predict. We now can examine whether the diagnostic tests are fulfilled for this choice of model.

C. Model Selection for WIN data

With the above analysis, we choose the model ARMA(1,0) for the transformed WIN data under λ=0.75.

D. Model Checking

1. plot the residuals vs the forecast window size, see Figure 12.

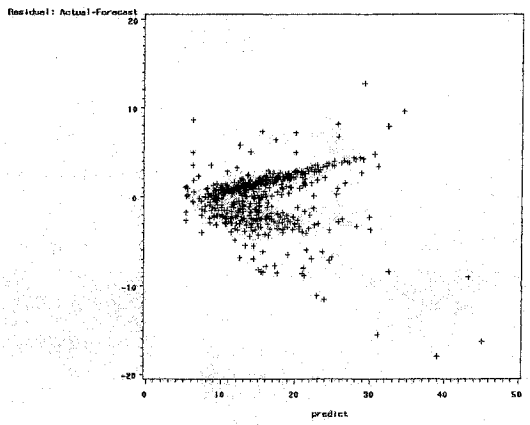

Figure 12: Residuals of Window Size for λ=0.75, p=1, q=0. Here the model is not acceptable since the residuals are not very independent from the forecast series of window size, with some pattern.

2. plot the residuals vs time, see Figure 13.

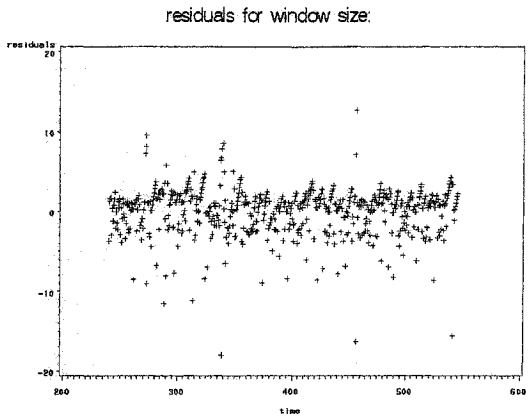

Figure 13: Residual time series of Window Size for λ=0.75, p=1, q=0. Here the residuals are not quite randomly distributed in time.

3. Histogram Plot of the Residuals

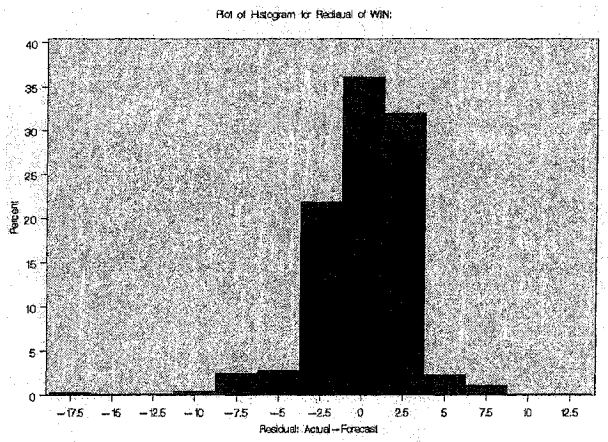

Figure 14: The Normal Probability Plot of Residuals for Window Size. This figure shows the histogram of the residuals are not fairly symmetrically distributed.

4. Normal Probability Plot of the Residuals

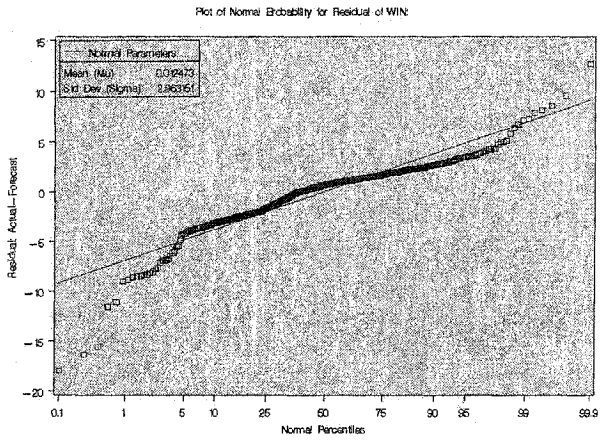

Figure 15: The Normal Probability Plot of Residuals for Window Size. It shows heavier tails than expected in a normal distribution.

E. Modelling Decision

After overall evaluation of the normality of residuals, the independence between residuals and forecasted window size, the rank of the orders of p and q, we choose the ARMA model with λ as 0.75, p_0 as 1, and q_0 as 0 for fitting the data of window size. The coefficients are shown in Table 6 and the corresponding model equation given below:

$$(1 - 0.7B)W_t = 8 + a_t$$

Chosen model for window size

p0	q0	AIC	lambda	PrdErr	avepe	sym
1	0	3087.46	0.75	36.7596	36.7596	1

Parameter	Estimate	Pr > |t|	Lag	Variable
MU	8.07364	<.0001	0	swin
AR1,1	0.67365	<.0001	1	swin

Table 6: Chosen model for window size data

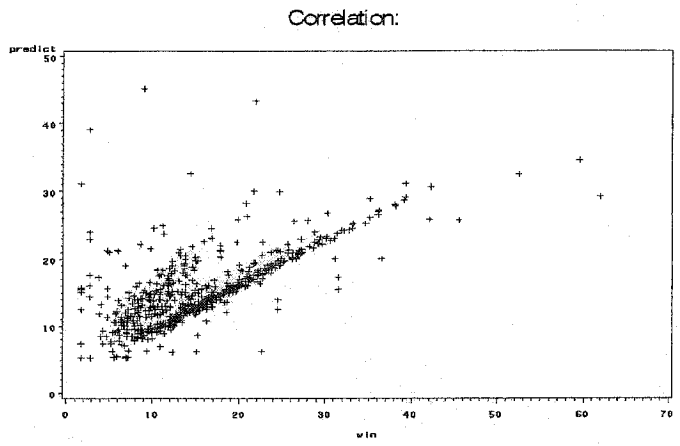

Figure 16: The Scatter Plot of Window Size and Predicted Window Size shows the rough linear correlation. This is because there are peaks between 272 and 273 second, 456.8 second and 457.3 second, 546.7 and 546.88 second. Actually it is a poor fit and more complex models needed.

Although Figure 16 shows some success at predicting Window Size based on an ARMA(1,0) model, that figure and other plots of residuals clearly show two-phase behaviors, which will be a part of our future works. Using higher orders of p and q did not change this observation. More complex non-linear models (e.g. Threshold autoregressive models) may be able to improve the model considerably but are not pursued here. To sum up, this is not a complete model, but appears to be the most appropriate choice within the class of ARMA(p,q) models.

5.3 Fitting ARMA to The Data of RTT

5.3.1 General Procedures

The procedure of fitting ARMA to RTT data is similar to that of fitting ARMA to Window Size data. Moreover, the effect of fitting RTT is much better than fitting WIN data.

5.3.2 Algorithm Implementation

A. Data preparation

1. Run Opnet program to get the simulation data;
   Figure 4 shows the output of Window Size and RTT, which lasts 17 minutes and includes burn-in phase.

2. Dump the output figure into a spreadsheet, remove burn-in phase and import the excel data into SAS data set: rtt_raw, see Figure 17. Figure 18 shows RTT data corresponding to the period from 4 minutes to 9 minutes in Figure 17 and provides 24371 records.

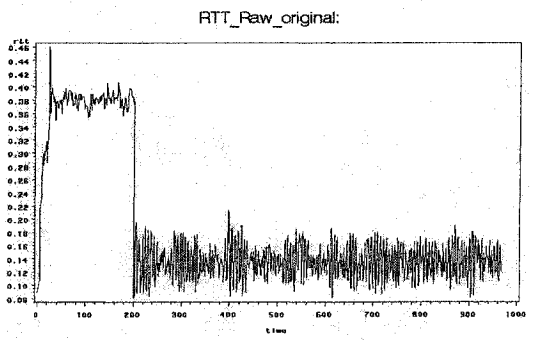

Figure 17: The original raw RTT data input from spreadsheet, covering 0 to 17 minutes.

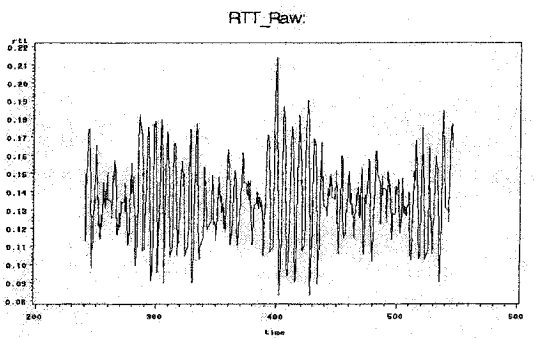

Figure 18: The dumped raw RTT Data has been imported into SAS data set, covering 4 to 9 minutes (546 seconds). There are approximately 24371 records without burn-in phase and explosive right tail.

3. Get the improved data set by discretizing time and averaging data. Since the data set rtt_raw contains 24371 records at unequally time points, we now discretize time to a constant increment of 0.5 seconds and average the multiple observations in each time bin. Then we obtain 613 records in data set of rtt_impr. There are gaps in this series corresponding to longer stretches where no RTT data were recorded. The data are shown in above Figure 19.

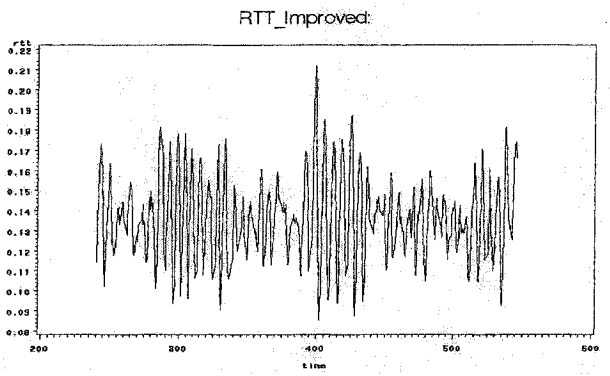

Figure 19: Improved RTT data with replacing multiple data of same time with the average.

4. Fill the gaps to get the data set of rtt_interp. The rtt_impr data set contains time series data at equally-spaced times but with gaps. We piecewise-constantly interpolate this data to produce a time series with equally-spaced observations. There are 613 observations with interval of 0.5 seconds in the data set of rtt_interp, see Figure 20.

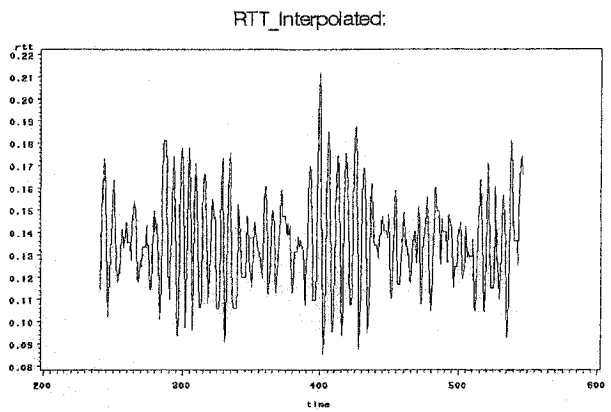

Figure 20: The RTT data after piecewise constant interpolation, with size of 613. After interpolation, we will use the Box-Cox transformation to remove the feature of any non-stationarity at the next step.

B. Model Selection

1. Get transformed data set, rtt_interp, by transforming data set of rtt_interp into data set of rtt_sas by the Box-Cox Transformation. We provide an example when RTT series is transformed using λ=0.6, see Figure 21.

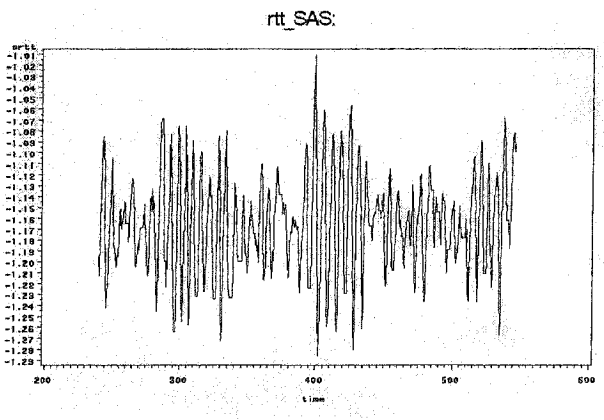

Figure 21: Transformed Data of RTT with λ=0.6.

The method we use is fitting an ARMA(10,10) model for different λ and plot the PredErr against λ. Loop: λ from 0.2 to 0.9 step 0.025

(a) Run PROC ARIMA(p=10,q=10) on different transformed data: rtt_sas.

(b) get inverse value of y, by

$$y^{-1} = InverseY = \begin{cases} (1+\lambda ForecastY)^{\frac{1}{\lambda}}, & \lambda \neq 0 \\ e^{ForecastY}, & \lambda = 0 \end{cases}$$

$$i.e. InverseRTT\_RTT = \begin{cases} (1+\lambda RTT)^{\frac{1}{\lambda}}, & \lambda \neq 0 \\ e^{WIn}, & \lambda = 0 \end{cases}$$

(c) add predict error under each p,q and transformation. The result is in Table 7. $prediction\ error = \frac{\sum (inverse-original)^2}{N}$

Obs	lambda	PrdErr
6	1.500	.000044447
7	0.850	.000044838
8	0.575	.000045067
9	0.550	.000045068
10	0.600	.000045070
11	0.525	.000045071
12	0.625	.000045074
13	0.500	.000045080
14	0.650	.000045081
15	0.675	.000045091
16	0.475	.000045092

Table 7: Part of The Table of λ and PrdErr for Transformed RTT, sorted by PrdErr.

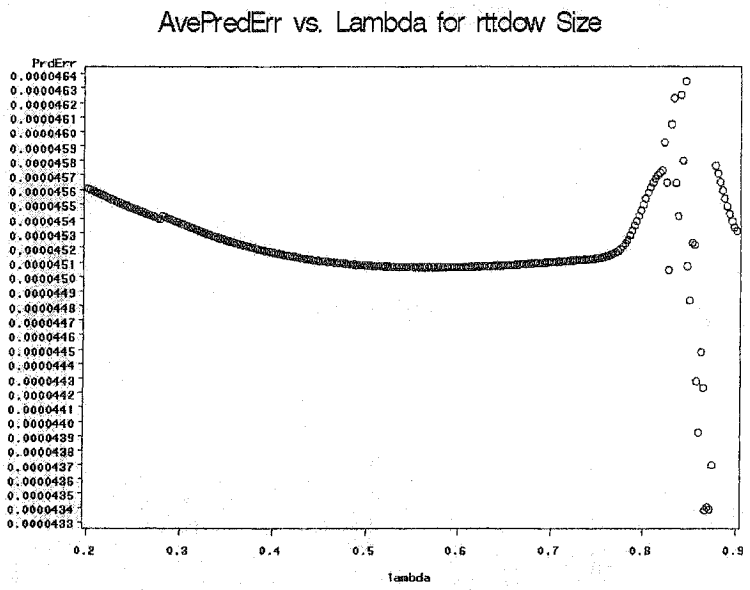

Figure 22: The Plot of Average Predicted Error against λ for ARMA(10,10). We choose λ=0.6 as the best value in the stable region. For $\lambda > 0.8$, there are many regions of instability.

From the plot in Figure 22 and Table 7, we choose $\lambda_0 = 0.6$, which is close to the bottom of the graph.

2. For $\lambda_0 = 0.6$, we now try to find a proper model by AIC and rank of p+q.

loop: i from 0 to max_N
loop: j from 0 to max_N

(a) run ARIMA(p=i,q=j) and get its AIC.

(b) For λ_0, we obtain a list of pairs of p, q and AIC. This is shown in table 9.

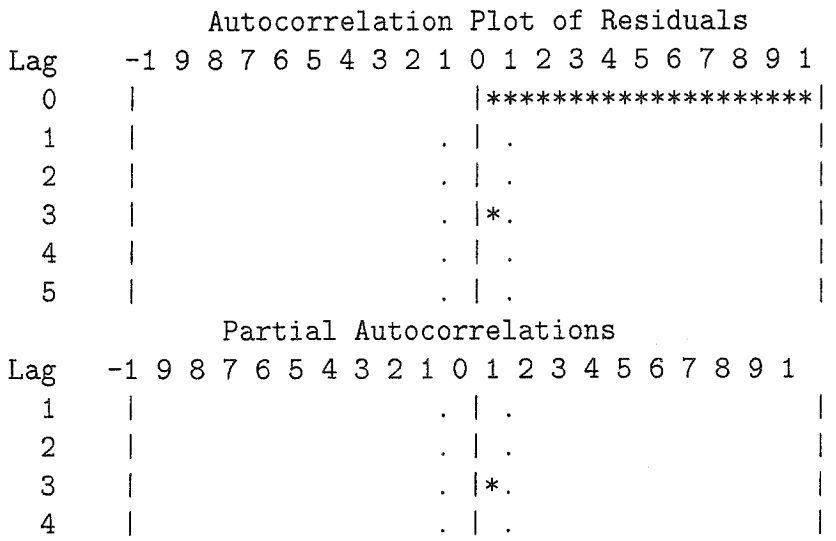

Table 8: ACF and PACF show the stationarity of rtt_sas with lambda=0.6 and the model will be a mixed ARMA,according to Section 3.2.3 C.

Obs	p0	q0	AIC	order(p+q)
1	3	3	-3383.83	6
2	6	2	-3383.13	8
3	7	6	-3382.36	13
4	8	2	-3382.32	10
5	4	3	-3382.25	7

Table 9: Part of The Table of p, q and AIC sorted by AIC, for $\lambda = 0.6$. We choose p=3 and q=3 for ARMA model because both AIC and order are small.

3. rerun ARIMA(3, 3) for $\lambda = 0.6$ with the smallest AIC and obtain the data sets testest, testmod, teststat and predict. We will examine whether the diagnostic tests are fulfilled for this choice of model.

C. Model Checking

1. plot the residuals vs the forecast RTT

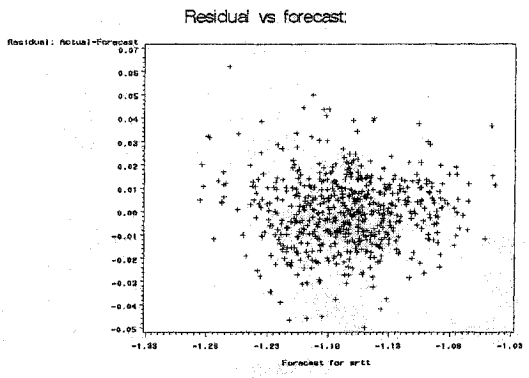

Figure 23: Residuals for Modelling RTT for λ=0.6, p=3, q=3. The model is acceptable since the residuals are independent from the forecast RTT without any pattern.

2. plot the residuals vs time

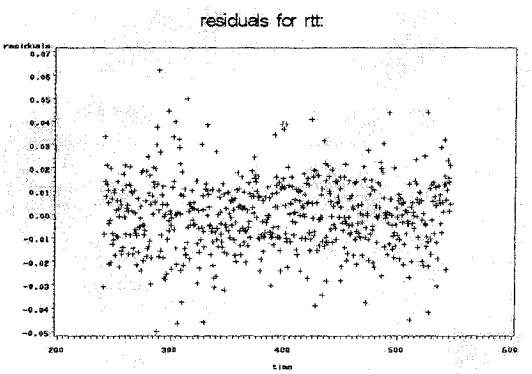

Figure 24: Residuals for RTT for λ=0.6, p=3, q=3. The model is acceptable since the residuals are randomly and normally distributed.

3. Normal Probability Plot of Residuals

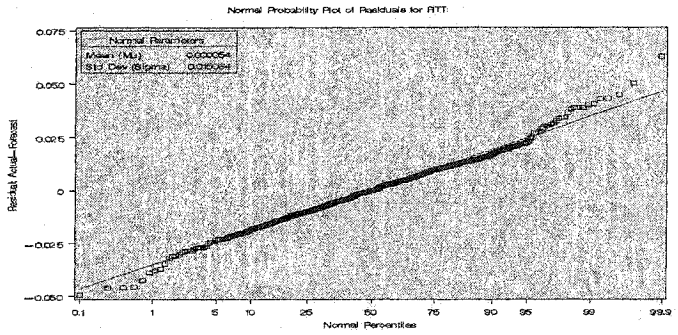

Figure 25: The Normal Probability Plot of Residuals for transformed RTT. The residuals form an alomost straight line in normal probability plot. Then we can be confident that they have a normal distribution.

4. Plot the Histogram of Residuals

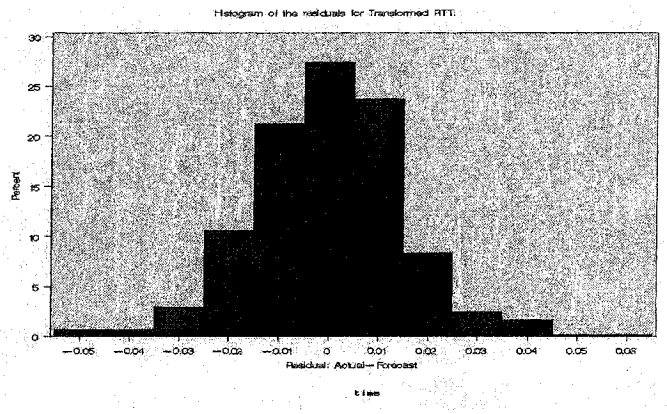

Figure 26: The Normal Probability Plot of Residuals for transformed RTT shows that histogram of the residuals is normally distributed. Again, the shapes of these graphs do not contradict the desired normality of residuals.

D.Modelling Decision

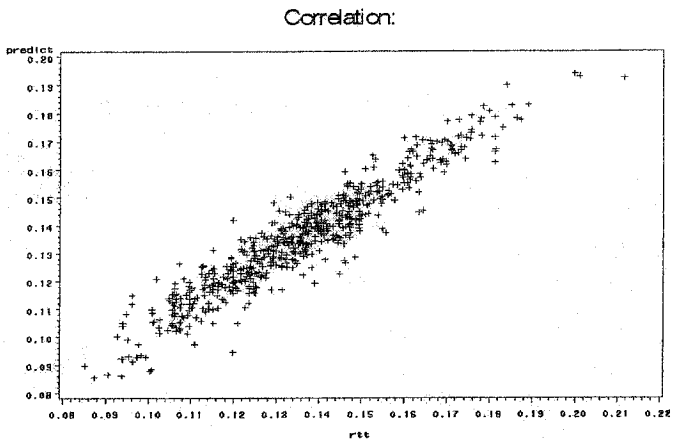

Figure 27: The Scatter Plot of RTT and Predicted RTT shows the strong linear correlation, which indicates the appropriate model ARMA(3,3) fit for RTT.

p0	q0	lambda	AIC	AVEPrdErr
3	3	0.6	-3383.83	.000045070

Parameters	Estimates	Pr>|T|
MU	-1.16525	<.0001
MA1,1	1.24671	<.0001
MA1,2	-0.13219	0.0781
MA1,3	-0.25893	<.0001
AR1,1	2.51878	<.0001
AR1,2	-2.32278	<.0001
AR1,3	0.75912	<.0001

Table 10: Chosen Model for RTT data

After overall evaluation of the normality of residuals, the independence between residuals and forecasted RTT, the rank of the orders of p and q, we choose ARMA model with λ as 0.6, p_0 as 3, and q_0 as 3 for fitting RTT data. The coefficients are shown in Table 10 and the corresponding model equation given below:

$$(1 - 2.5B^1 + 2.3B^2 - 0.8B^3)R_t = -1.2 + (1 - 1.2B^1 + 0.1B^2 + 0.3B^3)a_t$$

5.4 Fitting VARMA Model to The Bivariate Data of Window Size and RTT

The step of Data Preparation to process the source data is the same as before and the figures listed before. Figure 2 shows the procedure we will use. After merging the original data sets of window size and RTT, deleting the redundant data of the same time, we use time-interpolation and Box-Cox transformation on the bivariate data. We would like to use VARMA with the same λ 's obtained as in the previous section, but the optimization process of VARMA procedure in the SAS did not converge in those cases. Here we use $\lambda_{WIN} = 0.65$, and $\lambda_{RTT} = 0.5$ to illustrate the method since we did obtain convergence in this case. By running VARMAX procedure to get a list of AIC with different values of p and q and comparing among the values of AIC, we make a decision of the best model with the value of p and q, according to the minimum AIC.

5.4.1 VARMAX fails for mixed models

Our original intention was to fit VARMA models to the bivariate (WIN, RTT) series. Our experience with PROC VARMAX in SAS was disappointing; only a restricted collection of models were feasible as the procedures failed to converge for choices other than those in table 11.

Obs	AIC	p	q
1	-4.843	1	0
2	-4.918	1	1
3	-5.363	2	1

Table 11: Information of AIC with different orders of p and q indicates that AIC is a criterion to modelling the simulation series. This method used on a simulated bivariate series is introduced at Appendix 7.2.2. The orders of p=2 and q=1 is the highest that we can successfully run PROC VARMAX . So we will further test the fit of VARMA(2,1).

5.4.2 Model Checking

1. Plot Normal Probability of Residual for transformed WIN of (WIN,RTT).

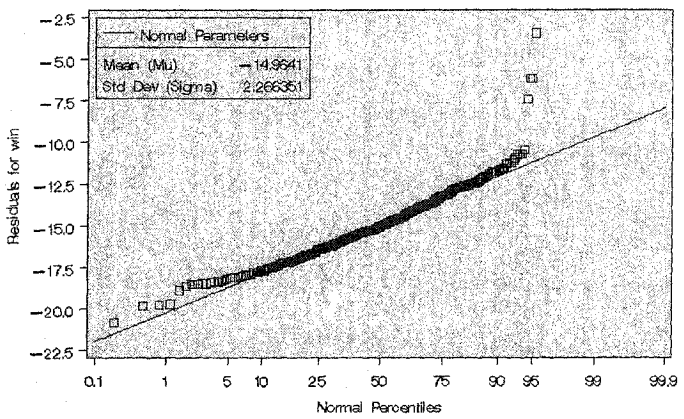

Figure 28: Test Normal Distribution of Residual Series for Transformed Window Size of the bivariate series shows its slight normal distribution. Here Window Size data is under Box-Cox transformation with lambda=0.65.

2. Plot Normal Probability of Residual for transformed RTT of (WIN,RTT).

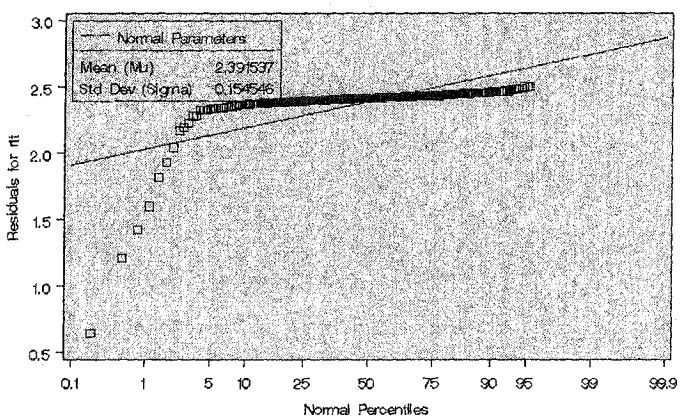

Figure 29: Test Normal Distribution of Residual Series for Transformed RTT of the bivariate series shows that it is not normally distributed. Here RTT data is under Box-Cox transformation with lambda=0.5.

3. Plot the Correlation between WIN and Predicted WIN of (WIN,RTT).

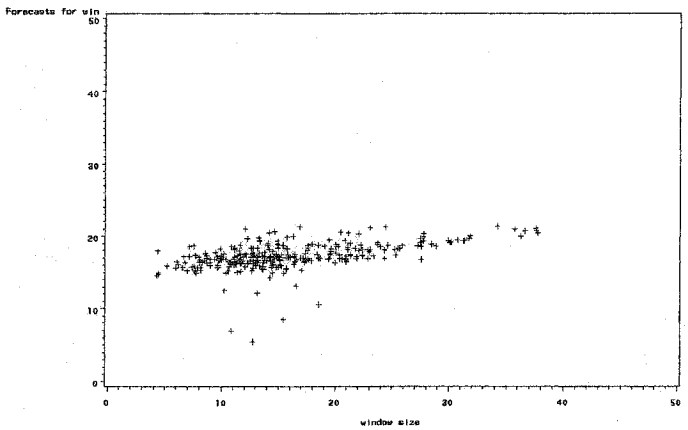

Figure 30: The Correlation between Original Window Size and Predicted Window Size which indicates it does not fit.

4. Plot the Correlation between RTT and Predicted RTT of (WIN,RTT).

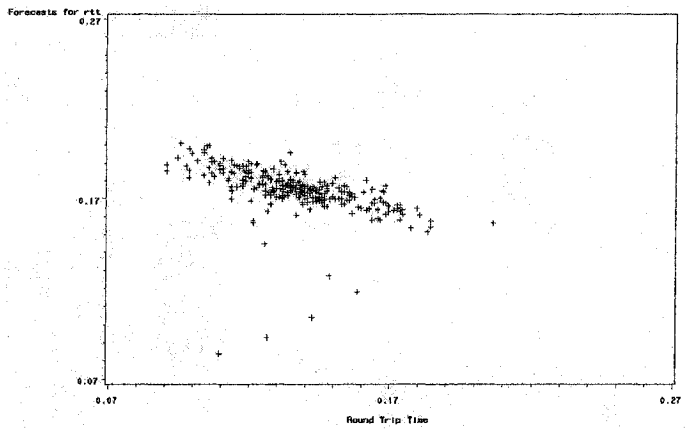

Figure 31: The Correlation between Original RTT and Predicted RTT which indicates it does not fit.

5.4.3 Model Decision

We find that it is hard to get mixed VARMA model for the bivariate here because of failures of the VARMAX procedure under SAS 8.2. Consequently we cannot pursue

VARMAX models further for $p > 0$ and $q > 0$. However, VARMA(p,0) (i.e. VAR(p)) were consistently attainable in SAS 8.2 and we look at these now for Bivariate Win data and RTT data.

5.5 Fitting VAR Models to Window Size and RTT

Figure 2 shows the procedure we will use. After merging the original data sets of window size and RTT, deleting the redundant data of the same time, we use time-interpolation and Box-Cox transformation on the bivariate data with the relative λ's obtained in the previous section(λ_{WIN}= 0.75 ,λ_{RTT} = 0.6). By running VARMAX procedure to get a list of AIC with different values of p and comparing among the values of AIC, we make a decision of the best model with the value of p, according to the minimum AIC.

5.5.1 Model Selection

Run VARMAX procedure with specifying different p to obtain AICs for VAR. The result is listed below.

Obs	AIC	VAR(p)	AvePredErr	
			Win	RTT
1	-5.24273	1	36.3722	.000119665
2	-6.07788	2	36.2502	.000050487
3	-6.11185	3	36.1667	.000048518
4	-6.17074	4	36.1759	.000045167
5	-6.16494	5	36.0073	.000045000
6	-6.15408	6	36.0687	.000044798
7	-6.14091	7	36.0886	.000044763
8	-6.12838	8	36.0922	.000044726
9	-6.12442	9	36.1124	.000044343
10	-6.11772	10	35.9726	.000044180

Table 12: Information of AIC with different orders of p. The orders of p=4 generate the minimum AIC. So we will further test the effectiveness of VARMA(4,0).

5.5.2 Model Checking

1. plot the residuals vs the predicted Win and RTT.

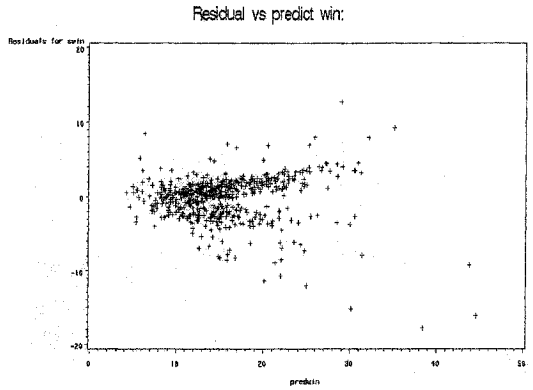

Figure 32: Residuals for transformed Win vs. forecast Win under VAR Modelling with p=4, q=0. The model to predict Win data is not satisfactory since the residuals are not very independent from the forecast WIN.

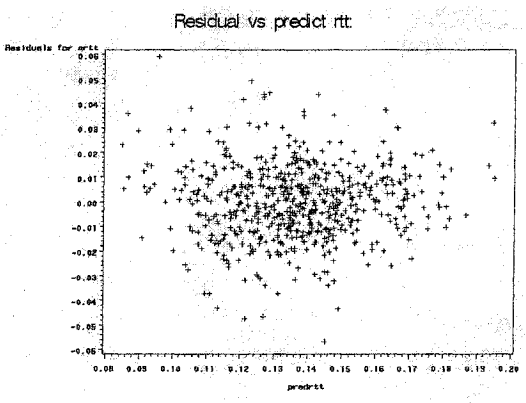

Figure 33: Residuals for transformed RTT vs forecast RTT under VAR Modelling with p=4, q=0. The model to predict RTT data is acceptable since the residuals are independent from the forecast RTT without any pattern.

2. plot the residuals vs time

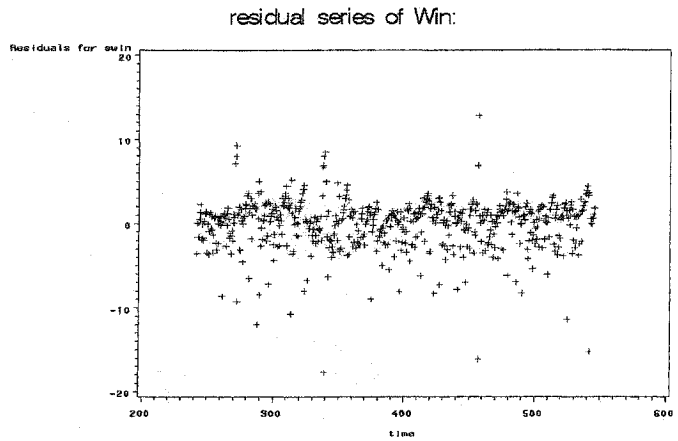

Figure 34: Residuals for Win under VAR modelling with p=4, q=0. The model is not satisfactory since the residuals are not randomly and normally distributed.

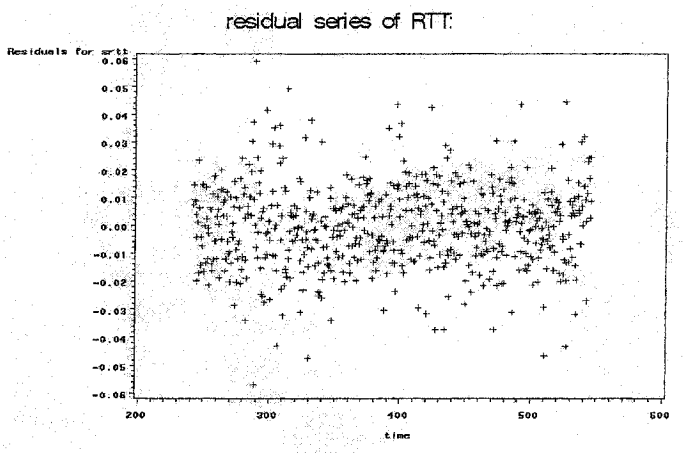

Figure 35: Residuals for RTT under VAR modelling with p=4, q=0. The model is acceptable since the residuals are randomly and normally distributed.

3. Normal Probability Plot of Residuals

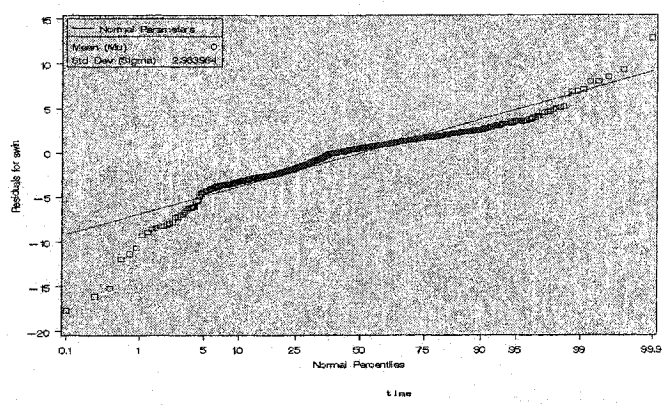

Figure 36: The Normal Probability Plot of Residuals for Transformed Win. It shows heavier tails than expected in a normal distribution.

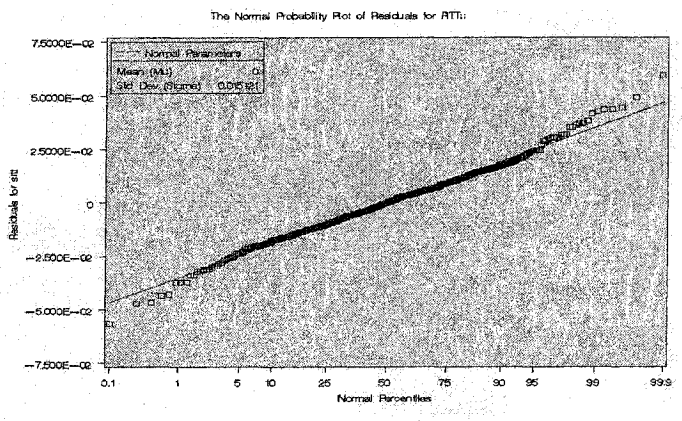

Figure 37: The Normal Probability Plot of Residuals for RTT: Since the residuals from the fitted model from a straight line in normal probability plot, we are confident that they have a normal distribution.

4. Plot the Histogram of Residuals

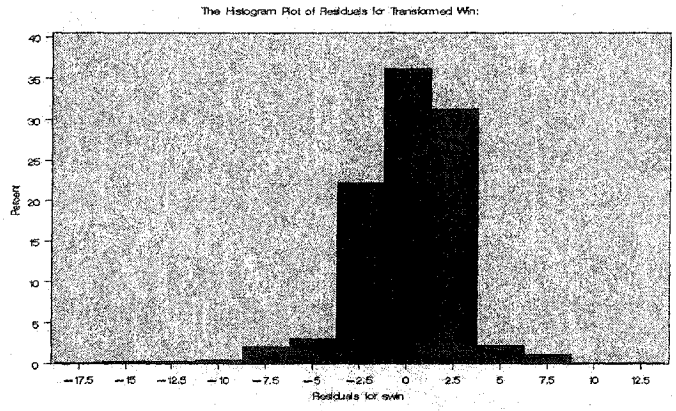

Figure 38: The Histogram of Residuals for Transformed Win shows that histogram of the residuals are not normally distributed.

5. Plot the Histogram of Residuals

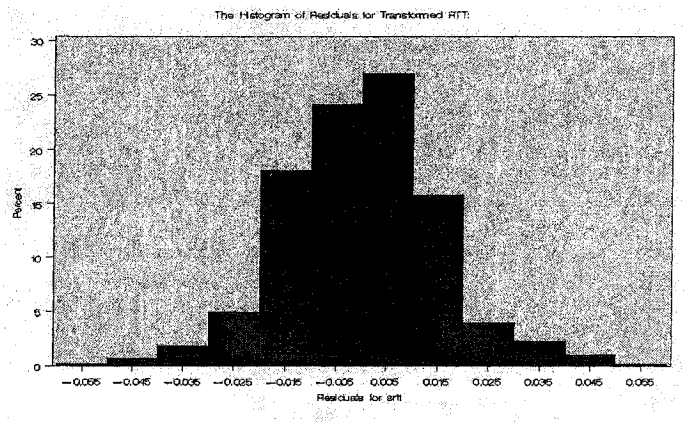

Figure 39: The Histogram of Residuals for Transformed RTT shows that histogram of the residuals are normally distributed.

From all the above figures, we can tell the appropriate behaviors of the residuals.

5.5.3 Model Decision

We pick VARMA(4,0) among models, based on the least AIC. The estimated parameters for the model we have got from the program are very close to the originals of the simulation series.

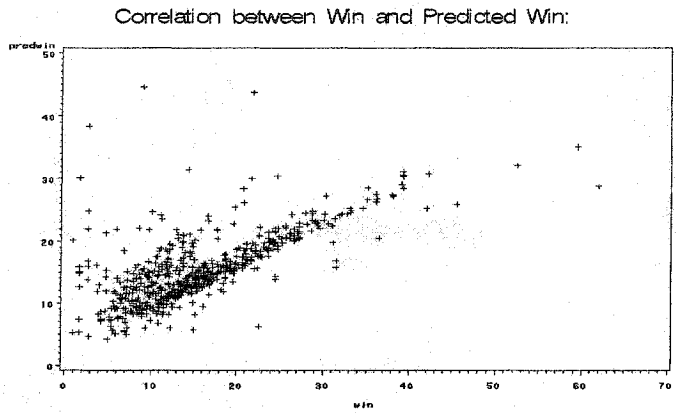

Figure 40: The Correlation between Original Window Size and Predicted Window Size which indicates VAR(4) the data does not fit well. Actually it is a poor fit and more complex models needed.

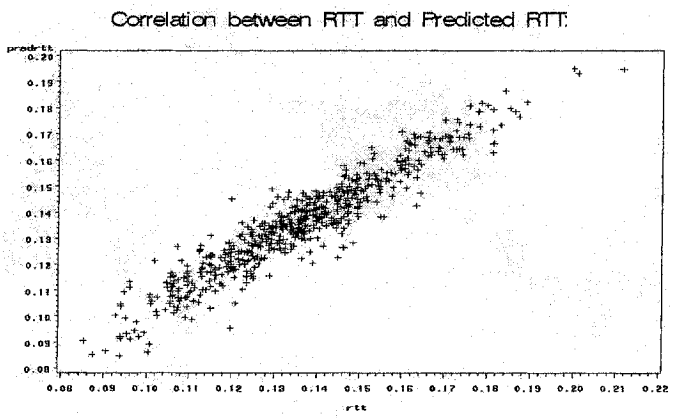

Figure 41: The Correlation between Original RTT and Predicted RTT which indicates VAR(4) fits well

Chosen Model Parameter Estimates

Equation	Parameter	Estimate	Prob>|T|	Variable
TransWin(t)	CONST1	9.20950	0.0669	1
	AR1_1_1	0.67104	0.0001	swin(t-1)
	AR1_1_2	16.64850	0.0331	srtt(t-1)
	AR2_1_1	0.02220	0.6555	swin(t-2)
	AR2_1_2	-20.08785	0.1221	srtt(t-2)
	AR3_1_1	-0.01619	0.7450	swin(t-3)
	AR3_1_2	6.32306	0.6254	srtt(t-3)
	AR4_1_1	-0.03146	0.4443	swin(t-4)
	AR4_1_2	2.30637	0.7671	srtt(t-4)
TranRtt(t)	CONST2	-0.46440	0.0001	1
	AR1_2_1	-0.00016	0.4357	swin(t-1)
	AR1_2_2	1.30320	0.0001	srtt(t-1)
	AR2_2_1	-0.00002	0.9254	swin(t-2)
	AR2_2_2	-0.60321	0.0001	srtt(t-2)
	AR3_2_1	-0.00018	0.4663	swin(t-3)
	AR3_2_2	0.16192	0.0145	srtt(t-3)
	AR4_2_1	0.00034	0.1028	swin(t-4)
	AR4_2_2	-0.26069	0.0001	srtt(t-4)

Table 13: Chosen VAR(4) Model for Transformed Bivariate Series

That is:

$$\begin{bmatrix} win_t \\ rtt_t \end{bmatrix} = \begin{bmatrix} 9.2095 \\ -0.4644 \end{bmatrix} + \begin{bmatrix} 0.67104 & 16.6485 \\ -0.00016 & 1.3032 \end{bmatrix} \begin{bmatrix} win_{t-1} \\ rtt_{t-1} \end{bmatrix} + \begin{bmatrix} 0.0222 & -20.08785 \\ -0.00002 & -0.60321 \end{bmatrix}$$

$$\begin{bmatrix} win_{t-2} \\ rtt_{t-2} \end{bmatrix} + \begin{bmatrix} -0.01619 & 6.32306 \\ -0.00018 & 0.16192 \end{bmatrix} \begin{bmatrix} win_{t-3} \\ rtt_{t-3} \end{bmatrix} + \begin{bmatrix} -0.03146 & 2.30637 \\ 0.00034 & -0.26069 \end{bmatrix} \begin{bmatrix} win_{t-4} \\ rtt_{t-4} \end{bmatrix}$$

The whole modelling process in the application would be as following: Transform two series: Window Size data by λ=0.75 and RTT data by λ=0.6; use the above method to get VAR model; retrieve both of the series then we can get the predicted series for Window Size and RTT. Based on the p_values above, we could say that RTT is more

likely a series of itself, while WIN is more likely a bi-variate series of the latest WIN and latest RTT.

5.6 Comparison of ARMA and VAR Model Fitting

Now we have three models for Win data and RTT data: ARMA(1,0) for Win data with λ=0.75, ARMA(3,3) for RTT data with λ=0.6, and VARMA(4,0) for bivariate transformed Win and RTT under the above transformations. Since modelling Window Size and RTT is our interest. We compare the average predicted error for Window Size and RTT in Table 14 and Table 15.

	ARMA(1,0)_Win	ARMA(3,3)_RTT	VAR(4)_Win&RTT	
No.parameters	2	7	18	
			WIN	RTT
AvePredError	36.7596	.00004507	36.1759	.00004517
lambda	0.75	0.6	0.75	0.6

Table 14: General Comparison of ARMA and VAR Model Fitting.

	Univariate Model	Bivariate Model
Window Size	36.7596	36.1759
Round Trip Time	.00004507	.00004517

Table 15: Comparison of Average Predicted Error for Window Size and RTT under both models.

We can also observe the same phenomenon by comparing Figure 16 to Figure 40, and Figure 27 to Figure 41. It is very slightly better to fit RTT by the univariate ARMA(3,3) model than by VAR(4) model, while it is better to fit Window Size by the VAR(4) than by AR(1). This suggests that when we model Window Size, modelling is more successful if we take RTT into account, although neither successfully explains the two-phase nature of the WIN data. On the other hand, when we evaluate RTT, Window Size does not help in predicting RTT. We can draw a conclusion that there exists a relationship between Window Size and RTT, but this may be strengthened if we can truly fit VARMA instead of just VAR models, or if we can truly fit the WIN data with a suitable model. Although the VAR(4) model for RTT did slightly worse than the ARMA(3,3) model, it does only use 4 parameters instead of 6.

5.7 Further Discussion about WIN Data

Figure 42 and Figure 43 shows a very brief section (0.5 minutes) of the WIN data. It shows how the Window Size follows a two or three-phase behavior. From a low value, the Window Size increases at least linearly (sometimes geometrically) until it hits a plateau (that may change over time, and then fluctuates about this plateau until cut in half or set back almost to zero. This reflects the fast start, increase and halving algorithms built into TCP congestion control. Since the series exhibits two or three distinctly different time series behaviors in sequence, it will probably be necessary to model the WIN data by a 3-phase threshold ARMA model in order to achieve

adequate predictive ability.

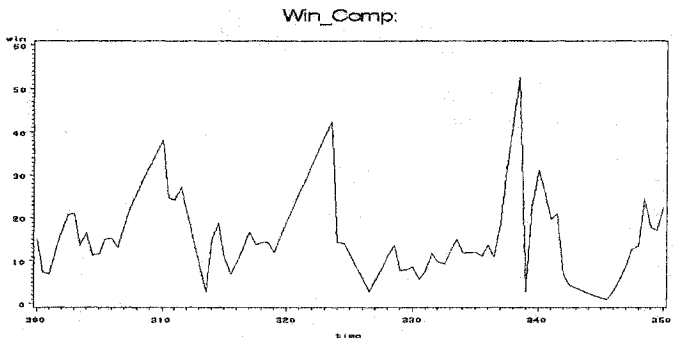

Figure 42: The WIN behaves cut-half which contradicts properties of time series.

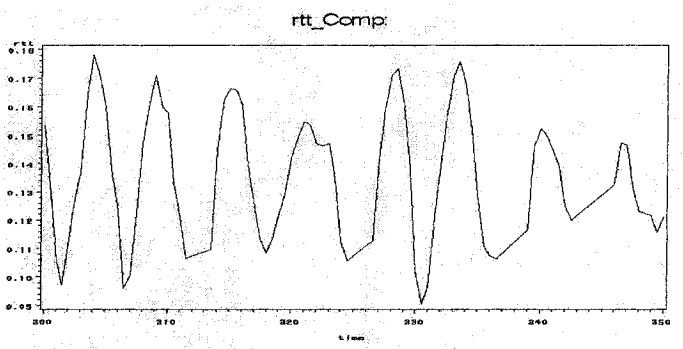

Figure 43: The RTT graph has characteristics which upholds the properties of time series

Chapter 6

Conclusions and Future Directions

In order to help TCP control the packet process such as window size for the source according to the congestion status of the network, especially RTT, we give a newly statistical measure to improve the congestion algorithm. Using both univariate and bivariate time series methodologies , we found that WIN prediction helped a little by VAR(4) model. RTT prediction was affected a little by VAR(4).

Although it is not suitable for WIN to be modelled as time series, it is still helpful to predict WIN data by taking RTT into account. For TCP/IP, RTT is more important than WIN data. We found that WIN data will affect the accurateness of RTT. Through modelling and analysis, we justified that the model of ARMA(1,0) with transformation parameter of λ=0.75 does not well fit Window Size data, the model of ARMA(3,3) with transformation parameter of λ=0.6 fits RTT data very well, and the model of VARMA(4,0) with the above transformation parameters improves Window Size prediction instead of RTT.

In the future, we will evaluate the degree of the efficiencies of the above modelling methods to improve TCP RTT calculation introduced at section 2.2.2 and to forecast TCP congestion. We will put the modelling method into use in the different circumstances, such as different nodes in same network, different loads in same network even different network. Our goal would be that how big should p plus q be to work in a

broad class of cases.

Chapter 7

Appendix

7.1 Modelling Method for the Simulation Series

7.1.1 ARMA Method for the Simulation Series

In order to test both the ARIMA procedure and the information criteria, we generate a simulation time series with data size of 1000, ARMA(3,2) for example:

$$u_t = .4 * u_{t-1} + .1 * u_{t-2} - .5 * u_{t-3} + a + .3 * a_{t-1} + .5 * a_{t-2}$$

Running Proc ARIMA under different pairs of p and q. We get ACF and PACF.

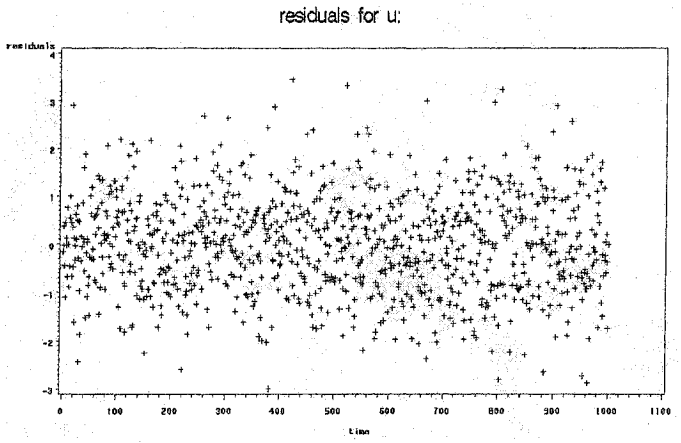

Figure 44: Simulated Data

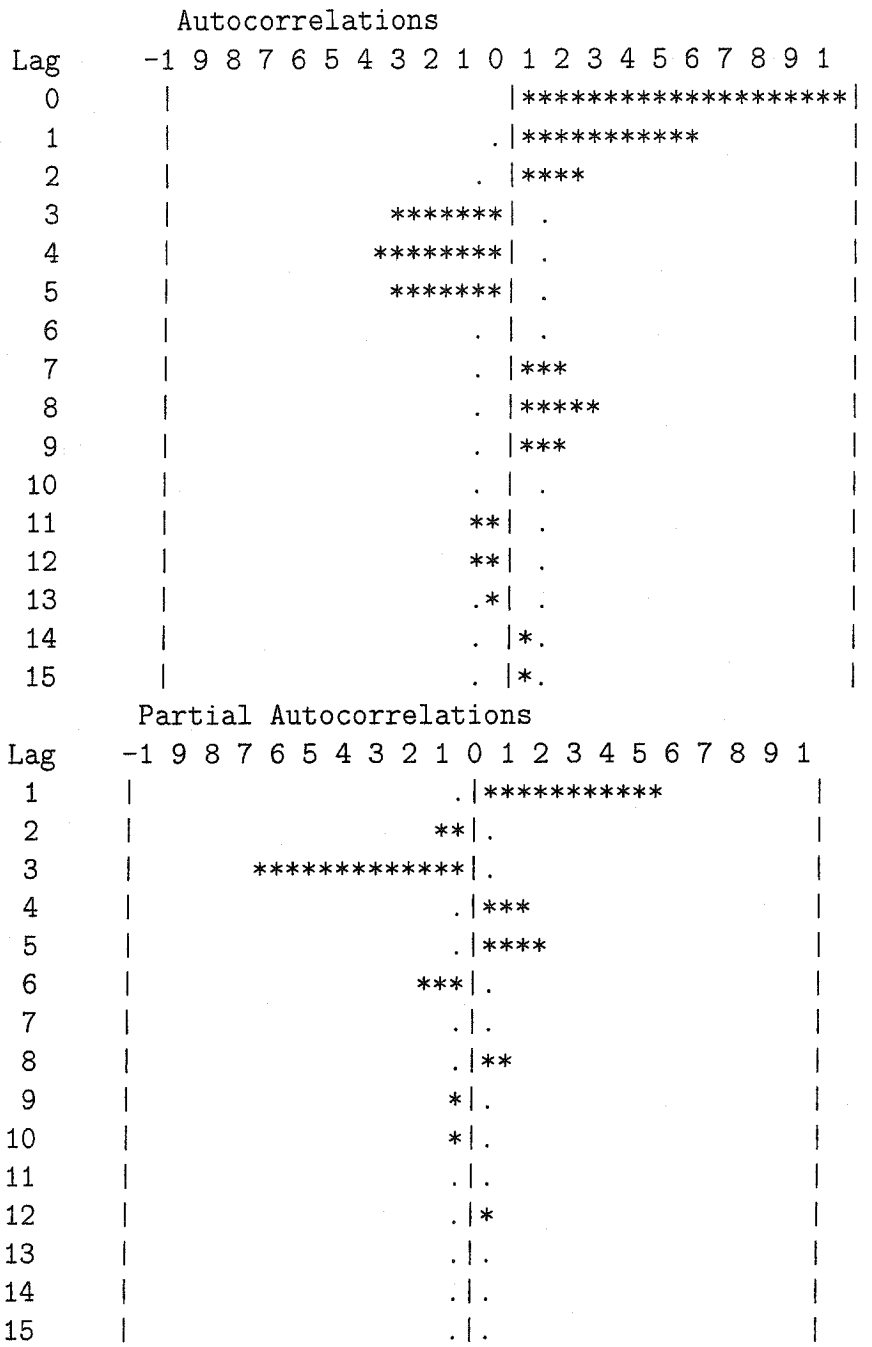

Table 16: ACF and PACF of Simulation Series. The model will be not an AR model since ACF is not tail-off(see Section 3.2.3C). Here it could not tell us the information about the orders of p=3 and q=2 for Simulated ARMA(3,2). We will look into AIC to find the model.

After running PROC ARIMA, we get information including p, q, order and AIC.

Obs	AIC	p	q	order	AIC	Rank
1	2867.31	5	4	9	2867.31	3
2	2869.08	6	4	10	2869.08	3
3	2870.94	6	5	11	2870.94	3
4	2870.95	7	4	11	2870.95	3
5	2871.07	10	9	19	2871.07	5
6	2871.72	8	7	15	2871.72	4
7	2872.61	10	2	12	2872.61	4
8	2872.74	7	5	12	2872.74	4
9	2873.22	4	2	6	2873.22	2
10	2873.71	9	2	11	2873.71	3
11	2873.90	3	2	5	2873.90	2
12	2874.25	9	3	12	2874.25	4
13	2874.25	9	4	13	2874.25	4
14	2874.25	9	5	14	2874.25	4
15	2874.25	9	6	15	2874.25	4

Table 17: Part of The Table of p, q, order and AIC sorted by AIC. We choose p=3 and q=2 because of smaller order, smaller AIC here and better estimators in Table 18.

The residual series is apparently independent from the forecast series indicating that the model fitted to the observation series.

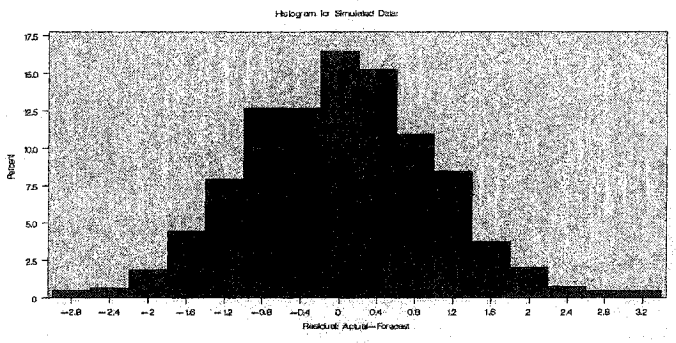

Figure 45: The Histogram of Residual Series for Simulated Series

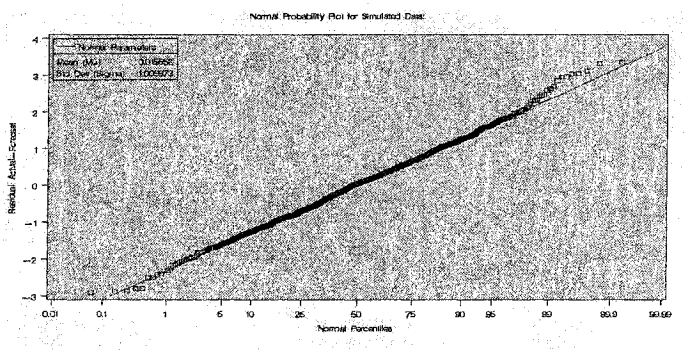

Figure 46: The Plot of the Residual Series for Simulated Series

By using the ARIMA procedure and its relative statements with different parameters of p and q, we find that the plots of ACF and PACF do not give us a clear vision for determining the orders of the model.

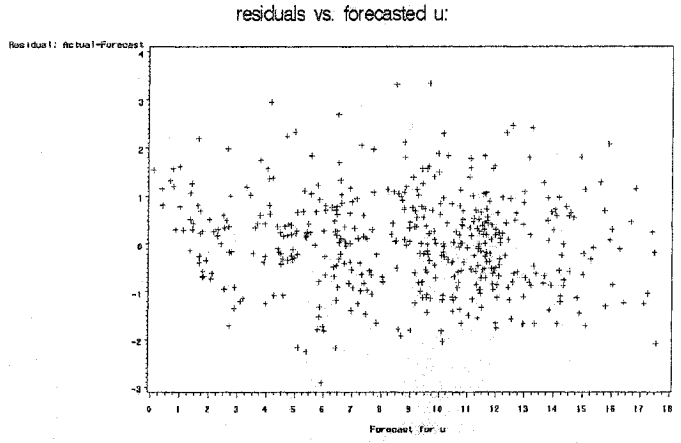

Figure 47: Residual vs forecast simulation Data

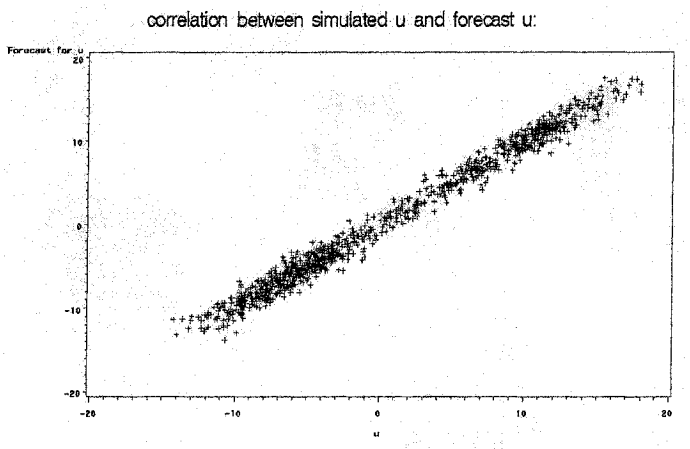

Figure 48: The Correlation between Simulated Data and Forecast Data indicates the model fits well.

To sum up, we say the ARIMA procedure, the order of p and q, AIC and ACF/PACF are useful for us to fit the model to the source series:

$$(1 - 0.34714B - 0.14213B^2 + 0.508B^3)U_t = (1 + 0.29124B + 0.43063B^2)a_t$$

Parameter	Estimate	Pr > |t|
MU	-0.0019328	0.9716
MA1,1	-0.29124	<.0001
MA1,2	-0.43063	<.0001
AR1,1	0.34714	<.0001
AR1,2	0.14213	0.0004
AR1,3	-0.50800	<.0001

Table 18: The Parameters of ARMA Model Fitting the Simulation Series

7.1.2 VARMA Method for the Simulation Series

A. Obtain Simulation Bivariate Series

In order to explore the performance of both the VARMAX procedure and the relative information criteria, we generated a simulated time series with data size of 1000, VARMA(1,1), and applied these methods to this ideal case.

$$\begin{bmatrix} y_{t1} \\ y_{t2} \end{bmatrix} = \begin{bmatrix} 1.2 & -0.5 \\ 0.6 & 0.3 \end{bmatrix} \begin{bmatrix} y_{t-1,1} \\ y_{t-1,2} \end{bmatrix} + \begin{bmatrix} a_{t1} \\ a_{t2} \end{bmatrix} - \begin{bmatrix} -0.1 & 0.3 \\ 0.2 & 0.4 \end{bmatrix} \begin{bmatrix} a_{t-1,1} \\ a_{t-1,2} \end{bmatrix}$$

B. Run VARMAX procedure with different pairs of p and q to obtain AICs

Obs	AIC	p	q	PrdError
1	2.070351	0	1	6.82291
2	1.26063	0	2	3.76088
3	1.26063	0	3	3.76088
4	0.206484	1	0	2.20688
5	0.017037	1	1	2.00194
6	0.023692	1	2	2.00056
7	0.059505	2	0	2.04145
8	0.023698	2	1	2.00119

Table 19: AIC and Predicted Error with different orders of p and q indicates that AIC is a useful criteria in modelling the simulation series.

C. Model Checking

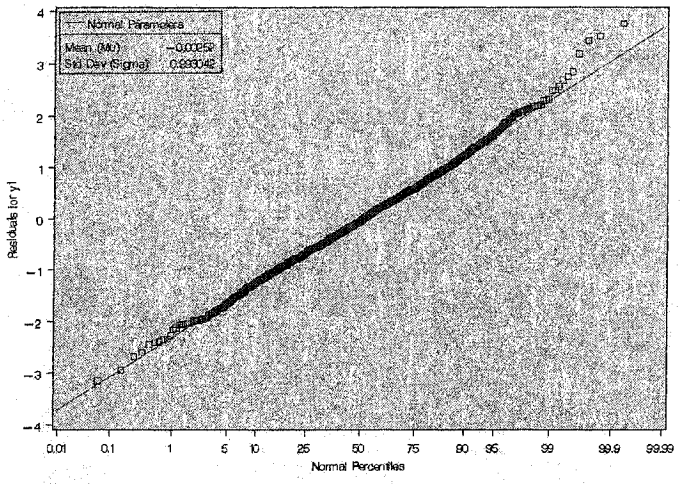

Figure 49: Normal Probability Plot of Residual Series of Simulated Series Y1

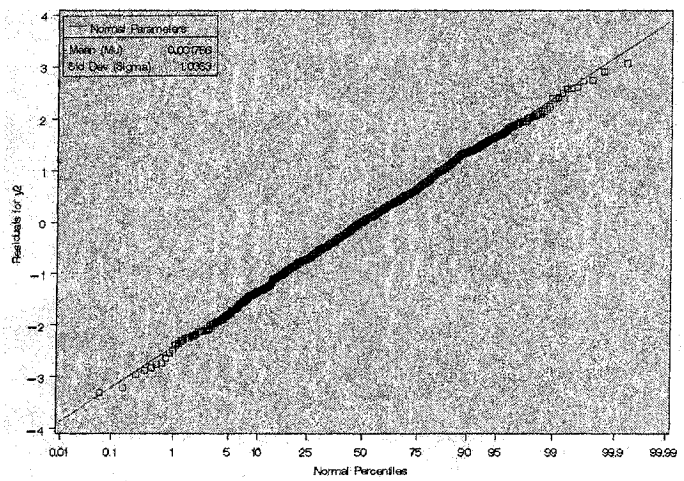

Figure 50: Normal Probability Plot of Residual Series of Simulated Series Y2

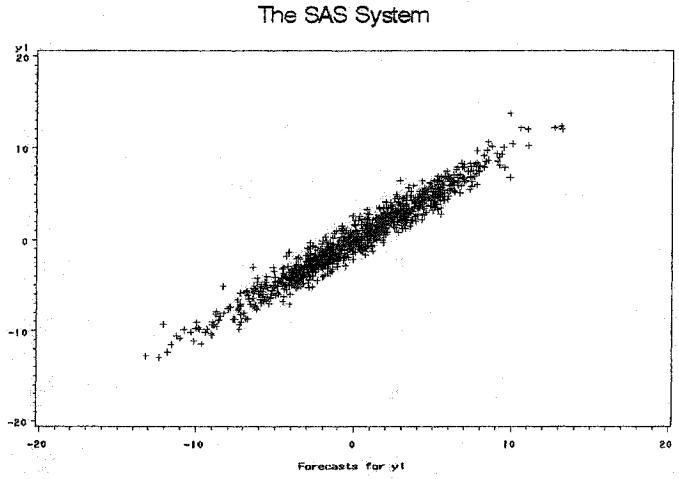

Figure 51: The Scatter Plot of Y1 and Predicted Y1 indicates it fits very well.

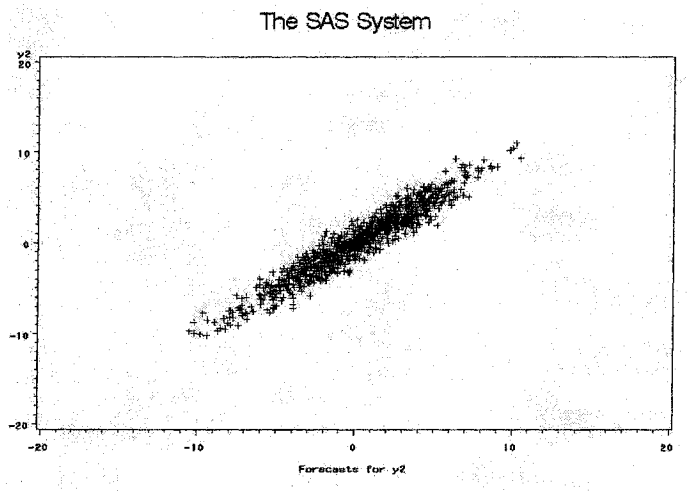

Figure 52: The Scatter Plot of Y2 and Predicted Y2 indicates it fits very well.

D. Model Decision

Comparing the values of AIC and prediction error among a group of VARMA models, we choose VARMA(1,1) as our model.

Model Parameter Estimates

Equation	Parameter	Estimate	Prob>|T|	Variable
y1(t)	CONST1	0.02257	0.3283	
	AR1_1_1	1.20999	0.0001	y1(t-1)
	AR1_1_2	-0.51823	0.0001	y2(t-1)
	MA1_1_1	-0.08733	0.0028	e1(t-1)
	MA1_1_2	0.31220	0.0001	e2(t-1)
y2(t)	CONST2	-0.01530	0.5074	
	AR1_2_1	0.59816	0.0001	y1(t-1)
	AR1_2_2	0.30271	0.0001	y2(t-1)
	MA1_2_1	0.21016	0.0001	e1(t-1)
	MA1_2_2	0.36721	0.0001	e2(t-1)

Table 20: Chosen Model for Estimated Simulation Series VARMA(1,1)

The estimated parameters for the model we have got from the program are very close to the originals of the simulation series. i.e.

$$\begin{bmatrix} z_{t1} \\ z_{t2} \end{bmatrix} = \begin{bmatrix} 1.20999 & -0.51823 \\ 0.59816 & 0.30271 \end{bmatrix} \begin{bmatrix} z_{t-1,1} \\ z_{t-1,2} \end{bmatrix} + \begin{bmatrix} a_{t1} \\ a_{t2} \end{bmatrix} - \begin{bmatrix} -0.08733 & 0.31220 \\ 0.21016 & 0.36721 \end{bmatrix} \begin{bmatrix} a_{t-1,1} \\ a_{t-1,2} \end{bmatrix}$$

which is the same as the original simulation series.

7.2 The Source Code

UNIVARIATE TIME SERIES MODELLING FOR RTT

```
/* UNIVARIATE TIME SERIES MODELING FOR RTT*/

%global min_AIC  p0  q0 kk pVal NN PredErr lambda;
*options mlogic mprint symbolgen mprint merror;/*debugging
techniques*/

 /*get the original data and trunc*/
 data rtt_raw;
   infile "c:\yli\thesis\simu_data_rtt.txt";
   input time 1-10 rtt 12-19;
   time=int(time*2)/2;
   run;
/*Add a field of time into rtt_raw to obtain rtt_impr*/
  data temp;
   do time=0 to 1000 by 0.5;
   output;
   end;
   run;quit;

data rtt_impr;
 merge rtt_raw temp;
 by time;
 run;quit;
/*Average rtt_impr*/ proc means data=rtt_impr NOPRINT;
 by time;
 var rtt;
 output out=rtt_impr_2 ;
 run;quit;
 /*keep data between 4 to 9 minutes*/
data rtt_impr;  set rtt_impr_2;
 if _STAT_^="MEAN" then delete;
 if time<=240 or time>546 then delete;
 run;quit;
/*Interpolate time interval*/ data rtt_interp; set rtt_impr;
  retain fill;
  if rtt^=. then fill=rtt; else rtt=fill;
  run;quit;
/*find the Lambda with smallest AIC on ARMA(10,10);*/
%macro fitting(var,denom);
 /*get value of lambda to the global varible*/
 data _NULL_; call symput("lambda",&var/&denom); run;
/*Get Box-Cox Transform data set*/
 data rtt_sas; set rtt_interp;
  lam=symget("lambda")+0.0;
  keep time rtt srtt lam;
  if &lambda=0 then do;
     srtt=log(rtt);
     end;
  else do;
     srtt=(rtt**(&lambda)-1)/(&lambda);
     end;
 run;
/*Count the total number*/
 data rtt_N; set rtt_sas;
   retain num 0;
   num+1;
   run;

 proc sort data=rtt_N  ;
   by DESCENDING num;
   run;

 data rtt_N; set rtt_N;
   if _N_=1 then output;
   else delete;
   run;

 data rtt_N; set rtt_N;
   call symput("NN",num);
   run;
/*Run ARMA proc to get Average Predcited Error for choosing appropriate Lambda*/
 proc arima data=rtt_sas; *find the relative model arma(p,q);
   identify var=srtt nlag=24 outcov=testcov;
   estimate p=10 q=10 outest=testest outmodel=testmod outstat=teststat  singular=1e-30;
   forecast lead=10 out=predict_final;
   run;quit;
/*Add time variable, rtt_sas and output of Predict*/
 data times; set rtt_sas; keep time; run;

 data predict; set predict_final; if _N_>&NN then delete; run; quit;

 data predict_first; merge times predict; run; quit;

 DATA predict;  MERGE rtt_sas predict_first; BY time; RUN ;
```

```
/*Calculate the predicted value for original RTT*/
 data predict; set predict;
   stdresidual=residual / std;
   if &lambda=0 then  predict=exp(forecast);
   else predict= (&lambda*forecast+1)**(1/(&lambda));
   err=(rtt-predict)*(rtt-predict);
   sum+err;
   run;quit;

 data predict; set predict;
   keep time rtt srtt forecast predict residual err sum;
   output;
   run;quit;
/*Plot residual vs predicted value, and RTT vs predicted value*/
 goptions reset=all;
  title "residual vs forecast:";
  proc gplot data=predict;
  plot residual*forecast;
  run;quit;

 goptions reset=all;
   title "Correlation:";
   symbol1 color=black   ;
   proc gplot data=predict;
      plot predict*rtt;
      run;quit;

   /*Obtain average Predicted Error*/
 data rtt_hat; set predict;
    keep time rtt predict err sum ;
    output;
    run;

 proc sort data=rtt_hat;
   by DESCENDING time;
   run;

 data rtt_hat; set rtt_hat;
   if _N_=1 then output;
   else delete;
   run;

 data rtt_hat; set rtt_hat;
   num=symget("NN")+0.0;
   avePe=sum/num;
   call symput("PredErr",avePe);
   run;
```

```
 /* Put the Average Predicted Error for comparison*/
 data arma0; *write down the fitting result;
   lambda=&lambda+0.0;
   PrdErr=symget("PredErr")+0.0;
   output;
   run;

 proc append base=arma data=arma0;
   run;

 data arma; set arma;
   if prderr=. then delete;
   run;

 proc print data=arma; run;/*print out the fitted AIC and levels*/

 data testcov; set testcov; upper=stderr;  run;

%mend fitting;

%macro start(sa,sb,sd);
%do r= &sa*&sd/10 %to &sb*&sd/10;
   %fitting(&r,&sd); *call macro fitting to find lambda;
%end;
%mend start;
/*Clear data sets of ARMA, PRedict_final*/ data arma; run; data
predict_final; run; /*Lambda from 0.2 to 1.5 step 0.0025*/
%start (2,15,400);

  /*Plot curve for lambda vs predErr*/
  goptions reset=all;
  symbol3 color=black value=circle;
  proc gplot data=arma;
  title "AvePredErr vs. Lambda for rtt:";
  plot prderr*lambda;
    run;quit;

   goptions reset=all;
    symbol1 color=black value=circle ;
    symbol2 color=black value=plus;
    symbol3 color=black value=circle;
    axis1 order=(0.2 to 1.8 by 0.05);
    axis2 order=(0.000041 to 0.000047 by 0.0000005);
    title "Rank Lambda vs. AvePrdErr for rtt:";
   proc gplot data=arma;
       plot prderr*lambda /haxis=axis1 vaxis=axis2;
```

```
   run;quit;

  data arma1; set arma;
    avepe=prderr+0;
    sym=1+floor( (p0+q0)/4);
    run;
/*Plot table of lambda vs predErr sorted by lambda and by predErr*/
proc sort data=arma1 out=armatemp; by lambda; run; quit; proc
print data=armatemp; run;

proc sort data=arma1 out=armatemp; by prderr; run; quit; proc
print data=armatemp; run;

symbol1 color=black value=circle ;
    proc gplot data=arma1;
       plot avepe*lambda;
    run;quit;

 /*Pick up the value of lambda to plot the datasets of RTT*/
proc sort data=arma1; by prdErr; run;
 data final0; set arma1;
      if _N_=1 then do;
         output ;
         call symput("p0", p0);
         call symput("q0", q0);
         call symput("lambda",lambda);
      end;
      else delete;
      run;

 data rtt_sas; set rtt_interp;
  lam=symget("lambda")+0.0;
  keep time rtt srtt lam;
  if &lambda=0 then do;
    srtt=log(rtt); end;
  else do;
    srtt=(rtt**(&lambda)-1)/(&lambda);
   end;
   run;

goptions reset=all;
   title "rtt_Raw:";
   symbol1 color=black  interpol=join ;
 proc gplot data=rtt_raw;
    plot rtt*time;
    run;quit;
goptions reset=all;
   title "rtt_Improved:";
   symbol1 color=black  interpol=join ;
   proc gplot data=rtt_impr;
    plot rtt*time;
    run;quit;
goptions reset=all;
   title "rtt_Interpolation:";
   symbol1 color=black  interpol=join ;
   proc gplot data=rtt_interp;
    plot rtt*time;
    run;quit;
goptions reset=all;
   title "rtt_SAS:";
   symbol1 color=black  interpol=join ;
   proc gplot data=rtt_sas;
    plot srtt*time;
    run;quit;

data temp; set arma;keep lambda prderr;run; proc sort data=temp;
by prderr; run; quit; proc print data=temp; run;quit;

%macro bypq(lambda);
/*Transform the data set*/ data rtt_sas; set rtt_interp;
  lam=&lambda+0.0;
  keep time rtt srtt lam;
  if &lambda=0 then do;
    srtt=log(rtt); end;
  else do;
    srtt=(rtt**(&lambda)-1)/(&lambda);
   end;
   run;

   /*Get the initial value of AIC*/
 *ods trace on / listing;
 *ods output ParameterEstimates=ppp;
  proc arima data=rtt_sas; *get the first model;
    identify var=srtt nlag=24 outcov=testcov;
    estimate p=1 q=0 outest=testest outmodel=testmod outstat=teststat  singular=1e-30;
*   forecast lead=10 out=predict printall;
    run;
*ods trace off;

  data AIC;  set teststat;  * get the initial AIC;
     call symput("p0",1);
     call symput("q0",0);
     keep _VALUE_;
     if _N_=1 then output; else delete;
```

```
    run;

data AIC; set AIC;
  call symput("min_AIC",_VALUE_);  *init min_AIC;
  run;

data testacc; run;
/*Try different p and q to compare AIC*/
 %do i=0 %to 10; *loop with all rank of p,q;
    %do j=0 %to 10;

      proc arima data=rtt_sas; *find the relative model arma(p,q);
        identify var=srtt nlag=24 outcov=testcov;
       estimate p=&i q=&j outest=testest outmodel=testmod outstat=teststat  singular=1e-30;
        run;
      /*Put current p and q into testacc*/
      data testemp; set teststat; p=&i; q=&j; run;

      data testacc; set testacc testemp; run;
     /*get a pair of p,q and AIC, as well as Minimum AIC with p0 and q0*/
      data AIC1;   set teststat; *if the AIC is smaller, record it and p,q in temp AIC;
        keep _VALUE_;
        if _N_=1 then do;
           output;
           min=symget("min_aic")+0.0;
           if _VALUE_ < min then do;  *get the smallest AIC;
             call symput("min_AIC", _VALUE_);
             call symput("p0", &i);
             call symput("q0", &j);
           end;
        end;
        run;

     data AIC;   set AIC;  *put the AIC into data file AIC;
       keep min;
       min=symget("min_AIC");
       output;
       run;

  %end;
  dm output 'clear';
  dm log 'clear';
 %end;

 data testacc; set testacc; if _STAT_^="AIC" then delete; order=p+q; run;
  /*Rerun arima to focus on minimum AIC*/
 proc arima data=rtt_sas; *use the fittest parameters to generate the output;
    identify var=srtt nlag=24 outcov=testcov;
    estimate p=&p0 q=&q0 outest=testest outmodel=testmod outstat=teststat  singular=1e-30;
    forecast lead=10 out=predict_final;
    run;
quit;
%mend bypq;
/*Fix lambda=0,6, try different p and q to find p0 and q0 with
minimum AIC*/
%bypq(0.6);

/*Print the table of p,q and AIC*/ data testacc_final; set
testacc; keep AIC p q; AIC=_VALUE_; run;quit;

Proc sort data=testacc_final;
 by AIC;
 run;quit;
proc print data=testacc_final;
 run;quit;

   goptions reset=all;
   title "rtt_SAS:";
   symbol1 color=black  interpol=join ;
   proc gplot data=rtt_sas;
       plot srtt*time;
    run;quit;
/*Final plotting for model checking*/
%macro check(lambda);
  /*Plot transformed data*/
 data rtt_sas; set rtt_interp;
  lam=symget("lambda")+0.0;
  keep time rtt srtt lam;
  if &lambda=0 then do;
    srtt=log(rtt); end;
  else do;
    srtt=(rtt**(&lambda)-1)/(&lambda);
   end;
   run;

  /*rerun ARIMA for plotting teh results*/
 ODS LISTING;
  ODS TRACE ON/listing;
  ods output ARPolynomial=rlt_AR;
  ods output MAPolynomial=rlt_MA;
 title "ACF of ARMA for rtt:";
  proc arima data=rtt_sas; *use the fittest parameters to generate the output;
    identify var=srtt nlag=24  crosscor=(Time) outcov=testcov;
    estimate p=3 q=3 outest=testest outmodel=testmod outstat=teststat  singular=1e-30;
```

```
      forecast lead=10 out=predict_final ;
   run;quit;
 ODS TRACE OFF;
/*Print the parameter estimates for the model*/
proc print data=final0; run;
proc print data=rlt_ar; run;
proc print data=rlt_ma; run; quit;

goptions reset=all;
  title "residuals vs. forecasted rtt:";
  symbol1 color=blue;
   proc gplot data=predict_final;
      plot residual*forecast;
   run;quit;
/*Get residual time sreies*/
data resi; set predict_final;
  keep time residuals;
  retain time 240;
  time=time+0.5;
  residuals=residual*1;
  output;
  run;
  quit;
/*Merge rtt_sas and predict_final to calculate the predicted value*/
data predict_first; set predict_final;
 retain no 0;
 no=no+1;
 output;
 run;quit;

data rtt_sas;set rtt_sas;
 retain no 0;
 no=no+1;
 output;
 run;quit;

DATA predict; MERGE rtt_sas predict_first; BY no;  RUN ;
/*Calculate the predicted RTT*/
data predict; set predict;
    stdresidual=residual / std;
    if &lambda=0 then  predict=exp(forecast);
    else predict= (&lambda*forecast+1)**(1/(&lambda));
    err=(rtt-predict)*(rtt-predict);
    sum+err;
    run;quit;

data predict; set predict;
   keep time rtt srtt forecast predict residual err sum;
   if time=. or residual=. then delete;
   run;quit;
   output;
   run;quit;

goptions reset=all;
 title "Residual vs forecast:";
   proc gplot data=predict;
       plot residual*forecast;
       run;quit;

goptions reset=all;
 title "Residual vs forecast:";
   proc gplot data=predict;
       plot residual*predict;
       run;quit;


goptions reset=all;
 title "Correlation:";
 symbol1 color=black   ;
 proc gplot data=predict;
     plot predict*rtt;
     run;quit;

proc sort data=arma;
   by prderr;
      run;
   quit;

title "residuals for rtt:";
proc gplot data=resi; plot residuals*time; run;quit;

data arma_rtt; set arma;
  keep p0 q0 AIC  Lambda  AvePrdErr;
  AvePrdErr=prderr;
  output;
  run; quit;

title "Stats for ARMA:";
proc print data=arma_rtt;
  var p0 q0 AIC Lambda AvePrdErr;
  run;
  quit;

title "Stats2 for ARMA:";
```

```
 proc print data=teststat;
   var _type_ _stat_ _value_;
   run;
   quit;

%mend;
/*Model checking by testing relative plots*/
%check(0.6);
```

UNIVARIATE TIME SERIES MODELLING FOR WINDOW SIZE

```
/* UNIVARIATE TIME SERIES MODELING FOR WINDOW SIZE*/

%global min_AIC  p0  q0 kk pVal NN PredErr lambda;
options mlogic mprint symbolgen mprint merror;/*debugging
techniques*/ /*get the original data and trunc*/ data win_raw;
 infile "c:\yli\thesis\simu_data_win.txt";
 input time 1-10 win 12-14;
 time=int(time*2)/2;
 run;
/*Add a field of time into win_raw to obtain Win_impr*/ data temp;
  do time=0 to 10000 by 0.5;
  output;
  end;
  run;quit;
data win_impr;
  merge win_raw temp;
  by time;
  run;quit;
  /*Average Window Size_impr*/
proc means data=win_impr NOPRINT;
  by time;
  var win;
  output out=win_impr_2 ;
  run;quit;
 /*keep data between 4 to 9 minutes*/
data win_impr;  set win_impr_2;
  if _STAT_^="MEAN" then delete;
  if time<=240 or time>546 then delete;
  run;quit;
/*Interpolate time interval*/ data win_interp; set win_impr;
  retain fill;
  if win^=. then fill=win; else win=fill;
  run;quit;
 /*find the Lambda with smallest AIC on ARMA(10,10);*/
%macro fitting(var,denom); *find the fittest model with smallest AIC and rank of p,q;
 /*get value of lambda to the global varible*/
  data _NULL_; call symput("lambda",&var/&denom); run;
 /*Get Box-Cox Transform data set*/
  data win_sas; set win_interp;
    lam=symget("lambda")+0.0;
    keep time win swin lam;
    if &lambda=0 then do;
      swin=log(win); end;
    else do;
      swin=(win**(&lambda)-1)/(&lambda);
      end;
      run;
   /*Count the total number*/
    data win_N; set win_sas;
    retain num 0;
    num+1;
    run;

  proc sort data=win_N  ;
  by DESCENDING num;
  run;

  data win_N; set win_N;
    if _N_=1 then output;   else delete;
    run;

  data win_N; set win_N;
    call symput("NN",num);
    run;
 /*Run ARMA proc to get Average Predcited Error for choosing appropriate Lambda*/
  proc arima data=win_sas; *use the fittest parameters to generate the output;
    identify var=swin nlag=24 outcov=testcov;
    estimate p=10 q=10 outest=testest outmodel=testmod outstat=teststat  singular=1e-30;
    forecast lead=10 out=predict_final;
    run;
 /*Add time variable, Win_sas and output of Predict*/
  data times; set win_sas; keep time; run;

  data predict; set predict_final; if _N_>&NN then delete; run; quit;
```

```
data predict_first; merge times predict; run; quit;

DATA predict; MERGE win_sas predict_first; BY time;RUN ;

/*Calculate the predicted value for original Win*/
data predict; set predict;
   stdresidual=residual / std;
   if &lambda=0 then  predict=exp(forecast);
   else predict= (  ( (&lambda*forecast+1)**(1/(&lambda)) )*100+0.5)/100;
   err=(win-predict)*(win-predict);
   sum+err;
   run;quit;

   data predict; set predict;
    keep time win swin forecast predict residual err sum;
    output;
    run;quit;
/*Plot residual vs predicted value, and Win vs predicted value*/
 goptions reset=all;
 title "residual vs forecast:";
    proc gplot data=predict;
        plot residual*forecast;
        run;quit;

     goptions reset=all;
 title "Correlation:";
 symbol1 color=black   ;
    proc gplot data=predict;
     plot predict*win;
     run;quit;

  /*Obtain average Predicted Error*/
 data win_hat; set predict;
   keep time win predict err sum ;
   output;
   run;

proc sort data=win_hat;
  by DESCENDING time;
  run;

data win_hat; set win_hat;
  if _N_=1 then output;   else delete;
  run;

data win_hat; set win_hat;
*  if avePe>=0 then do;
    num=symget("NN")+0.0;
    avePe=sum/num;
   call symput("PredErr",avePe);
*   end;
   run;

 /* Put the Average Predicted Error for comparison*/
data arma0; *write down the fitting result;
    p0=symget("p0")+0.0;
    q0=symget("q0")+0.0;
    AIC=symget("min_AIC")+0.0;
    lambda=&lambda+0.0;
    PrdErr=symget("PredErr")+0.0;
    output;
    run;

 proc append base=arma data=arma0;
    run;

 data arma; set arma;
   if prderr=. then do;
     delete;
   end;
   run;

 proc print data=arma; /*print out the fitted AIC and levels*/
     run;

 data testcov; set testcov; upper=stderr;
   run;
   dm output 'clear';
   dm log 'clear';
%mend fitting;
%macro start(sa,sb,sd);
/*data arma; run; data arma1; run;*clear;*/
%do r= &sa*&sd/10 %to &sb*&sd/10;
   %fitting(&r,&sd);
%end;
%mend start;
/*Clear data sets of ARMA, PRedict_final*/ data arma; run; data
predict_final; run; /*Lambda from 0.3 to 1.8 step 0.0025*/
%start (3,18,400);

  /*Plot curve for lambda vs predErr*/
   goptions reset=all;
   symbol3 color=black value=circle;
  proc gplot data=arma;
```

```
 title "AvePredErr vs. Lambda for Window Size";
 plot prderr*lambda;
   run;quit;

  goptions reset=all;
   symbol1 color=black value=square ;
   symbol2 color=black value=plus;
   symbol3 color=black value=circle;
  title "Rank Lambda vs. AvePrdErr for Window Size";
 proc gplot data=arma1;
      plot avepe*lambda=sym;
   run;quit;

 symbol1 color=black value=circle ;
   proc gplot data=arma1;
      plot avepe*lambda;
   run;quit;
/*Pick up the value of lambda to plot the datasets of Win*/

 proc sort data=arma1;
    by prdErr;
    run;
  data final0; set arma1;
     if _N_=1 then do;
        output ;
        call symput("p0", p0);
        call symput("q0", q0);
        call symput("lambda",lambda);
     end;
     else delete;
     run;

data win_sas; set win_interp;
 lam=symget("lambda")+0.0;
 keep time win swin lam;
 if &lambda=0 then do;
   swin=log(win); end;
 else do;
   swin=(win**(&lambda)-1)/(&lambda);
  end;
  run;

  data win_sas; set win_sas;
    if swin<=0 then delete;
    run;
goptions reset=all;
   title "Win_Raw:";
   symbol1 color=black  interpol=join ;
   proc gplot data=win_raw;
       plot win*time;
    run;quit;
goptions reset=all;
   title "Win_Improved:";
   symbol1 color=black  interpol=join ;
   proc gplot data=win_impr;
       plot win*time;
    run;quit;

goptions reset=all;
   title "Win_Interpolation:";
   symbol1 color=black  interpol=join ;

   proc gplot data=win_interp;
       plot win*time;
    run;quit;
    goptions reset=all;
   title "Win_SAS:";
   symbol1 color=black  interpol=join ;
   proc gplot data=win_sas;
       plot swin*time;
    run;quit;

%macro bypq(lambda);
/*Transform the data set*/ data win_sas; set win_interp;
  lam=&lambda+0.0;
  keep time win swin lam;
  if &lambda=0 then do;
    swin=log(win); end;
  else do;
    swin=(win**(&lambda)-1)/(&lambda);
   end;
   run;
 /*Get the initial value of AIC*/
 *ods trace on / listing;
 *ods output ParameterEstimates=ppp;
  proc arima data=win_sas; *get the first model;
    identify var=swin nlag=24 outcov=testcov;
    estimate p=1 q=0 outest=testest outmodel=testmod outstat=teststat  singular=1e-30;
*   forecast lead=10 out=predict printall;
    run;
*ods trace off;

  data AIC;  set teststat;  * get the initial AIC;
      call symput("p0",1);
```

```
    call symput("q0",0);
    keep _VALUE_;
    if _N_=1 then output; else delete;
    run;

data AIC; set AIC;
  call symput("min_AIC",_VALUE_);  *init min_AIC;
  run;
data testacc; run;
 /*Try different p and q to compare AIC*/
%do i=0 %to 5; *loop with all rank of p,q;
    %do j=0 %to 5;

   proc arima data=win_sas; *find the relative model arma(p,q);
       identify var=swin nlag=24 outcov=testcov;
       estimate p=&i q=&j outest=testest outmodel=testmod outstat=teststat  singular=1e-30;
       run;
      /*Put current p and q into testacc*/
   data testemp; set teststat; p=&i; q=&j; run;
   data testacc; set testacc testemp; run;

   data AIC1;   set teststat; *if the AIC is smaller, record it and p,q in temp AIC;
        keep _VALUE_;
        if _N_=1 then do;
           output;
           min=symget("min_aic")+0.0;
           if _VALUE_ < min then do;  *get the smallest AIC;
             call symput("min_AIC", _VALUE_);
             call symput("p0", &i);
             call symput("q0", &j);
           end;
        end;
        run;

     data AIC;   set AIC;  *put the AIC into data file AIC;
       keep min;
       min=symget("min_AIC");
       output;
       run;

   %end;
   dm output 'clear';
   dm log 'clear';
  %end;

  data testacc; set testacc; if _STAT_^="AIC" then delete; order=p+q; run;
  /*Rerun arima to focus on minimum AIC*/
   proc arima data=win_sas; *use the fittest parameters to generate the output;
   identify var=swin nlag=24 outcov=testcov;
   estimate p=&p0 q=&q0 outest=testest outmodel=testmod outstat=teststat  singular=1e-30;
   forecast lead=10 out=predict_final;
   run;
   quit;
 %mend bypq;
 /*Fix lambda=0,6, try different p and q to find p0 and q0 with minimum AIC*/
  %bypq(0.75);

proc print data=final0;
run;
proc print data=rlt_ar;
run;
proc print data=rlt_ma;
run;
quit;

  goptions reset=all;
   title "residuals vs. forecasted window size:";
   symbol1 color=blue;
   proc gplot data=predict_final;
      plot residual*forecast;
      run;quit;

  goptions reset=all;
  title "residuals vs. forecasted window size:";
  symbol1 color=blue;
   proc gplot data=predict_final;
      plot residual*forecast;
      run;quit;

  goptions reset=all;
  title "residuals of window size:";
  symbol1 color=blue;
  data resi; set predict_final;
     keep time residuals;
     retain time 240;
     time=time+0.5;
     residuals=residual*1;
     output;
     run; quit;

 data predict_first; set predict_final;
     retain no 0;
     no=no+1;
     output;
```

```
    run;quit;

data win_sas;set win_sas;
    retain no 0;
    no=no+1;
    output;
    run;quit;

DATA predict;
  MERGE win_sas predict_first ;
  BY no;
  RUN ;

data predict; set predict;
   stdresidual=residual / std;
   if &lambda=0 then  predict=exp(forecast);
   else predict= (  ( (&lambda*forecast+1)**(1/(&lambda)) )*100+0.5)/100;
   err=(win-predict)*(win-predict);
   sum+err;
   run;quit;

data predict; set predict;
    keep time win swin forecast predict residual err sum;
    output;
    run;quit;

goptions reset=all;
proc gplot data=predict;
    plot residual*predict;
    run;quit;
goptions reset=all;
proc gplot data=predict;
    plot predict*win;
    run;quit;
proc gplot data=predict;
    plot win*forecast;
    run;quit;
goptions reset=all;
  title "Correlation:";
  symbol1 color=black   ;
proc gplot data=predict;
  plot predict*win;
  run;quit;

title" For Window Size:";
proc univariate data=resi plots;
   var resid;
   run; quit;

proc sort data=arma;
   by lambda; run; quit;

title "residuals for window size:";
proc gplot data=resi;
   plot residuals*time;
   run;quit;

data arma_win; set arma;
   keep p0 q0 AIC  Lambda  AvePrdErr;
   AvePrdErr=prderr;
   output; run; quit;

proc sort data=arma_win;
   by lambda;
   run;

title "Stats for ARMA:";
proc print data=arma_win;
  var p0 q0 AIC Lambda AvePrdErr;
  run;  quit;

title "Stats2 for ARMA:";
proc print data=teststat;
  var _type_ _stat_ _value_;
  run;   quit;
```

Varma modelling for win and RTT

```
* Title: Varma modeling for win and RTT
* Name: Yingzi Li
*date: May 11, 2004; options mlogic mprint symbolgen mprint
merror;/*debugging techniques*/
%global PredErr Lambda min_AIC p0 q0;
 /*get the original data and trunc*/
  data rtt_raw;  /*get the original data and trunc*/
    infile "c:\yli\thesis\simu_data_rtt.txt";
    input time 1-10 rtt 12-19;
    time=int(time*2)/2;
    run;
  data win_raw;  /*get the original data and trunc*/
    infile "c:\yli\thesis\simu_data_win.txt";
    input time 1-10 win 12-14;
```

```
  time=int(time*2)/2;
  run;
 /*Add a field of time into raw to obtain impr for WIN*/
data temp;
  do time=0 to 10000 by 0.5;
     output;
  end;
  run;quit;
data win_impr;
  merge win_raw temp;
  by time;
  run;quit;
/*Average impr*/
 proc means data=win_impr NOPRINT;
  by time;
  var win;
  output out=win_impr_2 ;
  run;quit;
 /*keep data between 4 to 9 minutes*/
 data win_impr;  set win_impr_2;
  if _STAT_^="MEAN" then delete;
  if time<240 or time>546 then delete;
  run;quit;
/*Interpolate time interval*/
 data win_interp; set win_impr;
  retain fill;
  if win^=. then fill=win; else win=fill;
  run;quit;
/*Add a field of time into rtt_raw to obtain rtt_impr*/
 data temp;
  do time=0 to 10000 by 0.5;
  output;
  end;
  run;quit;
 data rtt_impr;
  merge rtt_raw temp;
  by time;
  run;quit;
 /*Average rtt_impr*/
 proc means data=rtt_impr NOPRINT;
  by time;
  var rtt;
  output out=rtt_impr_2 ;
  run;quit;
/*keep data between 4 to 9 minutes*/
 data rtt_impr;  set rtt_impr_2;
  if _STAT_^="MEAN" then delete;
  if time<240 or time>546 then delete;
  run;quit;
/*Interpolate time interval*/
 data rtt_interp; set rtt_impr;
  retain fill;
  if rtt^=. then fill=rtt; else rtt=fill;
  run;quit;
 /*data varma_raw1 ;
     merge win_raw(in=a) rtt_raw(in=b) ;
     by time;
     if a and b then output varma_raw1;
     run;*/

/*Merge bivariate series*/
 proc sql;
  create table varma as
  select t1.time as time,t1.win as win, t2.rtt as rtt
  from win_interp as t1 inner join rtt_interp as t2
     on t1.time=t2.time;
  quit;
 /*
title 'VARMA_SAS:';
  symbol1 i=join l=1;
  symbol2 i=join l=2;
  axis2 label=(a=-90 r=90 " ");
 proc gplot data=varma_sas;
    plot swin*timeID;
   plot2 srtt*timeID;
   run;
   quit;
 data varma_sas; set varma_sas;
   keep timeID win rtt swin srtt;
   if swin=. or srtt=. then delete;
   label swin='Trans Window Size'
         srtt='Trans Round Trip Time';
   run;
   quit;
proc gplot data=varma_sas;
    plot swin*srtt;
    run;quit;*/
/*Transform the bivariate sets*/
 data varma_sas; set varma;
  keep time win rtt swin srtt;
  swin=(win**0.75-1)/0.75;
  srtt=(rtt**0.6-1)/0.6;
  put swin srtt;
  run;  quit;
```

```
/*find p and q with smallest AIC*/
      ods trace on / listing;
      ods output infocriterion=IC2;
         proc varmax data=varma_sas ;
            nloptions tech=qn;
            model swin srtt / p=4 q=0;
            output out=outSet2;
            run; quit;
      ods trace off;
 /*get merged dataset for predict error;*/
 data pred; set outset2;
   retain noo 0;
   noo=noo+1;
   output;
   run;quit;

 data varma_sas1; set varma_sas;
   if _N_=1 then delete;
   run;

 data varma_sas1; set varma_sas1;
    retain noo 0;
    noo=noo+1;
    output;
    run;quit;

 DATA predict;
   MERGE varma_sas1 pred ;
   BY noo;
   RUN ;
 data predict_1; set predict;
    if time=. then delete;
    if res1=. then delete;
    output;
    run;quit;
 /*Calculate Predict Errors and Predict values for Win and RTT*/
 data predvarma; set predict_1;
    keep time win predwin swin for1 err1 rtt predrtt srtt for2 err2 res1 res2;
    predwin=  (0.75*for1+1)**(1/0.75) ;
    err1=(win-predwin)*(win-predwin);
    predrtt= (0.6*for2+1)**(1/0.6) ;
    err2=(rtt-predrtt)*(rtt-predrtt);
    output;
    run;quit;

 /*get total number of data for average predict error;*/
```

```
data predvarma; set predvarma;
   if time=. then delete;
   if res1=. then delete;
   output;
   run;quit;

data temp_N; set predvarma;
   keep num 0;
   num+1;
   output;
   run;quit;
proc sort data=temp_N  ;
by DESCENDING num;
run;

data temp_N; set temp_N;
  if _N_=1 then output;   else delete;
  run;

data temp_N; set temp_N;
  call symput("NN",num);
  run;

/*get avePredErr for  rtt;*/
data rtt; set predvarma;
  keep num rtt srtt predrtt for2 err2 res2 sum2;
  retain num 0;
  num+1;
  sum2+err2;
  output;
  run;quit;

proc sort data=rtt;
  by descending num;
  run;

data rtt_final; set rtt;
  if _N_=1 then output; else delete;
  run;quit;

data rtt_final; set rtt_final;
  drop nu;
  avePrederr=sum2/num;
  output;
  run;quit;
proc print data=rtt_final;
 run;
```

```
/*get aveapredErr for win;*/
 data win; set predvarma;
   keep num win swin predwin for1 err1 res1 sum1;
   retain num 0;
   num+1;
   sum1+err1;
   output;
   run;quit;
 proc sort data=win;
   by descending num;
   run;quit;
 data win_final; set win;
   if _N_=1 then output; else delete;
   run;
 data win_final; set win_final;
    avePrederr=sum1/num;
    output;
    run;quit;

 proc print data=win_final;
 run;

/*Plot residual vs predict win;*/
 goptions reset=all;
 title "Residual vs predict win:";
 proc gplot data=win;
    plot res1*predwin;
    run;quit;

/*Plot residual vs predict rtt;*/
 goptions reset=all;
 title "Residual vs predict rtt:";
 proc gplot data=predvarma;
    plot res2*predrtt;
    run;quit;

/*Plot residual vs predict transformed win;*/
  goptions reset=all;
  title "residuals vs. forecasted window size:";
  symbol1 color=blue;
  proc gplot data=outset2;
    plot res1*for1;
    run;quit;

 /*Plot residual vs predict transformed rtt;*/
 goptions reset=all;
 title "residuals vs. forecasted RTT:";
 symbol1 color=blue;
 proc gplot data=outset2;
   plot res2*for2;
   run;quit;

/*Plot residual series of win;*/
 goptions reset=all;
 title "residual series of Win:";
 symbol1 color=blue;
 proc gplot data=predict_1;
    plot res1*time;
    run;quit;

/*Plot residual series of rtt;*/
 goptions reset=all;
 title "residual series of RTT:";
 symbol1 color=blue;
 proc gplot data=predict_1;
    plot res2*time;
    run;quit;
 /*Histogram Plot for Win and RTT*/
 goptions ftext=SWISS ctext=BLACK htext=1 cells;
 symbol v=SQUARE c=BLUE h=1 cells;
 proc univariate data=Work.Predvarma  vardef=N  noprint;
    var RES1;
    histogram / caxes=BLACK cframe=CXF7E1C2 waxis= 1
              cbarline=BLACK cfill=BLUE pfill=SOLID
              vscale=percent hminor=0 vminor=0
              name='HIST';
    run;
    symbol;
    goptions ftext= ctext= htext=;

 goptions ftext=SWISS ctext=BLACK htext=1 cells;
 symbol v=SQUARE c=BLUE h=1 cells;
 proc univariate data=Work.Predvarma  vardef=N  noprint;
    var RES2;
    histogram / caxes=BLACK cframe=CXF7E1C2 waxis= 1
              cbarline=BLACK cfill=BLUE pfill=SOLID
              vscale=percent hminor=0 vminor=0
              name='HIST';
   run;
   symbol;
   goptions ftext= ctext= htext=;
```

```
/* Probability Plots for WIN*/
goptions ftext=SWISS ctext=BLACK htext=1 cells;
symbol v=SQUARE c=BLUE h=1 cells;
proc univariate data=Work.Predvarma  vardef=N  noprint;
  var RES1;
  probplot / caxes=BLACK cframe=CXF7E1C2 waxis= 1
           hminor=0 vminor=0   name='PROB'
           normal( mu=est sigma=est color=BLUE l=1
                  w=1);
 inset normal;
 run;
 symbol;
 goptions ftext= ctext= htext=;

/* Probability Plots */
goptions ftext=SWISS ctext=BLACK htext=1 cells;
symbol v=SQUARE c=BLUE h=1 cells;
proc univariate data=Work.Predvarma  vardef=N  noprint;
   var RES2;
   probplot / caxes=BLACK cframe=CXF7E1C2 waxis= 1
           hminor=0 vminor=0   name='PROB'
           normal( mu=est sigma=est color=BLUE l=1
                  w=1);
   inset normal;
   run;
   symbol;
   goptions ftext= ctext= htext=;

/*Plot 'Correlation between Win and Predicted Win:';*/
 symbol1 i=join c=blue w=1;
 symbol2 i=join c=red w=1;
 axis1 order=(0 to 310 by 50);
 axis3 order=(0 to 0.5 by 0.05);
 axis2 order=(0 to 50 by 10);
 goptions reset=all;
 title 'Correlation between Win and Predicted Win:';
 proc gplot data=predvarma;
     plot predwin*win /haxis=axis2 vaxis=axis2;
     run;quit;

/*Plot 'Correlation between Win and Predicted rtt:';*/
goptions reset=all;
title "Correlation between RTT and Predicted RTT:";
symbol1 color=black   ;
proc gplot data=predvarma;
     plot predrtt*rtt /haxis=axis3 vaxis=axis3 ;
     run;quit;

goptions reset=all;
title 'Correlation between Win and Predicted Win:';
symbol1 i=join c=blue w=1;
symbol2 i=join c=red w=1;
axis1 order=(0 to 310 by 50);
axis3 order=(0 to 1.5 by 0.1);
axis2 order=(0 to 120 by 10);
proc gplot data=predvarma;
   plot predwin*time=1  win*time=2/  overlay haxis=axis1 vaxis=axis2;
   run;quit;
title "Correlation between RTT and Predicted RTT:";
proc gplot data=predvarma;
   plot predrtt*time=1  rtt*time=2/  overlay  haxis=axis1 vaxis=axis3 ;
   run;quit;
```

Bibliography

[1] Anderson, Time Series Analysis and Forecasting, utterworths 1975

[2] Chris Chatfield, The Analysis of Time Series Fifth edition, Chapman & Hall 1996

[3] C.Chatfield, The Analysis of Time Series, Chapman and Hall/CRC, 2004

[4] D.E.Comer, Internetworking with TCP/IP Vol1: Principles, protocols and architectures, Prentice Hall, New Jersey 07456

[5] F.Baccelli, D.R.McDonald, J.Reynier A mean-Field Model for multiple TCP Connections through a Buffer Implementing RED

[6] G.E.Box. and G.M.Jenkins, Time Series Analysis Forecasting and Control (2002)

[7] G.E.P. Box and G.M.Jenkins, Time series analysis : forecasting and control, Publisher San Francisco : Holden-Day, c1976

[8] G.T.Wilson, M.Reale and A.S.Morton, Developments in Multivariate Time Series Modeling, March 2001

[9] H.Jiang and C.Dovrolis, Passive Estimation of TCP Round-Trip Times, ACM Computer Communication Review, July 2002.

[10] H.Lutkepohl, Introduction to Multiple Time Series Analysis, Springer-Verlag, 1991

[11] H.Ohsaki, M.Morita and M.Murata, Measurement Based Modelling of Internet Round-Trip time Dynamics using System Identification, NETWORKING 2002, LNCS 2345, pp. 264-276,2002

[12] H.Ohsaki, M.Morita and M.Murata, On Modelling Round-Trip Time Dynamics of the Internet using System Identification, ICOIN 2002, LNCS 2343, pp.359-371,2002

[13] J.Gani and M.B.Priestly, Essays in Time Series and Allied Processes, Journal of Applied Probability,Special Volume(1986)

[14] J.Widmer, Models for TCP Throughput

[15] J.D.Hamilton, Time Series Analysis, Princeton University Press 1994

[16] K.Washburn, J.Evans, TCP/IP Running a Successful Network, Addison Wesley Inc., 1995

[17] L.Fourie, M.Maskery, D.McDonald, Red Without Feedback Delay

[18] L.Ljung, System Identification Theory for the User. Prentice-Hall, Inc. (1987)

[19] L.S.Brakmo and L.L.Peterson, TCP Vegas: End to end Congestion Avoidance on a Global Internet, IEEE Journal on Selected Areas in Communications, Vol. 13, No.8, October 1995

[20] M.B. Priestley, Spectral Analysis and Time Series, Prediction and Control, Academic Press 1981

[21] P.J.Brockwell and R.A.Davis, Introduction to Time Series and Forecasting, Springer 1996

[22] P.J.Brockwell, R.A.Davis, Introduction to time series and forecasting, New York:Springer, c2002.

[23] R.A.Johnson and D.W.Wichern, Applied Multivariate Statistical Analysis, Prentice Hall 1992

[24] R.H.Shumway, Applied Statistical Time Series Analysis, Prentice Hall 1988

[25] R. Jain. A Delay-Based Approach for Congestion Avoidance in Interconnected Heterogeneous Computer Networks. ACM COmputer COmmunication Review, Oct. 1989

[26] SAS Institute, SAS User's Guide, SAS/ETS Time Series Analysis and Forecasting

[27] S.Bohacek and B. Rozovskii, A Diffusion Model of Roundtrip Time, Computational Statistics and Data Analysis

[28] S.M.Pandit Time Series And System Analysis With Applications, John Wiley & Sons 1983

[29] T.C. Hsia, System Identification, Lexington Books, 1977

[30] V.Welch, NCSA, 1996

[31] W. S. Wei, Time Series Analysis(Univariate and Multivariate Methods), Addison-Wesley Publishing Company,Inc.1989

[32] Walter Vandaele, Applied Time Series and Box-Jenkins Models, Academic Press, Inc.1983