UNIVERSITY OF ALBERTA

Prediction of Machine Degradation Based on Vibration Analysis

BY

Fan Jiang ©

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

Engineering Management

DEPARTMENT OF MECHANICAL ENGINEERING

Edmonton, Alberta

Fall 2002

0-612-81415-7

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR:	Fan Jiang
TITLE OF THESIS:	Prediction of Machine Degradation Based on Vibration Analysis
DEGREE:	Master of Science
YEAR THIS DEGREE GRANTED:	2002

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

Fan Jiang
#3B 8915-112 ST
Edmonton, AB, T6G 2C5

April 28, 2002

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled: *Prediction of Machine Degradation Based on Vibration Analysis* submitted by Fan Jiang in partial fulfillment of the requirements for the degree of Master of Science.

M J Zuo

M. J. Zuo (Supervisor)

Peter C Fly

Peter Flynn

Don Koval

Don Koval

April 28, 2002

Abstract

In the manufacturing and processing industries, a substantial portion of operating costs goes to maintenance. An effective maintenance system should be able to monitor the operating conditions of a machine, issue advanced warnings of possible faults, predict the remaining life of a deteriorating machine, and schedule appropriate maintenance actions to prevent fatal equipment breakdowns. A machine's vibration is an excellent indicator of deteriorating mechanical condition. Current vibration monitoring systems used in the industry can only display vibration levels and equipment deterioration trends but have no functions of forecasting or predicting the equipment's remaining life.

In this thesis, we study the methods of forecasting especially for equipment degradation prediction. Some of the most often used forecasting methods are studied and analyzed to find their potentials to be applied to degradation prediction based on vibration analysis. A new method combining Akaike Information Criterion (AIC) and generalization error is proposed for forecasting model selection. Three cases are studied using this new method. The results show that the proposed method can provide a better forecasting model than using AIC only.

Support vector machine, a new tool for classification and regression, has been successfully applied to time series prediction. In this thesis, SVM is used for deterioration prediction. SVM is compared with time series models for vibration data prediction. The results show that SVM is capable of making a better prediction than time series models. However, SVM is very computation intensive. Discussions and future research studies are also provided.

Acknowledgement

I would like to express my sincere thanks to my supervisor, Dr. Ming-Jian Zuo, for his illuminating guidance, aid and advice throughout my M.Sc study.

I wish to thank Dr. Jing Lin for his thoughtful discussions and kind help.

I wish to thank my committee members: Dr. Flynn as my co-supervisor, and Dr. Koval, for their time and efforts to examine my thesis.

I also wish to express my deep gratitude to my wife, Xiaojin Jiang, for her understanding, encouragement, and support.

Table of Contents

List of Tables

List of Figures

Notation

$\{x_t, t = 1, 2, \ldots, n\}$	time series from $t = 1$ to n
μ	mean of time series x_t
σ_x^2	variance of time series x_t
$\hat{\mu}$	estimated mean of time series x_t
$\hat{\sigma}_x^2$	estimated variance of time series x_t
γ_k^2	autocovariance at lag k for time series x_t
$\hat{\gamma}_k^2$	estimated autocovariance at lag k for time series x_t
ρ_k	autocorrelation function (ACF) at lag k
ϕ_{kk}	partial autocorrelation function (PACF) at lag k
w_t $(t = 1, 2, \ldots, n)$	white noise series
σ_{w}^2	variance of white noise series
p	order of autoregressive model
d	times of differencing
q	order of moving average model
$\phi(B)$	autoregressive operator of order p
B	backward shift operator
$\theta(B)$	moving-average operator of order q
h	capacity of a model
l	size of training set
m	size of validation set
$C(h, l)$	difference between generalization error and training error
$R(t)$	average tool wear at time t
k	number of coefficients in a regression model
$\hat{\sigma}_k^2$	estimated variance for a regression model with k coefficients
s	weight factor of generalization error
∇	differencing operator
$R_{emp}[f]$	denotes the empirical risk functional
$C(.)$	cost function or loss function
$\lvert\xi\rvert_\varepsilon$	ε-insensitive loss function
ε	predetermined error value for ε-insensitive loss function
Φ	nonlinear mapping from the input space to feature space
ω	slope obtained from linear regression
b	intercept obtained from linear regression
$R_{reg}[f]$	regularized risk functional
λ	regularization constant
ξ_i, ξ_i^*	slack variables

Abbreviations

AR	autoregressive
MA	moving average
ARMA	mixed autoregressive and moving average
ARIMA	autoregressive integrated moving average
SVM	support vector machine
ERM	empirical risk minimization
SRM	structural risk minimization
RSS	residual sum of squares
AIC	Akaike's Information Criterion
GE	generalization error
comb	combined factor of AIC considering generalization
Inf	infinity
SSE	sum of squared errors
MAPE	mean absolute percentage error
RBF	radial basis function

Chapter 1. Introduction

In the manufacturing and processing industries, a substantial portion of operating costs goes to maintenance. Reducing maintenance cost is essential for operating cost minimization. For some non-critical equipment, the best maintenance plan might be to let it run to failure. The most often used maintenance strategy is periodic maintenance, e.g.: periodic oil changes every six thousand kilometers. For these critical equipment whose failure may cause heavy economic losses or human lives, condition monitoring is applied in addition to scheduled periodic maintenance. Condition monitoring will reduce operating and maintenance costs because wear and defects in moving parts can be discovered and repaired before the machine breaks down. Commonly used monitoring techniques include vibration analysis, oil analysis, and temperature analysis. Mechanical faults that may develop in time such as imbalance, misalignment, and bearing failure usually generate increased vibration. A machine's vibration signal is an excellent indicator of deteriorating mechanical condition. Vibration analysis has become an important part of industrial predictive maintenance programmes.

Vibration monitoring systems used in the industry today use different warning levels to determine whether the equipment is in normal condition or not. When the vibration level, e.g.: peak-to-peak amplitude, is above a certain warning level, a decision will be made subjectively from experience whether to continue operating or stop right away. Jardine et al. (1999) propose a method to optimize condition based maintenance decisions subject to vibration monitoring. In their study, the proportional hazard model (PHM) is used to estimate statistically the risk of the equipment failing within the next inspection interval. The key vibration signals are recorded to estimate the risk of an item failing. Their work focuses on decision optimization but no prediction of future risks is discussed. Most of the current vibration monitoring systems can only display vibration levels and equipment deterioration trends but have no function of forecasting or predicting the equipment's remaining life.

An effective maintenance system should be able to monitor the operating conditions of a machine, issue advanced warnings of possible faults, predict the remaining life of a deteriorating machine, and schedule appropriate maintenance actions to prevent fatal equipment breakdowns. If equipment's remaining life can be accurately predicted, appropriate preventive maintenance tasks can be scheduled in time to prevent equipment breakdowns. Certain research works have been conducted to forecast the trend of machine degradation by monitoring the amplitudes of its fault related vibration features. Tse and Atherton (1999) apply neural networks in machine vibration prediction. They compare the prediction results from traditional time series methods and from neural networks. The results show that neural networks may provide better predictions. Their work is discussed in detail in Chapter 2 with other degradation predicting methods.

In this thesis, we study the methods of equipment degradation forecasting based on vibration analysis. Forecasting is very important in many types of organizations since predictions of future events must be incorporated into the decision-making process. In forecasting events that will occur in the future, a forecaster must rely on information concerning events that have occurred in the past. It means that the best forecast of the future is based on what has happened in the past. Note that in some textbooks, forecast is defined as prediction of future events. But in this thesis, the terms of forecasting and prediction are treated as synonyms.

Forecasting methods can be divided into two main categories: qualitative and quantitative. Qualitative forecasting methods generally adopt the opinions of experts to predict future events subjectively. They are used when historical data either are not available or are scarce. For example, consider a situation in which new equipment is being introduced with no historical vibration data available. To forecast future vibration for the new equipment, a company may rely only on experts' opinions.

Quantitative forecasting methods can be divided into univariate models and causal models. Univariate models are solely on the basis of the past values of the recorded data series and assume that the data pattern will continue in the future. When a univariate model is used, historical data are analyzed in an attempt to identify a data pattern. By assuming that the pattern will continue in the future, we can forecast what is likely to happen in the future. Suppose that we have a series of vibration data. After analyzing the series, we decide that a linear model can be used to describe the pattern of this set of data. Then, this linear model will be fitted from existing data points and used for future vibration level forecasting. Causal models involve the identification of other variables that are related to the variable to be predicted. A statistical model that describes the relationship between the related variables and the variable to be predicted can be developed. For example, the sales of a new product might be related to its price, advertising expenditures, and competitors' prices. The relationship can be figured out between the sales of this product and its price, its advertising expenditures, and competitors' prices. Having determined this relationship, future sales of this new product can be determined based on future conditions of its price, advertising expenditures, and competitors' prices. In this thesis, we consider univariate models only.

The objective of this thesis is to provide future vibration forecasting based on given historical vibration data. Different forecasting methods are studied to identify potential candidates for our study. Time series models and support vector machines are major forecasting tools under investigation. Different forecasting models are applied to recorded vibration data to verify their effectiveness.

Chapter 2 is a general and extensive review of forecasting methods. This chapter reviews some of the most often used forecasting methods in the literature. These methods include traditional time series models, such as autoregressive (AR), moving average (MA), and autoregressive integrated moving average (ARIMA); machine learning methods or black box methods, such as neural networks (Tse and Atherton, 1999) and support vector machines (SVMs) (Muller et al., 1999); general path models (Lu and Meeker, 1993), and continuous state system reliability analysis (Zuo et al., 1999). These methods, including their algorithms and assumptions are briefly introduced. Their application assumptions and limitations are also discussed.

In Chapter 3, we propose a new time series method for selecting the optimal prediction model. The method combines Akaike's Information Criterion (AIC) with generalization error for time series model order determination. AIC and its expanded forms are widely accepted method for order(s) selection. But AIC measures the goodness of fit of existing data while we are interested in forecasting into the future. From application experience, we cannot solely depend on AIC for time series model order(s) determination. Generalization error directly measures a model's prediction ability. Minimizing generalization error is always the ultimate goal of building a forecasting model. The method presented in this thesis considers both AIC and generalization error for optimal time series model order(s) selection. Three examples including one vibration trend forecasting from real world data are studied. The applications show that we are able to obtain a better forecasting model with smaller generalization error by applying the proposed method than solely depending on AIC.

Chapter 4 studies Support Vector Machines (SVM) for short term forecasting. SVM is a statistical tool for solving pattern recognition and regression problems. It has been reported to have good performance in predicting time series. In this Chapter, basic concepts and algorithms of SVM are introduced. Then, we report our application of SVM to machine vibration prediction. Three examples are studied comparing forecasting results from SVM and from time series forecasting models such as AR and ARIMA. Vibration signals from a gearbox were collected and analyzed. The vibration trend data used in Chapter 3 is applied in this chapter for short term forecasting comparison. Existing SVM software tool (based on Matlab environment) is revised to conduct the SVM short term prediction.

Finally, Chapter 5 provides a summary of this thesis and suggestions of future research directions in equipment vibration prediction.

Chapter 2. Literature Review on Forecasting Methods

As discussed in Chapter 1, forecasting or prediction is a fundamental problem in many real world applications, such as weather forecasting and stock market analysis. Most forecasting methods are based on fitting a model to a set of known data (Makridakis et al, 1984). In this chapter we provide a literature review of forecasting methods that may be applied in vibration analysis. These methods include time series models, such as autoregressive (AR), moving average (MA), mixed AR and MA (ARMA), and integrated AR and MA (ARIMA); machine learning methods or black box methods, such as neural networks (Tse and Atherton, 1999) and support vector machine (SVM) (Muller et al., 1999); and general path models (Lu and Meeker, 1993). Their pros and cons are also discussed.

2.1 Time Series Models

A time series is a set of observations generated sequentially in time. Most time series forecasting methods are based on the following assumptions:

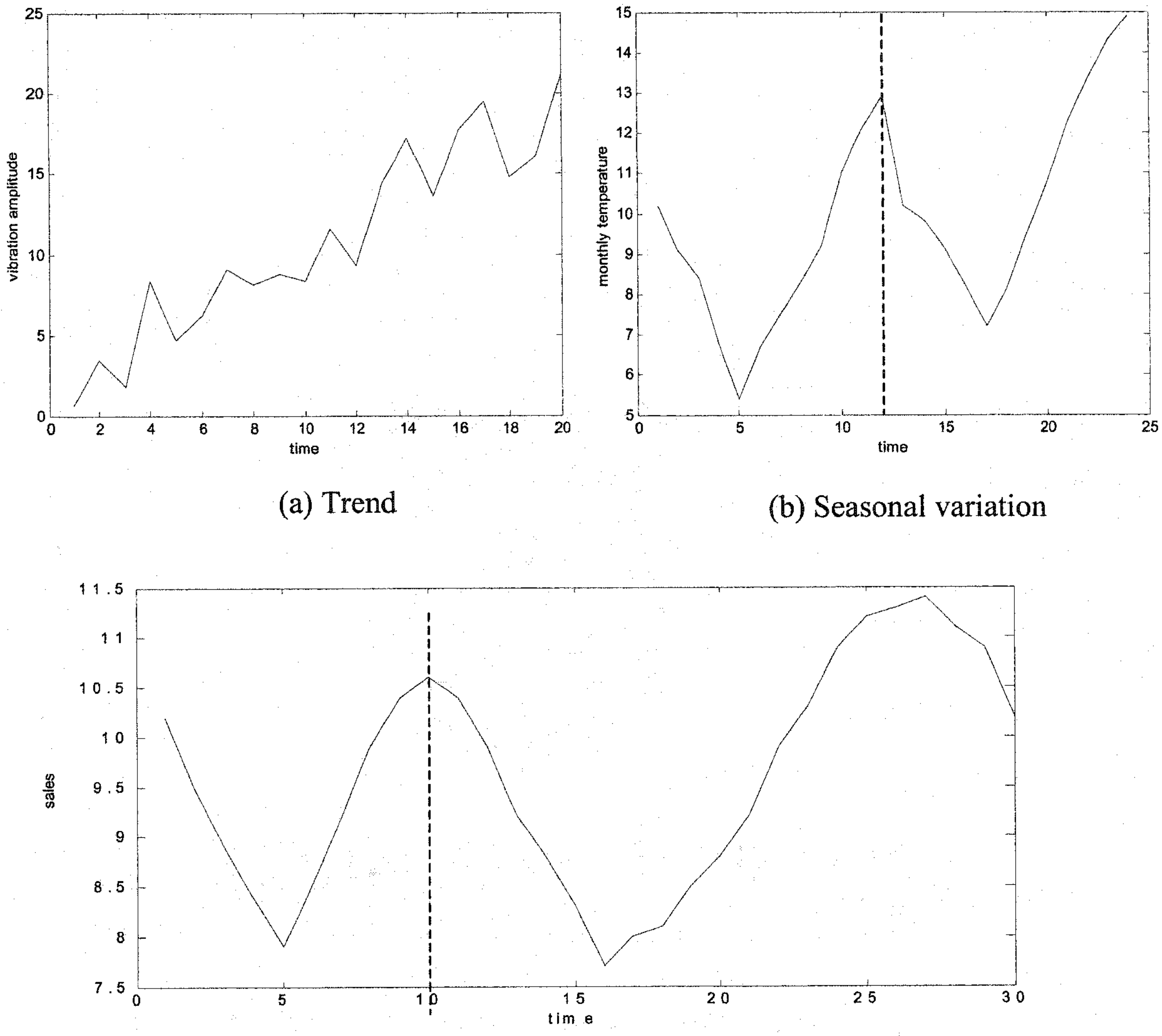
1. Data points are collected at equal time intervals.
2. There are no missing data.

Time series models are widely used in economic and business planning (Bowerman and O'Connell, 1993), machine degradation prediction (Tse and Atherton, 1999), weather forecasting, and sunspot activity analysis (Brockwell and Davis, 1987). Many researchers use time series analysis to develop mathematical models that provide plausible descriptions of sample data (Shumway and Stoffer, 2000a). However, our interest lies in forecasting. As a result, we require the model obtained not only to fit sample data well, but, more importantly, to provide good forecasting.

Time series forecasting methods vary widely in sophistication and applications. The two basic approaches to forecasting time series are: the univariate methods and causal (multivariate) methods. In univariate methods, forecasts of a time series are derived solely on the basis of the historical behavior of the series itself. Causal methods take into account interrelations among various factors and series. They require more information on the process and therefore are more complex. We only discuss the univariate time series methods in this thesis.

Univariate time series prediction models include overall trend models, smoothing models, seasonal models, decomposition models, and Box-Jenkins methodology (Bowerman and O'Connell, 1993). The Box-Jenkins methodology is studied the most because of its overwhelming advantages (Bowerman and O'Connell, 1993). A Box-Jenkins model is developed in terms of statistical concepts under the assumption that the processes being modeled are dynamic and subject to statistical fluctuations. As statistical models, these methods require a minimum amount of data before they can be applied. To apply Box-Jenkins models, the rule of thumb is to have at least 50 data points or 4 to 5 seasons of seasonal data, whichever is larger (Bowerman and O'Connell, 1993). The time series models to be studied in this thesis, including AR, MA, ARMA, ARIMA, are all variations of Box-Jenkins models.

A time series model assumes that the series is composed of two components: pattern and random error. The pattern of a time series can be: trend, cycle, and seasonal variations. Trend refers to the upward or downward movement that characterizes a time series, as indicated in Figure 2.1 (a). Cycle refers to recurring up and down movements around trend levels, as plotted in Figure 2.1 (c). The fluctuations can have non-uniform duration measured from peak to peak. One of the common cyclical fluctuations is the business cycle: expansion and contraction. Seasonal variations are periodic patterns in a time series that repeat themselves after a certain period, as indicated in Figure 2.1 (b). For example, the average monthly temperature is seasonal in nature since it directly measures changes in the weather.



(a) Trend

(b) Seasonal variation

(c) Cycle

Figure 2.1 Time series exhibiting trend, seasonal, and cyclical components

A time series, say, $\{x_t, t = 1, 2, \ldots, n\}$, can be represented by:

$$x_t = \hat{x}_t + w_t \tag{2.1}$$

where $\hat{x}_t$ is the estimated value from a model, and w_t is the residual. One method to test the goodness of the model obtained is to check whether the residual is random. Statistical tests can be applied to the residuals (Shumway and Stoffer, 2000a). Before time series models can be discussed in details, the concepts of the autocorrelation function (ACF) and the partial autocorrelation function (PACF) need to be introduced.

2.1.1 Autocorrelation and Partial Autocorrelation Function

Autocorrelations are statistical measures that indicate how a time series is related to itself over time. Given a series $\{x_t,\ t = 1, 2, \ldots, n\}$, its mean μ and variance σ_x^2 can be estimated by:

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_t\ , \tag{2.2}$$

$$\hat{\sigma}_x^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_t - \hat{\mu})^2\ . \tag{2.3}$$

In the given series, $\{x_t, t = 1, 2, \ldots, n\}$, take two points, x_t and x_{t+k}, that are separated by a lag of k. The covariance between these two values are called autocovariance and is defined as:

$$\gamma_k = \operatorname{cov}[x_t, x_{t+k}] = E[(x_t - \mu)(x_{t+k} - \mu)] \tag{2.4}$$

where E represents the mathematical expectation. If $\gamma_k = 0$, we can conclude that points x_t and x_{t+k} are not related to each other. When $k = 0$, we have:

$$\gamma_0 = E[(x_t - \mu)^2] = \sigma_x^2 \tag{2.5}$$

The autocorrelation function (ACF) is a function of lag k. ACF at lag k for series, $\{x_t\}$ is defined as:

$$\rho_k = \frac{\gamma_k}{\gamma_0} \tag{2.6}$$

From equation (2.6), we can see that ρ_k has no measuring unit and $-1 \le \rho_k \le 1$.

Partial-autocorrelations are another set of statistical measures used to evaluate the relationships among the series values. The partial autocorrelation function (PACF) is an extension of autocorrelation, where the dependence on the intermediate elements within the lag is removed. If the lag is 1 (i.e., there are no intermediate elements within the lag), the partial autocorrelation will be equivalent to autocorrelation, which is:

$$\phi_{11} = \frac{\gamma_1}{\gamma_0} = \rho_1 \tag{2.7}$$

For $k = 1, 2, 3, \ldots$, successively, we have

$$\phi_{22} = \frac{\begin{vmatrix} 1 & \rho_1 \\ \rho_1 & \rho_2 \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 \\ \rho_1 & 1 \end{vmatrix}} = \frac{\rho_2 - \rho_1^2}{1 - \rho_1^2} \tag{2.8}$$

$$\phi_{33} = \frac{\begin{vmatrix} 1 & \rho_1 & \rho_1 \\ \rho_1 & 1 & \rho_2 \\ \rho_2 & \rho_1 & \rho_3 \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 & \rho_2 \\ \rho_1 & 1 & \rho_1 \\ \rho_2 & \rho_1 & 1 \end{vmatrix}} \tag{2.9}$$

Most time series textbooks discuss the ACF and PACF in details. Refer to Shumway and Stoffer (2000) or Brockwell and Davis (1987) for details about ACF and PACF.

2.1.2 Autoregressive (AR) Models

Autoregressive models are created with the idea that the next value of the series, x_t, can be expressed as a function of the latest p values, $x_{t-1}, x_{t-2},\ldots, x_{t-p}$, where p determines the number of the latest data points needed to forecast the next value. This p value can be obtained by studying ACF and PACF values (Shumway and Stoffer, 2000). This will be discussed in detail in Chapter 3.

When the mean, μ, of the series $\{x_t\}$ is zero, an autoregressive model of order p, denoted by ***AR*(*p*)**, has the following form:

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \ldots + \phi_p x_{t-p} + w_t \tag{2.10}$$

where ϕ_1, ϕ_2, ..., ϕ_p are constants and w_t is a white noise series with mean zero and variance σ_w^2. The definition of white noise w_t ($t = 1, 2, \ldots, n$) is given as:

$$E[w_t] = 0 \text{ and} \tag{2.11}$$

$$\text{cov}[w_s, w_t] = \begin{cases} \sigma_w^2, s = t \\ 0, s \neq t \end{cases} \quad (0 \le s, t \le n) \tag{2.12}$$

If μ of the series is not zero, we can replace x_t by x_t - μ for $i = 1, 2, \ldots, t$ in (2.10). Then we have:

$$x_t - \mu = \phi_1(x_{t-1} - \mu) + \phi_2(x_{t-2} - \mu) + \ldots + \phi_p(x_{t-p} - \mu) + w_t$$

$$x_t = \mu(1 - \phi_1 - \phi_2 - ... - \phi_p) + \phi_1 x_{t-1} + \phi_2 x_{t-2} + ... + \phi_p x_{t-p} + w_t$$

Define $\alpha = \mu(1 - \phi_1 - \phi_2 - ... - \phi_p)$. We then get:

$$x_t = \alpha + \phi_1 x_{t-1} + \phi_2 x_{t-2} + ... + \phi_p x_{t-p} + w_t \qquad (2.13)$$

Equation (2.13) should be applied when the series has a non-zero mean value. Usually some transformations such as removing the mean should be performed before using a time series model. After the transformation, Equation (2.10) can be used for the new series.

Equation (2.10) can also be expressed compactly in the following form

$$\phi(B)x_t = \mathrm{w}_t \qquad (2.14)$$

where

$\phi(B)$ is an autoregressive operator of order p and has the form

$$\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p$$

B is the backward shift operator and $Bx_t = x_{t-1}$

w_t is the white noise

Equation (2.14) means that any given value x_t in the series is directly dependent on the p immediately preceding values.

The definition of a stationary time series is given by Brockwell and Davis (1987). If a time series is said to be stationary, its mean, variance and other statistical properties do not change over time. AR model may be used to model stationary or nonstationary series. But the accuracy of prediction will be unsatisfactory if the time series is nonstationary (Tse and Anderson, 1999). To achieve stationarity, a computational process called "regular differencing" (RD) is often used (Shumway and Stoffer, 2000). RD can be represented by differencing operator ∇, which is defined as:

$$\nabla x_t = x_t - x_{t-1} \qquad (2.15)$$

Differencing twice, denoted by $\nabla^2 x_t$, is defined as:

$$\nabla^2 x_t = (x_t - x_{t-1}) - (x_{t-1} - x_{t-2}) = x_t - 2x_{t-1} + x_{t-2} \qquad (2.16)$$

Figure 2.2 shows a simulated series from an AR(1) model with the form of: $x_t = 0.95x_{t-1} + w_t$, where w_t is the white noise following the normal distribution with mean of 0 and variance of 1.

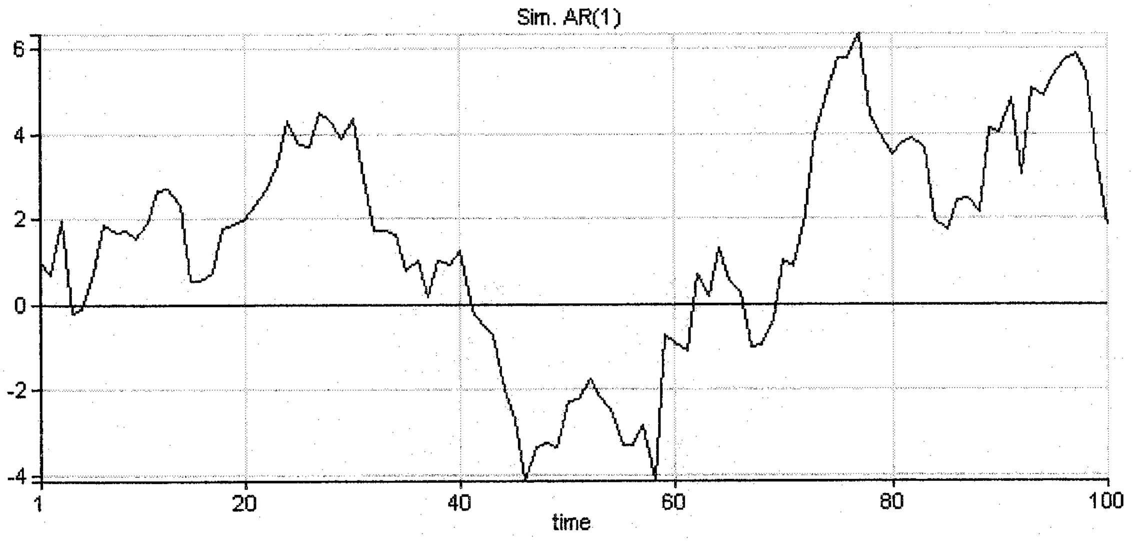


Figure 2.2 Simulated series from AR(1)

2.1.3 Moving-average (MA) Models

A moving-average model of order *q*, ***MA*(*q*)**, has the following form:

$$x_t = w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + \dots + \theta_q w_{t-q} \tag{2.17}$$

where there are q lags in the moving average and θ_1 , θ_2 ,..., θ_q are parameters that determine the overall pattern of the process, and w_i (t-$q \le i \le t$–1) are the residuals from the fitted model. They should behave like white noise if the model is suitable.

Similar to AR, the moving-average model can be expressed compactly in the following form

$$x_t = \theta(B)w_t \tag{2.18}$$

where

$\theta(B)$ is a moving-average operator of order q and has the form

$$\theta(B) = 1 - \theta_1 B - \theta_2 B^2 - \dots - \theta_q B^q$$

The origin of the term moving-average is derived from the fact that the moving-average model formula is simply a weighted average of a fixed number of past random errors that "moves forward" in time as t increases. The model contains $q + 2$ unknown parameters, μ, θ_1, $\theta_2,\ldots,$ θ_q, σ^2_{w}, which in practice have to be estimated from the data. Figure 2.3 shows a simulated series from MA(1) model with the form of: $x_t = 0.75w_{t-1} + w_t$, where w_t is the white noise following the normal distribution with mean of 0 and variance of 1.

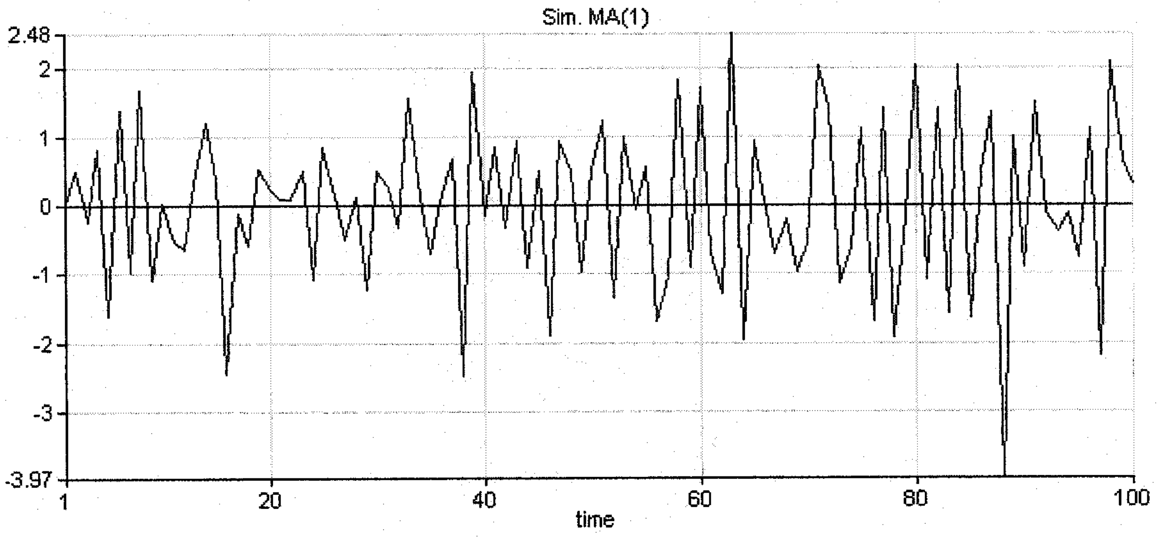


Figure 2.3 Simulated series from MA(1)

2.1.4 Mixed AR and MA (ARMA) models

Sometimes one has to include both autoregressive and moving-average terms in the model in order to model a time series well. These models that contain both AR and MA parameters are called "ARMA" models. A time series $\{x_t, t = 1, 2, \ldots, n\}$ is said to be ***ARMA(p, q)*** if x_t is stationary and

$$x_t = \alpha + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \ldots + \phi_p x_{t-p} + w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + \ldots + \theta_q w_{t-q} \qquad (2.19)$$

where $\alpha = \mu(1 - \phi_1 - \phi_2 - \ldots - \phi_p)$. In compact form, it is:

$$x_t = \frac{\theta(B)}{\phi(B)} w_t \qquad (2.20)$$

The definitions of $\phi(B)$ and $\theta(B)$ are given under equations (2.14) and (2.18), respectively. The model contains $p + q + 2$ unknown parameters, μ, ϕ_1, $\phi_2,\ldots,$ ϕ_p, θ_1, $\theta_2,\ldots,$ θ_q, σ^2_{w}. Figure 2.4 shows a simulated series from ARMA(1, 1) model with the form of: $x_t = 0.55x_{t-1} - 0.85w_{t-1} + w_t$, where w_t is the white noise following the normal distribution with mean of 0 and variance of 1.

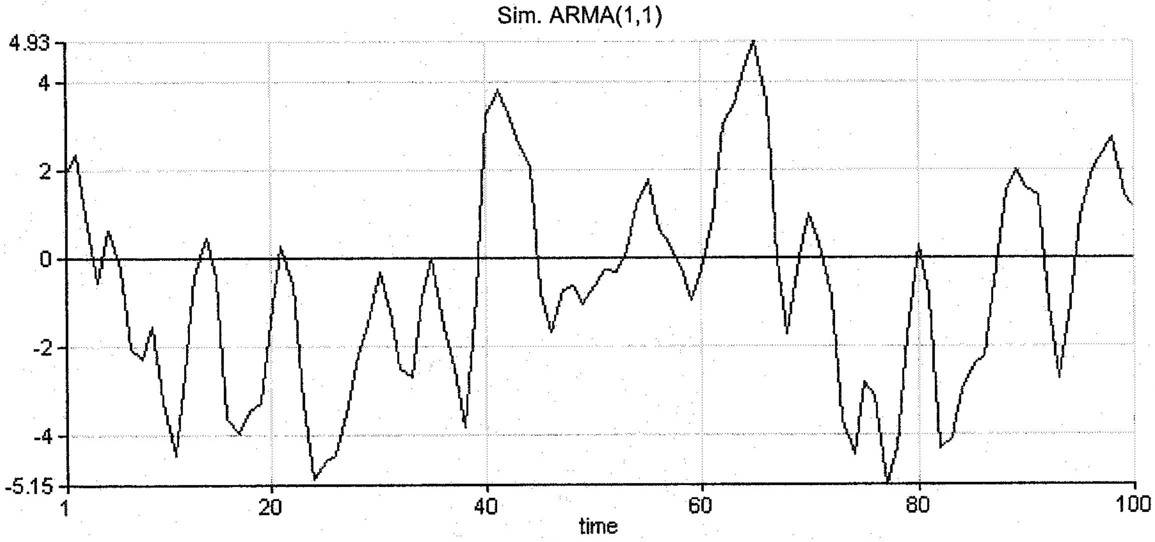


Figure 2.4 Simulated series from ARMA(1, 1)

2.1.5 Autoregressive Integrated Moving-average (ARIMA) Models

The three models discussed above are suitable only for stationary data. When the series is nonstationary, we should apply autoregressive integrated moving-average models. It is denoted by ***ARIMA(p, d, q)*** and is expressed as follows:

$$\phi(B)(1-B)^d x_t = \theta(B)\mathrm{w}_t \tag{2.21}$$

where $\phi(B)$ is an autoregressive operator of order p, $\theta(B)$ is a moving-average operator of order q, and d is the order of the differencing, which is usually 0, 1, or at most 2 in practice. The ARIMA models broaden the class of ARMA models to include differencing. The ARIMA models are often used because they include AR, MA, and ARMA models as special cases. For example, ARIMA(1, 0, 0) is equivalent to AR(1) model. Similar to AR, MA, and ARMA, ARIMA models are also linear models. In all these models, the prediction of the next value is represented as a linear combination of past observations. Figure 2.5 shows a simulated series from ARIMA(1, 1, 1) model with the form of: $(1 + 0.45B)(1 - B)x_t = (1 - 0.65B)w_t$, where w_t is the white noise following the normal distribution with mean of 0 and variance of 1.

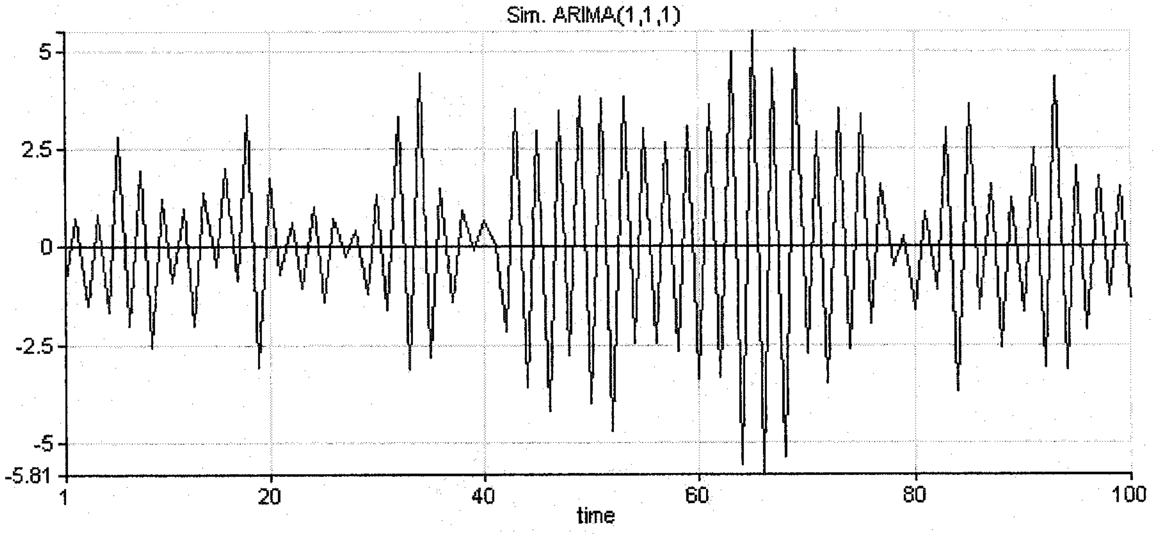


Figure 2.5 Simulated series from ARIMA(1, 1, 1)

2.1.6 Steps to Build Time Series Models

There are a few basic steps to fitting time series models to a series of data. These steps involve (1) plotting the data, (2) possibly transforming the data, (3) identifying the dependence orders of the model, (4) parameter estimation, and (5) model validation.

First, as with any data analysis, we should plot the data and inspect the graph for any anomalies. If the plotting of time series indicates that the series has a constant but nonzero mean, we can transform the series by removing the mean. If we observe a linear trend in the plotting, we can first fit a linear regression model to data series. Then remove the linear trend by deducting the obtained linear regression model from the original series. If the variability in the data grows with time, it will be necessary to transform the data to stabilize the variance. In such a case, log transformation is particularly useful:

$$y_t = \ln x_t \tag{2.22}$$

Other possibilities are the Box-Cox class of power transforms (Shumway and Stoffer, 2000a), as represented in equation (2.23):

$$y_t = \begin{cases} (x_t^{\lambda} - 1)/\lambda, & \lambda \neq 0 \\ \ln x_t, & \lambda = 0 \end{cases} \tag{2.23}$$

Methods for choosing the power λ are discussed by Johnson and Wichern (1992). In this thesis, we only use log transformation to stabilize the variance. There are other transformation methods such as filter, exponential smoothing, and moving standard deviation. Refer to Shumway and Stoffer (2000b) for more details.

After transforming the data, the next step is to identify preliminary model order(s): values of the autoregressive order, p, the order of differencing, d, and the moving average order, q. A plot of the data will typically suggest whether any differencing is needed. If there is no obvious pattern in the series, we do not have to make differencing. Practically, no more than twice differencing is necessary. When the preliminary values of d have been settled, the next step is to look at the ACF and PACF of data series after differencing to determine p and q. Another method to determine the model order(s) is to apply Akaike's Information Criterion (AIC). How to select model order(s) will be further discussed in Chapter 3.

After determining the model order(s), we can use least square method to estimate the parameters. The final step is to validate the model we build. Model validation requires the concepts of training errors and generalization errors.

2.2 Training Error and Generalization Error

To fit a model to a set of data points, we have two objectives: (1) minimize the deviations of the model from empirical data, and (2) minimize the deviations of the model from the data points yet to be observed. These two types of deviations are referred to as training errors and generalization errors, respectively. These two objectives sometimes conflict with each other. In this section, we will discuss training error, generalization error and their relationships.

2.2.1 Training Error

Training error measures the deviation (error) of the model from existing data points. For a time series $\{x_t, t = 1, 2, \ldots, n\}$, we may assume training error as:

$$\text{Training error} = \sum_{t=1}^{n}(x_t - \hat{x}_t)^2 \tag{2.24}$$

where $\hat{x}_t$ is the estimation at point t given by a fitted model. In this thesis, the training error is the same as residual sum of squares, which will be discussed further in Chapter 3.

For example, suppose that we have a set of data: $\{(1, 0.4), (2, 1.3), (3, 1.9), (4, 2.8), (5, 3.2), (6, 4.1)\}$. The best linear regression model for this set of data is: $y = 0.717x - 0.227$, shown in Figure 2.6. With equation (2.24), the training error for this linear model on the given data set is:

$(0.717*1\text{-}0.227\text{–}0.4)^2 + (0.717*2\text{-}0.227\text{-}1.3)^2 + (0.717*3\text{-}0.227\text{-}1.9)^2 + \ldots + (0.717*6\text{-}0.227\text{-}4.1)^2 = 0.008186 + 0.008534 + 0.000613 + 0.024994 + 0.025296 + 0.0005669 = 0.068190$

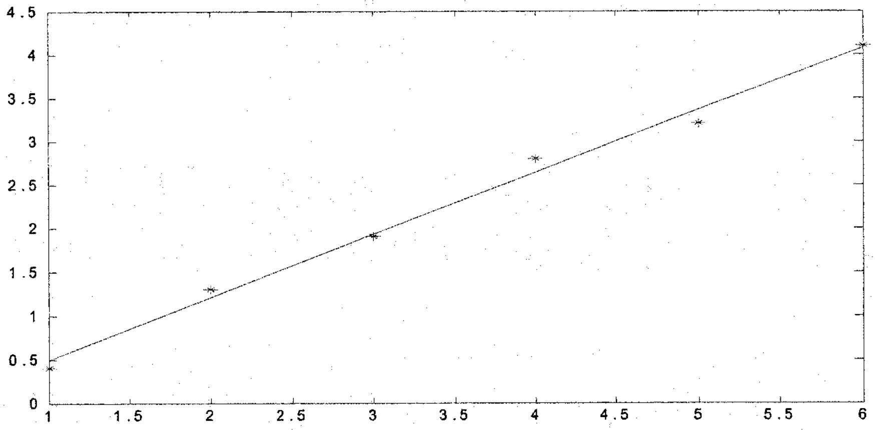


Figure 2.6 Linear regression from given data set

Linear regression applies Empirical Risk Minimization (ERM). ERM tries to minimize training error only, e.g.: using the least square method. Generally, the more free parameters a model has, the better fitting we will get for empirical data, i.e., the model will generate the smaller training error. When the model is complex enough, the model will fit all empirical data perfectly. Then, we will have a zero training error.

2.2.2 Generalization Error

In addition to the given data points, a good model should have good prediction ability to data points that have yet to be observed, which is called the generalization ability. Assume that we obtain m additional data points $\{x_{n+1}, x_{n+2}, \ldots, x_{n+m}\}$ after a model has been built using series $\{x_t, t = 1, 2, \ldots, n\}$. The generalization error can be defined as:

$$\text{Generalization error} = \sum_{i=1}^{m} (x_{n+i} - \hat{x}_{n+i})^2 \qquad (2.25)$$

Consider the example used in the previous section. Suppose that 4 additional points are obtained: {(7, 5.1), (8, 5.7), (9, 6.8), (10, 7.3)}. These 4 data points together with the previous 6 data points are plotted in Figure 2.7. The generalization error for the linear model built on the original series can be calculated by using equation (2.25) as:

$$(0.717*7-0.227-5.1)^2+(0.717*8-0.227-5.7)^2+(0.717*9-0.227-6.8)^2+(0.717*10-0.227-7.3)^2$$
$$= 0.09404 + 0.03591 + 0.32761 + 0.12619 = 0.58378.$$

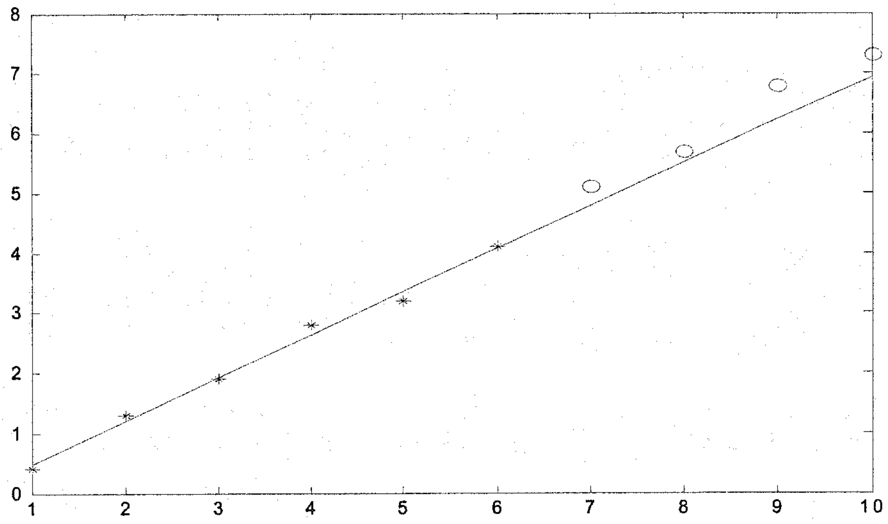


Figure 2.7 Linear regression line and additional data

The generalization error can only be calculated after additional data points are obtained. Since we do not know the future in advance, generalization error is hard to control. Meanwhile, we cannot totally depend on the training error as a predictor for generalization error. If a model has enough free parameters, it will fit the given data points perfectly. However, such a model may have a poor ability of generalization. This is the well-known phenomenon of “over-fitting”, which means that there are too many free parameters in the model. Cortes (1995) discusses the relationships among training error, generalization error, model complexity, and training set size in her thesis. The measure that she uses to describe the model complexity is called capacity, e.g. the number of free parameters.

2.2.3 Relationship Between Training Error and Generalization Error

A low training error does not guarantee a low generalization error. As we already know, we still want to use training error as an estimator of generalization error because generalization error can hardly be obtained. The difference between the generalization and the training errors is influenced by training set size, *l*, and the capacity of a model, *h*. The capacity of a model is a determining factor for this difference. Models with a large capacity, e.g.: large number of free parameters, relative to the size of the training set are likely to obtain a low training error. However, they might just be memorizing or over-fitting the patterns and hence exhibit a poor generalization ability. On the other hand, when the capacity is too small for the given sample data, models may under-fit the data and exhibit both high training and generalization errors. In between these capacity extremes, there is an optimal capacity for which the lowest generalization error may be obtained.

Learning curves describe the variation of the expected training and generalization errors as a function of the training set size *l* for a fixed model. Figure 2.8 (Cortes, 1995) shows a typical example of such learning curves. The training error is zero for small sizes *l* of the training set. In this case, the model with a certain capacity gets all the patterns right but over-fits the data and exhibits poor generalization ability. The generalization error improves as additional data points are used for training. When training set size goes infinity, generalization error will be equal to training error, i.e.:

$$\lim_{l\to\infty} C(h,l) = 0 \tag{2.26}$$

where $C(h, l)$ represents the absolute difference between generalization error and training error.

The influence of the capacity on the size of the difference $C(h, l)$ can be illustrated by plotting the expected training and generalization errors for a fixed training set size as a function of the capacity h of the model, as shown in Figure 2.9 (Cortes, 1995). High capacity models exhibit a low training error but a high generalization error due to over-fitting. A low capacity model may exhibit both high training and generalization errors.

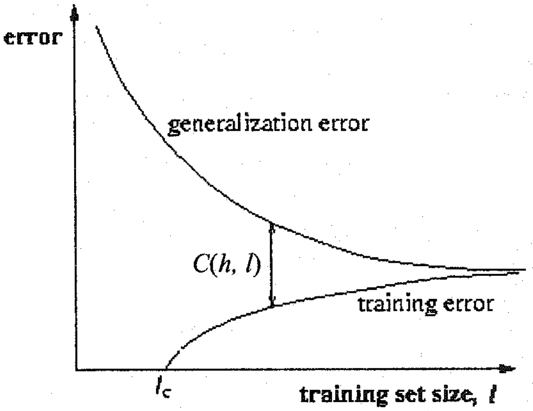


Figure 2.8 Learning curves for a model

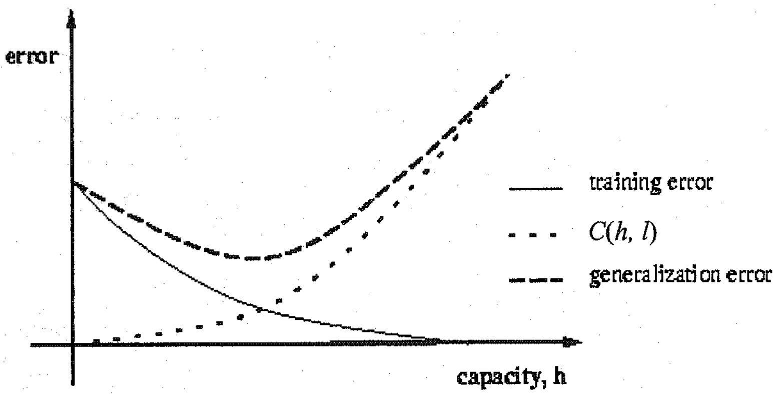


Figure 2.9 Error under influence of capacity

There is a principle: "The simplest explanation is the best". If two models achieve the same training error, we will choose the one with the smaller capacity. As a result, a good training algorithm for minimizing the generalization error should not just minimize the error on the training set. It should also provide a means for controlling the capacity of the model so that this optimal capacity can be reached. Empirical Risk Minimization (ERM) and capacity control are combined in Structural Risk Minimization (SRM) (Cortes, 1995). For more information about SRM, please refer to Cortes (1995).

2.3 Support Vector Machines

Support vector machines (SVMs) result from statistical learning theory and are proposed by Vapnik (Vapnik, 1997). They are learning machines for solving pattern recognition and regression problems. They are now gaining popularity due to many attractive features and promising empirical performance (Mukherjee, et al., 1997, Muller et al. 1999, Scholkopf et al., 1998).

2.3.1 Research and Applications of Support Vector Machines

Traditional approaches like linear or nonlinear regression only try to minimize training error. SVM considers both training error and capacity control. In this sense, it embodies SRM. It is this difference that equips SVMs with a greater potential to generalize, which is the goal of statistical learning. Additional details on SVM will be discussed in detail in Chapter 4.

Hearst (1998) gives a general introduction and discussion of SVMs followed by some real world applications from other researchers. She also lists current developments and open issues for further research. This is a good introductory paper for SVM.

Muller et al. (1999) applies SVM to time series prediction. In their work, they compare SVM prediction with radial basis function networks. Two benchmark time series: Mackey Glass and Santa Fe Competition are evaluated by applying different prediction methods. In both cases SVMs show an excellent performance. However, determining the proper parameters of SVMs, regularization constant (λ) and maximum unpunished error (ε), is always a problem as stated in the paper. The paper suggests bootstrap method and cross validation method to solve this problem but no detail is provided. Nevertheless, both bootstrap and cross validation are computation intensive.

Mukherjee et al. (1997) extensively test and compare SVMs on the chaotic Mackey Glass time series with different approximation techniques, including polynomial and rational approximation, local polynomial techniques, Radial Basis Functions, and Neural Networks. According to their results, SVMs provide the best prediction. They also study the sensitivity of SVM to its parameters and the embedding dimension. However, no explanations are provided on how they choose the kernel (refer to Section 4.1.2), λ, ε, and parameters of the selected kernel function to minimize the generalization error.

Drucker et al. (1997) do explain how to choose the optimal regularization constant λ in their paper. They generate 200 training data points and 40 validation data points. The best λ value that minimizes generalization error on the validation set is found by trial and error. However, they consider selecting λ only, instead of both λ and ε. Scholkopf et al. (1998) propose a new SVM algorithm, called ν-SV regression (ν-SVR), which adjusts automatically the parameter ε. But it brings another new parameter v to be determined by the user.

Gestel et al. (2001) apply Least Square SVM (LS-SVM) to financial time series prediction. The examples include prediction of the US short-term interest rate. The results show that in one step ahead forecasting, SVM gives the best results among all the models compared.

2.3.2 Available Software for Support Vector Machines

Computer programs need to be developed to implement SVM. Fortunately, there are many software packages available for SVM regression and classification, such as Collobert and Bengio's ***Torch***, Stefan Ruping's ***mySVM*** for Windows and Unix, Chih-Jen Lin's ***Looms***, Musicant's ***ASVM***, MATLAB ***Support Vector Machine Toolboxes*** by Gavin Cawley and Steve Gunn, ***BSVM*** by Chih-Wei Hsu and Chih-Jen Lin, and Alex Smola's ***Quadratic Optimizer for SV Pattern Recognition*** [48]. With the help of these free software resources, we can reduce the amount of duplication work. Some of these packages, such as Stefan Ruping's ***mySVM,*** even provide the source code. A user can easily realize his own needs by modifying the available code. In this thesis, we use two software packages: Matlab SVM toolbox by Steve Gunn and Stefan Ruping's *mySVM.*

The Matlab SVM toolbox by Steve Gunn (1998) is very user friendly. The toolbox includes a set of Matlab files with one running as user interface. It incorporates nine kernel functions: linear, polynomial, Gaussian Radial Basis Function (RBF), multiplayer perceptron, linear spline, linear Bspline, trigonometric polynomial, and exponential RBF. The regression or classification outcome will show as a plot in the user interface, which is very helpful in analyzing the regression results. Like other free resources, it does not have a detailed manual and sometimes the system is not stable. In addition, because it is a toolbox on top of Matlab, the running speed is slow.

Stefan Ruping's *mySVM* is written in C++ language. With the given source code, it is easy to add functions on top of *mySVM.* Cross validation is a build-in function of its training algorithm, which is an important feature that Gunn's toolbox does not have. It is fast because it is an independent system so we can apply it to problems with larger sample sizes. However, *mySVM* is not so visually attractive as the Matlab SVM toolbox.

2.4 Other Methods

In this section, we examine a few other methods that have potential in vibration analysis and degradation prediction.

2.4.1 Online Tool Wear Identification

In a machining process, a work piece usually moves at a high speed relative to the cutting tool. Because of the friction between cutting tool and work piece, the cutting tool will gradually wear out. A tool wear process can be viewed as a degradation process. It usually follows a non-decreasing, non-linear trend, consisting of three distinct periods: initial wear period, normal wear period, and accelerated wear period. Wang (1994) proposes a method for monitoring tool wear and identifying new tool wear function under changed working conditions. This method is also a forecasting method because it is capable of predicting future tool wear level.

The amount of total tool wear, $R(t)$, is assumed to be a third order polynomial function of time t: $R(t) = b_3t^3 + b_2t^2 + b_1t$. $R(t)$ may have different coefficients (b_1, b_2, and b_3) under different working conditions. Exponential Weighted Moving Average (EWMA) control chart is used to detect whether the average tool wear function has deviated from the existing tool wear function, i.e., out of control. When the process is out of control, an identification algorithm is used to update the tool wear function. This algorithm is based on the prior information on this function and the latest available tool wear measurement data.

To estimate the parameters of a tool wear function, both part measurements and prior information are used. Based on part measurements, curve fitting with the least square method can be used to identify the parameters of $R(t)$. But it cannot reflect promptly to working condition change. The new method proposed by Wang (1994) aims at making the best estimation of the parameters of the tool wear function $R(t)$, in terms of minimizing not only the sum of the squared errors between the part measurements and the estimations, but also the difference between the estimated parameters and the prior parameters of the tool wear function. The objective function is:

$$OBJ = \sum_{i=mc}^{mi} (X_{ti} - \hat{X}_{ti})^2 + \sum_{i=1}^{3} k_i (\theta_p - \hat{\theta})^2 \tag{2.27}$$

where *mc* is the measurement point number that working condition starts to change, *mi* is the current measurement point number, $mi \geq mc$, X_{ti} is the part measurement at time t_i, $\hat{X}_{ti}$ is the estimated part measurement at time t_i, k_i is the weighting factor for parameter deviation from the prior value, θ_p is the parameter vector of prior information of tool-wear function, and $\hat{\theta}$ is the estimation of parameter vector of tool wear function.

Equation (2.27) is composed of two parts: (1) the sum of square errors between the part measurements and estimated value, and (2) the sum of square errors between the prior parameters and the estimated parameters. A weighting factor *k* is used to combine the two terms in the objective function. The *k* value should be selected with care, which is important to ensure the effectiveness of the algorithm. Finally, on-line optimal tool replacement/adjustment decision can be made based on the updated tool wear function.

The method by Wang (1994) can be used for trend monitoring and future degradation prediction. To implement this method, prior information on the trend is needed. But sometimes the prior information is not obtainable. Meanwhile, how to select a proper weighting factor *k* deserves further investigation. The selection of the number of the most recent data points used to apply the least square method is not discussed either. The number of points to be considered is a factor that can greatly change the optimization result. We can apply the method by Wang (1994) when we have some prior knowledge on the vibration trend. This can be obtained either by experience of similar equipment or experiment.

2.4.2 Continuous State System Reliability Analysis

As equipment is used, it will deteriorate and finally fail. The reliability of the equipment will drop as time goes. Binary reliability theory assumes that the device under consideration is in one of two possible states: working or failed. Multi-state reliability analysis allows the device under consideration to have more than two possible states. The possible state can be: perfectly working, completely failed, or some intermediate states due to degradation. The states discussed in multi-state reliability analysis are discrete. Continuous state reliability models assume that the state of the device can be represented by a continuous variable. Continuous state system reliability analysis is another method of degradation forecasting. After the continuous state reliability model is obtained, the device's remaining life distribution becomes available.

Zuo et al. (1999) suggest four models for continuous state system reliability analysis. The authors believe that gradual degradation of system performance is caused by the degradation of some system parameters. The process of degradation is usually a continuous state process. One of these four models they propose is introduced as follows. Firstly, the observation data are collected. This work requires that at each time point t_i, there must be a certain number of observations to estimate statistical characteristics. In their example, 7 data points are available at each t_i. Then a distribution that can adequately represent the degradation data at each t_i is picked. The distribution can be Weibull, Gamma, or others. After a distribution is selected, we can estimate the parameters θ_i of the selected distribution for each t_i. Note that θ_i is a vector because each distribution may have more than one parameter. So we have $\theta_{i,1}$, $\theta_{i,2}$, ..., $\theta_{i,n}$. Next, θ_i are fitted as a function of t, $\theta_i(t)$, from $\theta_{i,1}$, $\theta_{i,2}$, ..., $\theta_{i,n}$. At last, we can use the developed model for remaining life estimation.

This work provides useful statistical models for system degradation prediction. The models are general and can be applied to vibration prediction. All the methods require that there are multiple observations at the given observation time points. However, in some real life cases, such as condition monitoring, only a single observation is available at each time point. In these cases, the models proposed are not applicable.

2.4.3 General Path Model

Lu and Meeker (1993) propose general path models to predict the growth of fatigue cracks. They believe that the development of fatigue cracks follows a certain general model (general path), which can be obtained from Paris Law. The parameters of this general path are random variables following certain distributions. That is why at each time point we have different observations. They believe that the degradation model obtained from existing knowledge is usually nonlinear with multiple variables. The random effects usually follow an unknown multivariate distribution. When a closed form of degradation model is available, a two-stage method is proposed to find reasonable estimates of the parameters of certain distribution. In the first stage, estimates are obtained by using least square method for each sample path (linear or nonlinear regression). Then we need to determine whether re-parameterization is needed so that multivariate normal assumption can be used. The multivariate transformation and test suggested by Andrews et al. (1971) can be used here. Following the procedure provided, one can obtain the two stage estimates of the basic model parameters.

When the closed form of degradation model cannot be obtained, Monte Carlo simulation may be used to estimate the time-to-failure distribution. Using the algorithm suggested, one can obtain the estimate of cumulative density function (Cdf). The comparison of suggested path model and conventional time-to-failure analysis is made. It shows that the degradation method provides a better estimate after stop time than other analyses such as fitting Weibull, Normal, or Lognormal distributions. This is because the degradation analysis method directly models the relationship between degradation and time. It takes into account of the amount of degradation in the censored observations when estimating Cdf. The traditional methods ignore the crack accumulation (degradation) of the sample, which does not fail before the stop time.

The general path model is a general method for degradation modeling and forecasting. However, the general degradation path must be known first. In their example, Paris Law is used to develop the general path of fatigue growth. It has similar limitations as the two other methods described earlier, i.e., prior knowledge is necessary in model building. When there is little prior knowledge or experience at all, these models may be hard to apply.

2.4.4 Application of Neural Networks on Vibration Forecasting

Neural networks are inspired by the way biological nervous systems process information. Neural networks use a set of processing elements (or nodes) loosely analogous to neurons in the brain. These nodes are interconnected in a network and weights are associated with each input to the nodes. As it is exposed to data, weights are adjusted through training. Patterns in data are then identified. In a sense, the network learns from experience just as humans do. This distinguishes neural networks from traditional computing programs that simply follow instructions in a fixed sequential order. There are different types of neural networks, such as Artificial Neural Networks (ANN), Recurrent Neural Networks (RNN), and Feed-Forward Neural Networks (FNN). Neural networks is a hot research topic. For more information on neural networks please refer to the book of Fausett (2002).

Tse and Atherton apply recurrent neural networks to machine deterioration prediction using vibration based fault trends (1999). This paper aims to an intelligent machine condition prognostic system, which can automatically predict the life span of a defective machine and advise the operator of appropriate remedy. The basic idea is to make forecasting by using a time series data. A prognostic method is developed to predict the rate of machine deterioration by using recurrent neural network.

Classical autoregressive models and neural networks can be used to achieve time series prediction. Five autoregressive models: vector autoregressive model, Burg algorithm, autoregressive model, bilinear model approach, and threshold autoregressive model are introduced and compared with two types of neural networks: feed-forward neural network (FFN) and recurrent neural networks (RNN). Normalized Akaike Information Criterion (NAIC) and variance are used together to evaluate the effectiveness of the models. A sunspot activity prediction example is studied (Tse and Atherton, 1999).

In industrial applications, Tse and Atherton (1999) apply the methods to two cases: corrosion of bearing in a cooling tower fan and defective gearbox of a compressor in a chemical plant. The future degradation level of a machine (measured by normalized combined vibration signals) instead of the life span is predicted.

Tse and Atherton (1999) use an existing neural networks method in the area of machine vibration prediction. The vibration level is used as the indicator of machine degradation level. Excessive vibration over certain limits is assumed to be a failure. Thresholds must be set up from which to tell whether the vibration is under normal condition or not.

However, this issue is not discussed in the paper. Neither time series models nor neural networks need the prior knowledge, while it is a requirement for the work of Wang (1994), Zuo et al. (1999), and Lu and Meeker (1993).

2.5 Summary

In this chapter, we have discussed some forecasting methods, which can be applied to degradation prediction based on vibration analysis. Section 2.1 introduces some time series models. In Section 2.2, two errors in forecast modeling are introduced: training error and generalization error. Support vector machine, a promising way for time series prediction is briefly introduced in Section 2.3. The current research applications of SVM and available SVM tools are introduced. In section 2.4, we introduced other possible candidates for vibration prediction. Their potential uses and limitations are also analyzed.

In this thesis, we aim to develop models for prediction of equipment deterioration based on condition monitoring. Vibration is used as an indicator of equipment degradation level. If we apply the condition monitoring to equipment, we may assume little or no prior knowledge. The methods applicable to our study include time series methods, support vector machines, and neural networks. In this thesis, time series statistical models and support vector machines are applied to vibration data for future vibration level prediction.

Chapter 3. AIC Considering Generalization Error to Determine the Best Forecasting Time Series Model

To fit a time series model, such as autoregressive (AR), moving average (MA), and autoregressive integrated moving average (ARIMA), to observed data, the order(s) of the selected model needs to be chosen first and then the parameters can be estimated. The least square method is usually applied to find the parameters of the selected model. The problem of choosing model order(s) is usually the main task for model building. There are a variety of techniques for choosing the model order(s). One method uses the autocorrelation function (ACF) and partial autocorrelation (PACF) of the data. We call the diagrams of ACF and PACF with different lags ACF and PACF correlograms. Order(s) of the model can be determined based on the obtained ACF and PACF correlograms by observation.

Akaike's Information Criterion (AIC) and its expanded forms provide another way for order(s) selection. The idea of applying AIC as a model order selection criteria is that the smaller the AIC value, the better the model will be. AIC provides a measure of goodness of fit of the model to the empirical data, not the forecasting ability. So there are occasions that the model with the smallest AIC value is not the forecasting model providing the smallest forecasting error. The split sample method can be used to measure the generalization error on the validation set. This chapter proposes a new method combining AIC and split sample validation. Application results show that this new method can find a model providing better prediction compared with applying AIC only.

3.1 ACF and PACF for Time Series Model Order(s) Selection

ACF and PACF are discussed in Chapter 2. The best model order(s) describing a time series can be obtained by studying the values of ACF and PACF. The behaviors of the

ACF and PACF for AR, MA, and ARMA models are shown in Table 3.1 (Shumway and Stoffer, 2000a).

Table 3.1 Behavior of the ACF and PACF for AR, MA, and ARMA models

	AR(p)	MA(q)	ARMA(p, q)
ACF	Tails off	Cuts off after lag q	Tails off
PACF	Cuts off after lag p	Tails off	Tails off

For an AR(p) model, the ACF will decay exponentially and the PACF will cut off after lag p. On the other hand, if we notice this phenomenon from correlograms illustrating ACF and PACF values, we can tell that it is an AR(p) process. For example, the ACF and PACF diagrams of a simulated AR(1) process without noise, $x_t=1+0.95x_{t-1}$, are plotted in Figure 3.1.

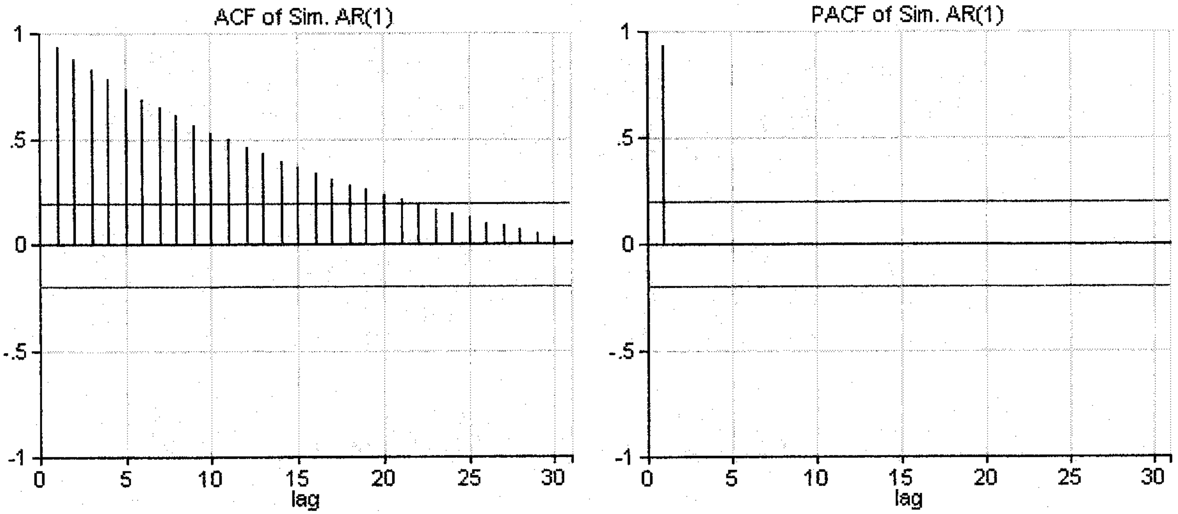


Figure 3.1 ACF and PACF correlograms of simulated AR(1)

In Figure 3.1, the two horizontal lines at center are upper and lower 95% confidence interval (CI) of ACF and PACF. The 95% confidence interval gives two bounds: upper and lower, from given data. It means that ACF or PACF values will fall into the two bounds at 95% of the time. If the ACF or PACF values are outside the confidence interval, we can regard them as significant. For example, in the ACF diagram of Figure 3.1, ACF at lag 3 is outside CI, we can conclude that ACF at lag 3 is significant.

Judging from ACF and PACF correlograms is subjective, which depends on experience. For example, if we have a series of data simulated from an ARMA model, it will be difficult to tell the model orders from correlograms since both ACF and PACF will be tailing off. Especially when there exist seasonal factors, there will be lots of terms in ACF and PACF with significant values. Moreover, when the raw data is noisy and when the model is complex, it is often very difficult to tell the model order(s) directly from ACF and PACF correlograms. For example, consider a simulated ARMA(1, 1) model:

$$x_t - 0.75 * x_{t-1} = w_t + 0.65 * w_{t-1}$$

where w_t is a white noise with normal distribution with mean of zero and variance of 2. The ACF and PACF correlograms are plotted in Figure 3.2.

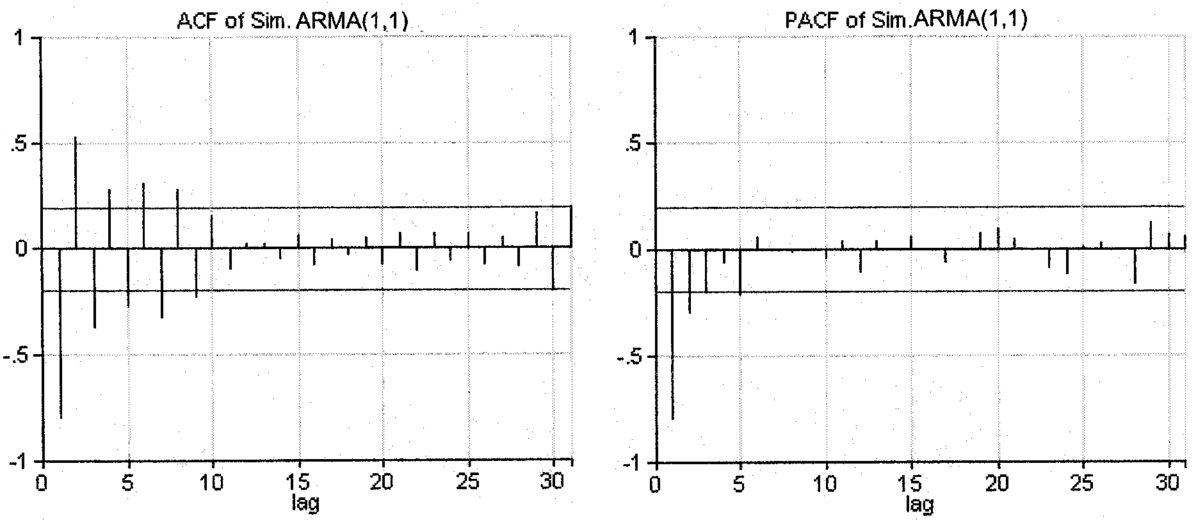


Figure 3.2. ACF and PACF correlograms of simulated ARIMA(1, 1)

From Figure 3.2, some people may argue it is an AR(2) or AR(3) model since ACF tails off while PACF cuts off after 2 or 3 lags. But in fact, as we know, it is simulated from an ARMA(1, 1) model. This example shows that it is hard to tell correctly what the model order is simply from the diagrams, especially to novices.

The method of judging time series model order(s) from ACF and PACF diagrams is direct and efficient. But it needs experience and is effective only to relatively simple models with low noise. Another method is to apply Akaike's Information Criterion as an evaluating criterion. It is discussed in the next section.

3.2 Akaike's Information Criterion and Its Revised Forms

Akaike's Information Criterion (AIC) and its expanded forms provide another way for order(s) selection. They have been widely accepted and used (Fugate et al., 2001, Tse and Atherton, 1999, Brie, et al., 1994).

Considering a group of data, $\{x_t\}$, we have an obtained model to calculate the estimated values of $\hat{x}_t$. Then the residual sum of squares under this model is:

$$RSS = \sum_{t=1}^{n} (x_t - \hat{x}_t)^2 \tag{3.1}$$

where n is the sample size.

Suppose that we consider a regression model with k coefficients for $\{x_t\}$ and denote the maximum likelihood estimator for the variance as:

$$\hat{\sigma}_k^2 = \frac{RSS_k}{n} \tag{3.2}$$

where RSS_k denotes the residual sum of squares under the model with k regression coefficients. Then, the goodness of fit for this particular model suggested by Akaike (1969, 1973, and 1974) is:

$$AIC = \ln \hat{\sigma}_k^2 + \frac{2k}{n} \tag{3.3}$$

The value of k yielding the minimum AIC specifies the order of the best model.

Having introduced what AIC is, let's have a look at the procedure of using AIC for model order(s) selection. Firstly, a certain type of model is selected, say, AR. Secondly, we try different orders of the model: 1, 2, …, p. The search range of p can be determined from ACF and PACF diagrams. For example, if ACF diagram tails off and PACF diagram has a cut off around 5, we can set the search range to be 5. Then we calculate the AIC value for each of the AR candidate model with specific order(s). At last, we pick the AR(k) with the smallest AIC as the optimal forecasting model based on the current data.

Now, we can have a deeper discussion on why AIC works. It is reasonable to minimize the error generated from training data, the average squared error $\hat{\sigma}_k^2$. As k increases, $\hat{\sigma}_k^2$ decreases monotonically. When k is large enough, the model will fit all the data exactly, which generates a zero training error. However, as explained in Chapter 2, such a model with so many free parameters has poor generalization ability. Such a model may lead to great estimation error at points other than the training data. It is the so-called overfitting problem that we must avoid. Let's look at the U.S. population data digitized from Shumway and Stoffer (2000a). The series contains the U.S. population every 10 years from 1910 to 1990, as plotted in Figure 3.3. We can use an eighth order polynomial function to fit the nine observations perfectly, which gives us a zero *RSS*. But look at its prediction; the model predicts that the population of the U.S. will be about 900 million in the year 2010. It does not make sense at all.

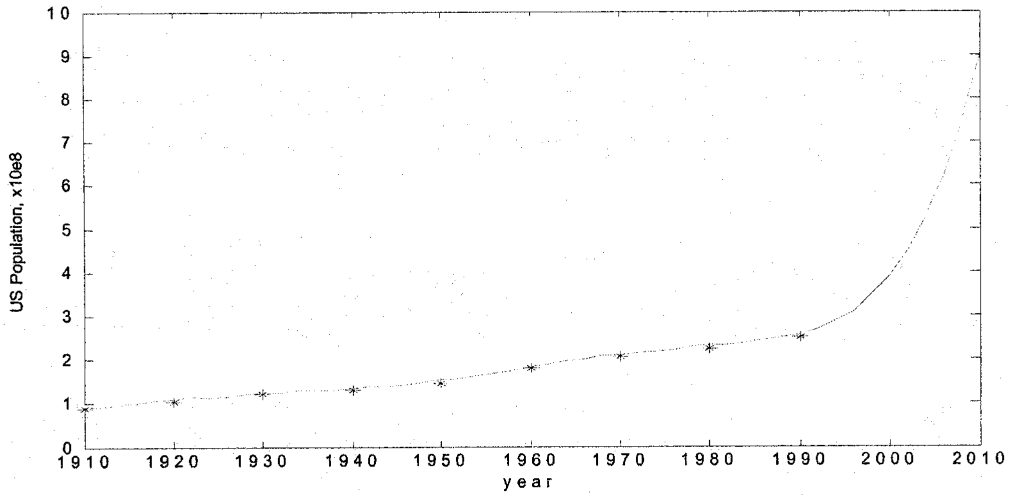


Figure 3.3. A perfect fit and a terrible forecast

From the example above, we know that adding more parameters may generate smaller *RSS* in training points but may also lead to bad forecasts because of the problem of overfitting. On the other hand, if the number of parameters is too small, the model will not be able to fit the data adequately either. Thus, there exists an optimal number of parameters somewhere in between these two extremes (Cortes, 1995).

Therefore, we ought to penalize models with too many parameters. The second term in equation (3.3) represents such a penalty. The choice for the penalty term given by equation (3.3) is not the only one and a considerable literature is available advocating different penalty terms. When the sample size is small and the relative number of parameters is large, a revised form of AIC: AICC is usually applied (Sugiura, 1978). It is defined as:

$$AICC = \ln \hat{\sigma}_k^2 + \frac{n+k}{n-k-2} \tag{3.4}$$

When the sample size is large, another revised form of AIC, called SIC, is proposed (Schwarz, 1978). It is derived based on Bayesian arguments.

$$SIC = \ln \hat{\sigma}_k^2 + \frac{k \log n}{n} \tag{3.5}$$

All these forms of the AIC evaluate each model on its own merits without having to look at the ACF or PACF correlograms. They are easy to use and now are widely accepted. Normalized Akaike Information Criterion (NAIC), very similar to AIC, is applied by Tse and Atherton (1999) as the criterion to evaluate the prediction models.

The AIC and its modified forms aim at minimizing the sum of squared errors in the training set. However, since we are looking for a forecasting model, we should make sure that the model has small generalization error.

The best forecasting model should generate the smallest generalization error. Of course, usually we do not know the generalization error because we do not know the "truth" of the future. Because the nature of AIC is a goodness of fit test, the model with the smallest AIC value is not necessarily the best forecasting model. Cross validation and split-sample validation are two methods directly measuring generalization errors. They will be discussed in the next section.

3.3 Cross validation and Split-Sample Validation

A method to validate a forecasting model is to collect more data and see how well the model fits the new data. This technique, however, is frequently not viable, because one cannot wait. A compromise is to leave out some of the data, fit the model to the remaining data, and see how well it fits to the data that was left out. This is called cross validation. Cross validation is a method used extensively in neural networks. It estimates the generalization error based on "resampling" (Weiss and Kulikowski, 1991; Shao, 1993; Plutowski, et al., 1994).

In k-fold cross validation, the data is divided into k subsets of (approximately) equal size. The model is trained k times, each time leaving out one of the subsets from training. The omitted subset is used to compute a generalization error. These k generalization errors are denoted by RSS_1, RSS_2, ..., RSS_k. If k equals the sample size, this is called "leave-one-out" cross validation. It means each time only one data point is left out for validation. "Leave-v-out" is another version of cross validation that involves leaving out all possible subsets with v data points.

Cross validation may leave out some past data points and use later values to estimate the past values. However, a univariate model always uses the past observations to predict later values. As a result, we cannot apply cross validation to univariate time series models. Another method called split-sample validation simply divides the data into two sets: the training set and the validation (testing) set. In the split-sample method, only a single subset (the validation set) is used to estimate the generalization error, instead of k different subsets; i.e., there is no "crossing". The split-sample method can be applied to univariate time series models because it does not violate the assumption that data points should be obtained from equal time intervals. In addition, the set containing earlier data points is always used for training while the other data set is used for validation. By dividing the sample into two sets, the generalization error on the validation set can be calculated using the model obtained from the training set. The smaller the generalization error is, the better the forecasting model will be.

One problem with the split-sample technique is that we are unable to use all the data available to us to fit the model. For time series models, we can only use the most current data points as the validation set. These data points cannot be used for model building. The most current data points are usually the most important for forecasting. The model obtained this way may not be satisfactory. The advantage of the split-sample method is that it uses generalization error as the criterion for model selection. To build a forecasting model, we should try to find a model that has the minimum generalization error.

In this chapter, we suggest a new method considering both AIC and generalization error for best forecasting time series model selection. This method combines the advantages of AIC and the split-sample method.

3.4 AIC Considering Generalization Error for Prediction Model Selection

3.4.1 The Method

AIC measures the goodness of fit of a model to data while the generalization error shows the prediction ability of the model. Minimizing the generalization error is usually the goal of the work of forecasting model building. Because of the characteristics of AIC and the split sample methods, we propose a method, which combines the merits of the two methods. The procedure of the proposed method is outlined below:

1. Plot the data.
2. Possibly transform the data.
3. Choose a certain time series model. It may be AR, ARMA or ARIMA. This may be based on examining the time series plotting and its ACF and PACF correlograms as discussed in Section 3.1.
4. Specify a search range(s) of model order(s).
5. Try a certain model order(s) within the search range(s), e.g.: AR(3).
6. Use the least square method to find the parameters of this specific model.
7. Calculate the AIC (or one of its the expanded forms) value of this model.

8. Split the sample and find the model parameters using only the data in the training set.
9. Use the newly fitted model to forecast validation set.
10. Calculate generalization error on the validation set. The generalization error is defined as:

    $$GE = RSS / m \qquad (3.6)$$

    where *RSS* is the residual sum of square obtained from the given model on validation set and *m* is the size of the validation set.
11. Calculate the combined value of AIC and GE, *comb*:

    $$comb = \text{AIC} + s^{*}\ln\text{GE} \qquad (3.7)$$

    where *s* is the weight factor of generalization error.
12. Go to step 5 to try a different order(s) until all candidates in the search range(s) are considered.
13. Pick the model order(s) that has the smallest *comb* value as the best forecasting model.
14. Diagnose the obtained model by residual randomness tests. If the obtained model does not pass the residual tests, go back to step 3 and rerun the process.

The key issues involved in applying this proposed procedure are discussed in the following section.

3.4.2 Discussions

3.4.2.1 Testing Sample Size

Given a data series with *n* data points, how do we split the sample into two set, a training set and a validation set to implement the proposed method? The size of the validation set determines the number of data points that are not used in the model building. If it is too big, we ignored too many data points. The most recent data are usually the most important for forecasting. If the size is too small, we do not have enough confidence to tell the forecasting abilities of different models. The validation set size depends on the size of the whole data set and the smoothness of the time series. From our experience, when the variance of the series is not very big, we can set the testing sample size to be around 10% of the total sample size. More research is needed to find out an optimal way of splitting up the sample under different conditions.

3.4.2.2 Weight Factor s

The weight factor s determines the balance between AIC and lnGE. It directly affects the *comb* value and therefore the model order(s) selection. Many factors may affect the weight constant s, such as validation sample size and smoothness of the series. If s is very small, the second term of equation (3.7) will be insignificant. Thus the proposed method will be just the AIC approach. The weight factor, s, should be selected in a way such that AIC and $s$lnGE are of similar magnitude. This way, no term dominates the other. We get the advantages of both the AIC approach and the split-sample approach. When the testing set size is properly selected and lnGE has similar magnitude as AIC, we can simply set s value to be 1.

3.4.2.3 Residual Tests

After building a time series model, we should check the randomness of the residuals on training set to validate the correctness of the obtained model. If the model is correctly built for the given series, the residuals should look like white noises. Statistical tests such as Cumulative spectrum test, Box-Pierce test, Fluctuation test, Outlier detection, and Normal test can test the randomness of the residuals (Brockwell and Davis, 1991, Shumway and Stoffer, 2000a). Cumulative spectrum test checks whether the variance of the residuals is constant. Box-Pierce test checks the independence of the residuals. Normal test is used to check whether the residuals follow the normal distribution. If the residuals pass these tests, we can conclude that the residuals are random noises and the model correctly describes the given data. However, each of these tests can be applied only when the size of training set is big enough, say, more than 40. An example of residual tests result is listed in Appendix 1.

As to the validation set, we can check the generalization errors. If a model usually over-predicts or under-predicts on the validation set, the model cannot be adopted.

3.4.2.4 Available Tools

To implement the proposed method, we have to apply a selected type of time series model twice on different data sets. The first time, we use all available data points. The second time, we apply the model only to the training set after splitting the sample. Fortunately, statistical time series software such as WinASTSA (Shumway and Stoffer, 2000b), SAS (SAS Institute Inc., 2001), R (R Development Core Team, 2000), and S-Plus (Venables and Ripley, 1999) can help us out of some tedious work.

WinASTSA is an interactive, menu driven time series analysis package, which runs under Windows operating systems. It is user friendly and easy to operate and it is free. More importantly, it has a search function, e.g.: "ARIMA Search", which is ideal for model orders selecting. The search select criteria include AIC and AICC. For more information about this software, please refer to its user's manual (Shumway and Stoffer, 2000b). Its drawback is when the data size is too big, the system becomes a little bit unstable. It is recommended to save the obtained results after each run.

3.5 Case Studies

3.5.1 A Simulated Second Order Polynomial Function

In this example, we use simulated data to test the effectiveness of the proposed method of AIC considering generalization error for best time series model selection. The simulated data is obtained from the following second order polynomial function:

$$y = 0.01 * x^2 - 0.1 * x + 5. \tag{3.8}$$

Gaussian random noise with standard deviation of 1 is added to the data values obtained. 80 data points are generated: $x_1, x_2, \ldots, x_{80}$. The original series $x_1, x_2, \ldots x_{80}$ is plotted in Figure 3.4. Because we know that the variance of the generated series is constant, so no transformation is applied to this series.

With these 80 data points, we will use the AR models to illustrate the AIC approach and the proposed approach. These 80 data points will be used to select the order of the AR model. After the order of the AR model is determined with different approaches, we will generate another 20 data points to compare the prediction performance of the selected models.

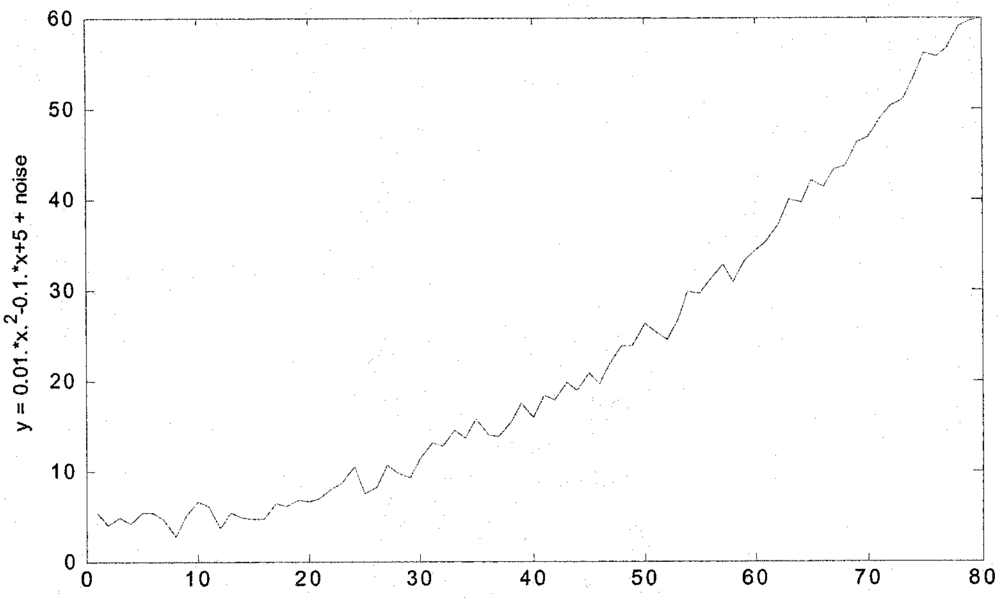


Figure 3.4. Simulated original time series data

Firstly, let's look at the AIC method. Following the steps to build time series models discussed in section 2.1.6, the next step is to select AR model order. A search method is applied. The search range is set to be from 1 to 10 because usually the model will not take an order more than 10. Here, AICC is set to be the selection criterion. It means that we calculate the AICC of models: AR(1), AR(2), ..., AR(10), and find the one with the smallest AICC value. With WinASTSA (Shumway and Stoffer, 2000b), the parameters of the 10 AR models are estimated and their AICC values are calculated. The AICC values of the 10 models are listed in Table 3.2.

Table 3.2. AICC values of selected AR models

	AR(1)	AR(2)	AR(3)	AR(4)	AR(5)	AR(6)	AR(7)	AR(8)	AR(9)	AR(10)
AICC	1.742	1.585	1.535	1.506	1.552	1.510	1.538	1.509	1.562	1.570

From Table 3.2, AR(4) provides the smallest AICC value. The AR(4) model is found to be:

$$x_t = 0.4+0.417x_{t-1}+0.172x_{t-2}+0.216x_{t-3}+0.251x_{t-4}+w_t \tag{3.9}$$

Based on Akaike's theory, equation (3.9) is the best AR model for prediction based on the existing 80 data points.

Now, let's consider the proposed method based on the proposed procedure in Section 3.4.1. Since we have obtained AICC values for all candidate models as discussed above, the next step is to split the sample into training set and validation set. We split the 80 data points into two groups, the training set contains 70 data points, x_1 to x_{70}, and the validation set contains 10 points, x_{71} to x_{80}. The validation set size is about 10% of total available data number. Then, the 10 candidates, AR(1) through AR(10), are refitted on the training set, x_1 to x_{70}, and predict 10 steps ahead, which are $\hat{x}_{71}, \hat{x}_{72}, \ldots, \hat{x}_{80}$. The prediction and original data are plotted in Figure 3.5. From Figure 3.5, we can see that AR(1) follows the trend of the original data well while all other models over-predict.

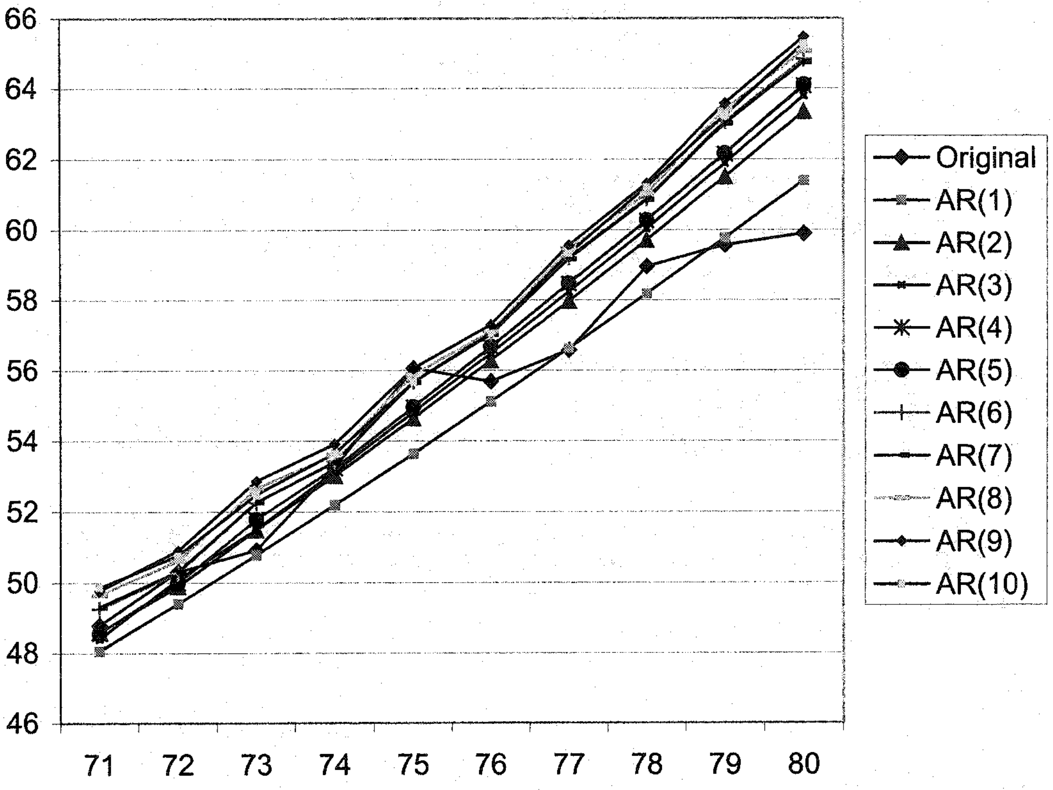


Figure 3.5. x_{71} to x_{80} and $\hat{x}_{71}$ to $\hat{x}_{80}$ from the 10 candidate models for simulated data

After obtaining the predictions, we can use equations (3.6) and (3.7) to calculate the generalization errors on validation set and the combined factor *comb* for all candidates. They are listed in Table 3.3. Here, we simply select the weight factor s to be 1 since lnGE and AIC have similar magnitude.

Table 3.3. *Comb* of the 10 models for simulated data

	AR(1)	AR(2)	AR(3)	AR(4)	AR(5)	AR(6)	AR(7)	AR(8)	AR(9)	AR(10)
AICC	1.742	1.585	1.535	1.506	1.552	1.510	1.538	1.509	1.562	1.570
lnGE	0.15	0.75	1.00	1.17	1.19	1.64	1.60	1.79	1.94	1.81
comb	1.89	2.33	2.53	2.68	2.74	3.15	3.14	3.30	3.50	3.38

Based on the proposed method, the model with the smallest *comb* value is the best model for prediction, which is AR(1). AR(1) model is found to be:

$$x_t = 0.22 + 1.02x_{t-1} + w_t \tag{3.10}$$

Recall that the best model suggested by AIC method is AR(4). To check which method is more effective, we use the original polynomial function to generate another 20 points, x_{81} to x_{100}. Then, we compare $x_{81} \sim x_{100}$ and $\hat{x}_{81} \sim \hat{x}_{100}$ from the 10 candidate models. They are presented in Figure 3.6. The Sum of Squared Errors (SSE) and Mean Absolute Percent Errors (MAPE) of the predictions away from observed values are listed in Table 3.4. Based on Table 3.4, AR(1) provides the smallest SSE, 46.22, and MAPE, 1.71%.

Table 3.4. SSEs and MAPEs of the 10 candidate models on $x_{81} \sim x_{100}$

	AR(1)	AR(2)	AR(3)	AR(4)	AR(5)	AR(6)	AR(7)	AR(8)	AR(9)	AR(10)
SSE	46.22	120.51	291.19	401.77	357.05	713.26	678.15	720.85	795.61	517.06
MAPE	1.71%	2.24%	3.49%	4.21%	3.96%	5.71%	5.58%	5.86%	6.22%	4.85%

Suggested by the AIC method, the best model is AR(4) and the worst model is AR(1) as indicated in Table 3.2. Based on the proposed method, the best forecasting model is AR(1) and the worst model is AR(9) as indicated in Table 3.3. As we can see from Table

3.4, AR(1) model generates the smallest SSE and MAPE values. It means AR(1) provides the best forecasting for $x_{81} \sim x_{100}$. On the other hand, the model providing the worst forecasting for $x_{81} \sim x_{100}$ is AR(9) because it creates the largest SSE and MAPE. We can conclude that the proposed *comb* value correctly represents the forecasting ability of different time series model, while AIC method finds neither the best nor the worst forecasting model. We can say that the proposed method is a more effective method than AIC method for time series model order(s) selection in this example.

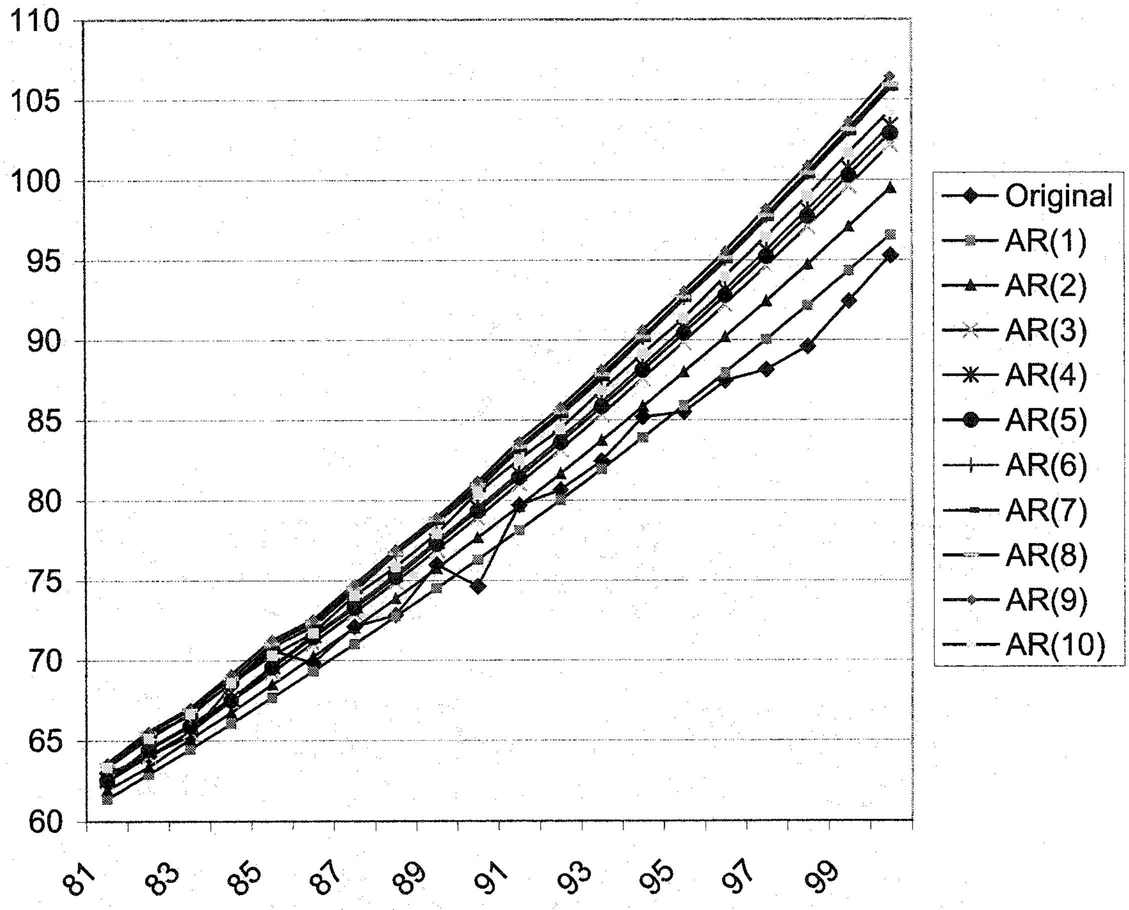


Figure 3.6. x_{81} to x_{100} and $\hat{x}_{81}$ to $\hat{x}_{100}$ from the 10 candidate models

Examining Figure 3.5 and Figure 3.6, AR(1) forecasts the trend pretty well in $x_{71} \sim x_{80}$, it also forecasts well in $x_{81} \sim x_{100}$. AR(2) ~ AR(10) over predict in $x_{71} \sim x_{80}$ and so do they in $x_{81} \sim x_{100}$. We can see that split-sample validation can give some early warning signals of missing the trend. It may help us see the forecasting ability that AIC is not able to provide. We can get rid of those models missing the trend in advance.

3.5.2 U.S. GNP Data

In this example, we use the data analyzed by Shumway and Stoffer (2000a, Chapter 2, page 145). The series, $\{y_t\}$, is quarterly U.S. GNP in billions of dollars from 1947 to 1991, 177 points altogether, which is seasonally adjusted. Figure 3.7 shows a plot of the data. If we take the first differencing of y_t to remove the trend, we will find the variance is increasing with time, plotted in Figure 3.8. Hence, we first take the log transform of the data to remove the trend in variance. The transformed data, $x_t = \ln(y_t)$, is plotted in Figure 3.9. The first differencing of x_t, $\nabla x_t = \nabla \ln(y_t)$ is defined as return or growth rate (Shumway and Stoffer, 2000a). The growth rate of U.S. GNP is plotted in Figure 3.10, which appears to have a constant variance.

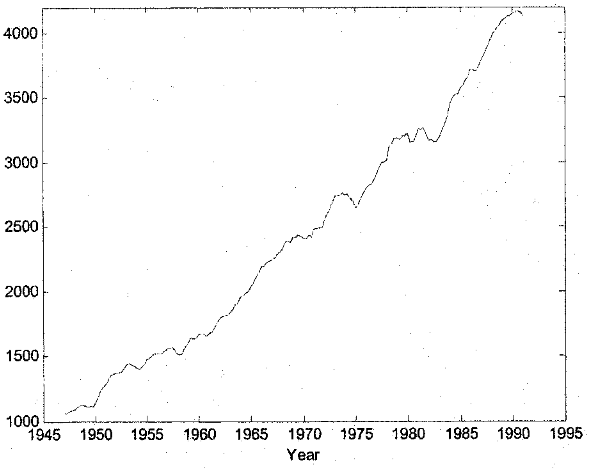


Figure 3.7. US GNP from 1947 to 1991

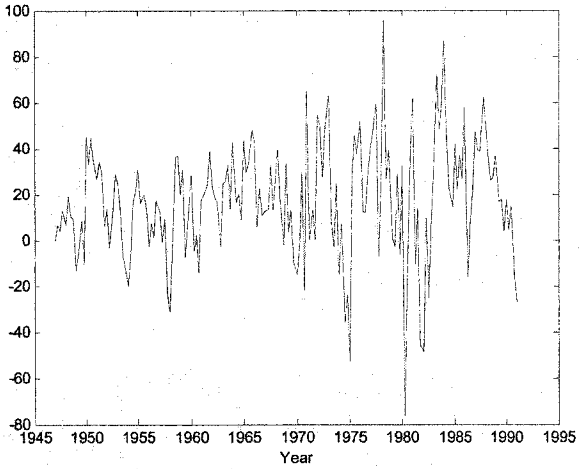


Figure 3.8. First differencing of GNP

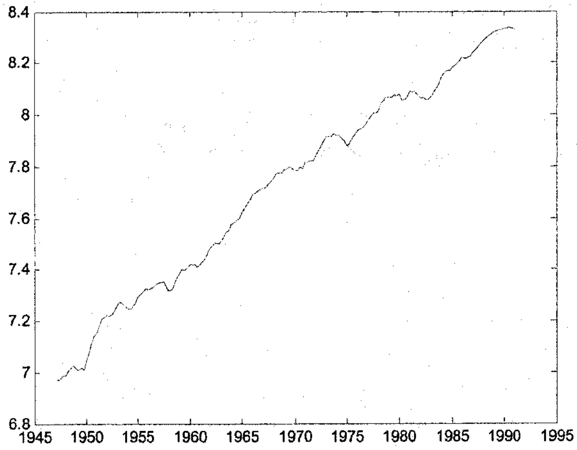


Figure 3.9. Log of US GNP data

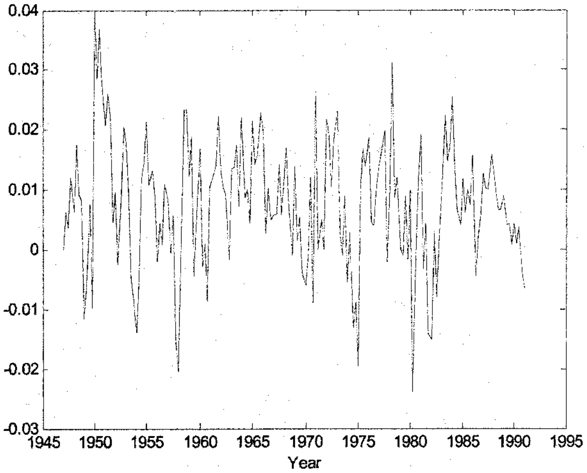


Figure 3.10. US GNP quarterly growth rate

To test the effectiveness of the proposed model order(s) selection method, we use the first 155 data points, x_1 to x_{155} for model building. The last 22 points are reserved to verify the forecasting abilities of different time series models. Please note that the purpose of reserving these 22 data points is to verify the effectiveness of different forecasting methods. It is different from reserving roughly 10% data points for split sample validation, which is a step of the proposed method.

Firstly, look at the AIC method. Having plotted the data and transformed the time series, we select ARIMA model for model building because it is a general model including AR, MA, and ARMA. Next, we specify ARIMA search ranges to be p: 0 – 10, d: 0 – 2, and q: 0 – 10. Ranges of parameters p, q are selected to be less than 10 because in reality, model orders are limited within 10. The range of d is selected to be within 2 because we seldom conduct differencing more than twice. The model selecting criterion is still AICC. We will get 363 candidate models in such search ranges. The parameters of each candidate are estimated and the AICC values are calculated. The best model suggested by AICC is ARIMA(10, 2, 4) with AICC of -8.16.

To use the proposed method, the next step is to split the 155 data points into two groups: x_1 to x_{142} for training and x_{143} to x_{155} for validation. Again, the size of validation set is selected to be about 10% of the total sample size. We refit the candidate models on the training set and then forecast $x_{143} \sim x_{155}$. The next step following the proposed procedure is to calculate the generalization errors on the validation set. We find that lnGE and AICC have similar magnitude. Thus, we choose the weight factor $s = 1$. Now, the *comb* values can be calculated by using equation (3.7). ARIMA(2, 1, 4) is found to have the smallest *comb*: -15.19. The AICC of ARIMA(2, 1, 4) is –8.05. While, the ARIMA(10, 2, 4) with the smallest AICC shows a much bigger *comb*: -12.09.

Due to page limitation, we are unable to discuss all these 363 candidate models. The following models are selected.

(1) ARIMA(10, 2, 4). It is selected because it has the smallest AICC value. It is found to be:

$$(1+.99B_1 -.16B_2 -.7B_3 +.09B_4 +.37B_5 +.13B_6 -.11B_7 -.02B_8 +.15B_9 +.09B_{10})\ \nabla x_t$$

$$= (1 + .47B_1 - .81B_2 - .95B_3 + .3B_4)\, w_t \tag{3.11}$$

(2) ARIMA(2, 1, 4) . It is selected because it has the smallest *comb* value. It is found to be:

$$(1-0.84B^1-0.16\, B^2)\, \nabla\, x_t = (1-0.49\, B^1-0.09\, B^2-0.2\, B^3-0.22\, B^4)\, w_t \tag{3.12}$$

(3) ARIMA(3, 1, 2) . It is selected because it has the second smallest AICC. It is found to be:

$$(1 -0.88\, B^1 -0.46\, B^2 +0.35\, B^3)\, \nabla\, x_t = (1 -0.56\, B^1 -0.53\, B^2)\, w_t \tag{3.13}$$

(4) ARIMA(0, 1, 2) . It is selected because this model is suggested by Shumway and Stoffer (2000a). It is found to be:

$$\nabla x_t = (1 + .42\mathrm{B1} + .36\mathrm{B2})\, w_t \tag{3.14}$$

The AICC values of these four models are listed in Table 3.5. Based on AIC theory, the smaller the AICC is, the higher rank the model has. As shown in Table 3.5, the model with the highest rank is ARIMA(10, 2, 4) based on the AICC model selection criterion.

Table 3.5 The AICC values of the four models for log GNP data

	ARIMA(10, 2, 4)	ARIMA(2, 1, 4)	ARIMA(3, 1, 2)	ARIMA(0, 1, 2)
AICC	-8.16	-8.05	-8.14	-7.93
Rank	1	3	2	4

The AICC, lnGE, *comb* values of the four candidate models are listed in Table 3.6. The smaller the *comb* value is, the higher rank the model has.

Table 3.6 The *Comb* values of the four models for log GNP data

	ARIMA(10, 2, 4)	ARIMA(2, 1, 4)	ARIMA(3, 1, 2)	ARIMA(0, 1, 2)
AICC	-8.16	-8.05	-8.14	-7.93
lnGE	-3.93	-7.14	-5.97	-5.07
Comb	-12.09	-15.19	-14.11	-13.00
Rank	4	1	2	3

From Tables 3.5 and 3.6, we can see that ARIMA(10, 2, 4) is the best model based on the AICC criterion while ARIMA(2, 1, 4) is the best model based on the proposed method.

Remember that we have reserved 22 data points to verify the forecasting abilities of different models. Now, we fit the four candidate models to the original series $x_1 \sim x_{155}$ and forecast 22 steps ahead. The predicted values are represented by $\hat{x}_{156} \sim \hat{x}_{177}$. The predictions and the measured data are plotted in Figure 3.11. The SSEs and MAPEs of the forecasting from the four models are listed in Table 3.7.

Table 3.7. SSEs and MAPEs of the four models for log GNP data

	ARIMA(10, 2, 4)	ARIMA(2, 1, 4)	ARIMA(3, 1, 2)	ARIMA(0, 1, 2)
SSE	1.81922	0.00879	0.05091	0.18726
MAPE	3.426%	0.160%	0.491%	0.970%
Rank	4	1	2	3

From Figure 3.11, we can see that the model ARIMA(0, 1, 2) suggested by Shumway and Stoffer (2000a) can only provide a near constant forecasting. This model may be good at fitting the existing data points but it has a poor forecasting ability. The best model suggested by AICC is ARIMA(10, 2, 4) but it also makes a constant forecasting. These two models are far from ideal to be good at forecasting.

Based on the AICC values, we have ranked the four candidates as listed in Table 3.5: ARIMA(10, 2, 4), ARIMA(3, 1, 2), ARIMA(2, 1, 4) and worst ARIMA(0, 1, 2). However, this ranking does not correctly reflect the forecasting abilities of these four models. From Table 3.6 and based on the forecasting errors, we know that the correct ranking of forecasting abilities should be: ARIMA(2, 1, 4), ARIMA(3, 1, 2), ARIMA(0, 1, 2), and ARIMA(10, 2, 4). The ranking based on the *comb* values coincides with the ranking of forecasting errors as listed in Table 3.7. It means that the proposed method correctly tells the forecasting ability of the four candidate models. So, *comb* is a better indicator for forecasting ability than AICC for these four candidate models. This example again shows the advantage of the proposed method in time series forecasting model order(s) selection.

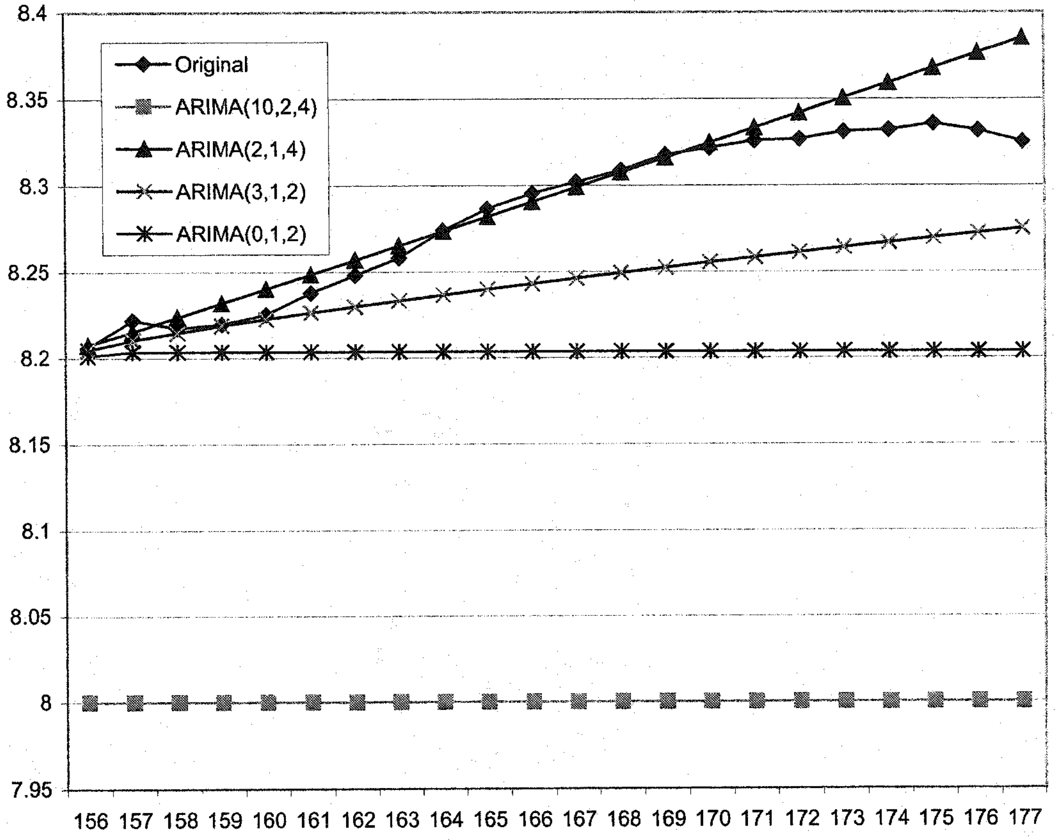


Figure 3.11. $x_{156} \sim x_{177}$ and $\hat{x}_{156} \sim \hat{x}_{177}$ of the four models for log GNP data

3.5.3 Vibration Trend of A Synchronous Motor

We have shown two successful applications of AIC considering generalization error. Now we use it to predict the vibration trends. The data used here are obtained from a paper by Miller (1998). Miller's company uses Bently Nevada Trendmaster® 2000 to monitor a 7000 hp AC synchronous motor. This motor's vibration is continuously monitored and recorded. In the paper, a figure is attached to show the development of the trend of vibration level during a two-day period. In the recorded vibration data, a portion of the vibration trend has amplitude over the danger level because of a damaged coupling. We digitize the portion of the vibration trend over the danger level as plotted in Figure 3.12. The unit in y-axis of Figure 3.12 is recorded as millimeter (mm) in the process of digitization. Please note that the amplitude unit of the digitized data is not the same as the unit used in Miller's figure. The x-axis of Figure 3.12 is the number of data points: 1 ~ 120.

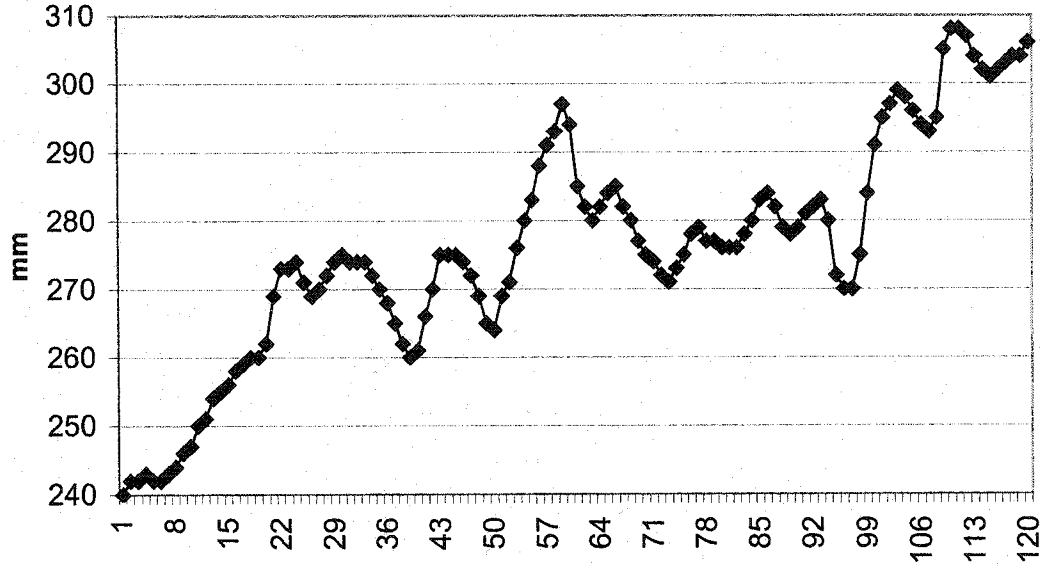


Figure 3.12. Vibration trend obtained from Miller's paper

Here, we have 120 points: $x_1 \sim x_{120}$. To test the forecasting abilities of different models, we need to reserve the last 10 data points. So, we use $\{x_t\} = x_1 \sim x_{110}$ to build the forecasting model.

By examining the plotting in Figure 3.12, we find that it has an increasing trend and thus this time series is not stationary. Meanwhile, we also notice that the recorded time series is not smooth. We select ARIMA model for forecast model building because it is general. Again, the proposed method for model order(s) selection is compared with the method of using AIC only by following the procedure introduced in Section 3.4.1.

Firstly, look at the AIC method. To this time series, differencing is firstly applied. We difference x_t once to remove the linear trend. The series after first differencing is plotted in Figure 3.13. The ACF and PACF correlograms of the series after first differencing are plotted in Figure 3.14. From this figure, we cannot tell the appropriate orders of model for this series. So, the ARIMA search method is used again. The search range is set to be p: 0 – 10, $d = 1$, and q: 0 – 10. The selecting criterion is AICC. We will get 121 candidate models. The parameters of each candidate are estimated and the AICC values are calculated. The best model suggested by AICC is ARIMA(2, 1, 10) with AICC of 2.634.

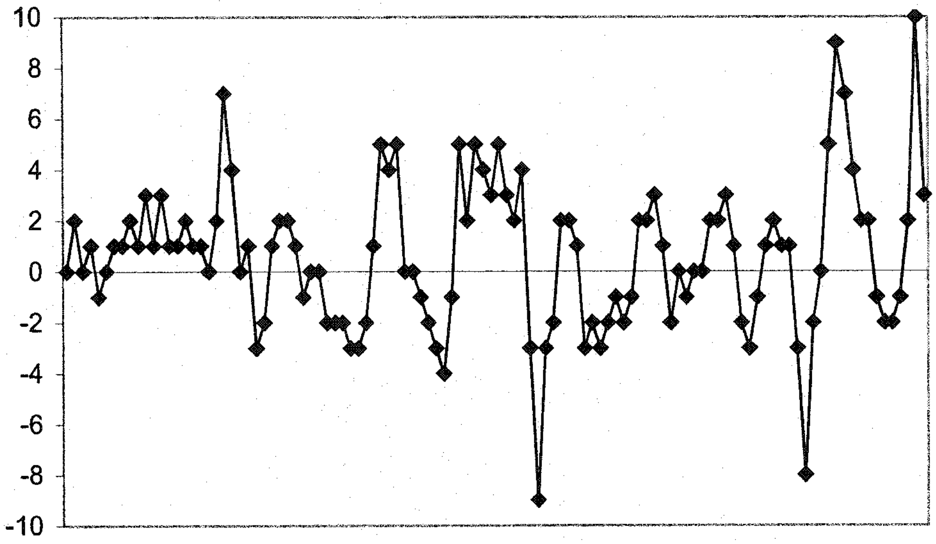


Figure 3.13 First differencing of vibration trend

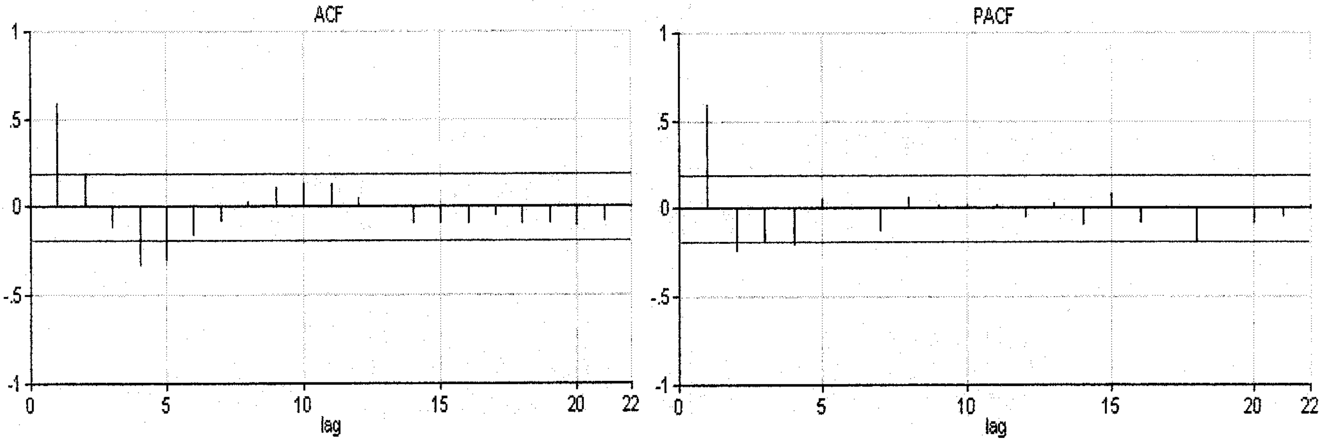


Figure 3.14 ACF and PACF of vibration trend series $x_1 \sim x_{110}$ after one differencing

Then, let's look at the proposed method. To apply the proposed method, we need to split the 110 data points into two groups: x_1 to x_{100} for training and x_{101} to x_{110} for validation. The size of the validation set is about 10% of the total sample size. We refit the candidate models on the training set and forecast 10 steps ahead to obtain $\hat{x}_{101} \sim \hat{x}_{110}$. Then, we calculate the generalization errors on the validation set. We find that lnGE and AICC have similar amplitudes. Thus, we can choose the weight factor s = 1. Now, the *comb* values can be calculated by using equation (3.7). Following the procedure of the proposed method, we find ARIMA(5, 1, 5) has the smallest *comb*: 7.417. The AICC of ARIMA(5, 1, 5) is 2.715.

Because the limitation of space, we won't be able to discuss all these 121 candidate models. We only compare the two models suggested by AICC and *comb*, respectively.

(1) ARIMA(2, 1, 10). It is selected because it has the smallest AICC value. The model is found to be:

$$\begin{aligned}&(1 -.27B^1 -.75B^2)\,\nabla\, x_t = \\ &(1+ .41B^1-.64B^2-.57B^3-.7B^4-.45B^5+ .24B^6+ .13B^7+ .17B^8+ .23B^9- .11B^{10})\, w_t \qquad (3.17)\end{aligned}$$

(2) ARIMA(5, 1, 5). It is selected because it has the smallest *comb* value. The model is found to be:

$$\begin{aligned}&(1+ .38B^1 - .27B^2 - .06B^3+ .33B^4+ .07B^5)\,\nabla\, x_t \\ &\quad = (1+ 1.15B^1 - .47B^2 + .07B^3- .01B^4- .23B^5)\, w_t \qquad (3.19)\end{aligned}$$

The AICC, lnGE, *comb* values of the above two candidate models are listed in Table 3.8.

Table 3.8 *Comb* of the two models for vibration trend data

	ARIMA(2, 1, 10)	ARIMA(5, 1, 5)
AICC	2.634	2.765
lnGE	4.856	4.653
Comb	7.490	7.418

To verify, we apply these two models to original series $x_1 \sim x_{110}$ and forecast 10 steps ahead. The estimated values are represented as $\hat{x}_{111} \sim \hat{x}_{120}$. The predictions from these two candidate models on the reserved data points are plotted in Figure 3.15. The SSEs and MAPEs of the predictions from the two models are listed in Table 3.9.

Table 3.9. SSEs and MAPEs of the two models for vibration trend data

	ARIMA(2, 1, 10)	ARIMA(5, 1, 5)
SSE	586.86	9.4528
MAPE	2.361%	0.274%

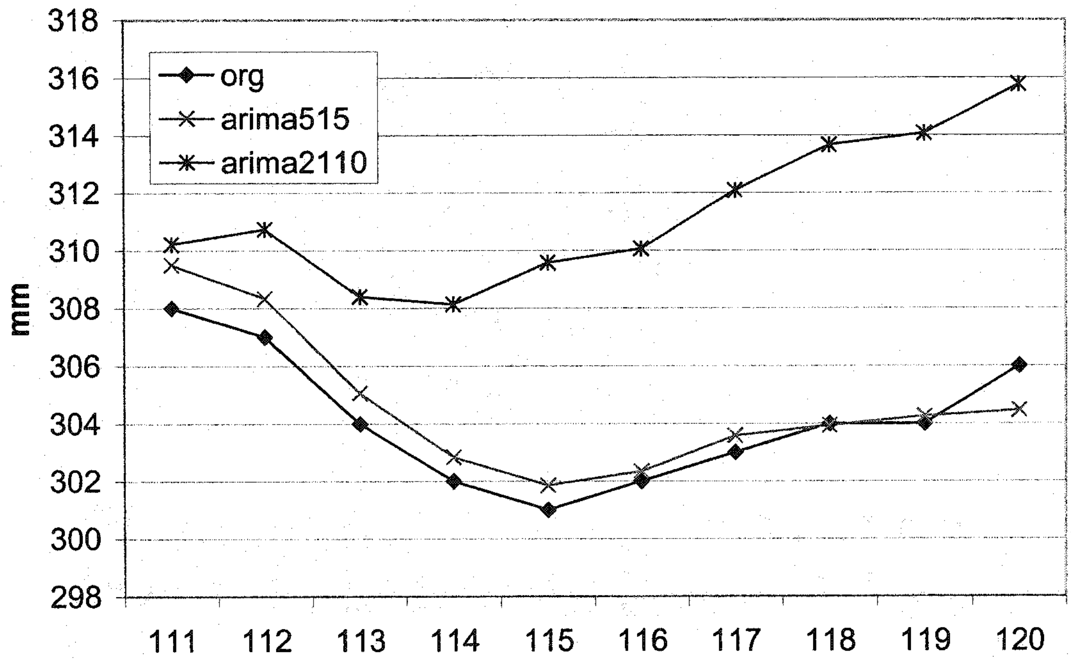


Figure 3.15 $x_{111} \sim x_{120}$ and $\hat{x}_{111} \sim \hat{x}_{120}$ of the two models for vibration trend data

From Figure 3.15, the best model suggested by AICC, ARIMA(2, 1, 10), does not correctly forecast development of vibration trend. While, the best model suggested by *comb* predicts well. From Table 3.9, we can see that ARIMA(2, 1, 10) suggested by AICC generates much larger SSE and MAPE. It means that ARIMA(5, 1, 5) suggested by the proposed method has a better forecasting ability than the model suggested by AIC. This proves that the proposed method is better than AIC in best forecasting time series model order(s) selection.

3.6 Discussions and Conclusions

Some regression methods, e.g.: second order polynomial regression, can be used to forecast time series. The problem of linear or nonlinear regression is that they cannot catch the cyclic and seasonal pattern if there is any in the time series. Meanwhile, time series models use the latest few points to predict, e.g.: AR(3) model just use the most recent 3 data points to predict the next. Usually, the latest data points contain the more valuable information on the series. The regression methods treat all data points to be the same without differentiating their importance. As a result, regression methods are not so convincing as time series models in vibration forecasting based on historical vibration recordings. In addition, what function form to use is also an issue for linear and nonlinear regression.

AIC and its expanded forms are widely accepted method for order(s) selection. AIC includes two terms, the Log-likelihood statistic and a penalty function controlling the model capacity. However, we cannot solely depend on AIC. The method presented in this chapter combines both AIC and split-sample validation. The applications show that we are able to obtain a better forecasting model with smaller forecasting error by applying the proposed method than solely depending on AIC. By splitting up the sample and measuring the generalization error on the validation set, we can always get an early warning whether a certain model is likely to over or under predict. The proposed method can be applied to time series models such as AR, ARMA, and ARIMA. By considering both AIC and generalization error, we are more likely to get the better prediction model.

More research is needed to find the optimal size of the validation set for a certain series. Meanwhile, guidance should be developed for selecting a good weight constant s.

Chapter 4. Support Vector Machines for Prediction

As introduced in Chapter 2, Support Vector Machine (SVM) is designed for regression and classification problems. It is a statistical tool but approaches problems similar to neural networks. SVM can be seen as a "half-way house" between the two, borrowing some terminology and ideals from each (Vapnik, 1997). Successful applications of SVM for time series predictions reported in the literature motivates this study to use SVM for vibration trend forecasting. In this chapter, the basic principles of SVM are studied introduced and three examples are used to compare prediction results from SVM and from time series models.

4.1 Basic Principles of SVM

SVM can be applied to multidimensional data analysis, say, (X, y), where X is a vector. So, SVM is not restricted univariate data only. Usually, the vibration data recordings are just two dimensional data pairs: (x_i, y_i), where x_i is the recording of time. Here, we introduce the principles of SVM for solving a two dimensional problem.

4.1.1 Cost Function

Suppose that we are given a set of training data (x_1, y_1), (x_2, y_2), …, (x_l, y_l), where l denotes the size of this data set. We need to find a function $y = f(x)$ based on this data set. The empirical risk associated with estimated function f is defined by Smola and Scholkopf (1998) as:

$$R_{emp}[f] = \sum_{i=1}^{l} C(f(x_i) - y_i) \tag{4.1}$$

where $f(x_i) - y_i$ represents the estimation error at point (x_i, y_i). $R_{emp}[f]$ denotes the empirical risk.

$C(.)$ is the cost function or loss function determining how we will penalize the estimation errors. The most popular cost function is a quadratic function, as used in the least square regression method. With this function, when there is an estimation error, the penalty is equal to the square of the estimation error, $C(f(x_i) - y_i) = [f(x_i) - y_i]^2$. This cost function is plotted in Figure 4.1.

Another form of the cost function is absolute value function. With this function, when there is an estimation error, the penalty is equal to the absolute value of the estimation error, $C(f(x_i) - y_i) = |f(x_i) - y_i|$. This cost function is also plotted in Figure 4.1.

Cortes and Vapnik (1995) propose a “soft-margin” cost function: ε-insensitive cost function. This cost function punishes only the data points that have estimation errors greater than a predetermined value: ε. It is defined as (Muller et al., 1999):

$$C(f(x_i) - y_i) = |\xi|_\varepsilon = \begin{cases} |f(x_i) - y_i| - \varepsilon & for\,|f(x_i) - y_i| \geq \varepsilon \\ 0 & otherwise \end{cases} \tag{4.2}$$

where $|\xi|_\varepsilon$ denotes the ε-insensitive cost function. Such a cost function with $\varepsilon = 0.5$ is plotted in Figure 4.1.

From Figure 4.1, we can see that when the estimation error is large, the quadratic cost function applies the biggest penalty while the ε-insensitive cost function applies the least penalty. When the estimation error is small, say less than ε, the absolute value cost function applies the largest penalty while the ε-insensitive cost function does not penalize at all. Different cost functions may be used by different users or under different conditions. The ε-insensitive cost function is the most often used cost function in SVM (Smola and Scholkopf, 1998). In this chapter, we also use the ε-insensitive cost function with SVM in our prediction model selection.

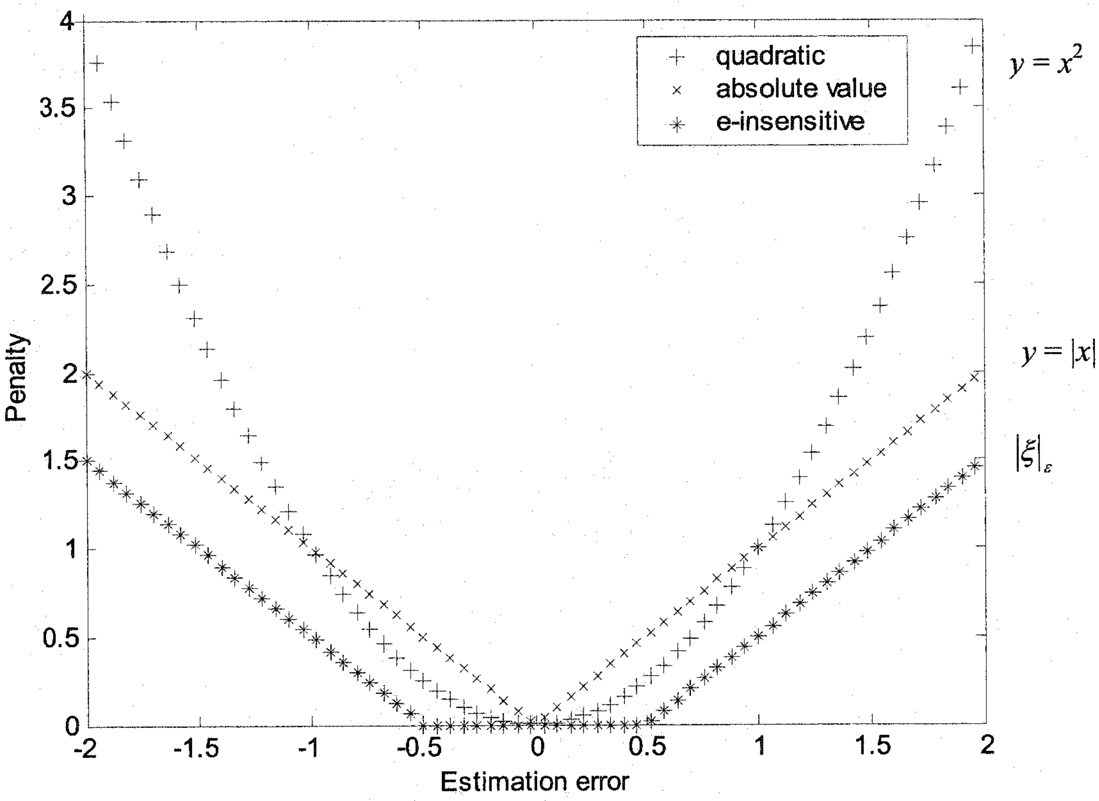


Figure 4.1 Quadratic, absolute value and ε-insensitive cost functions

4.1.2 Kernel Function

Suppose that the input training data are represented as: (x_1, y_1), (x_2, y_2), …, (x_l, y_l), where x_i is a scalar. The input space will be a 2 dimensional space, (x, y). Sometimes, the problem may be nonlinear in nature in the input space. For example, in Figure 4.2(a), the data in the input space follow a quadratic function, $y = x^2$. We can change the representation of data by using: $\Phi(x) = x^2$. The original quadratic problem becomes a linear problem: $y = \Phi(x)$. The obtained linear problem is represented in Figure 4.2(b). Now, we have a new space $(\Phi(x), y)$ where we can perform linear regression. We call the new space the feature space.

Another example about the input space and the feature space is introduced by Cristianini and Shawe-Taylor (2000). The Newton's law of gravitation expresses the gravitational force between two bodies with masses m_1, m_2, and separation r:

$$f(m_1, m_2, r) = C * m_1 * m_2 / r^2 \qquad (4.3)$$

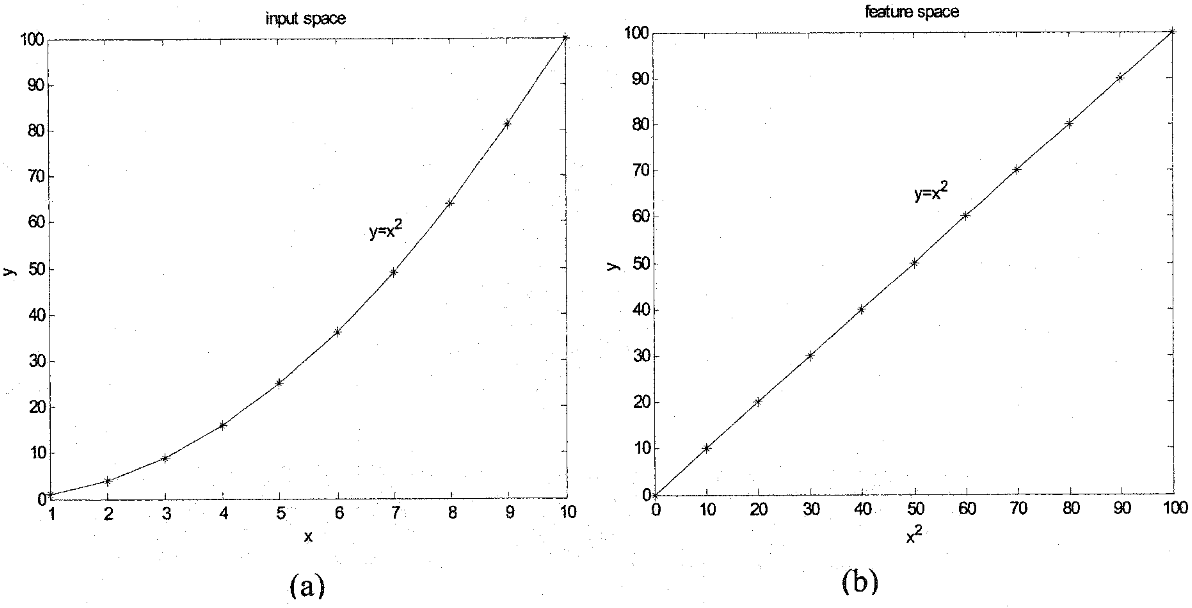
where C is a constant. This law is expressed in terms of the observable quantities, mass and distance. Function (4.3) is not a linear function. But a simple change of coordinates:

$$(m_1, m_2, r) \rightarrow (x, y, z) = (\ln m_1, \ln m_2, \ln r) \qquad (4.4)$$

gives the representation:

$$g(x, y, z) = \ln f(m_1, m_2, r) = \ln C + \ln m_1 + \ln m_2 - 2*\ln r = \ln C + x + y - 2z \qquad (4.5)$$

After this transformation, equation (4.5) is a linear function.



(a)

(b)

Figure 4.2 The idea of input space and feature space

General speaking, we have input data $\mathbf{x} = (x_1, \ldots, x_n)$, we can change the representation of data by:

$$\mathbf{x} = (x_1, \ldots, x_n) \rightarrow \Phi(\mathbf{x}) \qquad (4.6)$$

This step is equivalent to mapping the data in the input space to a new space: $F = \{\Phi(\mathbf{x})\}$. This new space is called the feature space. Note that the input space and the feature space can have different dimensions (Smola and Scholkopf, 1998, Hearst, 1998).

In SVM, the basic idea is to map the data from the input space into a feature space via a nonlinear mapping, i.e.: rewrite the data in a new representation. In the feature space, we can perform linear regression:

$$f(x) = (\omega \cdot \Phi(x)) + b \tag{4.7}$$

where Φ is the nonlinear mapping from the input space to the feature space and ω and b are the slope and the intercept, respectively, obtained from linear regression in the feature space. Kernels are usually used for this nonlinear to linear mapping (Smola and Scholkopf, 1998, Hearst, 1998). In the example shown in Figure 4.2, the mapping is simply $\Phi(x) = x^2$. In SVM, the available kernels include radial basis function kernel (RBF), dot (linear) kernel, and polynomial kernel (Smola and Scholkopf, 1998, Ruping, 2000). For more information about kernel functions, refer to Smola and Scholkopf (1998), Cristianini and Shawe-Taylor (2000) and Boser et al. (1992).

4.1.3 Regularized Risk

In a regression problem, we need to find a function, $y = f(x)$, based on a given data set in order to estimate y with a given x value. As discussed in Chapter 2, the more free parameters the function f has, the smaller the empirical risk will be. When the number of free parameters in f is large enough, all data points will be perfectly fitted, which results in a zero empirical risk. However, this model overfits the data and the generalization ability is poor. As we try to reduce the empirical risk, we should also try to control the capacity of f.

As discussed in Chapter 3, AIC includes two terms: one measures the training error and the other penalizes the increase of the number of free parameters. In SVM, we also try to control the capacity of the model in addition to minimizing the training error. This technique is called regularization. In SVM, the regularized risk can be defined in the following (Vapnik, 1982, Muller, et al. 1999):

$$R_{reg}[f] = \lambda\, R_{emp}[f] + \|\omega\|^2 = \lambda \sum_{i=1}^{l} C(f(x_i) - y_i) + \|\omega\|^2 \tag{4.8}$$

where $R_{reg}[f]$ represents the regularized risk, l denotes the training set size, $C(.)$ is a cost function, ω is the slope of fitted linear function in (4.7), and $\lambda > 0$ is a regularization constant.

The goal of SVM regression is to minimize this regularized risk. We can see that the smaller the λ, the less weight will be applied to the empirical risk and correspondingly, more weight to the squared value of the slope ω.

4.1.4 The Optimization Model in SVM

The objective of SVM is to minimize the regularized risk. Since SVM always conducts linear regression in the feature space, the estimation function f is a linear function. With the ε-insensitive cost function, the optimization model is (Vapnik, 1995):

$$\min \|\omega\|^2 + \lambda \sum_{i=1}^{l} (\xi_i + \xi_i^*) \tag{4.9}$$

$$\text{subject to: } \begin{cases} y_i - (\omega \cdot x_i) - b \le \varepsilon + \xi_i \\ (\omega \cdot x_i) + b - y_i \le \varepsilon + \xi^* \\ \quad \xi_i, \xi_i^* \quad \ge 0 \end{cases} \tag{4.10}$$

where $(x_i, y_i, 1 \le i \le l)$ is the training data point, ε is the pre-specified value for ε–insensitive cost function, ξ_i and ξ_i^* can be calculated by using equation (4.2), and the term $\lambda \sum_{i=1}^{l} (\xi_i + \xi_i^*)$ represents the empirical risk. The pre-selected constant $\lambda > 0$ determines the balance between empirical error and the $||\omega||^2$ of the obtained model. The slope and the intercept of the linear regression mode are ω and b, respectively. The decision variables that we try to find from the optimization model (4.9) and (4.10) above are ω and b.

This optimization model is a quadratic programming problem because all constrains are linear and the objective function is a quadratic function. The combined conjugate gradient and projection method and interior point primal-dual path-following algorithms can be used to solve the quadratic programming problem (Smola, 1996).

Determining the values of the parameters λ and ε are important to the optimization problem in SVM. They are combined together to control both capacity and empirical risk. The parameter ε determines the upper limit of estimation errors that will not be punished in the data while λ is the weight on empirical risk. When λ is infinity, it means we do not allow any empirical error. How to obtain the optimal values of λ and ε is still being studied by researchers.

Cross validation discussed in Chapter 3 may be used to determine the values of λ and ε. However, it is computation intensive. Moreover, the best parameters suggested by cross validation are not necessarily the optimal parameters for prediction. It is because cross validation aims to minimize the generalization error between the lower and the upper limits of the data set. Cross validation uses all given data points to calculate a generalization error without differentiating the importance of most recent observations and past observations. However, most recent observations tell more about the current condition. Thus, we do not use cross validation to find the best SVM prediction parameters. In this chapter, we use the split sample method discussed in Chapter 3 to determine the best λ and ε values for SVM prediction.

4.2 Comparison of AR, ARIMA and SVM in Time Series Prediction

In this section, we apply SVM to data series of Australian expenditure on financial services. The prediction results from SVM are compared with the results from AR and ARIMA models.

4.2.1 Criterion Used to Evaluate the Prediction Models

Because we need to predict the future degradation levels based on historical vibration signals, we are interested in how accurate the obtained models are in prediction rather than how good the models fit to existing data. Thus, we do not use AIC as the criterion to evaluate the models. In this Chapter, we adopt the criterion for comparing generalization errors by Mukherjee et al. (1997). We term it the generalization criterion:

$$\text{Generalization Criterion} = \frac{1}{M}\sum_{n=l+1}^{l+M}\left\|x_{n+1} - \hat{f}_l(x_n)\right\|^2 / Var \tag{4.11}$$

where l is the number of training points; M is the number of validation points; x_{n+1} is the measured data at point $(n + 1)$; $\hat{f}_l(x_n)$ is the prediction value at point $(n + 1)$ based on n training data, and Var is the variance of the time series, which is a constant. $\frac{1}{M}\sum_{n=l+1}^{l+M}\left\|x_{n+1} - \hat{f}_l(x_n)\right\|^2$ represents the average generalization error on the validation set. If it is divided by the variance, the result will be a non-unit value. Equation (4.11) is used to measure the prediction abilities of different models.

4.2.2 Australian Expenditure on Financial Services

In this example, we use a set of time series data associated with software ITSM (Brockwell and Davis, 1994). It is Australian expenditure on financial services from September 1969 to March 1990 in millions of dollars. The series is quarterly recorded so that all the data points are separated by an equal time interval. There are 86 data points all together: $x_1 \sim x_{86}$. The original series is plotted in Figure 4.3.

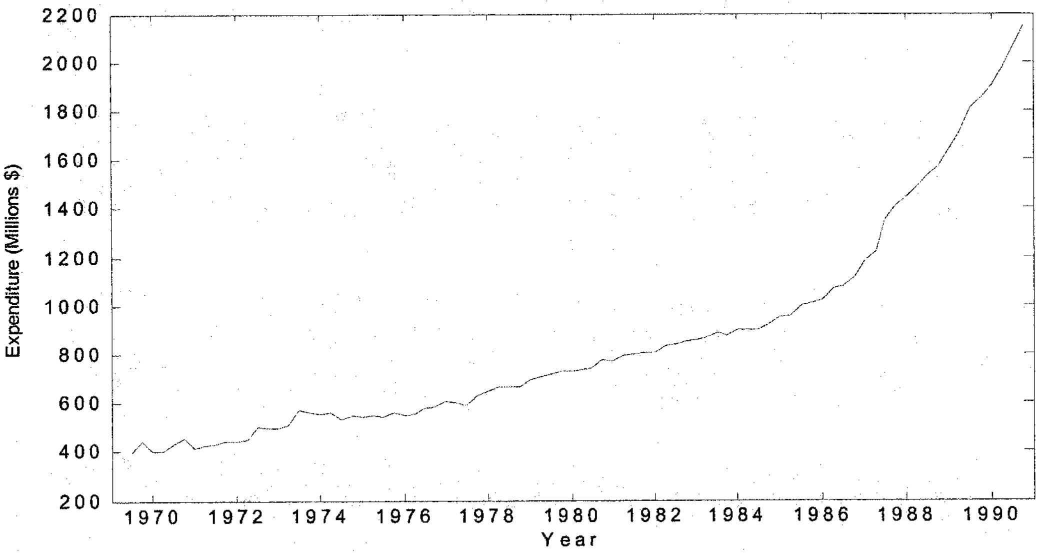


Figure 4.3. Australian expenditure on financial services from 1969 to 1990

To compare the prediction abilities of different models, we reserve the last 26 points, $x_{61} \sim x_{86}$, for validation. Three models: AR, ARIMA, and SVM are compared for one-step ahead and four-step ahead predictions of the given series.

4.2.3 One-Step Prediction

Firstly, we only predict one step ahead, i.e.: we only forecast one point into the future. For example: for data point 61, we can predict its value using the past 60 points. While, for point 81, there will be 80 training data points.

To use time series models, the software tool WinASTSA (Shumway and Stoffer, 2000) is selected for AR and ARIMA model building and prediction. We use AICC as the model order(s) selection criterion. Search method is applied to select the smallest AICC value in the range specified. For AR model, the search range is selected to be from AR(0) to AR(10). ARIMA search range is p: 0 – 10, $d = 1$, and q: 0 – 10.

Whenever there is a new data point added into the original series, the best prediction model order(s) for the new time series might be different from the one before the point is added. For example, to series $x_1 \sim x_{60},$ the best AR model suggested by the smallest AICC value is AR(3). But to series $x_1 \sim x_{61}$, the best AR model with the smallest AICC is AR(4). The same thing may happen when applying ARIMA model to this set of data. In this example, we always search for the best model with the smallest AICC value after each new point is added in. The prediction results from AR and ARIMA models are plotted in Figure 4.4.

Now, let's look at SVM prediction. For this set of data, we choose Linear Spline as the kernel function. Linear Spline kernel is selected because preliminary trial-and-error shows that it seems to be able to provide a small generalization error. More research work is needed to find the optimal kernel function for a certain series of data.

To use SVM, we should firstly determine the values of two parameters: λ and ε. Cross validation is suggested by some researchers to find the optimal λ and ε values. However, it is computation intensive. Moreover, the best parameters suggested by cross validation are not necessary the optimal parameters for prediction. Here, we use split sample method discussed in Chapter 3 to determine the best parameters for SVM prediction. In this example, we divide the sample into two groups, the latest point as validation point and the rest data points as training set. The generalization error calculated from validation point is used as the criterion for parameter selection. A Matlab program is designed based on Gunn's SVM Matlab toolbox to perform split sample validation. The Matlab source code is attached in Appendix 2.

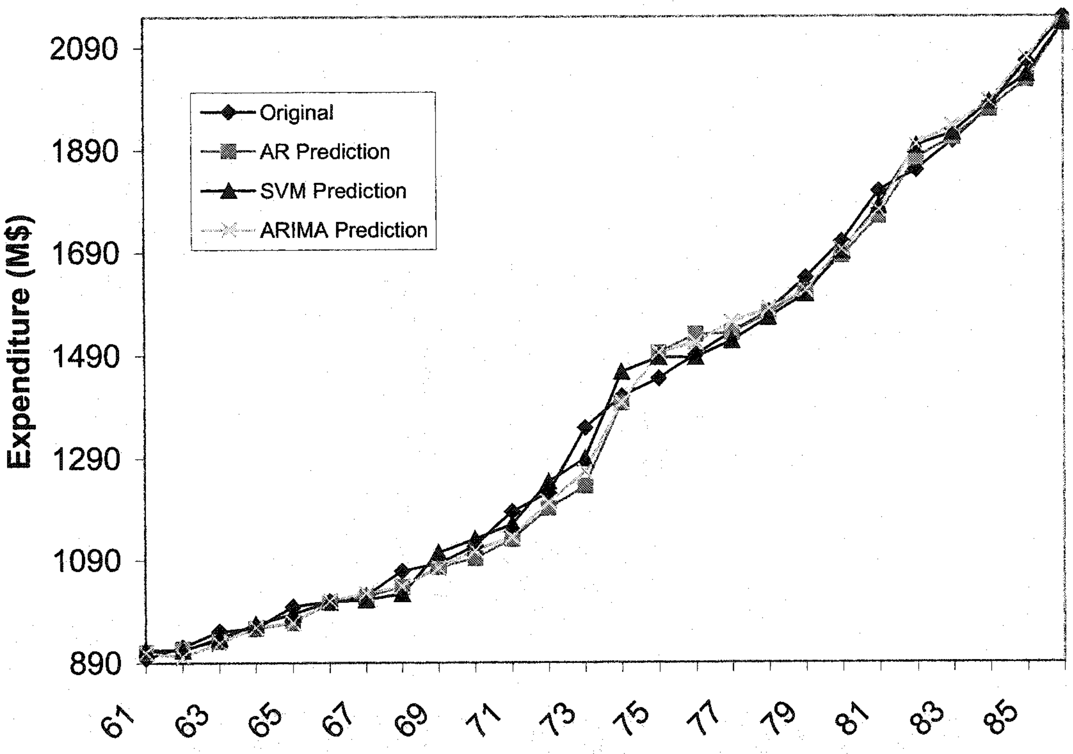


Figure 4.4 Actual and one-step predicted values by the three models for $x_{61} \sim x_{86}$

Consider the series $x_1 \sim x_{60}$, we firstly conduct rough two-dimensional search in a wide range of ε: 0 ~ 10 and λ: 1000 ~ 5000. We feel that we can find out the regularity in such a range. Because of page limitation, only the generalization errors in the search range of ε: 0 ~ 10, with step size 2, λ: 1000 ~ 3800 with step size 200 are listed in Table 4.1.

In Table 4.1, we find that generally speaking, the larger the λ is, the smaller the generalization error will be. Thus, we choose λ to be infinity. After that, we do one-dimensional search for optimal ε value. We set the search range of ε to be 1 ~ 10 with a step size of 0.25 because from the wide range search, we find that the optimal ε seems to fall into this range. When $\varepsilon = 8.25$, we will get the smallest generalization error: 22.74. We list part of the search results in range 7 ~ 9.5 in Table 4.2. In Table 4.2, we use "GE" to represent generalization error.

Table 4.1 Different λ andε values and the corresponding generalization errors by split sample validation for Australian expenditure data

ε λ	0	2	4	6	8	10
1000	332.7203	272.8139	263.3193	320.6817	148.2284	166.2801
1200	324.7234	277.7436	262.5996	318.2552	234.8618	161.8865
1400	319.1619	271.7072	261.5484	317.6412	233.4041	159.0014
1600	313.6534	248.4431	260.3041	315.3501	233.4399	153.5671
1800	305.7603	222.5028	258.9776	313.6148	233.4341	163.0286
2000	306.9746	215.0332	257.9354	312.6698	233.2457	159.7263
2200	293.7236	225.3643	257.1809	311.8356	232.0489	156.6574
2400	285.6493	196.7569	256.4277	311.0026	230.1057	154.7586
2600	283.1843	176.778	255.6757	310.6717	229.6603	153.7619
2800	280.4305	176.7823	255.1115	306.8412	227.7209	152.6994
3000	270.8314	176.7934	254.9602	299.2553	225.7146	148.0147
3200	255.7169	176.6768	254.1474	298.1312	223.7047	143.8129
3400	241.0833	176.5358	254.2498	298.6313	220.8177	142.8473
3600	226.9307	176.3116	254.2783	298.9918	217.8861	142.0207
3800	213.259	176.1237	254.0963	299.191	214.9761	141.8815

Table 4.2 Generalization errors according to different ε values when λ is infinity

ε	7	7.25	7.5	7.75	8	8.25	8.5	8.75	9	9.25	9.5
GE	36.27	39.33	44.60	35.69	26.97	22.74	23.02	27.85	37.22	48.72	61.92

Based on split sample validation, we find that the optimal parameters for SVM regression are: $\varepsilon = 8.25$ and $\lambda =$ infinity. After selecting λ and ε values based on series $x_1 \sim x_{60}$, we keep them for all the one-step-ahead predictions on validation set. This method of finding λ and ε values is not ideal. Future research work is needed for finding optimal λ and ε values for SVM prediction.

After selecting λ and ε values, we can conduct SVM prediction. Existing SVM software tool based on Matlab environment (Gunn, 1998) is revised to obtain SVM prediction. The Matlab source code is attached in Appendix 3. In this chapter, we use a Pentium-1000 PC with 256 MB SDRAM memory for SVM regression. For easier comparison, the actual and predicted values from three different models are plotted in Figure 4.4. The generalization errors obtained using equation (4.11) are listed in Table 4.3. The computation time of different models are also presented in Table 4.3.

Table 4.3: One-step forecasting error and computation time of the three models

	AR Model	ARIMA Model	SVM
Generalization Criterion	0.023727	0.0175915	0.014926
Computation Time (seconds)	8	12	289

From the results, we can see that SVM provides the smallest prediction error; ARIMA model comes next, and AR model performs the worst. However, all these predictions are pretty close and SVM is much more time consuming.

4.2.4 Four-Step Prediction

We use the three models to make four-step prediction. For each data set, we predict 4 steps ahead. For example, we use $x_1 \sim x_{60}$ to predict $x_{61} \sim x_{64}$ and $x_1 \sim x_{61}$ to predict $x_{62} \sim x_{65}$. At last, use $x_1 \sim x_{81}$ to predict $x_{82} \sim x_{86}$.

To AR and ARIMA prediction, the processes of four-step prediction are similar to one-step prediction as discussed in Section 4.2.3. To SVM, the kernel function, λ and ε are selected to be the same as used in Section 4.2.3. Following the steps, we can calculate the generalization error for the 4-step predictions using equation (4.11).

All together, we have 23 sets of validation points with 4 data points in each set, i.e.: $x_{61} \sim x_{64}$, $x_{62} \sim x_{65}$, ..., $x_{82} \sim x_{86}$. There are 23 generation errors obtained from these 23 validation sets. We take the average of these 23 generation errors as the average 4-step generation error. The generalization errors and computation time of different models are shown in Table 4.4.

Table 4.4: Four-step generalization error and computation time of the three models

	AR Model	ARIMA Model	SVM
Generalization Criterion	0.597511	0.4646842	0.104173
Computation Time (seconds)	6	10	261

Based on Table 4.4, SVM still provides the smallest generalization error for the validation sets in four-step forecasting. The advantage of SVM over AR and ARIMA is more significant than the one-step-ahead prediction above. But it is still very computation intensive. From discussions above, SVM is a better prediction tool than the two time series models in this example.

4.3 Use SVM for Vibration Trend Prediction

As discussed in Chapter 2, SVM has been reported to have superior performance in many applications, such as financial time series prediction. The example studied in section 4.2 shows that SVM provides a better prediction than time series models. However, no report has been found to apply SVM in vibration trend prediction. In this section, we will apply SVM to vibration data series prediction. The evaluation criterion is equation (4.11), the same as the one used in Section 4.2.

4.3.1 Gearbox Vibration Data

With the success of SVM to predict the expenditure on financial services, we then use it to predict the vibration trends using vibration data collected from a gearbox.

4.3.1.1 The Experiment

The gearbox under monitoring initially has a crack in one of the teeth. The crack develops during the experiment, which causes the vibration level to increase. Finally, one tooth is broken at the end of the experiment. Accelerometers are used to collect the vibration signals from this deteriorating gearbox. The gearbox is monitored periodically. Each time, 4096 data points in time domain are sampled and recorded. Altogether, there are 20 sets of 4096 data recordings. The obtained signal in time domain at the beginning and the end of the experiment are plotted in Figure 4.5 and Figure 4.6, respectively. From the two figures, we can see that the impact is serious when there is one tooth broken. The unit of vibration amplitude is recorded as the voltage: mill-volt (mv) of the signal.

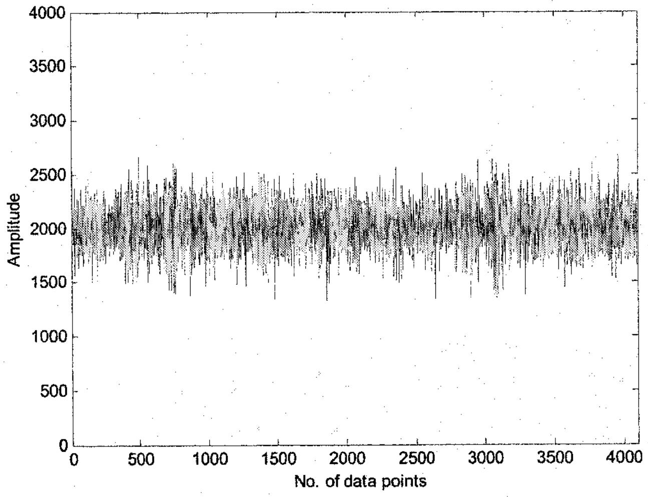


Figure 4.5. Vibration signal at the beginning of experiment

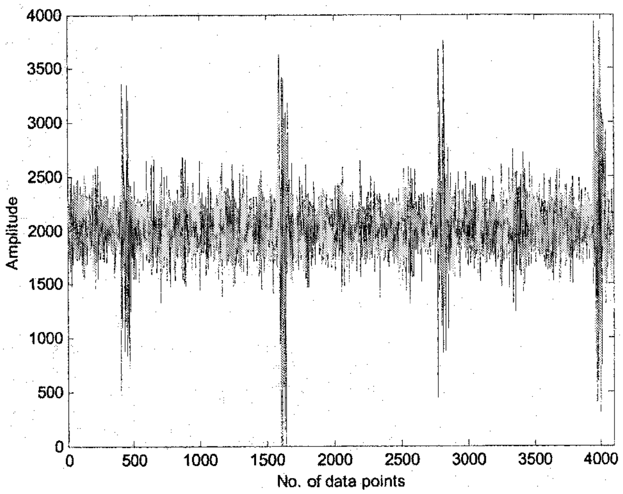


Figure 4.6. Vibration signal at the end of experiment

4.3.1.2 Enveloping

The energy of the envelope from the original signals is used as the indicator of fault appearance and degradation condition of the gearbox. Enveloping method is widely used in vibration analysis and condition based monitoring (Yu et al. 1994, Barkov and Barkova, 1996). To obtain the envelope of signal, we firstly remove the mean of signal and then take its absolute value. After that, we make fast Fourier transformation (FFT) to transform signal from the time domain to the frequency domain. In the frequency domain, we filter out the signal above a predetermined cutoff frequency to get the envelope spectra. The reason to filter out the high frequency components is to enlarge the envelope, which appears at low frequency. But if the cutoff frequency is too low, some characteristics of enveloping will be lost. Thus, the cutoff frequency is determined by the nature of the signal. With this balance in mind, the cutoff frequency is set to be one tenth of the maximum frequency in this experiment. It means only frequency components lower than one tenth of the maximum frequency will be left after filtering. In Figure 4.7, we plot out the signal at the beginning of experiment and its transform after enveloping. The Matlab program of enveloping is listed in Appendix 4.

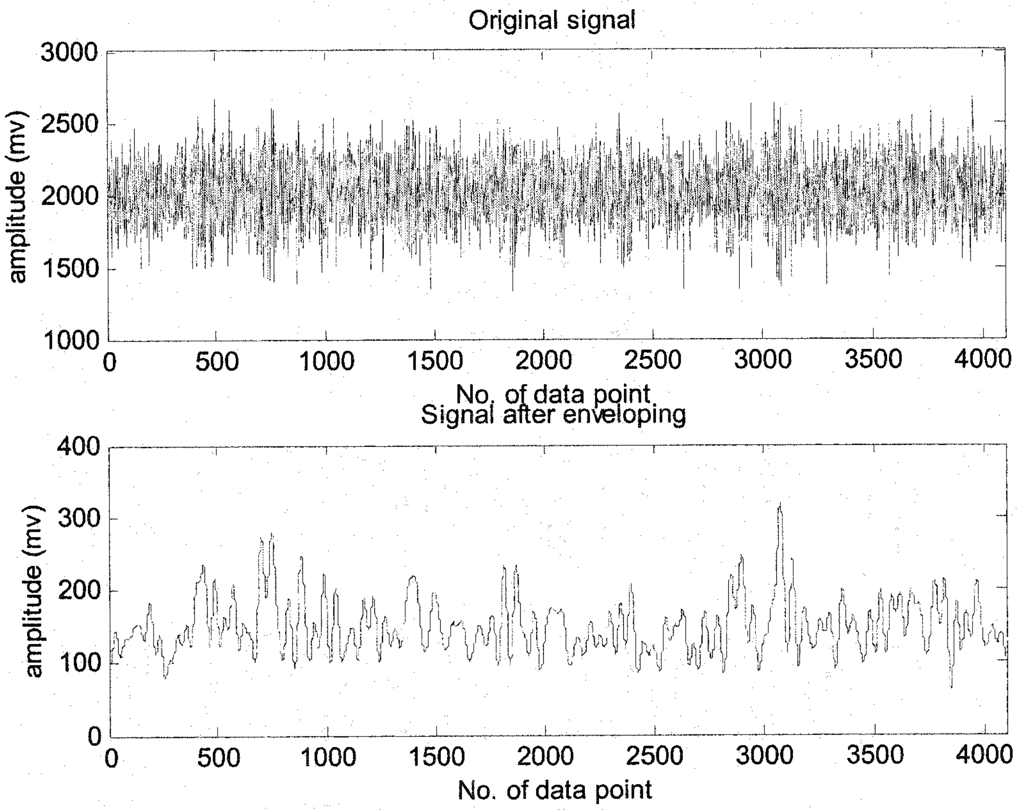


Figure 4.7. Original signal and its transform after enveloping

After filtering, we can calculate the rooted mean square (RMS) value of the obtained set of data to represent the energy of envelope. For each set of 4096 recorded data points, we can obtain a single energy of the envelope representing the current gearbox working condition. Since we have 20 sets of data, we get 20 points after enveloping: $(x_1, y_1) \sim (x_{20}, y_{20})$, where y_i represents the energy of the envelope at a certain time. Because the gearbox is monitored with equal time interval, we can simply set $x_i = i$, $i = 1 \sim 20$.

4.3.1.3 One-Step-Ahead Prediction

We only considered one-step-ahead short-term prediction in this example. The reason was that we had only 20 data points altogether. A long term prediction based on such limited data points does not make much sense. Meanwhile, in real world application, it is not so practical to look too far into the future for on-line monitoring.

With such limited number of data points, we cannot use AR, ARIMA or other time series models any more because the rule of thumb to apply the time series models is that there are at least 50 data points for training (Shumway and Stoffer, 2000a). Instead, we use linear regression and 2nd order polynomial regression. Both of these two methods use time as independent variable as SVM does. To compare the prediction abilities of linear regression and 2nd order polynomial regression with SVM, the 20 data points were separated into two sets: the training set with 12 data points and the validation set with 8 data points. The generalization error is calculated from the validation set for each model obtained from the training data points.

Firstly, we perform one-step-ahead prediction using linear regression. We use the training data available to build linear regression model and predict one-step-ahead. The generalization error is calculated based on the difference between the estimated and observed value using equation (4.11). For example, with $(x_{1,} y_1) \sim (x_{12,} y_{12})$, the linear regression model is found to be:

$$y = 0.01372x + 18.45066 \tag{4.12}$$

Using this model, we can predict $\hat{y}_{13}$ = 18.629. The observed value at that point is 18.647. Using equation (4.12), we find the generalization error at point x = 13 to be 0.0545.

When a new data point is added into the old data series, a new series becomes available. Each time, we fit the linear regression model again to the new series. Then, we will get different linear models to predict the next step. The prediction results from linear regression models are plotted in Figure 4.8.

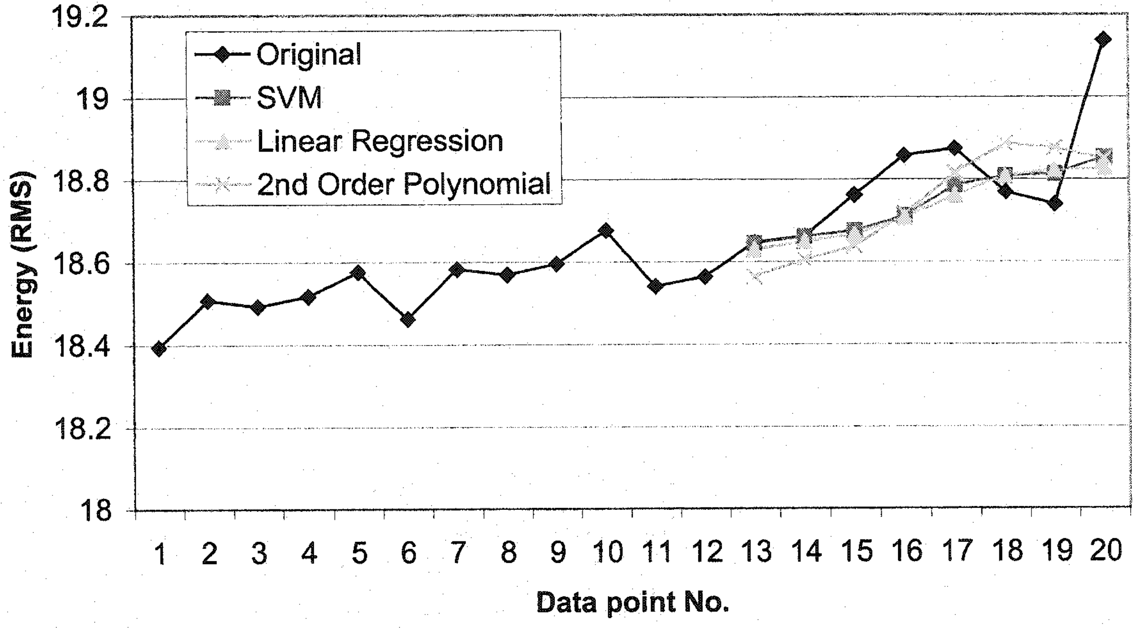


Figure 4.8. Comparison of the three models predicting gearbox vibration data

Secondly, we perform one-step-ahead prediction using the 2nd order polynomial regression. Similar to linear regression, we build the 2nd order polynomial model and use the obtained model to predict one-step-ahead. Using equation (4.11), we find the generalization error based on the difference between the estimated and observed values. For example, with $(x_1, y_1) \sim (x_{12}, y_{12})$, the 2nd order polynomial regression model is found to be:

$$y = -0.002146x^2 + 0.04162x + 18.3856 \qquad (4.13)$$

Using this model, we can predict $\hat{y}_{13}$ = 18.564. The observed value at that point is 18.647. Using equation (4.11), we can calculate the generalization error at point $x = 13$ is 1.2138.

Similar to linear regression, each time we fit the 2nd order polynomial regression model to the new series when there is a new data point added in. We have different 2nd order models for the next step prediction. The prediction results from 2nd order polynomial regression models are also plotted in Figure 4.8.

Now, let's look at SVM prediction. To build a SVM model, the linear (dot) kernel is selected as kernel function. Linear kernel is selected because preliminary trial and error shows it seems to be able to provide a small generalization error. Meanwhile, the original series plotted in Figure 4.8 seems to roughly follow a straight line.

Then, we use split sample method to determine the best λ and ε values for SVM prediction. The sample is divided into two groups: the latest point as validation point and the rest data points as training set. The generalization error calculated from validation point is used as the criterion for parameter selection.

Consider the series $(x_{1,}y_1) \sim (x_{12,}y_{12})$, we firstly conduct rough two-dimensional search in a wide range of ε: 0.03 ~ 0.06 and λ: 100 ~ 5000. The value of ε cannot set to be too big. If it is too big, too much deviation will not be punished. Because of page limitation, only the generalization errors in the search range of ε: 0.03 ~ 0.06, with step size 0.005, λ: 100 ~ 2100 with step size 200 are listed in Table 4.5.

Table 4.5 Different λ and ε values and the corresponding generalization errors by split sample validation for gearbox vibration data

λ ε	0.03	0.035	0.04	0.045	0.05	0.055	0.06
100	0.0038	0.0032	0.0026	0.0022	0.0016	0.0021	0.0025
300	0.0038	0.0032	0.0026	0.0022	0.0016	0.0021	0.0025
500	0.0038	0.0032	0.0026	0.0022	0.0016	0.0021	0.0025
700	0.0038	0.0032	0.0026	0.0022	0.0016	0.0021	0.0025
900	0.0038	0.0032	0.0026	0.0022	0.0016	0.0021	0.0025
1100	0.0038	0.0032	0.0026	0.0022	0.0016	0.0021	0.0025
1300	0.0038	0.0032	0.0026	0.0022	0.0016	0.0021	0.0025
1500	0.0038	0.0032	0.0026	0.0022	0.0016	0.0021	0.0025
1700	0.0038	0.0032	0.0026	0.0022	0.0016	0.0021	0.0025
1900	0.0038	0.0032	0.0026	0.0022	0.0016	0.0021	0.0025
2100	0.0038	0.0032	0.0026	0.0022	0.0016	0.0021	0.0025

In Table 4.5, we find that the λ value seems to have little effect to the generalization error in this example. Thus, we simply choose λ to be infinity. After that, we do one-dimensional search for optimal ε value. We set the search range of ε to be 0.045 ~ 0.055 with a step size of 0.001 because it seems that the optimal ε falls into this range. The generalization errors according to different ε values are listed in Table 4.6. In Table 4.6, "GE" denotes the generalization error.

Table 4.6 Generalization errors according to different ε values when λ is infinity

ε	0.045	0.046	0.047	0.048	0.049	0.05	0.051	0.052	0.053	0.054	0.055
GE	0.0022	0.0021	0.002	0.0019	0.0018	0.0016	0.0017	0.0018	0.0019	0.002	0.0021

From Table 4.6, we can see that when $\varepsilon = 0.05$, we will get the minimum generalization error based on split sample validation. Thus, we pick $\varepsilon = 0.05$ and $\lambda =$ infinity for SVM building and next step prediction.

When a new data point is added into the old data series, a new series will be generated. For different data series, different optimal λ and ε are selected before making a one-step-ahead forecasting. The λ and ε values are updated as listed in Table 4.7. In Table 4.7, "Inf" represents "Infinity".

Table 4.7 Updated λ and ε values for gearbox vibration data

Point #	13	14	15	16	17	18	19	20
λ	Inf	Inf	10	Inf	Inf	Inf	10	Inf
ε	0.05	0.06	0	0.02	0.095	0.02	0.04	0.08

After determining the parameters of λ and ε, Gunn's SVM Matlab toolbox is used for SVM model building and one-step-ahead prediction. The one-step-ahead prediction results are plotted in Figure 4.8.

The generalization errors of the three models calculated from equation (4.11) are listed in Table 4.8. From Table 4.8, we find SVM creates the smallest generalization error. It means SVM is the best forecasting tool in the three models for this example.

Table 4.8 One-step generalization errors of and computation time the three methods for gearbox vibration data

	Linear Regression	2nd order polynomial	SVM
Generalization Criterion	0.9223	1.2461	0.7657
Computation Time (seconds)	0	0	6

4.3.2 Vibration Trend of A Synchronous Motor

In this example, we use the vibration trend data (Miller, 1998) analyzed in Chapter 3. This time, we compare the one-step-ahead prediction using SVM and time series models.

As we have 120 points all together, the first 100 points, $x_1 \sim x_{100}$, are used for model building. The rest 20 points, $x_{101} \sim x_{120}$, are reserved to compare the prediction results of different models.

Firstly, let's consider time series model. From the analysis in Section 3.5.3, we know that ARIMA(5, 1, 5) is able to provide the best forecasting. So we just apply the result here and use ARIMA(5, 1, 5) for all the 20 points one-step prediction. But for each new data points added in, we re-train ARIMA(5, 1, 5) to get its coefficients. For example, as shown in Chapter 3, for data set $x_1 \sim x_{100}$, ARIMA(5, 1, 5) is define as:

$$(1+ .38B^1 - .27B^2 - .06B^3 + .33B^4 + .07B^5)\, \nabla x_t$$
$$= (1+ 1.15B^1 - .47B^2 + .07B^3 - .01B^4 - .23B^5)\, w_t \qquad (4.14)$$

For data set $x_1 \sim x_{101}$, we rebuild the model and for this series, ARIMA(5, 1, 5) has the form of:

$$(1- .21B^1 - .74B^2 + .35B^3 + .48B^4 - .47B^5)\, \nabla x_t$$
$$= (1+ .71B^1 - .39B^2 - .32B^3 + .22B^4 - .53B^5)\, w_t \qquad (4.15)$$

We can see that, these two models are significantly different. In fact, for these 20 predictions, we trained ARIMA(5, 1, 5) 20 times to get 20 different sets of model parameters.

Next, let's look at SVM prediction. The linear spline kernel is selected as kernel function after trial and error. Then, we use split sample method to determine the best λ and ε values for SVM prediction. The sample is divided into two groups: the latest point as validation point and the rest of the data points as training set. The generalization error calculated from validation point is used as the criterion for parameter selection.

Consider the series $x_1 \sim x_{100}$, we firstly conduct rough but wide range search. Part of the obtained generalization errors is listed in Table 4.9.

Table 4.9 Different λ andε values and the corresponding generalization errors by split sample validation for motor vibration data

λ ε	0	0.2	0.4	0.6	0.8	1.0	1.2	1.4
$1*10^5$	297.4886	285.6247	274.0306	263.3316	251.4014	240.1442	231.078	239.8465
$2*10^5$	266.4069	267.1193	265.0143	279.9995	286.1106	290.1384	288.7285	236.4374
$3*10^5$	238.9853	245.2469	251.5684	258.7454	265.6227	272.0243	283.8693	283.5852
$4*10^5$	214.1178	219.552	226.1879	232.9631	239.4706	244.0779	258.925	271.5592
$5*10^5$	190.2558	195.7078	202.0092	208.4468	214.948	221.2528	235.0626	243.1418
$6*10^5$	167.8129	173.3443	179.2208	185.2564	187.2768	199.5389	210.8125	212.6026
$7*10^5$	146.6999	152.3046	157.8071	163.4752	166.0317	178.7045	183.1117	184.0863
$8*10^5$	126.9047	132.2958	137.7887	141.3626	147.0167	156.272	156.5635	157.6192
$9*10^5$	108.5451	113.5775	118.6695	118.4892	129.1754	131.7438	132.0744	133.2471
$10*10^5$	91.6194	96.3926	101.1395	102.1426	108.9836	109.3911	109.6686	118.6611
$11*10^5$	76.2785	80.6634	82.6471	87.2296	88.7756	89.0946	97.9025	99.3299
$12*10^5$	62.4056	66.2772	65.356	70.4983	70.642	78.7382	76.1811	70.8493

From Table 4.9, we find that the larger the λ is, the smaller the generalization error will be. Thus, we select λ to be infinity. After that, we conduct one-dimensional search for optimal ε value. We set the search range of ε to be 0 ~ 1.0 with a step size of 0.1. The obtained generalization errors are listed in Table 4.10. In Table 4.10, "GE" denotes generalization error. When $\varepsilon = 0.8$, we will get the smallest generalization error.

Table 4.10 Generalization errors according to different ε values when λ is infinity

ε	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
GE	6.8458	5.2419	3.8518	2.6755	1.7127	0.9637	0.4286	0.0964	0.0019	0.1581	0.4694

Based on split sample validation, the optimal parameters for SVM regression are: $\varepsilon = 0.8$ and λ = infinity. After selecting λ and ε values based on series $x_1 \sim x_{100}$, we keep them for all the one-step-ahead predictions on validation set. The Matlab source code for one-step prediction based on Matlab toolbox (Gunn, 1998) is attached in Appendix 5. The original values, prediction results from ARIMA(5, 1, 5) and SVM are plotted in Figure 4.9. We use equation (4.11) to calculate generalization errors from both of these two methods. The calculation results are listed in Table 4.11. The computation time of SVM and ARIMA(5, 1, 5) is also listed in Table 4.11.

Table 4.11 One-step generalization error and computation time of the two models for motor vibration trend

	ARIMA(5, 1, 5)	SVM
Generalization Error	0.037018	0.058356
Computation Time (seconds)	10	912

From Table 4.11, we can find that SVM provides a larger generalization error than ARIMA(5, 1, 5) in this example. However, they are pretty close as we can see from Figure 4.9. If we update the parameters λ and ε after each new point or a certain number of points added in, we might get a better prediction result.

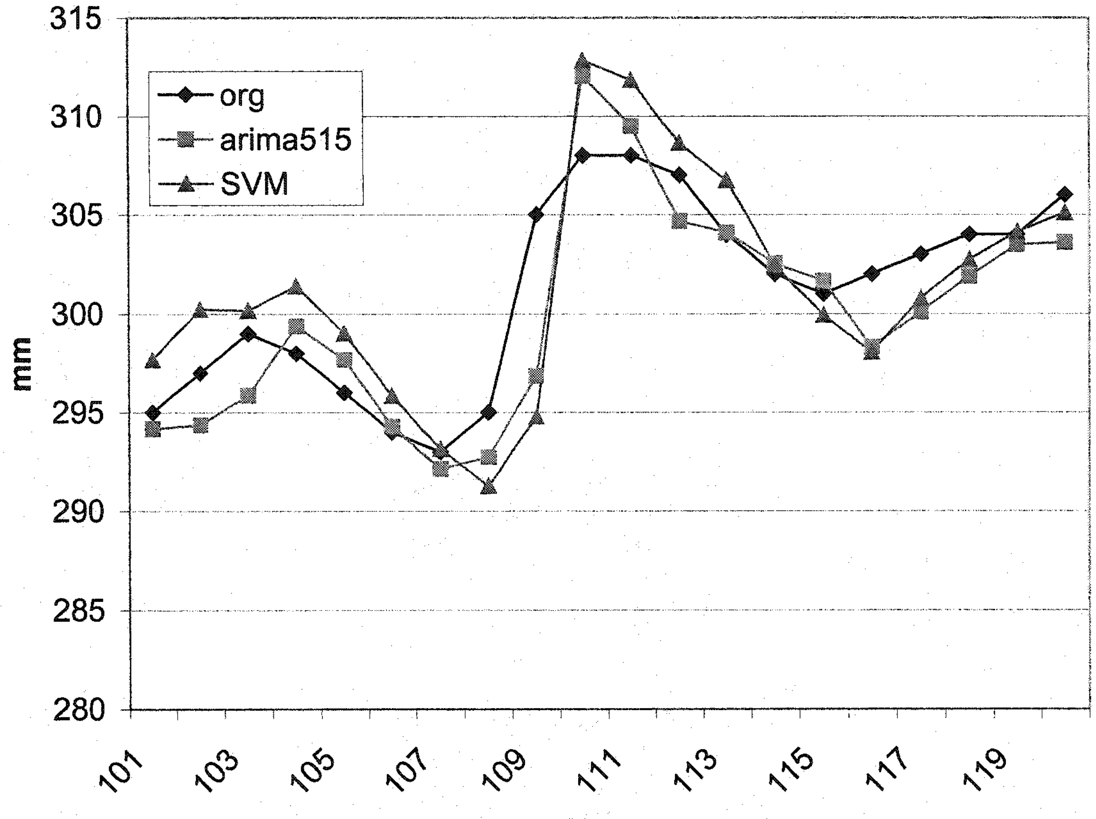


Figure 4.9 Comparison of the two methods predicting motor vibration trend

4.4 Conclusions and Discussions

This chapter applies SVM to future vibration level prediction. The applications on Australian expenditure on financial services data and gearbox vibration data show that SVM is able to provide a better prediction with smaller generalization error than linear regression, 2nd order polynomial models, and time series model. Because SVM attempts to minimize the generalization error instead of training error, it is aiming to provide a better forecasting. However, in the example of motor vibration trend prediction, SVM produces a larger generalization error than the time series model. The prediction result from SVM might get improved if we update the parameters λ and ε after each new point or a certain number of points are added into the series.

The time series models can be applied only when each data point is measured with the same time interval. If the data are not sampled with equal time interval, time series model do not apply. In this case, the possible candidates include linear, nonlinear regression methods and SVM. SVM provides a method for model capacity control to avoid the problem of overfitting. Therefore, SVM is likely to provide a better prediction than linear and nonlinear regression methods. In the three examples studied in this chapter, the sizes of data set are small. Further work is needed to test large data sets. We also try to use SVM for long term forecasting. It does not show much advantage.

In SVM, the parameters λ and ε are combined together to control both capacity and empirical risk. When λ is infinity, it means that we do not allow empirical error if we apply ε-insensitive cost function. The solution will be the flattest linear function for which ε-insensitive empirical risk function is zero in the feature space. Finding the best parameters of SVM is still a research issue deserving further investigation. To obtain the optimal λ and ε, cross validation method is suggested by many researchers. However, no detailed research work is found discussing how to select λ and ε. Cross validation is computation intensive. Meanwhile, we also find the model with the smallest generalization error from cross validation may not be the best forecasting model. The reason is that cross validation aims to minimize the generalization error between the lower and upper limits of the data set. However, what we are interested is forecasting, which is to predict points outside the upper limit.

In this chapter, we suggest to use split sample validation to find the optimal λ and ε values for prediction. It seems to provide reasonable good result as shown in the three examples studied. However, it does not guarantee that we will find the real optimal values. Future research work is needed to find the best SVM parameters for prediction.

In the examples studied in this chapter, all the optimal λ values are found to be infinity. As discussed before, λ determines the weight of the empirical risk. The optimal λ might be different for different data sets and for different cost function we use. Thus, it is just by chance that the optimal λ values are infinity in the examples studied here.

When a new data point is observed and added to the original data series, a new series will be generated. The best SVM parameters λ and ε for this new series might not be the same as for the old one. If we have to find out the optimal λ and ε each time a new data is added in, the work will be tremendous. A possible solution for this problem is to keep the old parameters for a certain period of time or a certain number of observations, say 10. After every 10 new data have been observed and added in, a program will automatically run to search for new optimal λ and ε for the updated series. The advantage of this method is the searching program can always run at the background since it is computational expensive. While in the front, we can keep taking new data points and conduct SVM prediction based on old parameters. Whenever the new optimal λ and ε are found by the background program, the old parameters will be immediately updated.

The draw back of SVM is that it is not computation efficient. As we compare in this chapter, SVM is hundreds times slower than linear regression or nonlinear regression.

Chapter 5. Conclusions and Future Work

An effective maintenance system should be able to monitor the operating conditions of a machine, issue advanced warnings of possible faults, and predict the remaining life of a deteriorating machine. If the equipment's remaining life can be accurately predicted, appropriate preventive maintenance tasks can be scheduled in time to prevent equipment breakdowns. Certain research works have been done to forecast the trend of machine degradation by monitoring the amplitudes of its fault related vibration features. This thesis studies forecasting methods for equipment degradation based on vibration analysis. It will provide useful information about how long the equipment can last and when it is most appropriate to shut it down for maintenance.

The well-used prediction methods include time series methods, neural networks, support vector machine, general path model, and continuous state system reliability analysis. All these methods have their advantages and challenges. In this thesis, time series methods and support vector machine are studied.

A new method for optimal prediction time series model order(s) selection is proposed. The method aims to combine the advantages of AIC and split-sample method for time series model order(s) determination. AIC measures the goodness of fit of existing data while split-sample method can verify the goodness of forecasting in the validation set. The applications show that we are able to obtain a better forecasting model with smaller forecasting error by applying this new method than solely depending on AIC. The proposed method combines the advantages of both AIC and split-sample validation so that it is effective in forecasting model selection. Application results of the proposed method in comparison with using AIC only also confirm the effectiveness of this method.

Support vector machine (SVM) is a new tool for regression and classification. It has been successfully applied to time series prediction. SVM is likely to provide a better prediction because it controls the model capacity at the same time as trying to minimize training error. But no work has been reported to apply SVM to predict future vibration level based on vibration recordings. In this thesis, we apply SVM to short term vibration prediction. Split sample validation is used for optimal kernel function and parameters λ and ε selection. The drawback of split sample validation is that it does not necessarily provide the optimal forecasting parameters. As new data points are added in, the parameters obtained from split sample validation for old series might not be optimal for the series at present time. We suggest to update the parameters λ and ε for every a certain number of observations, say 10. This will update λ and ε values while not consuming much time.

Three examples are studied to compare the forecasting results from SVM and from other forecasting methods such as linear or nonlinear regression, AR and ARIMA. Two out three examples show that SVM provides better short term predictions. To the example that SVM does not perform so well, better prediction result from SVM is expected if we update the parameters λ and ε after each new point or a certain number of points are added in. One of the prerequites of using time series models is the data points should be separated by equal time interval. It limits the applications of time series models. When the time interval is not unique, we can select SVM as our prediction tool. But another problem with SVM is that it is computation extensive.

Future research issues include the following:

1. How to select the best weight factor s between AIC and generalization error. Weight factor s directly affects *comb* value and therefore model order(s) selection. It relates to the magnitude of AIC and the logarithm of the generalization error from the validation set. The method of AIC considering generalization error is effective only when s is properly selected.

2. How to split up sample to obtain a proper size for the validation set. If the size is too big, too many data points are not used in model building. But if it is too small, it may be hard to verify the goodness of forecasting. The size of the validation set also influences the generalization error and therefore the selecting of weight factor s.

3. How to find the optimal λ and ε for SVM to provide a best prediction. This problem is inherent in SVM. Cross validation is a possible solution but far from ideal. It only aims to minimize the generalization error between the upper and lower limits of the data set. A method needs to be developed to find optimal SVM model parameters for prediction.

4. How to find the optimal kernel function and its parameters for SVM prediction. Kernel function selection is another factor that may affect the prediction result of SVM. Little research work has been found discussing how to select an optimal kernel function.

5. Test the models on more testing points. The number of testing points in this thesis is quite limited. If more data points are available for testing and comparing, the justification will be more convincing.

Appendices

Appendix 1. Residual tests for Example 1 ARIMA(2,1,4) residuals (Obtained from WinASTSA)

```
Residual tests for 2nd Polynomial ARIMA(2,1,4) residuals
T = 67
Cumulative spectrum test
  max. diff.   p-value
   .090         N.S.
Probable constant variance.

Box-Pierce test
  lag  chi sq.  p-value
   1     .23     N.S.
  20   14.32     N.S.
Probable independence.

Fluctuation test
   z          p-value
   1.08        N.S.
Probable random series.

Outlier detection
  max. |z|     p-value
   2.30         N.S.
No outliers.

Normal test
   corr.      p-value
  .99730        N.S.
Probable Gaussian distribution.
```

Appendix 2. Matlab Source Code for Split Sample Validation for SVM

```
% split_s, This is contained in a file named split_s.m
% This program is for split sample validation test to select the best SVM model
% parameters.

clear all      %clear memory

tic            % starts a stopwatch timer.

X1 = X;
Y1 = Y;
N = length(X);         % size of the input vector
k = 1;                 % predict how many data points in validation set

X2 = X(1 : N-k);       % save training set to X2
X = X(1 : N-k);
Y = Y(1 : N-k);
Y2 = Y1(N-k+1 : N);
step = X(2) - X(1);    % calculate step size
n = 1;                 % counter
result = 0;            % initial value of sum of generalization errors
X = svdatanorm(X, ker);

for e = 0 : 2 : 10     % search range for e
  m = 1;
  for C = 10000 : 200 : 11000     % search range from C
    tstY = 0;                  % test Y value, obtained from SVM regression
    e_error = 0;               % individual generalization error of each fold

    [nsv beta bias] = svr(X, Y, ker, C, 'einsensitive', e); % SVM regression
%     tstX = (1-(k-1)*step : step : 1)';
%     tstX = (1 : step : 1 + (k-1)*step)';

    tstX = (1 : step : 1+k*step)';                % validation set
    tstY = svroutput(X, tstX, ker, beta, bias);   % SVM prediction

    e_error = Y2 - tstY;                          % calculate error
    result(m, n) = sum(e_error.^2);               % sum of the squared error

    m = m + 1;
  end
  n = n + 1;
```

```
end

t = toc % prints the elapsed time since tic was used.

X = X1;
Y = Y1;

result
```

Appendix 3. Matlab Source Code for Australian Expenditure on Financial Services Data Set SVM Regression

```
% svm_reg_fin, This is contained in a file named svm_reg_fin.m
% This program performs SVM regression, one-step-ahead forecasting and
% generalization error calculation for Australian expenditure on financial services
% data set

clear all;              % clean memory

ker = ' spline;         % Set the kernel to be lineal spline
C = ‘inf’;              % Capacity set to be infinity
loss = 'einsensitive'; % loss function set to be e-insensitive
e = 0.01;               % e value set to be 0.01

X1 = X;                 % save X to variable X1
Y1 = Y;                 % save Y to variable Y1

f_size = 86;            % data set size is 86
tr_size = 60;           % training set size is 60
ts_size = f_size - tr_size;   % testing set size = data set size – training set size

tstY = 0;               % initial value of prediction vector set to be zero
e_error = 0;            % initial residual value set to be zero
testX = X1(tr_size + 1 : f_size);   % build testing data set
testY = Y1(tr_size + 1 : f_size);   % build testing data set
step = X(2) - X(1);                 %calculate the step size

sigma = 0;              %initial value of generalization error is set to be zero

tic % starts a stopwatch timer.

for i = tr_size + 1 : f_size
   trainX = X1(1 : i - 1);     % build training data set
   trainY = Y1(1 : i - 1);     % build training data set

   X = svdatanorm(trainX, ker);   % standardize X
   Y = trainY;

   [nsv beta bias] = svr(X, Y, ker, C, 'einsensitive', e); %SVM regression
   tstY(i) = svroutput(X, 1 + step, ker, beta, bias); % one step ahead prediction

   e_error(i) = Y1(i) - tstY(i);      % calculate estimation error
   var_Y = var(trainY);               % calculate the variance of training set
   sigma = sigma + e_error(i).^2 / var_Y; %calculate generalization error
end

t = toc                                  % prints the elapsed time since tic was used.

sigma = sigma / ts_size                      % get the final generalization error
tstY = tstY(tr_size + 1 : f_size)';

figure;                                      % create a new figure
plot(testX, testY, '*-', testX, tstY, '-r');     % plot out the result

save result_fin;                             % save the result into a file
```

Appendix 4. Matlab Source Code for Enveloping Implementation

```
% filt, This is contained in a file named filt.m
% This program performs spectra enveloping

fid = fopen(file, 'r');          % open data file saving recorded signal
x = fscanf(fid, '%f\n');         % open data file saving recorded signal

x1 = x;                          % save original signal to x1
N = length(x);                   % get the length of the signal
X = x - mean(x);                 % remove mean
X = abs(x);                      % take absolute value

f = fft(x);                      % Fast Fourier Transformation (FFT)
L = fix(N / 10);                 % set the filter band width
f(L : N – L + 1) = 0;            % low pass filtering
s=real(ifft(f));                 % inversed FFT, back to time domain

figure; subplot(2,1,1); plot(x1); title(‘Original signal’)    % plotting
subplot(2,1,2); plot(s); title(‘Signal after enveloping’);    % plotting

energy(i)=sum(s.^2);                         % calculate energy of envelope spectra
```

Appendix 5. Matlab Source Code for Vibration Trend SVM Regression

```
% svm_reg_vt, This is contained in a file named svm_reg_vt.m
% This program performs SVM regression, one-step-ahead forecasting and
% generalization error calculation for motor vibration trend data

clear all;              % clean memory

ker = ' spline';        % Set the kernel to be lineal spline
C = 'inf';              % Capacity set to be infinity
loss = 'einsensitive'; % loss function set to be e-insensitive
e = 0.01;               % e value set to be 0.01

X1 = X;                 % save X to variable X1
Y1 = Y;                 % save Y to variable Y1

f_size = 120;           % data set size is 120
tr_size = 100;          % training set size is 100
ts_size = f_size - tr_size;  % testing set size = data set size – training set size

tstY = 0;               % initial value of prediction vector set to be zero
e_error = 0;            % initial residual value set to be zero
testX = X1(tr_size + 1 : f_size);   % build testing data set
testY = Y1(tr_size + 1 : f_size);   % build testing data set
step = X(2) - X(1);                 %calculate the step size

sigma = 0;              %initial value of generalization error is set to be zero

tic % starts a stopwatch timer.

for i = tr_size + 1 : f_size
   trainX = X1(1 : i - 1);     % build training data set
   trainY = Y1(1 : i - 1);     % build training data set

   X = svdatanorm(trainX, ker);    % standardize X
   Y = trainY;

   [nsv beta bias] = svr(X, Y, ker, C, 'einsensitive', e); %SVM regression
   tstY(i) = svroutput(X, 1 + step, ker, beta, bias); % one step ahead prediction

   e_error(i) = Y1(i) - tstY(i);        % calculate estimation error
   var_Y = var(trainY);                 % calculate the variance of training set
   sigma = sigma + e_error(i).^2 / var_Y; %calculate generalization error
end
```

```
t = toc                                  % prints the elapsed time since tic was used.

sigma = sigma / ts_size;                 % get the final generalization error
tstY = tstY(tr_size + 1 : f_size)';

figure;                                  % create a new figure
plot(testX, testY, '*-', testX, tstY, '-r');  % plot out the result

save result_VT;                          % save the result into a file
```

References

[1] Akaike, H., "Fitting Autoregressive Models for Prediction", *Ann. Inst. Stat. Math*, Vol. 21, pp. 243 – 247, 1969.

[2] Akaike, H., "Information Theory and an Extension of the Maximum Likelihood Principal", *2nd Int. Symp. Information Theory*, pp. 267 – 281, B.N. Petrov and F. Csake eds. Budapest: Akademia Kiado, 1969.

[3] Akaike, H., "A New Look at Statistical Model Identification", *IEEE Trans. Automat. Contr.*, AC – 19, pp. 716 – 723, 1974.

[4] Andrews, D. F., Gnanadesikan, R., and Warner, J. L., "Transformations of Multivariate Data", *Biometrics*, vol. 27, pp. 825 – 840, 1971.

[5] Barkov, A. V., and Barkova, N. A., "Diagnostics of Gearing and Geared Couplings Using Envelope Spectrum Methods", *20th annual meeting of the Vibration Institute*, 1996, http://www.vibrotek.com/articles/gears/index.htm.

[6] Boser, B. E., Guyon, I. M., and Vapnik, V. N., "A Training Algorithm for Optimal Margin Classifiers", *Proceeding of Fifth Annual Workshop Computational Learning Theory*, ACM Press, New York, pp. 144 – 152, 1992.

[7] Bowerman, B.L., and O'Connell, R.T., *Forecasting and Time Series: An Applied Approach*, Duxbury Press, Belmont, Calif., 1993.

[8] Brie, D, Dal Ponte, D, Tomczak, M, and Richard, A, "Estimation of the Order of the AR Part of ARMA Models with Application to Frequency Estimation" *SignalProcessing VII: Theories and Applications*, pp. 664 – 667, 1994.

[9] Brockwell, P.J., and Davis, R.A., *Time Series: Theory and Methods*, Springer-Verlag, 1987.

[10] Brockwell, P., and Davis, R.A., *Time Series: theory and methods*, Springer-Verlag, New York, 1991.

[11] Cortes, C., and Vapnik, V., "Support Vector Networks", *Machine Learning*, Vol. 20, pp. 273 – 297, 1995.

[12] Cortes, C., *Prediction of Generalization Ability in Learning Machines*, PH.D Thesis, University of Rochester, 1995.

[13] Cristianini, N., and Shawe-Taylor, J., *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, 2000.

[14] Drucker, H., and Burges, "Support Vector Regression Machines", *Neural Information Processing Systems* 9, eds: M.C. Mozer, J.I. Jordan, T. Petsche, pp. 155-161, MIT Press (1997) with C.J.C. Burges, L. Kauffman, A. Smola, and V. Vapnik.

[15] Fausett, L. V., *Fundamentals of Neural Networks*, 2002.

[16] Fugate, M.L., Sohn, H, and Farrar, C.R., "Vibration-based Damage Detection Using Statistical Process Control", *Mechanical Systems and Signal Processing*, Vol. 15, Issue 4, pp. 707 – 721, 2001.

[17] Gestel, T.V., Suykens, J.A.K., Baestaens, D.E., Lambrechts, A. Lanckriet, G., Vandaele, B., Moor, B.D., and Vandewalle, J., "Financial Time Series Prediction Using Least Squares Support Vector Machines Within the Evidence Framework", *IEEE Transactions on Neural Networks*, Vol. 12, NO. 4, pp. 809 – 821, 2001.

[18] Goutte, C., "Note on Free Lunches and Cross-Validation", *Neural Computation*, Vol. 9, pp. 1211 – 1215, 1997.

[19] Gunn, S.R., "Support Vector Machines for Classification and Regression", *Technical Report*, Department of Electronics and Computer Science, University of Southampton, 1998.

[20] Hearst, M. A., "Trends & Controversies: Support vector machines", *IEEE Intelligent Systems*, July/August 1998, pp. 18 – 28, 1998.

[21] Jardine, A.K.S., Joseph, T., and Banjevic, D., "Optimizing Condition-based Maintenance Decisions for Equipment Subject to Vibration Monitoring", *Journal of Quality in Maintenance Engineering*, Vol. 5, No. 3, pp. 192 – 202, 1999.

[22] Jiang, F., Zuo, M.J. and Lin, J., "Forecasting of Machine Degradation Based on Vibration Trends and Support Vector Machines", *Proceedings of CSME*, Kingston, Canada, 2002, Accepted.

[23] Johnson, R.A., and Wichern, D.W., *Applied Multivariate Statistical Analysis, 3rd ed.*, Englewood Cliffs, NJ: Prentice-Hall, 1992.

[24] Kendall, M.G., *Time-Series*, Griffin London, 1973.

[25] Lu, C.J., and Meeker, W.Q., "Using Degradation Measures to Estimate a Time-to-Failure Distribution", *Technometrics*, Vol. 35, No. 2, pp. 161 – 174, 1993.

[26] Makridakis, S., Anderson, A., Carbone, R., Fildes, R., Hibon, M., Lewandowski, R., Newton, J., Parzen, E., and Winkler, R., *The Forecasting Accuracy of Major Time Series Methods*, John Wiley & Sons, 1984.

[27] Miller, M., "Machine Condition Information – A Valuable Resource for Strategic Management Decisions", *Orbit*, Bently Nevada, pp. 30 – 31, June, 1998.

[28] Mukherjee, S., E. Osuna, and F. Girosi, "Nonlinear Prediction of Chaotic Time Series using a Support Vector Machine", In J. Principe, L. Gile, N. Morgan, and E. Wilson (Eds.), *Neural Networks for Signal Processing VII — Proceedings of the 1997 IEEE Workshop*, New York. IEEE, 1997.

[29] Muller, K.-R., A. Smola, G. Ratsch, B. Scholkopf, J. Kohlmorgen, and V. Vapnik, "Predicting Time Series with Support Vector Machines", In B. Scholkopf, C.J.C. Burges, and A.J. Smola (Eds.), *Advances in Kernel Methods — Support Vector Learning*, Cambridge, MA. MIT Press. Short version appeared in *ICANN'97, Springer Lecture Notes in Computer Science*, pp. 243–254, 1999.

[30] Plutowski, M., Sakata, S., and White, H., "Cross-validation Estimates IMSE", *Advances in Neural Information Processing Systems 6,* Cowan, J.D., Tesauro, G., and Alspector, J. (eds.), San Mateo, CA: Morgan Kaufman, pp. 391 – 398, 1994.

[31] R Development Core Team, *An Introduction to R: Notes on R: A Programming Environment for Data Analysis and Graphics*, 2000.

[32] SAS Institute Inc., *SAS User's Guide*, 2001.

[33] Scholkopf, B., Bartlett, P., Smola, A., and Williamson, R., "Support Vector Regression with Automatic Accuracy Control", *Proceedings ICANN 98*, 1998.

[34] Schwarz, F., "Estimating the Dimension of a Model", *Ann. Stat.*, Vol. 6, pp. 461 – 464.

[35] Shao, J., "Linear Model Selection by Cross-validation", *Journal of the American Statistical Association*, Vol. 88, pp. 486 – 494, 1993.

[36] Shumway, R.H. and Stoffer, D.S., *Time Series Analysis and Its Applications*, Springer, 2000a.

[37] Shumway, R.H. and Stoffer, D.S., *The ASTSA Manual,* http://anson.ucdavis.edu/~shumway/tsa.html, 2000b.

[38] Smola, A.J., *Regression Estimation with Support Vector Learning Machines*, Master's Thesis, Physik Department, Technische University Munchen, 1996.

[39] Smola, A.J., and Scholkopf, B., "A Tutorial on Support Vector Regression", *NeuroCOLT2 Technical Report Series*, NC2-TR-1998-030, 1998.

[40] Sugiura, N., "Further Analysis of the Data by Akaike's Information Criterion and the Finite Corrections", *Commun. in Statist, A, Theory and Methods*, Vol. 7, pp. 13 – 26, 1978.

[41] Tse, P.W. and Atherton, D.P., "Prediction of Machine Deterioration Using Vibration Based Fault Trends and Recurrent Neural Networks", *Journal of Vibration and Acoustics*, Vol. 121, pp. 355 – 362, 1999.

[42] Vapnik, V., *The Nature of Statistical Learning Theory*, Springer, N.Y., 1995.

[43] Vapnik, V.N., *Statistical Learning Theory*, John Wiley, 1997.

[44] Venables, W.N., Ripley, B.D., *Modern Applied Statistics with S-PLUS*, Springer-Verlag, New York, 1999.

[45] Wang, X.D., *Tool Wear Function Identification and Tool Replacement / Adjustment Decision Making*, M.Sc. Thesis, Department of Mechanical Engineering, University of Alberta, Edmonton, Alberta, Canada, 1994.

[46] Weiss, S.M. and Kulikowski, C.A., *Computer Systems That Learn*, Morgan Kaufman, 1991.

[47] Yu, A., Barkov A., and Yudin I., "Automatic Diagnostics and Condition Prediction of Rolling Element Bearings Using Enveloping Methods", *18th annual meeting of the Vibration Institute,* June, 1994, http://www.vibrotek.com/articles/new94vi/index.htm.

[48] Zuo, M.J., Jiang, R., and Yam, R.C.M., "Approaches for Reliability Modeling of Continuous-State Devices", *IEEE Transactions on Reliability*, Vol. 48, No. 1, pp. 9 – 18, 1999.

[49] http://www.support-vector.net/software.html