EARTHQUAKE PREDICTION USING NEURAL NETWORKS

by
NAZIFE DIMILILER
B.Sc. in E.E., Rutgers University, 1992

Submitted to the Institute for Graduate Studies in Science and Engineering in partial fulfillment of the requirements for the degree of Master of Science in Computer Engineering

1995

ACKNOWLEDGEMENTS

I would like to express my grattitute and endless apprecition to both my supervisors Dr. Ethem Alpaydin and Dr. Aysin Baytan Ertüzün for their continued academic and moral support. I am especially indepted to Dr. Aysin B. Ertüzün for giving me many chances for completing my thesis.

I would like to thank Prof. Cemil Gürbüz and Dr. Fikret Gürgen for participating in my jury, and for their invaluable help and support throughout the project.

I am also deeply greateful to Dr. Levent Akin and Serif Baris, who has made valuable suggestions and comments on my work.

I am indepted to both my supervisors, Dr. Ethem Alpaydin, and Dr. Aysin B. Ertüzün, my jury members Prof. Dr. Cemil Gürbüz and Dr. Fikret Gürgen, and also to Dr. Levent Akin for their sincere interest and endless kindness.

I would, also, like to thank my family and friends who let me be, and put up with me.

Finally I want to phrase my grattitude to my supervisors again for making it possible for me to complete my thesis. Thank you.

ABSTRACT

Earthquake Prediction is a mainly unsolved problem. A large number of different approaches have been tried and only a small number of attempts were fruitful. A few of these are explained briefly in this thesis. One of the most succesful earthquake prediction sytems in use today is the Canada-Nevada, CN, algorithm. It is discussed and contrasted to the neural networks implemented in the project.

For this project the earthquake prediction problem is treated as a time series prediction problem and neural networks that have been used for ordinary time series prediction with some success have been applied to the problem. The data used was treated as a two dimensional time series with two variables; the magnitude of the present earthquake, and the time elapsed since the previous earthquake. The neural network architectures implemented were the multilayer perceptron network with sigmoidal activation function, NADINE, and a multilayer network with chaotic activation function. The results were not succesful because of the complex nature of input data and the earthquake generation process.

ÖZET

Tam olarak çözülmemiş bir problem olan deprem tahmin problemini yapay nöron ağlarına uyguladık. Manyitüt ve iki deprem arasındaki zamandan oluşan verilerin bir zaman serisini meydana getirdikleri ve bu serinin deprem sırası hakkında tüm gerekli bilgiyi içerdiğini kabul ederek, daha önce çeşitli zaman serilerini tahmin etmede ve modellemede kullanılan bazı nöron ağlarını çalıştırdık.

Tezde deprem tahmin etme amacıyla kullanılan algoritmaların en başarılılarından olan Canada-Nevada algorithması kısaca açıklanıyor ve bu algoritmanın başarılı olurken, tezde uygulanların başarısız olmasının nedenleri tartışılıyor. Ayrıca nöron algorithmalarıyla birlikte Box Jenkins yöntemi de uygulanıyor.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

A_c	Activation of the node c
B^m	Backward shift operator for m steps
D	Fractal Dimension
Dev	Deviation from long term trend
E	Total Error
$E[\,]$	Expectation operator
F^m	Foreward shift operator for m steps
G_i^{-1}	ith root of the AR weight equation $\Phi(B)$
H_i	Coefficient of ith root of the AR weight equation $\Phi(B$
$\boldsymbol{J}$	Jacobian Matrix
K	Increase in seismic activity
$L()$	Likelihood function operator
M	Magnitude of an Earthquake
$N()$	Counting function
$Omax$	Maximum area of fractures
$O(.)$	Order of a solution
$P()$	Probability distribution function
$Q()$	Quiscence function
$\Re$	Set of real numbers
$R()$	Ratio of the number of main shocks in two magnitude ranges
$S()$	Conditional sum of squares
$Scon()$	Spatial Concentration of seismicity
$Taq()$	Spatial contrast of activity
T_t	Interevent time between two events
$Tvar()$	Temporal variation of seismicity
$Y()$	Clustering of earthquakes
W	Weight vector of an adaptive system
a_t	White noise process value at time t
$\hat{a}_t$	Estimated white noise value at time t
b	The slope of the completeness curve
$b_i()$	Number of aftershocks
c_k	Estimated autocovariance for lag k
d	Degree of differencing for model identification
$d()$	Deficiency of activity function
d_p	Degree of a polynomial
$e()$	Local error in NADINE
h	A period of time
ht	Depth of the epicenter of an earthquake
$g()$	An arbitrary function
$l()$	Log likelihood function operator, $l=ln[L()]$
m	Embedding dimension
p	Order of AR process
$p(f)$	Power spectrum operator for frequency f
$p_i(t)$	Evolution of the principal axis of an ellipsoid in time t

q	Order of MA process
$r_k()$	Estimated autocorrelation function for lag k
s	Time, can be discrete or continous
s(.)	A nonlinear function
t	Time, can be discrete or continous
v	Fraction of intervals in which an earthquake occurs
$var[\ ]$	Variance opeartor
w	Weight of a connection in neural network
w_t	Value of the stationary process, formed by applying the operator ∇ until stationarity conditions are satisfied, at time t, in Box-Jenkins algorithm
z_t	Time series value at time t
$\tilde{z}_t$	Time series value at time t minus the mean of the process
Δ	Short time duration
Γ	A scalar function
$\phi(B)$	Weights $(1, \phi_1, \phi_2, ..., \phi_p)$ of an AR process
$\theta(B)$	Weights $(1, \theta_1, \theta_2, ..., \theta_q)$ of an MA process
Ξ_i	Projection onto axis i
$\psi(B)$	Weights, $(1, \psi_1, \psi_2, ...)$ of a general linear process
α	Fractal attractor
β	Weight of an earthquake
γ_k	Theoretical autocovariance function for lag k
$\chi_j(z)$	A local function
δ	A set of weights for NADINE
$\gamma(B)$	Theoretical autocovariance generating function
$\gamma_{za}(k)$	Theoretical covariance function between z and a for lag k
φ_{lj}	Partial autocorrelations of a function
λ	Lyapunov exponent
μ	Mean of a signal
ρ_k	Theoretical autocorrelation function for lag k
σ_x^2	Variance of signal x.
ω	A set of weights for NADINE
ξ	Vector of the general set of parameters
ζ	A magnitude value used to normalize the weights
∇^d	Bakward difference operator of order d
$\bar{\cdot}$	Operator for the mean of the signal
$\hat{\cdot}$	Estimated value of a signal
$\tilde{\cdot}$	The value of a signal minus its mean
$\{.\}$	Time series

1. INTRODUCTION

One of the central problems of science in the study of naturally occuring phenomena is forecasting the future values of a system using the previous and present knowledge about its behavior. There are two main approaches to choose for prediction purposes. These are the model based approach and the statistical one. The model based approach, which is usually expected to be more reliable, requires a priori and usually extensive knowledge about the underlying dynamics. Most often this is not available, or the economical or physical laws giving rise to the system are not fully understood. In such a situation, a stochastic prediction was the only choice until recently. Stochastic prediction schemes basically involve a step where they analyze the sequence of observations in order to infer from the statistics or dynamics of the process some valuable knowledge on the future evolution of the sequence. Another possible approach is using the neural networks. Neural networks, like the statistical approach, does not require extensive knowledge about the underlying dynamics. However any knowledge about the system can be used to guide the choice of the network architecture. This approach assumes that information about the generating function, is implicit in the data produced, and the network is trained adaptively on a set of data. After the network has learned the mapping in the training set to a sufficient error margin, the network is used for prediction. Sometimes it may be necessary to preprocess the data to attain logical results, but neural networks, unlike the conventional statistical methods that require human intervention and judgement at every stage, build a valid model autonomously. The aim of this project is to use neural networks for earthquake prediction, treating the data as a time series.

Nature tends to produce very complicated irregular and usually nonstationary and chaotic processes, and this is one of the main problems faced by a prediction model for naturally occuring phenomena. Furthermore it is usually impossible to isolate the system as there are a large number of intermingled factors affecting the system. This implies that a large number of variables will be needed to describe the process. Measurement and inclusion of all these variables may be practically impossible. Therefore some appropriate metric, should be involved in order to limit the input variables to the most important ones. Some information will thus be lost.

The time series analyzed in this thesis consists of a four dimensional earthquake data. The ten year earthquake catalog of the area in western Turkey, between lattitudes 39.50-41.50N and longitudes 25.00-28.50E, starting from 1/1/1970 was supplied by Prof. Cemil Gürbüz and Serif Baris from Kandilli Observatory and Earthquake Research Inst. Each

earthquake in this catalog is described by a time of occurence to the nearest minute, three dimensional coordinates, lattitude, longitude and depth, and magnitude of the event. In order to simplify the system and since it was not very reliably measured the depth variable was ignored in the subsequent analysis.

The basic assumption underlying this project is that the four dimensional earthquake sequence, as described by a time of occurence, two dimensional coordinates, lattitude and longitude, and magnitude, contains enough information about the dynamical system producing it for building a model capable of making valid predictions. This may not be necessarily true and the failure of building an acceptable model might be due to the lack of information. Another simplification that we have performed was to limit the region analyzed to a small area so that only the time of occurence and magnitude of an earthquake would be used to describe it. This proved to be a bad choice as it ignored effectively a large portion of the underlying dynamical system that gave rise to the earthquake sequence in that region.. In other words by studying a smaller region without any consideration to the faulting structure we have ignored some of the earthquakes that were generated by the same dynamical system that generated the earthquakes in the study region. Thus valuable information was lost. Actually even the whole area that was to be analyzed, might be too small for it to accurately contain all the information pertaining to the sequence of earthquakes. In fact even though the complete set of data had complete information on small earthquakes starting with magnitude 2.6 according to the Richter scale, it was found that it did not contain enough information on earthquakes with high magnitudes. The highest magnitude that was at all included in the data set was 5.5. The faulting system of the area, as well as other physiacal properties such as the structure of litosphere and the stress and strain regime of the area, must be used as guide when determining the boundaries of the unit region to be studied. This extra information is absolutely necessary to make sure that the information content of the data collected from the region is not incomplete. The time span considered is also important and the competeness analysis must be performed to make sure that there is enough information on both high and low magnitudes. If the information is not comlete then a longer time span or a different geometry for the region must be considered.

There have been various attempts to earthquake prediction problem, mostly based on identification of the precursary signs of earthquakes, with magnitudes above a threshold, that occured in the past and announcing that there is a probability of an earthquake of a given magnitude range occuring in a specified time span. Usually this time span is on the order of years and the magnitudes of the predicted earthquakes are above a threshold so that they would actually have an effect, that is they would actually cause economical damage or

harm life in some way. Small earthquakes are generally not predicted because they do not have any direct effect on human life and because they occur very frequently.

We have not encountered any application of the problem of earthquake prediction as a continuation of a time series to adaptive neural networks, or to any other methods, in the literature. There have been various succesful applications of seismic processes to neural networks but they are all classification problems [1,2] This is to be expected since neural networks are well known for their powerful interpolation capability. Given a smooth enough mapping an adaptive neural network can reconstruct it from the time series. They have also been succesful in modelling or identifying nonlinear dynamic systems [3,4]. Such networks do not perform as well when the problem is manipulating new dynamic information, or extrapolating. They usually require additional information in order to extrapolate which might either be unavailable or unknown. Therefore practical application of neural networks to time series prediction of naturally occuring sequences have not been very succesful. Another approach to earthquake prediction has been to visually analyze plots of a set of variables related to succesive earthquakes [5]. These variables provide a mean for representing time difference and distance between succesive events, clustering, rate of clustering and diffusion of seismicity. The analysis, effectively, identifies the precursory signs of large earthquakes and these precursory signs are used for further prediction.

The earthquake prediction problem is described in greater detail in the second section which discusses two different approaches to earthquake prediction or modelling. One of the approaches is to fit a curve or a line through the earthquake data [6-9]. Namely these are the time magnitude predictable scheme [6-8], that looks for a relation between the interevent time between the previous event and the present event and the magnitude of the present event, together with the bayesian probabilistic prediction scheme that uses teh bayesian theory to predict the probability of an earthquake at a given time[9]. The other main approach is the pattern classification scheme [10-12]. In this scheme the precursory conditions before major earthquakes are identified in the training phase, and if these conditions are satisfied an earthquake is predicted. It was found that the application of the pattern recognition schemes has been favorable in most cases [13].

Section three is devoted to a discussion of different approaches to time series estimation. A traditional forecasting method, Box-Jenkins [14], is discussed in this section. It is to be used as a check mark for the performance of the neural network application. As another non-neural application nonlinear and linear parametric prediction schemes are discussed for the case of chaotic series. The chaotic series applications are included because

the earthquake sequence data is known to possess a chaotic exponent as discussed in the section four. Third section also introduces two of the architectures implemented in this project as well as two alternative ones which were not implemented because of time restriction. The time series prediction problem applied to the presented neural networks is discussed in this section for general data.

Finally in the fourth section the earthquake occurence sequence is analyzed to reveal its properties. Fractal dimension of the temporal distribution of the earthquake magnitudes, and completeness analysis are presented in this section. Also the fourier transform as well as the autocorrelation functions and partial autocorrelations, that will be used in the Box-Jenkins prediction method, are given in this section. The results of the application of the time series prediction are given in the fifth section together with explanations for the chosen architecture and parameters.

A plot of the epicenters of all earthquakes in the data set is presented in the introduction of section 4. A very brief explanation of the dynamics of the region is also given in the same section.

2. EARTHQUAKE PREDICTION PROBLEM

There have been various attempts for predicting the earthquake occurence sequence and identifying the underlying dynamical system giving rise to this sequence in the geophysics society. Some researchers have found that the mathematical system underlying the seismicity of the regions they studied is nonstationary and nonlinear [5,6]. This would imply that a prediction scheme or a model for identification based on a limited data set may be valid only for a relevant length of time. That is, building a satisfactory model does not guarantee prediction in the same error margins for a long time under ideal conditions, unless the model is adaptive. Furthermore it implies that if the modelling attempts do not take into account this nonstationarity of the system, a more complicated system with complex dynamics will be built in order to account for the variations due to nonstationarity.

Earthquake occurence system is a very complex one depending on a large number of natural and human induced factors. Ideally when making a prediction or identifying the underlying system producing the earthquakes, all relevant seismic activity should be taken into account. Furthermore the tectonic regime of the region is also very important in at least the initial model building phase. Variables such as change in magneticity of the region can give some implication of a coming earthquake. In fact anything that might indicate that there is abnormal action underneath the crust might be helpful in predicting the possibility of an earthquake occuring in future. Theoretical as well as practical limitations, restrict researchers to a much smaller set than the set mentioned above. It is not possible to determine all relavant factors because the dynamics change in time and the system is not completely understood. Also it does not make too much sense to measure all the suspected factors. Thus, most of the models built for understanding the dynamics of the earthquake generation process or for predicting the future evolution of it, are too simplified. Another deficiency of the reasearch, done up to this point is, perhaps, that most of the work done has been devoted to earthquakes with magnitudes greater than a certain threshold which is usually taken to be around 5. Using such a magnitude cutoff restricts the scope of the study and therefore the scope of possible understanding of the underlying dynamical system. removing the aftershock sequence of major earthquakes is a very commonly used simplification approach. Even though in some cases the number or, intensity of aftershocks are used as a parameter, exclusion of these shocks must result in some information loss.

Model building is very helpful in understanding the dynamics of a system, and therefore making predictions about its future evolution. For this reason a model of the earthquake generation process in the Aegean area, which also includes the Marmara region

studied in this thesis, is presented. This model is not very helpful in prediction, and it seems to be too restricted. In section 2.2, one of the most widely used pattern recognition algorithms for earthquake prediction is presented. This algorithm uses the same type of input, the data from earthquake catalogs, as the type used in this thesis. There are two versions of this algorithm, one for prediction in regions with transform faulting and the other one for prediction in subduction zones. The fact that the faulting structure is an important factor for building a model of earthquake generation process was mentioned before. The original pattern recognition algorithm mentioned above, Canada-Nevada algorithm [10], or CN algorithm, was originally applied to the Canada, Nevada regions, and it was found to be succesful in prediction. Afterwards it was applied to a number of other regions with some success. But there were regions where it did not show a good performance as well. It turned out that, the algorithm performed well for regions whose faulting system are similar to that of the Canada Nevada region. That is the CN algorithm was succesful for prediction in regions with transform faulting, but unsuccesful in subduction zones. The reason was that in these regions with different faulting structures, the the precursory conditions were different. The CN algorithm will be explained in some detail in order to give some idea about the features that can be used for earthquake classification. M8 [11], algorithm is an adaptation of CN to subduction zones.

2.1 Time and Magnitude Predictable Model for Earthquake Generation

Modelling the earthquake generation process is very desirable since it will, probably, allow a better understanding of the actual generation process and also will help in predicting the future earthquakes. There have been some attempts for finding the relationship between some variables pertaining to earthquakes. Papazachos [7] has analyzed the area 34-43N and 18-30E, that includes the whole Aegean and the surrounding areas in Greece, Albania, southern Yugoslavia, southern Bulgaria and western Turkey, and determined the following relation between interevent times and magnitudes of events that occured in the region.

$$\begin{aligned} \log T_t &= 0.36\, M_{min} + 0.35\, M_p + const1, \\ M_f &= 0.95\, M_{min} - 0.49 + const2. \end{aligned} \tag{2.1.1}$$

where T_t is the repeat time, or interevent time, measured in years and M^p and M^f are, respectively, the surface wave magnitude of the preceeding and the following mainshocks, M_{min} is the smallest earthquake considered and *const1* and *const2* are constants that vary from source to source. The magnitude threshold for this analysis was 5.5 and data from 1855 to 1990 was used. The region was divided into 49 smaller regions each corresponding to a different source . The amount of data used was very small, in some cases only two events occured in the region analyzed. The region analyzed by Papazachos is one of the most active regions of Eurasia. Thrust type faulting occurs along the southern and western parts, inner Aegean experiences normal faulting and strike-slip faulting occurs along the northern Anatolia seismic belt and its extension [7]. Papadimitriou [8], has applied the same method to the western coast of South and Central America and has arrived at the same result.

The area considered in this thesis is included in the analysis above but in the time span we considered there was only a few earth movements above 5.5. The rest of the data did not show any such distribution and the number of events above Richter scale 5.0 were very few. The original catalog used was not complete for the bigger magnitudes. This might have been the reason for the failure to find a relation in the same form as that of Papazachos and Papadimitrou.

A similar approach as the above has also been used by other researchers. For instance Ferraes [9] has used Bayes' theorem in order to construct a relation for predicting next earthquake with a magnitude greater than 6 according to Richter scale. Ferraes has applied his algorith to the Omatepec segment which belongs to a subduction zone. In [6] the earthquake generation process is simulated in order to help in understanding the underlying dynamics. This kind of experimention is very helpful in the initial stages of model building for the prediction problem.

2.2 Earthquake Detection using Pattern Recognition Techniques

There are some applications of pattern recognition to the prediction problem of earthquakes such as the CN and M8 algorithms [10,11], both produced by the same group. These algorithms make use of precursary signs of earthquakes, such as clustering or quiscence, increasing of seismic activity to a certain level, migration, expansion of seismicity, simultaneous changes, increase in the level of seismic activity of the region,

increase in clustering of earthquakes and a period of relative quiscence among other things as they occured for previous events.

The idea behind the CN algorithm is basically to evaluate 9 functions that describe the precursory indications of a strong earthquake then a time of increased probability of the occurence of a strong earthquake (TIP,) is announced for a given time. These functions were chosen to account for experimental results and expectation. For example it was observed that a major earthquake was preceeded by a variation in the intensity of the earthquake flow, in the form of quiscence or increase in the level of earthquakes. Therefore these traits was represented by three functions. Also it is suspected that the underlying system is nonlinear so that traits such as clustering in time and space were incorporated into the algorithm. The functions used in CN are listed below [10,12]. The magnitude range considered is $M_l \leq M \leq M_u$. The integral traits of the earthquake flow are defined in a sliding time window $t\text{-}s \leq t_i \leq t$. Let t_i and M_i be the time and magnitude of the main shock respectively. The number of earthquakes with magnitudes greater than or equal to a specified threshold is

$$N(t|M_l, s) \tag{2.2.1}$$

The functions are listed briefly.

1. Clustering of earthquakes. Let $b_i(M_a, h)$ be the number of aftershocks with magnitude $M \geq M_a$ within the period h for each main shock i. The measure of clustering will be the maximum of all $b_i(M_a, h)$.

$$Y_{max}(t|M_l, s, M_a, h) = max_i\, b_i(M_a, h) \tag{2.2.2}$$

2. The weighted number of earthquakes within intervals of time and magnitude. This function is weighted according to M_i.

$$\Sigma(t|M_l, M_u, s, \zeta, \beta) = \Sigma 10^{\beta(M_i - \zeta)} \tag{2.2.3}$$

where ζ is used to normalize the weights so that only the magnitude range will be used This is done in order to allow the algorithm to be applied to different areas with same type of behavior. The parameter β is determined by one of the two conditions: the weight of an earthquake is roughly proportional either to the area of the rapture in the source or to its linear dimension. If the energy released by an earthquake is dependent on its magnitude according to the relation $log(E) = cons + BM$, where M is the magnitude, then $\beta = (2B/3)$, or $\beta = (B/3)$.

3. The ratio of the number of earthquakes for two overlapping magnitude ranges (M_1, M_2) and $M \geq M_1$.

$$R(t|M_{1l}, M_{2l}) = 1 - (N(t|M_{2l}, s) / N(t|M_{1l}, s)) \qquad (2.2.4)$$

4. Deficiency of activity or quiscence. The following integral is taken over only positive values.

$$Q1(t||M_1, s, u) = \int_{t-s-u}^{t-s} [n(M_1 s - N(t|M_1, s)]dt \qquad (2.2.5)$$

5. Quiscence is defined in one more way. This is taken to be the last minimum of $N(t)$. Only the last 15 years are considered in this function

$$Q2(t|M_1, s) = [N(t_1|M_1, s) - N(t_2|M_1, s)] - [N(t|M_1, s) - N(t_2|M_1, s)] \qquad (2.2.6)$$

6. Deviation from the long term trend. This defines the temporal variation of seismicity.

$$Dev(t|M_1, s) = N(t|M_1, t - t_0) - N(t - s|M_1, t - s - t_0)(t - t_0) / (t - s - t_0) \qquad (2.2.7)$$

7. Increase in activity. This function calculates the difference between the number of earthquakes at two succesive intervals *(t-s,t)* and *(t-2s,t-s)*.

$$K(t|M_l,s) = N(M_l,s) - N(t-s|M_l,s) \tag{2.2.8}$$

8. Maximum area of fractures in the source within u years. $\beta=2/3Y$.

$$O_{max}(t|m,M',s,u,\alpha,\beta) = \max_{[t-u,t]}[\Sigma(t|m,M_l,M_u,s,\alpha,\beta)]2/3[N(t|t|M,s) - N(t|M_u,s)] \tag{2.2.9}$$

9. Spatial contrast of activity is defined as simultaneous quiscence and activation in adjacent regions. Quiscence is diagnosed if $N(t|M_l,s) \leq N_q$ and activation is diagnosed if $N(t|M_l,s) \leq N_a$, where N_q and N_a are threshold number of events. Spatial contrast of activity is measured as time since the region iunder study and some adjacent region were in opposite states for more than one year and defined as

$$T_{aq}(t|M_l,s,p) \tag{2.2.10}$$

The original application of this algorithm requires extensive knowledge of the underlying system and of course needs to be trained for each area seperately even though the functions have been normalized so that it would be applicable to a wide range of regions with different seismic activity level. The problem is that it cannot be applied to every area directly and needs to be used by an expert. It was shown that even though it is applicable to California Nevada region and a large number of other regions on different continents it cannot be applied to the subduction zones such as some regions in Japan and Mexico [13]. The reason is that the nine functions computed for the California Nevada region is not applicable to these areas therefore a new extensive study of these functions need to be done. For instance one of the nine functions used in the CN algorithm is the number of

aftershocks but in the studied regions in Mexico a main earthquake is usually followed by seismic quiescence. Therefore one of the nine functions used in the CN is eliminated and prediction, or classification , should be done using only the remaining eight functions. Since the CN algorithm did not work as it was in such regions it had to be re-tailored to fit the new data set. For this reason the same group that devised CN produced the subduction-zone version of CN: M8 [11].

The M8 algorithm has been applied to Japan and other areas with some success but it can only predict earthquakes with magnitudes greater than 7.5. Pattern recognition techniques has been applied to detection and recognition of earthquakes but no such tecnique for adaptively predicting earthquakes using neural networks was found in the literature.

3. TIME SERIES PREDICTION METHODS

In this section a number of time series prediction methods are introduced. The subject of this thesis is to solve the earthquake prediction problem by using neural networks while treating the earthquake data as a time series. In an earthquake catalog each earthquake, that is each datum of the time series, is described by three coordinates, lattitude, longitude, and depth, time of occurence and magnitude. It is assumed that these information are sufficient for describing an earthquake event fully, and that prediction of possible future behavior of the system can be made by using only the time series data up to the present time.

It was mentioned in Section 2 that there have been various attempts to solve the earthquake prediction and analysis problem using pattern recognition techniques as well as other more conventional techniques such as line fitting [7-12]. No references to attempts, using artificial intelligence or any other approach, for predicting future earthquakes based solely on the past earthquake sequence was found in literature. Therefore this section introduces a number of conventional methods, such as the Box-Jenkins method, as well as the neural algorithms that can be used for such a prediction task. Unfortunately only a few of the methods presented in this section are actually applied to the problem because of time restriction. The result of the applications will be given in the last section together with discussion on possible improvements and other viable alternatives.

3.1 Box-Jenkins Forecasting System

There are a lot of mathematical models describing the evolution of physical phenomena in use today. Most of these are based on some physical laws or at least a theory validating the assumptions made for building the particular model. These systems are usually assumed to be totally deterministic, in other words calculating the future value of some time dependent quantity with arbitrary error margin is assumed to be possible. Most of the time this is just an approximation because there are always some unknown and/or uncontrollable factors, such as the variable wind velocity when launching a missile, that affect the system generating the time series. As stated before it is not possible to isolate the

system. Therefore probably none of the naturally occuring phenomena or systems is totally deterministic.

Even though totally deterministic models are just approximations of the real world they are usually fairly good approximations. There are problems where no assumption of determinism, and no use of theoretical rules, can be made. One such problem is predicting the monthly sales of a newsprint. In this case there are many unknown phenomena and it is not possible to build a deterministic model to desribe it. Stochastic models, or probability models are used for problems of this kind. A stochastic model calculates the probability of a value lying between two specified limits. Box-Jenkins Model [14], is a particular stochastic model building system and has been widely used as a tool for time series prediction and control in business and economics world for a long time. It is a well established sytem and therefore has also been used as a conventional forecasting sysem in order to test or show the validity of new forecasting algorithms.

Box-Jenkins model fits an Autoregressive Integrated Moving Average, ARIMA, process to the data set it is given. Both AR and MA are time series models themselves. An AR process models a process output as a linear combination of past values of the output plus a white noise error. An MA process can be considered, on the other hand, as a linear combination of past errors or equivalently as the output of a linear filter whose input is a white noise signal. The white noise signal or error is defined by Box and Jenkins, as random drawings from a fixed distribution usually assumed Normal and having mean zero and variance σ_a^2. Throughout this section the term stationary will be used to define processes with a constant mean. This loose definition, however will be replaced by the more formal definition, that the variance of a linear process should converge, whenever there is need to define the stationarity formally. The terms trend and level of a process will be used to refer to the mean and the slope, respectively, of a time series or of a part of it.

Throughout this subsection bold faced symbols will be used to denote vectors. Also frequent use of the backward shift operator B and the backward difference operator ∇, defined below will be made to simplify the representation.

$$\begin{aligned} Bz_t &= z_{t-1}, \ \textit{and} \\ B^m z_t &= z_{t-m}, \\ \nabla z_t &= z_t - z_{t-1} = (1-B)z_t. \end{aligned} \tag{3.1.1}$$

Let $\tilde{z} = z - \mu$, where μ is the mean, replace z in the sequel, so that the resulting process has a zero mean. Consider a general linear process:

$$\begin{aligned} \tilde{z}_t &= a_t + \psi_1 a_{t-1} + \psi_2 a_{t-2} + \cdots \\ &= a_t + \sum_{j=1}^{\infty} \psi_j a_{t-j} \\ &= \psi(B) a_t \end{aligned} \tag{3.1.2}$$

where $\psi(B) = 1 + \sum_{j=1}^{\infty} \psi_j B^j = \sum_{j=0}^{\infty} \psi_j B^j$, with $\psi_0 = 1$. This process can also be seen as the output of a linear filter whose input is white noise process a_t. The white noise process is a sequence of uncorrelated random variables with mean zero and constant variance as defined below:

$$\begin{aligned} E[a_t] &= 0 \\ var[a_t] &= \sigma_a^2. \end{aligned} \tag{3.1.3}$$

The autocovariance function of white noise is therefore zero for all lags except zero lag

$$\begin{aligned} \gamma_k &= E[a_t a_{t+k}] \\ &= \sigma_a^2, \ \ if \ k = 0, \\ &= 0, \ \ if \ k \neq 0. \end{aligned} \tag{3.1.4}$$

Then its autocorrelation function ρ_k is 1 when $k=0$ and zero otherwise. The autocovariance of (3.1.2) is:

$$\begin{aligned}\gamma_k &= E\left[\tilde{z}_t \tilde{z}_{t+k}\right] \\ &= E\left[\sum_{j=0}^{\infty}\sum_{h=0}^{\infty}\psi_j \psi_h a_{t-j} a_{t+k-h}\right] \\ &= \sigma_a^2 \sum_{j=0}^{\infty}\psi_j \psi_{j+k}.\end{aligned} \tag{3.1.5}$$

Then the variance of the same process is given by:

$$\gamma_0 = \sigma_z^2 = \sigma_a^2 \sum_{j=0}^{\infty}\psi_j^2. \tag{3.1.6}$$

The autocovariance generating function of a linear process is defined as:

$$\begin{aligned}\gamma(B) &= \Sigma \gamma_k B^k \\ &= \sigma_a^2 \psi(B)\psi(F).\end{aligned} \tag{3.1.7}$$

For stationarity the infinite series $\psi_j, j \le 0$ must converge. To see this consider the power spectrum of (3.1.2)

$$\begin{aligned}p(f) &= 2\sigma_a^2 \psi(e^{-i2\pi f})\psi(e^{i2\pi f}) \\ &= 2\sigma_a^2 \left|\psi(e^{-i2\pi f})\right|^2, \quad 0 \le f \le 1/2.\end{aligned} \tag{3.1.8}$$

Then the variance is:

$$\sigma_z^2 = \int_0^{1/2} p(f)df = 2\sigma_a^2 \int_0^{1/2} \psi(e^{-i2\pi f})\psi(e^{i2\pi f}). \qquad (3.1.9)$$

The requirement for stationarity of a process is that its variance, that is the integral in (3.1.9), must converge. And for the integral in (3.1.9) to converge the infinite series $\psi_j, j \leq 0$ must converge for B within or on the unit circle. Note that the operator B in the autocovariance generating function is replaced by $e^{-i2\pi f}$ to arrive at (3.1.8) and (3.1.9). These results will be referred to frequently for describing the special time series AR, MA, ARMA and ARIMA.

A purely AR process is in the form:

$$\begin{aligned} &\phi(B)\tilde{z}_t = a_t, \ \textit{where} \\ &\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p, \end{aligned} \qquad (3.1.10)$$

where $\tilde{z}_t$ is the deviation of the process from some origin. If the process is a stationary process μ can be its mean so that the resulting time series $\tilde{z}_t$ is has zero mean. Such a model contains $p+2$ unknown parameters: p weights ϕ_i, the mean μ, and the white noise variance σ_a^2. In the sequel p will be considered as the order of the AR process. An AR process can be stationary or nonstationary depending on the weights ϕ_i. The process defined in (3.1.10) can be transformed into a linear filter format as follows:

$$\tilde{z}_t = \frac{1}{\phi(B)} a_t = \phi^{-1}(B)a_t. \qquad (3.1.11)$$

$\phi(B)$ can be factorized to obtain $\phi(B) = (1 - G_1 B)(1 - G_2 B) \cdots (1 - G_p B)$. Then expanding in partial fractions,

$$\tilde{z}_t = \phi^{-1}(B) = \sum_{j=1}^{p} \frac{K_j}{(1 - G_j B)} a_t. \tag{3.1.12}$$

Comparing (3.1.12) to (3.1.2), it can be seen that, for $\psi(B) = \phi^{-1}(B)$ to be convergent for $|B| \leq 1$, then $|G_j| \leq 1$, where $j=0,2,...,p$. Thus the roots of the equation $\phi(B)=0$, also referred to as the zeros of the polynomial $\phi(B)$, must lie outside the unit circle.

For calculating the autocovariance function multiply (3.1.10) throughout by $\tilde{z}_{t-k}$, and take expectations of both sides. Note that $\tilde{z}_{t-k}$ contains shocks a_j only upto time t-k, and that previous shocks are uncorrelated with the present or future shocks. Then the autocovariance function is

$$\gamma_k = \phi_1 \gamma_{k-1} + \phi_2 \gamma_{k-2} + \cdots + \phi_p \gamma_{k-p}, \quad k > 0 \tag{3.1.13}$$

Dividing (3.1.13) throughout by γ_0 gives the autocorrelation function of the process

$$\rho_k = \phi_1 \rho_{k-1} + \phi_2 \rho_{k-2} + \cdots + \phi_p \rho_{k-p}, \quad k > 0 \tag{3.1.14}$$

The above difference equation can be written as follows

$$\phi(B)\rho_k = 0 \tag{3.1.15}$$

where $\phi(B) = 1 - \phi_1 B - \cdots - \phi_p B^p$, and B operates on k instead of t. Note that $\phi(B)$ can be expressed as $\phi(B) = \prod_{i=1}^{p}(1 - G_i B)$, so that $G_1^{-1}, G_2^{-1}, \ldots, G_p^{-1}$ are the roots of the characteristic equation $\phi(B)$ in (3.1.15). Then the general solution of (3.1.15) is:

$$\rho_k = H_1 G_1^k + H_2 G_2^k + \cdots + H_p G_p^k \tag{3.1.16}$$

It was shown that for stationarity it is required that $|G_i| \leq 1$. Therefore there are two possible situations for distinct G_i.

1. A root G_i is real and will geometrically decay to zero as k increases. This is refered to as a damped exponential.
2. A pair of root G_i, G_j are complex and follow a damped sine wave as desribed by

$$Hd^k \sin(2\pi fk + F) \tag{3.1.17}$$

Another important operation in time series analysis is the partial autocorrelation function Even though an AR process has an autocorrelation function which is infinite in extent, it can be represented by p nonzero functions of autocorrelations. Let ϕ_{kj} denote the jth coefficient in an autoregressive process of order k, ϕ_{kk} is the last coefficient. Then

$$\rho_j = \phi_{k1}\rho_{j-1} + \phi_{k2}\rho_{j-2} + \cdots + \phi_{kk}\rho_{j-k}, \quad k \geq j \geq 0 \tag{3.1.18}$$

This leads to the Yule-Walker equations

$$\begin{bmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-1} \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-2} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & 1 \end{bmatrix} \begin{bmatrix} \phi_{k1} \\ \phi_{k2} \\ \vdots \\ \phi_{kk} \end{bmatrix} = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_k \end{bmatrix} \tag{3.1.19}$$

The equation (3.1.19) can be solved for succesive values of k to obtain the partial autocorrelation functions ϕ_{kk}. The quantity ϕ_{kk} is regarded as a function of lag k, and for an AR process of order p it is nonzero only for k less than or eqaul to p. This is equivalent to saying that for a pth order AR process the partial autocorrelation function has a cutoff after lag p.

An MA process of order q is written as:

$$\begin{aligned} \tilde{z}_t &= a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \cdots - \theta_q a_{t-q} \\ &= (1 - \theta_1 B - \theta_2 B^2 - \cdots - \theta_q B^q) a_t \\ &= \theta(B) a_t \end{aligned} \tag{3.1.20}$$

Such an MA process of order q has $q+2$ unknown parameteers: q weights θ_i, the mean μ and the white noise variance σ^2 for the a_t which are generally estimated from the data. Comparison between (3.1.20) and (3.1.2) shows that $\psi(B) = \theta(B)$, and $\theta(B)$ is a finite series, therefore there are no restrictions on the coefficients to ensure stationary.

The autocovariance function of a MA(q) process is defined as

$$\begin{aligned} \gamma_k &= E\left[(a_t - \theta_1 a_{t-1} - \cdots - \theta_q a_{t-q})(a_{t-k} - \theta_1 a_{t-k-1} - \cdots - \theta_q a_{t-k-q})\right] \\ &= (-\theta_k + \theta_1\theta_{k+1} + \theta_2\theta_{k+2} + \cdots + \theta_{q-k}\theta_q)\sigma_a^2, \quad k = 1,2,\ldots,q, \quad and \\ &= 0, \quad k > q \end{aligned} \tag{3.1.21}$$

From (3.1.21) it is easily seen that the variance of this process is

$$\gamma_0 = (1 + \theta_1^2 + \theta_2^2 + \cdots + \theta_q^2)\sigma_a^2 \tag{3.1.22}$$

Thus the autocorrelation generating function of a MA(q) process is given by

$$\rho_k = \frac{-\theta_k + \theta_1\theta_{k+1} + \cdots + \theta_{q-k}\theta_q}{1 + \theta_1^2 + \cdots + \theta_q^2}, \quad k = 1,2,\ldots q \tag{3.1.23}$$

The autocorrelation function for lags greater than q is zero for a MA(q) process. In other words the autocorrelation function of a MA(q) process has a cut-off at lag q. It is possible to write the autocorrelation functions in the form of (3.1.19). However the Yule-Walker equations for a MA(q) process are not linear unlike the AR(p) process. Therefore iterative algorithms are employed to make crude estimates of the coefficients of the MA(q) process using the autocorrelations.

It is possible to convert between these two models by sacrificing parsimony. An AR process of finite order p is modeled by an infinite MA series and a MA process of finite order can be expressed as an AR process of infinite order. In other words a finite MA pocess is equivalent to an infinite AR process and vice versa. This observation explains the duality between the properties of the AR and MA proceses. For example a finite MA process has an autocorrelation function which is zero beyond a certain lag, however its partial autocorrelation functioin is infinite in extent.

In prediction tasks the the character and order of the process are generally not known . Therefore it is advantageous to model an unknown series as a mixed ARMA process to achieve parsimony.

$$\phi(B)\tilde{z}_t = \theta(B)a_t \tag{3.1.24}$$

which employs $p+q+2$ unknown parameters, $p+q$ weights and the mean of the process, and the variance of the white noise. This process is refered to as ARMA(p,q) process. It can be thought of as a special AR(p) process $\phi(B)\tilde{z}_t = e_t$, with e_t following a qth order MA process $e_t = \theta(B)a_t$. Similarly it can also be thought of as a qth order MA process $\tilde{z}_t = \theta(B)b_t$, where $\phi(B)b_t = a_t$, so that $\phi(B)\tilde{z}_t = \theta(B)\phi(B)b_t = \theta(B)a_t$. It can easily be seen that the moving average term on the right of (3.1.24) will not affect the stationaroty conditions established for the AR process in the preceeding text.

The autocorrelation and autocovariance function of the mixed ARMA process can be derived by multiplying (3.1.24) throughout by $\tilde{z}_{t-k}$. Then the autocovariance function is given by

$$\gamma_k = \phi_1\gamma_{k-1} + \cdots + \phi_p\gamma_{k-p} + \gamma_{za}(k) - \theta_1\gamma_{za}(k-1) - \cdots - \theta_q\gamma_{za}(k-q) \qquad (3.1.25)$$

where $\gamma_{za}(k) = E[\tilde{z}_{t-k}a_t]$ is the covariance function between z and a. But z_{t-k} depends only on shocks that occured up to time $t-k$, therefore $\gamma_{za}(k)$ is nonzero only for k less than or equal to 0. Therefore the autocorrelation function is defined as

$$\rho_k = \phi_1\rho_{k-1} + \phi_2\rho_{k-2} + \cdots + \phi_p\rho_{k-p}, \quad k \geq q+1 \qquad (3.1.26)$$

Thus for an ARMA(p,q) process there will be q autocorrelations $\rho_q, \rho_{q-1}, \ldots, \rho_1$, that depend directly on the q MA parameters θ as well as on the p AR parameters ϕ. Also, p of these values $\rho_q, \rho_{q-1}, \ldots, \rho_{q-p+1}$ provide the starting values for the difference equation (3.1.26). This difference equation, more compactly written as $\phi(B)\rho_k = 0$, determines all the autocorrelations at higher lags. If the order of the AR part of the ARMA process is higher than that of the MA part, i.e if $q-p<0$, the whole autocorrelation function ρ_j, for $j=0,1,2,\ldots$, will consist of a mixture of damped exponentials and/or damped sine waves. The nature of the exponentials and sine waves is determined by the polynomial $\phi(B)$ and the starting values. If on the other hand the order of the MA part of the ARMA process is

greater than or equal to that of the AR part, that is if $q\text{-}p \geq 0$, then $q\text{-}p+1$ initial values $\rho_0, \rho_1, \ldots, \rho_{q-p}$ will not follow this general pattern.

The ARMA process defined in (3.1.25) can also be written as

$$a_t = \theta^{-1}(B)\phi(B)\tilde{z}_t \tag{3.1.27}$$

$\theta^{-1}(B)$ is an infinite series in B. Hence the partial autocorrelation function of an ARMA is infinite in extent. Eventually the partial autocorrelation function of an ARMA process behaves like that of a purely MA process, and it is dominated by a mixture of damped exponentials and/or damped sine waves.

So far all the model processes discussed are stationary but many series encountered in practice, for example in industry and business, exhibit nonstationary behavior. These series however exhibit homogeneous behavior such that when differences in level and trend are allowed for the behavior of the series may be similar. Such a behavior may be represented by a general AR operator, $\varphi(B)$, which has one or more zeros on the unit circle. If the zeros are allowed to be outside the unit circle the system will be unstable and exhibit exponential behavior.

$$\varphi(B) = \phi(B)(1-B)^d \tag{3.1.28}$$

where $\phi(B)$ is the stationary AR operator defined before. Therefore homogeneous nonstationary behavior can be represented by the following model.

$$\begin{gathered} \varphi(B)z_t = \phi(B)(1-B)^d z_t = \theta_0 + \theta(B)a_t, \ \ or \\ \phi(B)w_t = \theta_0 + \theta(B)a_t, \ \ where \\ w_t = \nabla^d z_t \end{gathered} \tag{3.1.29}$$

This model is called ARIMA of order (p,d,q). The term $\phi(B)$ is called the autoregressive operator and is assumed to be stationary, in other words it is assumed to have its roots lie outside the unit circle. $\varphi(B)=\nabla^d\phi(B)$ is the generalized nonstationary autoregressive operator and has d of its roots lie on the unit circle. 'I' in the name ARIMA stands for Integrated, and it is used to explain that summation, which is the inverse of differencing operator, must be performed in order to isolate z_t on the left of (3.1.29). $\theta(B)$ is the moving average operator and is assumed to have its roots lie outside the unit circle for invertibility. It was explained before that a linear process is stationary if the variance of the process is finite, or equivalently the coefficients $\psi(B)$ converge. Note that the series $\psi(B)=\theta(B)$ is finite thus without any limitations on the MA parameters $\theta(B)$ the variance of the MA process is finite.. The additional term θ_0 allows inclusion of a deterministic function of time in the model. Without θ_0 the model is capable of representing series with stochastic trends such as random changes in level and slope of the series. Allowing θ_0 to be nonzero is equivalent to allowing the mean of the process to be nonzero. As an example consider a system where $d=1$. When θ_0 is nonzero the mean of the difference w_t is

$$E[w_t]=E[\nabla^d z_t]=\mu_w=\theta_0/(1-\phi_1-\phi_2-\cdots-\phi_p) \qquad (3.1.30)$$

Most of the time though there is not sufficient cause for assuming a deterministic trend for the practical systems, therefore it is not included unless it absolutely needed. ARIMA processes can be expressed in three different forms discussed in the preceding text. The value z_t can be expressed in terms of previous values of z's and present and previous values of a's. It can also be expressed in terms of current and previous shocks, that is only in terms of a_{t-j} for $j\geq 0$. Alternatively it can be expressed in terms of the previous values of z_{t-j}, for , and the present shock at only.

Difference equation representation of z_t is more convenient for use in forecasting than the others. It is in the same form as that of the ARMA process (3.1.24) except that the stationary AR operator, $\phi(B)$, is replaced by the general nonstationary one, $\varphi(B)$.

$$z_t = \varphi_1 z_{t-1} + \cdots + \varphi_{p+d} z_{t-p-d} - \theta_1 a_{t-1} - \cdots - \theta_q a_{t-q} - a_t, \quad where$$
$$\varphi(B) = \phi(B)(1-B)^d = 1 - \varphi_1 B - \varphi_2 B^2 - \cdots - \varphi_{p+d} B^{p+q} \tag{3.1.31}$$

The random shock form of the ARIMA model is the same as the expression for the output of a linear filter when input is white noise.

$$z_t = \psi(B) a_t \tag{3.1.32}$$

The linear filter weights can be found by applying the general AR operator, $\varphi(B)$, on both sides of (3.1.32) and equating the righthand side to the righthand side of (3.1.29). Then the linear filter weights, $\psi(B)$, are calculated using (3.1.33)

$$(1 - \varphi_1 B - \cdots - \varphi_{p+d} B^{p+d})(1 + \psi_1 B + \psi_2 B^2 + \cdots) = (1 + \theta_1 B + \cdots \theta_q B^q) \tag{3.1.33}$$

Note that for j greater than the larger of $p+d-1$ and q, $\psi(B)$ must satisfy $\varphi(B)\psi_j = \phi(B)(1-B)^d \psi_j = 0$. If the inverse of the linear filter operator is applied to both sides of (3.1.32) the signal z_t is expressed in terms of previous zs and current shock at.

$$\psi^{-1}(B) z_t = \pi(B) z_t = \left(1 - \sum_{j=1}^{\infty} \pi_j B^j\right) z_t = a_t \tag{3.1.34}$$

The values of the π werights can be found in terms of the ARIMA coefficients by applying MA operator to both sides of (3.1.34) and equating the lefthand side to that of the (3.1.29).

These results are used in identifying and modelling the characteristics of the data in an formalized way. Model building using Box-Jenkins system is an iterative process consisting of the following general steps.

i. Postulate a general class of models by inspecting the theoretical as well as the practical aspects of the situation.
ii. Identify the subclass of these models that is dictated by the data
ii. Estimate the parameters of the model. In other words fit the model to the data.
iii. Perform diagnostic checks. If the model does not fit the data properly repeat the procedure from step i.

These stages will be explained in more detail in the subsequent sections. Each one of these sections will refer to the basic theory given in the preceding paragraphs.

3.1.1 Model Identification

Two of the stages, identification and estimation, in Box-Jenkins modelling algorithm necessarily overlap. During identification process at least a crude estimation is made about the parameters. And at the estimation stage more information is collected so that if possible a simpler model than the one derived in identification stage can be employed. Model identification involves the task of finding the appropriate subclass of models from the general ARIMA processes

$$\phi(B)\nabla^d z_t = \theta_0 + \theta(B)a_t \qquad (3.1.35)$$

that satisfy the given contraints in the form of time series data. In the rest of the section the parameter θ_0 will be assumed to be 0, for reasons explained before. The approach is first to difference z_t, that is apply the differencing operator, as many times as needed to produce a stationary mixed autoregressive moving average process in the form below.

$$\phi(B)w_t = \theta_0 + \theta(B)a_t, \quad where$$
$$w_t = (1-B)^d z_t = \nabla^d z_t. \tag{3.1.36}$$

Then the resulting ARMA process is identified. In both steps use of autocorrelation and partial autocorrelation function is made.

For a stationary mixed ARMA process of order $(p,0,q)$ the autocorrelation function satisfies the following equation:

$$\phi(B)\rho_k = 0, \quad k > q \tag{3.1.37}$$

If the weights satisfy the relation

$$\phi(B) = \prod_{i=1}^{p}(1-G_i B) \tag{3.1.38}$$

then the autocorrelation function for this difference equation is of the form:

$$\rho_k = H_1 G_1^k + H_2 G_2^k + \cdots + H_p G_p^k, \; k > q-p \tag{3.1.39}$$

The stationarity condition is that the roots of $\phi(B)$ lie outside the unit circle, which implies that the roots G_i, $1 \le i \le p$, lie inside the unit circle. This means that if none of the roots lie close to the unit circle the autocorrelation function will die out rapidly for moderate or large k. The tendency for the estimated autocorrelation function not to die is taken as an indication of nonstationarity since it implies that a root close to unity exists. Such a process should be treated as nonstationary in z_t but possibly stationary in ∇z_t or higher differences.

After chosing the degree, d, of differencing necessary to achieve stationarity, the next step is to estimate the autoregressive order p, and the moving average order q. For this purpose use of estimated autocorrelation together with partial autocorrelation functions is made. The autocorrelation function of an autoregressive process of order p tails off and its partial autocorrelation has cutoff after lag p. The moving average process of order q, on the other hand has its autocorrelation function cutoff after lag q, while its partial autocorrelation tails off. If both the autocorrelations and partial autocorrelations tail off, a mixed process is suggested. Also the autocorrelation function of such a mixed process of the given orders has its autocorrelation function contain a mixture of exponentials and damped sine waves after the first (q-p) lags and its partial autocorrelation function be dominated by a mixture of exponentials and damped sine waves after the first (p-q) lags.

It should be kept in mind that in these stages estimated autocorrelations and partial autocorrelations are used as the theoretical ones are not available. These estimated values can be autocorrelated with each other and can have large variances. Also they should not be expected to very closely approximate the theoretical ones [14]. One of the reasons for these deficiencies is that the fixed precision causes the error to accumulate. It is not possible to remedy the situation so caution should be exercized when interpreting the estimated values. In particular it was found that large estimated autocorrelations can be found after the theoretical autocorrelation function has damped out[14].

Initial estimates for the parameters of the model identified, is also made at this stage. For example consider a MA(q) process with autocorrelation function (3.1.23). The autocorrelation function is zero onle for the first q lags. The expressions for $\rho_1, \rho_2, \ldots, \rho_q$ in terms of $\theta_1, \theta_2, \ldots, \theta_q$, and this supplies the q equations for q unknowns needed to find the coefficients of the MA process. The preliminary estimates for the coefficients can be made by substituting the estimated autocorrelations instead of the theoretical ones. Then a preliminary estimation of the white noise standard deviation can be made using the estimated MA coefficients and the estimated γ_0 as follows

$$\gamma_0 = \sigma_a^2(1 + \theta_1^2 + \cdots + \theta_q^2) \tag{3.1.40}$$

For an AR process the parameters are found by solving the Yule-Walker equations (3.1.19). The estimated values should be substituted for the theoretical ones. After applying the

differencing operator ∇ as needed for making the process stationary, an ARIMA process can be represented as an ARMA process as can be seen in (3.1.29). An ARMA process is indicated by the fact that both autocorrelations and partial autocorrelations functions tail off. Also from (3.1.26) it can be seen that theoretical autocorrelations of ARMA follow the difference equation satisfied by the autocorrelations of the pure AR process $\phi(B)w_t = a_t$. For example if the autocorrelation function of the dth difference appears to be falling off exponentially except for a change in ρ_1. A *(1,d,1)* order process described as $(1-\phi_1 B)w_t = (1-\theta_1 B)a_t$, with $w_t = \nabla^d z_t$ should be suspected. Then the parameters are found by substituting the estimated autocorrelation values in the autocorrelation expressions for this process. It should be noted that these estimated values should be used to make the scope of the search for more efficient estimation of parameters narrower. For a problem more than one model may seem to be possible at this stage, then the only possible solution is to folllow the Box-Jenkins algorithm for one model and see if this is sufficient. If it is not the oher model is entertained. Usually estimation stage can give a clue about which model is better suited to the data.

3.1.2 Model Estimation

After having identified the orders of the ARIMA process the parameters need to be estimated. Box and Jenkins modelling estimates the parameters based on likelihood and Bayesian methods.

Suppose there is a sample of N observations z whose known probabality distribution $p(z/\xi)$ depends on some unknown parameters ξ. The vector ξ denote a general set of parameters and particularly can be taken to refer to the $p+q+1$ parameters ϕ, θ, σ of the ARIMA nodel. With the outcome of an experiment in the form of observations z in hand, the problem is to find the set of parameters ξ that gave rise to the observations. The likelihood function which fixes z but lets the ξ be the variable is used for finding the values of the parameters. The relative value rather than the absolute value of the likelihood function is needed, so that $L(\xi|\mathbf{z})$ is usually regarded as containing an arbirtrary multiplicative constant. Box and Jenkins uses the log likelihood function $l(\xi|\mathbf{z}) = \ln L(\xi|\mathbf{z})$ instead of the likelihood function, because the former contains an arbitrary additive constant and it is more convenient to work with and additive arbitrary constant than a multiplicative one.

The importance of the likelihood function stems from the idea that all the information about the mathematical system giving rise to the observed data is contained in the likelihood function. The Bayesian meaning of the likelihood function is that it is the posterior distribution of parameters that arising from the data.

Graphical as well as analytical study of the likelihood function has to be employed in order to understand the situation better. Box and Jenkins found that for moderate to large amounts of data the likelihood function is unimodal and can be approximated by a quadratic function around the maximum value. In these cases the likelihood function is described by its maximum value and the second derivative at the maximum. The parameters are found to meet the conditions at the maximum value and the second derivative of the function at the maximum value gives an idea about the spread of the likelihood and can be used to calculate the standard errors of the estimates.

Suppose that there are $N=n+d$ observations z, of a time series generated by a (p,d,q) ARIMA process. A series w, of $n=N-d$ elements, is generated from z such that $w_t = \nabla^d z_t$. In this way the nonstationary ARIMA process is transformed into a stationary ARMA process in the form of (3.1.24). In this case the process can be expressed as

$$a_t = \tilde{w}_t - \phi_1 \tilde{w}_{t-1} - \phi_2 \tilde{w}_{t-2} - \cdots - \phi_p \tilde{w}_{t-p} + \theta_1 a_{t-1} + \theta_2 a_{t-2} + \cdots + \theta_q a_{t-q} \qquad (3.1.41)$$

where $\mu = E[w_t]$, and $\tilde{w}_t = w_t - \mu$. When $d>0$ it is clear that $\mu=0$, otherwise it is sufficient for most purposes to substitute the average of the differenced signal as the mean of the process, i.e. $\mu = \overline{w} = \sum_{t=1}^{n} w_t / n$.

In order to start working on the difference equation (3.1.42), the ideal p values of $\boldsymbol{w}$ and q values of $\boldsymbol{a}$ prior to the commencement of the observed series are needed. Then the values of $\boldsymbol{a}$ conditional on this set of initial values can be calculated. Let the terms $\boldsymbol{w}_*, \boldsymbol{a}_*$ denote the initial values input to the difference equation (3.1.42). Then the succesive values of the shock $a_t(\boldsymbol{\phi}, \boldsymbol{\theta} | \boldsymbol{w}_*, \boldsymbol{a}_*, \boldsymbol{w})$ can be calculated. If a's are normally distributed,

$$p(a_1,a_2,\ldots,a_n) \propto \sigma_a^{-n} exp\left\{-\left(\sum_{t=1}^{n} a_t^2/2\sigma_a^2\right)\right\} \tag{3.1.42}$$

Then the log likelihood function conditioned on the initial values is

$$l_*(\phi,\theta,\sigma_a) = -n\ln\sigma_a - S_*(\phi,\theta)/2\sigma_a^2 \tag{3.1.43}$$

where the conditional sum of squares function is defined as

$$S_*(\phi,\theta) = \sum_{t=1}^{n} a_t^2(\phi,\theta|w_*,a_*,w) \tag{3.1.44}$$

It can be seen that the data dependency of the log likelihood function is only throough the sum of squares function. This implies that, under the normal distribution assumption sum of squares can be studied to understand the log likelihood funcion. In fact for any fixed σ_a, l_* is a linear function of S_*.

The starting point for parameter estimation is thus finding the initial values of w's and a's. There are a few approaches that can be shosen for calculating suitable values that can be used as starting values. One procedure is to replace the elements of w_*, and a_* by their unconditional expectations. The unconditional expectations of the random shock process a_* is zero. If $\mu = 0$ and the model does not contain any deterministic elements, the unconditional expectation of w_* is also zero. It was found that this procedure can give rise to poor approximation when the roots of $\phi(B) = 0$ are close to the boundary of the unit circle so that the process is approaching nonstationarity [14]. Another, more reliable, approach is to set all initial a's to zero and calculating the values of a's starting with a_{p+1}. The actual values of w's will be used throughout this procedure.. Note that using the latter procedure only n-p=N-p-d values of a_t will be available for calculating the sum of squares. But for large enough data sets this loss of information is not very important.

The unconditional log likelihood function of the process is derived in [14] and is given below

$$l(\phi,\theta,\sigma_a) = f(\phi,\theta) - n\ln\sigma_a - s(\phi,\theta)/2\sigma_a^2 \tag{3.1.45}$$

where the unconditional sum of squares function is denoted by

$$S(\phi,\theta) = \sum_{t=-\infty}^{n} E\left[a_t \middle| \phi,\theta,\omega\right]^2 \tag{3.1.46}$$

$f(\phi,\theta)$ is a function of the AR and MA parameters and is not important when n is not small. Usually the unconditional sum of squares dominates the equation (3.1.44). Then minimizing the sum of squares in (3.1.45), referred to as the least squares estimates in [14], very close approximations to the maximum likelihood estimates can be made.

The time series generated by (3.1.41) can also be generated by

$$\phi(B)\tilde{w}_t = \theta(B)e_t \tag{3.1.47}$$

where $F=B^{-1}$ is the forward shift operator. The equation (3.1.46) now supplies the backward forecasts $E\left[\tilde{w}_{-j} \middle| \boldsymbol{f},\boldsymbol{q},\boldsymbol{w}\right]$. In practice the estimates beyond a point $t=-Q$ become esentially equal to zero. Thus a further approximation results in

$$\tilde{w}_t = \phi^{-1}(B)\theta(B)a_t = \sum_{j=0}^{\infty} \psi_j a_{t-j} \cong \sum_{j=0}^{Q} \psi_j a_{t-j} \tag{3.1.48}$$

This means that the simpler rules for solving the sum of squares function of a moving average process of order Q can be used to approximate the solution of a mixed ARMA process.

The general procedure is to make use of the equation (3.1.46) to generate the backward forecasts and then to use the eqaution (3.1.41) to generate the $E[a_t]$'s with all the signals replaced by their expectation. Graphical aids can be of very great help in both identification and estimation stages, but they are not included in the present discussion. Detailed explanation of their application can be found in [14].

3.1.3 Model Diagnostics Checking

Two important problems in modelling are overfitting and lack of fitting. Use of autocorrelation functions as well as simple inspection can be made to pinpoint these problems in the application. This stage is essentially similar to the testing step in neural network terminology. The purpose is to test the goodness of fit. The model is analyzed to see whether it is adequate or not for the purposes and also to see the deficiencies of the modelling. After this stage if the requirements are not met the process is repeated for an altered form of the original model or for a new model.

After having identified a correct model the previous stages, a more elaborate model is actually fitted to the data in this stage. This technique is called overfitting and its purpose is to find out if additional parameters are actually needed to fit the data more closely. It is important to notice that if the results show that the additions are not needed, it simply means that the additions are not needed. By using this technique only the deficiency of a model can be proved. It does not validate a model.

Another method employed for diagnostic checking is based on the analysis of the residuals. The residuals are the a_i parameters estimated by the process. For example consider a model

$$\begin{aligned} &\phi(B)\tilde{w}_t = \theta(B)a_t, \quad where \\ &w_t = \nabla^d z_t. \end{aligned} \tag{3.1.49}$$

Assume that such a process has been fitted with estimated weight parameters then the residuals for this case are in the following form

$$\hat{a}_t = \hat{\theta}^{-1}(B)\hat{\phi}(B)\tilde{w}_t \tag{3.1.50}$$

As the series length increases, the estimated $\hat{a}_t$ values approach the actual white noise a_i's. This is because if the model is adequate the estimated values approximate the actual values with a small error.

$$\hat{a}_t = a_t + 0\left(1/\sqrt{n}\right) \tag{3.1.51}$$

where n is the length of the time series. The study of these estimated shocks, $\hat{a}_t$, will indicate the existence and nature of inadequecy of the model. Particulary the estimated autocorrelations of $\hat{a}_t$ can be used to see deficiencies of the model. Caution should be exercized in putting aside a model based solely on the autocorrelation function as the parameters of the process themselves are also estimated so that errors in estimating these parameters also add up. Particularly deviation of the estimated autocorrelations $r_k(\hat{a})$ from their theoretical zero value by less than $n^{1/2}$ should not be taken as lack of fit unless k is a moderately high lag. A similar approach is called a partmanteau lack of fit test and it considers the first K autocorrelations, $r_k(\hat{a})$, and checks the distribution of $Q = n\sum_{k=1}^{K} r_k^2(\hat{a})$. K should be large enough so that the linear filter weights ψ_j, are effectively zero for $j \geq 0$. If the fitted model is appropriate certain distribution conditions should be met, otherwise the average values of Q will be inflated.

Sometimes the parameters of a process change over a prolonged time period and the model becomes inadequate in modelling even though the form and the degree of the model is appropriate.

There are other tests that can be performed to check the ability of the model to fit the system. These are very complicated and detailed explanation is needed in order to make

them clear. Therefore they are not treated here, [14] contains detailed analysis of every stage of Box-Jenkins model.

3.2 Chaotic Time Series Prediction

Box and Jenkins method is useful in building stochastic models from time series. This procedure is explained in general terms in the preceding section. This section deals with the building of deterministic models. In this situation the deterministic component is desired to be extracted from the noisy observation data. Then a dynamical system model that contains the minimum number of variables and satisfies the constraints in term of the observed data should be built.

In this section, prediction methods for a special class of processes, the chaotic processes, will be discussed. Chaotic processes, or time series, had been treated as pathological cases for a long time before they actually gained any credibility and applicability in science. Some of the processes dismissed as random were found to be deterministic chaotic processes. The most exciting novelty introduced by the chaotic dynamics is that it made it possible to define and predict the evolution of these processes.

Chaotic processes are characterized by a number of measures such as a positive lyapunov exponent, fractal dimension and Kolmogorov entropy. Several relations exist among these mesaures and usually only the lyapunov spectrum is used for characterizing a chaotic process or at least for identifying one. A positive Lyapunov exponent implies that points that are arbitrarily close, so close that it is not possible to resolve them, in the beginning evolve into completely different outcomes after a finite number of iterations. In other words trajectories diverge, on average, at an exponential rate characterized by the largest lyapunov exponent. One definition of lyapunov exponent is due to [15]

$$\lambda_i = \lim_{t \to \infty} \frac{1}{t} \log_2 \frac{p_i(t)}{p_i(0)} \qquad (3.2.1)$$

The equation (3.2.1) was derived by considering the long term evolution of an infinitesimal n dimensional hypersphere of initial conditions. Note that the dimension of the hyperspace is the same as the dimension of the phase space. The sphere will be deformed into a n dimensinal hyperellipsoid because of the locally deforming nature of the flow. The term p_i(t) denotes the principal axis of this ellipsoid. Notice that if an axes is on the average expanding then the corresponding lyapunov exponent is positive and if the axes is on the average contracting the lyapunov exponent is negative. Dissipative systems have at east one negative exponent and the sum of all exponents is also negative. This limits the motion of trajectories of the process to occur in a limit hypervolume, the attractor. The exponential expansion seems to be incompatible with the motion on a bounded attractor. But a folding proess merges widely seperated trajectories and thus limits the motion to the attractor. Therefore it is claimed that each positive exponent computed by (3.2.1) indicates the direction in hich simultaneous expansion and folding takes place. This simultaneous operations decorrelates nearby points of the process. The equation (3.2.1) is not the only way of calculating the Lyapunov spectrum, there are a number of algorithms proposed to calculate this rate of exponential divergence in literature [16-18].

A chaotic process can also be described by its dimension which is fractal. The dimension of a process is the dimension of the attractor in the phase space of the dynamical system generating the process. There are a large number of algorithms for calculating te dimension of an experimental time series in literature one treatment can be found in[19]. A fractal dimension is defined as the Hausdorff dimension that is stricly greater than the topological dimension, which is classical integer dimension. It turns out that most of the time the fractal dimension is a non integer value. This mesaure will be used in section 4.1 to describe the spatiotemporal clustering of earthquakes.

These measures are employed to understand the underlying system better and to also help in model building and preduction schemes. For example it is claimed that the lyapuov exponents give an idea how far into future succesful forecasts can be made and that the fractal dimension gives a clue about the number of variables needed to make a prediction.

There are two main approaches to the prediction problem of deterministic time series. The first one is to compute the actual generating function, that is compute a global predicton function that approximates the actual generating function. Ideally such a solution is desirable not only because it will allow the user to predict he future behavior of the system but it will also help in understanding the dynamics of the process. Choosing this approach requires an immense amount of data and tedious work for practically occuring processes. There are cases when only some help in making predictions is needed. In cases like this it is

more sensible to calculate local functions for prediction task. These local functions are usually valid only over a specified region of the attractor. In the sequel applicaton of both local and global methods for predicting chaotic time series will be discussed. A geometric interpretation of the local prediction methods used in this section is that local portions of the curve $z(t)$ in the past that resemble the present situation are found and the prediction is done using the values the process had taken immediately after those events. This can be fitting a function to a local portion of the observed data that is similar to the present situation. For gloabal prediction methods, on the other hand, a function should be fitted to the whole set of data.

The local method is also divided into two main approaches. These are linear and nonlinear approximation approaches. Both methods have their own advantage and disadvantages as will become more clear in the subsequent discussion.

3.2.1 Linear Prediction

In this subsection a method, proposed by Paul Linsay, for forecasting chaotic time series using linear interpolation will be discussed [20]. The method introduced in this section is a local method that uses $m+1$ equations when the attractor dimension is m.

A common method for prediction calculates the Jacobian of the strange attractor in the vicinity of the target point and then uses the Jacobian to calculate the future evolution of the target point. It is not an easy task to calculate the Jacobian and most of the time nearest neighbor algorithms are employed to compute a prediction function. Unfortunately nearest neighbor algorithms need two to five times the minimum number of data needed to compute the Jacobian, in the form of nearest neighbors, to compute a function whose prediction performance is comparable to that of a Jacobian function. The same procedure must be repeated for every target point.

Consider a time series $\{z\}$, and an unknown map $f(z)$. Suppose that a repeated application of $z_t=f(z_{t-1})$ produces a chotic time series. The problem is to predict the value z_{t+1} of the first iterate of z_t. he first step to the solution of this problem is to find the k closest points $\{z_t(1),z_t(2),\ldots,z_t(k)\}$ to the value z_t on the attractor. The term $z_t(i)$ is used to denote the ith closest neighbor to the value z_t. Also find the first iterate of each of the elements of the k closest neighbor set $\{z_{t+1}(1),z_{t+1}(2),\ldots,z_{t+1}(k)\}$. Then compute a set of weights λ that satisfy the following equation

$$z_t = \sum_{i=1}^{k} w_i z_t(i) \tag{3.2.2}$$

The weight should be chosen to satisfy the normalization condition

$$\sum_{i=1}^{k} w_i = 1 \tag{3.2.3}$$

Let the Jacobian matrix in the vicinity of z_t be denoted by J and let b be a constant vector. Then if the function $f(.)$ is nearly linear around z_t, the next iterate of z_t is described as

$$z_{t+1} = f(z_t) \cong Jz_t + const \tag{3.2.4}$$

Applying the function $f(.)$ to both sides of (3.2.2) under the constraints (3.2.3) and (3.2.4) the next iterate z_{t+1}, is found in terms of the next iterates of the k closest neighbors.

$$z_{t+1} \cong \sum_{i=1}^{k} w_i z_t(i) \tag{3.2.5}$$

Then the prediction is simply made by calculating the coefficients satisfying the equation (3.2.2) and applying them to the first iterates of the closest neighbors.

The value of k is determined by the embedding dimension, m, of the process. In equation (3.2.2) there are m equations for the k unknowns and the normalization condition (3.2.3) contributes one more equation to the set. Thus $k=m+1$. If more neighbors than $m+1$ are used for prediction ad. hoc. conditions must be added to determine the weights λ_i. Note that the fractal dimension of the attractor of the process indicates the minimum

embedding dimension that should be used. Particulary the embeding dimension m should be greater than the fractal dimension.

If the Jacobian matrix matrix method was to be used as in (3.2.4), then $d(d+1)$ unknowns with an equal number of equations would have to be solved both for $\boldsymbol{J}$ and $\boldsymbol{b}$. This prediction can be made more accurate if the number of neighbors used is increased, the usual amount is two to five times $m+1$, and the best approximation of $\boldsymbol{J}$ and $\boldsymbol{b}$ are found through the least squares application. This, of course, requires more computing power and time.

An improvement in speed of the proposed method can be achieved by using delay embedding of a single variable. This means that the d dimensional point is denoted by $\{z_{t-d-1}, z_{t-d}, \ldots, z_{t-1}, z_t\}$. Thus the first $d-1$ coordinates of coordinates of z_{t+1} are equal to the last $d-1$ coordinates of z_t, therefore the next iteration of only one coordinate should be calculated.

3.2.2 Nonlinear Prediction

Martin Casdagli employs an inverse problem approach for nonlinear prediction of time series in [21]. The inverse problem is defined as the construction of a functional model, or a mapping, based on the data given. This map can then be used to make predictions into the future. The construction is achieved by interpolating or finding approximate functions that covering the observed data.

Casdagli has found that this approach can also be used in order to differentiate low dimensional chaos from randomness. Using this method not only predictions into the future can be made bu invariant mesaures such as the lyapunov spectrum of the process can also be found.

Let $f: \Re^m \rightarrow \Re^m$ denote a smooth map of iterates $\mathbf{z}_n = f^n(\mathbf{z}_0)$, where $1 \leq n \leq \infty$ lying on a strange attractor α. Generally the function $f(z_t)$ is not known and a smooth map $\hat{f}_\infty: \Re^m \rightarrow \Re^m$ is to be constructed based on the iterates z_n so that $z_{n+1} = \hat{f}_\infty(z_n)$, for $1 \leq n \leq \infty$ is satisfied. It can be seen that such an inverse solution has a unique solution and $\hat{f}_\infty\big|_\alpha = f\big|_\alpha$. There is no requirement on the behavior of the estimated function $\hat{f}_\infty$ outside the attractor α. It is not however practical to employ the above model

which approximates using an infinite number of iterates. A more realistic approach is to use only a finite number of data points and interpolate among these points in an attemp to approximate the generation function $f(z_n)$. Then the inverse problem is

$$z_{n+1} = \hat{f}_N(z_n), \quad \text{for } 1 \le n \le N \tag{3.2.6}$$

It is easy to see that the solution of the inverse problem operating on only a finite number, N, of data points is not unique. The predictor error of the estimated function

$$\varepsilon(\hat{f}_N) = \lim_{m\to\infty} \frac{1}{M} \sum_{n=N}^{N+M-1} \frac{\left\| z_{n+1} - \hat{f}_N(z_n) \right\|}{\lim_{M\to\infty} \left(M^{-1} \sum_{m=1}^{M} \left\| z_m - \lim_{M-\infty} M^{-1} \sum_{m=1}^{M} z_m \right\| \right.} \tag{3.2.7}$$

The denominator in (3.2.7) is a normalizing factor. The estimated process is said to be of order β, if $\varepsilon(\hat{f}_N) = O(N^{-\beta/D})$ where D is the information dimension of the process.

Consider a series of scalar variable $z(t)$ sampled at discrete intervals of time $t=n\tau$. If the underlying dynamics is that of a strange attractor lying on an I-dimensional manifold then the sequence $z(n\tau)$ is classified as a chaotic time series. In such a case Taken's embedding theorem states that for generic τ and embedding dimension $m \le 2I+1$, there exists a smooth map $f: \Re^m \to \Re$ that satisfies (3.2.8) for $1 \le n \le \infty$.

$$f\left(z\left((n+m-1)\tau\right),\ldots,z(n\tau)\right) = z((n+m)\tau) \tag{3.2.8}$$

The minimum value of the embedding dimension satisfying (3.2.8) will be denoted by m^* and called the minimal embedding dimension of the process. The inverse problem related to this situation is to compute a smooth function $\hat{f}_N$ according to (3.2.8), using N data points

$z(n\tau)$. Before a method is actually applied to compute the estimated function there are some practical problems that need to be tackled. The first problem is to find a practical performance measure. Given a time series of length $N+M$, the equation (3.2.7) is approximated by

$$\varepsilon^2(\hat{f}_N) = \frac{1}{M}\sum_{n+N}^{N+M-1}\left(z((n+1)\tau) - \hat{f}_N(z(n\tau),\ldots,z((n-m+1)\tau))\right)^2 / \sigma \quad (3.2.9)$$

where σ is the variance of the time series. The next step is to calculate m^*. A simple procedure is to start with $m=1$ and compute $\varepsilon(\hat{f}_N)$, and to repeat until increasing the embedding dimension does not improve $\varepsilon(\hat{f}_N)$ or an acceptably small value is reached. Having determined the minimal embedding function and a performance mesaure, an interpolation technique should be chosen so that the smallest value for $\varepsilon(\hat{f}_N)$ should be achieved with the smallest number of computational means. The final problem, that will be ignored in this thesis is minimizing the effects of noise pollution of the time series. refernces for this subject can be found in [21].

In the sequel a few approximation techniques will be discussed briefly. The previous requirement that $\hat{f}_N$ should be smooth is relaxed. Also a new operator Ξ_i is introduced to denote the projection onto the ith coordinate.

Global techniques choose the coordinate functions $\Xi_i\hat{f}_N : \Re^m \to \Re$, $1 \le i \le m$ from a standard function basis. In particular polynomial predictors of a given degree can be chosen to model the process. Then the free parameters of model are found by using a least squares algorithm for minimizing (3.2.10).

$$\sum_{n=1}^{N-1}\left(\Xi_i z_{n+1} - \Xi_i\hat{f}_N(z_n)\right)^2 \quad (3.2.10)$$

Choosing such an approach is advantageous because the resulting model, which is in a standard form, will be easy to analyze conceptually. Also Weirstrass approximation

guarantees that the estimated function $\hat{f}_N$, will converge to f as the number of data points, N, and the degree of the polynomial, d_p, are increased. On the other hand even though methods for decreasing this computational requirements are available this approach is computationally very expensive.

There is a variant to the method discussed in the previous paragraph that is popular for model building. This method is called the rational predictor and as the na,e suggests it fits a ratio of polynomials to the given data. The degrees of the polynomials need not be equal. In this case the function to be minimized for finding the optimal estimates for the free parameters is

$$\sum_{n=1}^{N-1} \left(\Xi_i z_{n+1} g_2(z_n) - g_1(z_n) \right)^2 \qquad (3.2.11)$$

where the ratio of polynomials to be fitted is g_1/g_2.

A viable alternative for global approach is the local one. This local approach, in its crudest form, requires much less computational resources than the global one. And significant improvements can be achieved by organizing the data. To construct a local predictor, first local areas of the data space are fitted by local functions and then all these functions are pieced together. For example if a value of $\Xi_i \hat{f}_N$ at a point z is required the k nearest neighbors of z are found and a polynomial of degree d is fitted through the corresponding points. Note that generally, the the degree d for local approximation is smaller than that of the global approximation.

The main disadvantage of the local prediction method is that the resulting overall prediction function is usually discontinous and thus difficult to analyze. This discontinuity is also dangerous when computing long term iterates because a local function is valid only at a portion of the phase space and should be used only where it is valid.

3.3 Neural Networks For Time Series Prediction

In this section a few of the neural network architectures that have been used for time series prediction in literature are explained briefly. Neural networks are well known for their interpolation capabilities. It has been shown that a multilayer percepron network, MLP, with two hidden layers is capable of modelling arbitrary nonlinear systems without memory [3,4,22]. Neural networks used for pattern recognition and classification tasks has performed very well and proved useful in practical applications.

Neural networks have also been used for modelling dynamical systems or predicting the continuation of a time series sequence with success as long as the process modelled or predicted is smooth enough or is not very complicated. All of the following networks were tested using experimental time series such as hennon mapping or Mackey-Glass chaotic time series. Very few application of real time data such as earthquake sequence or sunspot series have been used for testing the performance of these network.

3.3.1. Using CNLS-Net to Predict the Mackey-Glass Chaotic Time Series

The Connectionist Normalized Local Spline Network (CNLS-net) combines normalized radial basis functions, a linear gradient term and a simple rapid solution of the training algorithm proposed by Mead et. al. [22]. The motivation was to modify the radial basis function (RBF) nets so that better interpolation capabilities would be embodied with reduced amount of training requirement for accurate learning.

The CNLS-net has a single hidden layer. Asume that $\chi_j(z)$ is a localized function of z about some z_j as in RBF networks. Also consider the following identity.

$$g(z) = \frac{\sum_{j=1}^{N} g(z)\chi_j(z)}{\sum_j \chi_j(z)} \qquad (3.3.1)$$

where $g(z)$ is an arbitrary function. It can be approximated by its Taylor expansion about z_j. Then the approximation of $g(z)$ is:

$$\varpi(z) = \sum_{j=1}^{N} \left[f_j + (z - z_j) \cdot r_j \right] \frac{\chi_j(z)}{\Sigma_j \chi_j(z)} \tag{3.3.2}$$

The basic difference between CNLS-net and the RBF nets is the inclusion of the linear term $(z\text{-}z_j).r_j$ and the radial basis function normalization. These additions intend to reduce the amount of training data needed for reasonable approximations. Since the training of f_i and d_j is linear, it is fast.

The test case used for this architecture was Mackey-Glass (M-G) equation:

$$\frac{dx}{dt} = \frac{az(t-\tau)}{\left[1 + z^c(t-\tau)\right]} - bz(t) \tag{3.3.3}$$

where, $a=0.2$, $b=0.1$, $c=10$, and $\tau=30$. At this value of the delay parameter τ, the M-G attractor has an information dimension of 3.6.

The performance indicator used was the root mean squared prediction error (RMSE) divided by the standard deviation. The M-G equation was treated in [22] to limit its range approximately in the interval (0,1) and in this form it had a standard deviation of 0.24. A constant fit through the mean value of the function leads to a value of 1.0.

The training and test patterns were composed of 6 inputs and an output unit. The input units were spaced at time intervals of 6 time units each and the output unit was 6 time units after the last entry in the corresponding input set. The embedding dimension, which is a very sensitive matter, was found by trial; and error in [22].

Mead et. al. used nonoverlapping training and test files which consisted of 1000-5000 points at fixed time spacing. Sequential sets of 500 patterns were chosen at random

the corresponding test set which always consisted of 500 patterns were the sets following the training sets. The selected training patterns were held fixed for the entire training period but were usually presented in random, varying sequences for successive training epochs.

The CNLS-net architecture chosen initially in [22], had 6 input nodes, 1 bias, 28 hidden nodes and one output node. This yields about 200 weights, which were trained in to mimic the M-G equation.

The CNLS-net has two continuously adjustable parameters. Mead et. al. claim that the learning rate has a broad minimum and learning showed some regions of instability. It was found in [22], that the range of acceptable performance was broad and that even though higher learning rates gave rise to faster training, the system was then more susceptable to instability. The basis functions, too, had a broad optimum width, which depended on the characteristic structure of the function to be fit under the chosen embedding. The embedding structure is determined by the number of inputs and the time delay between succesive nodes. For this experiment Mead et. al. found it to be 30-45. It was suspected to have a relation with the delay parameter τ of the M-G equation.

Mead et. al. found that the trainability and prediction accuracy of the net were influenced by the random initial choice of basis function centers. Some of the effects of this were cancelled by the marginal stability of the training algorithm.

3.3.2 Clusnet Architecture for Prediction

The input to the ClusNet, proposed in [23], consists of sequentially delayed values of the signal whose future value is to be predicted. The mapping is designated as:

$$z(t+T) = \Gamma\big(z(t), z(t-\Delta), z(t-2\Delta), \ldots, z(t-(m-1)\Delta)\big) \qquad (3.3.4)$$

where bold faced letters denote vectors. Instead of learning global dynamics of the system, this approach employs an instance based method. When a new vector $z(t)$ is presented as a basis for a prediction of $z(t+T)$, similar vectors in storage are located and a linear

interpolation is done to the future of these values. Unfortunately this class of algorithms require a large amount of storage and processing time.

The clustering neural network Clusnet is a network that classifies a given set of n input vectors into N clusters. The learning phase determines the centroids of the clusters by minimizing the total Euclidean distances of the vectors from their respective centroids. For prediction the cluster that the input vector belongs to has to be determined.

The Clusnet architecture consists of two layers, the input layer is a static one with a fixed number of nodes. The only function of the input layer is to pass the data to the next layer. There are as many input nodes as the embedding dimension or the size of the input vector.

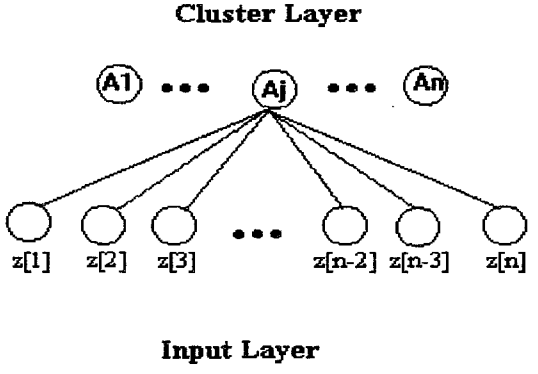

Figure 3.1 The Clusnet Architecture

The cluster layer should accomodate an adequate number of nodes, corresponding to the necessary number of clusters for accurate classification, dictated by the system that is being analyzed. This layer is a dynamical layer and the number of nodes is modified during the learning process until an acceptable point on the error surface is reached. The two layers are fully connected but there is no connection between nodes in the same layer. The weights between nodes in the input layer and a cluster node corresponding to centroid c is denoted by the vector w^c. The values of the weights are determined by the learning algorithm.

Consider z^j, an input vector fed into the network. The activation of the cluster c is defined as:

$$A_c = \left(z^j - w^c\right)^2 \tag{3.3.5}$$

where the square operation is the scalar product of the vector with itself. The cluster nodes compete with each other and the one with the smallest activation wins so that the output of each of the cluster nodes after the input is presented and the calculations are done as defined as follows:

$$\begin{aligned} O_i &= 1, \text{ if } i \text{ wins} \\ &= 0, \text{ otherwise} \end{aligned} \tag{3.3.6}$$

Such a competition will partition the input space so that similar points in the input space will be responded to by the same cluster node. The weight vector of the whole network is denoted by W. It is modified, or learned, dynamically as the learning continues. The number of clusters need not be known a priori but cluster nodes are assigned as need arises according to the criteria given to the network.

The Clusnet learning algorithm occurs in two stages. The first one is the initial cluster center allocation stage. During this stage the network is presented with a set of inputs. It partitions these into m clusters denoted by m cluster nodes. An assignment is correct if an input vector is assigned to the cluster center that is closest to it.

Naturally every input vector should not normally cause a new addition to the cluster nodes in the second layer. As new vectors are added to a cluster, the corresponding weights should change to minimize a specified error term. This stage requires one pass of the input vectors. However when this stage is completed some of the input vectors may no longer belong to the same cluster they were initially assigned to, as the weight vectors have changed. That is some assignments are not correct, therefore the network is not in an equilibrium state.

The second stage is the equilibrium stage. This stage starts with the set of weights, cluster centers, and input classifications determined by the first stage. It may reassign vectors from one cluster to another and change the corresponding weights appropriately. This decreases the number of misclassified vectors. The procedure continues until no vector

is assigned to a cluster incorrectly. When all vectors are assigned correctly the network is in equilibrium stage.

The total error of the network after stage two converges is defined as:

$$E = \sum_{k=1}^{N} \sum_{j=1}^{n_k} \left(w^k - z^j \right)^2 \tag{3.3.7}$$

where n_k is the number of input vectors in cluster k, and N is the total number of clusters. Then let

$$\frac{\partial E}{\partial w_i^l} = \sum_{j=1}^{n_k} \left(w^l - z^j \right) = 0 \tag{3.3.8}$$

The solution of the above equations is:

$$w^l = \frac{1}{n_l} \sum_{j=1}^{n_l} z^j \tag{3.3.9}$$

The weights can be determined if the assignment of the input vectors to the clusters are known.

Stage 1. Initial Cluster allocation stage

Create a class node C_1 with a weight vector $\mathbf{w}^1 = z^1$. When k clusters have already been created and an input vector z^j is presented to the network, the activation of all the nodes of the network are denoted by $A_1,...,A_k$ are computed. The node with the smallest activation value is found. Assume node m is the one that is closest to the input vector. If

A_m is less than a user defined constant ε this input vector is added to the cluster m and the corresponding weight vector is adjusted to minimize the following error term:

$$w^m = \frac{1}{n_m}\left\{(n_m - 1)w^m + z^j\right\} \tag{3.3.10}$$

In the above equation n_m is the number of vectors assigned to cluster m including the new input z^j.

If the input vector is not close enough to the closest cluster center m, that is the activation function A_m is not less than the constant ε, then a new cluster center, c_{k+1}, has to be created and its weight vector is set equal to the input vector:

$$w^{k+1} = z^j \tag{3.3.11}$$

All input vectors are passed through the network with the same procedure. The parameter ε will dictate the number of clusters that will be formed. A large number of ε will result in fewer clusters whereas a low value of ε will cause formation of more clusters.

Stage 2. The equilibrium Stage

After all the input vectors are presented and assigned to one of the cluster centers, in the equilibrium stage, the weight vectors are updated until the network is in the equilibrium.

Let $A_1,...,A_N$ be the activation of the cluster nodes when z^j is presented and let the weight vectors be denoted be $w_1,...,w_N$. If a vector z^j is found to be closer to a cluster c_{new} than the cluster c that it is initially assigned to, the activations functions are related as below:

$$A_c > A_{c_{new}} \tag{3.3.12}$$

where

$$A_c = \left(w^c - z^j\right)^2$$
$$A_{c_{new}} = \left(w^{c_{new}} - z^j\right)^2 \qquad (3.3.13)$$

To remedy this situation vector z^j is assigned to the cluster c_{new} by updating their corresponding weights using the equation below:

$$w'^c = \frac{1}{n_c - 1}\left\{n_c w^c - z^j\right\} \qquad (3.3.14)$$

Assume that the embedding scheme is given, then the learning phase consists of the presenting the input-output pairs (z^i, y^i), $1 \le i \le n_k$ to the ClusNet. The prediction task is then to return the correct value of y^j given the state vector z^j. Typically the input vector has several components:

$$z^i = \left[z_1^i, z_2^i, \ldots, z_S^i\right] \qquad (3.3.15)$$

where each component may be a delayed sample of the time series or any other independent measure.

For simplicity assume that there exists a scalar function $\Gamma: \Re^S \to \Re^1$. Taylor series expansion of the input vector z that belongs to the cluster c with center z_C, around the cluster center upto the linear term is:

$$y = \Gamma(z_C) + (z - z_C) \vee \Gamma(z_C) \qquad (3.3.16)$$

where the operator, ∇, denotes the gradient. The average value of y is

$$\bar{y} = \Gamma(\bar{z}) \tag{3.3.17}$$

Since the function Γ is unknown, otherwise there would be no need to construct an adaptive network for prediction, its gradient is also unknown. Write the expansion as

$$\Omega = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_s \end{pmatrix} \tag{3.3.18}$$

where s is the size of the input layer of the Clusnet. The w_is can be determined for each cluster using linear algebra techniques.

3.3.3 NADINE-A Feedforward NN for Arbitrary NonLinear Time Series

Madaline Networks applied to the modelling of time series realize a constrained Volterra series because of the fixed nature of the nonlinearity [24,25]. Nadine, proposed by Ahmed et. al. in [24], can model arbitrary Volterra series and therefore arbitrary nonlinearities with memory.

Nadine can be realized by layers of Adaptive Linear Combiners (ALC), where the outputs of one layer are used as the weights rather than the activations of the next layer. The ALC implements a constraint Volterra series in which the kernels do not admit arbitrary shapes

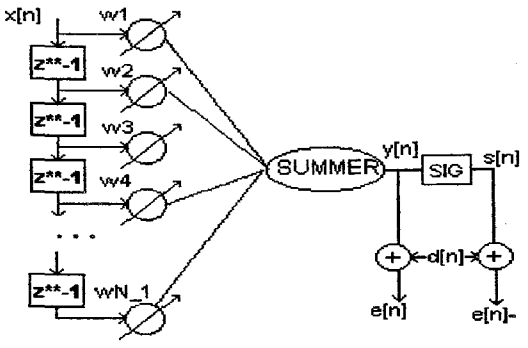

Figure 3.2 Adaptive Linear Combiner

The network can be trained using a backpropagation style learning without actually propagating adaptation information between the layers. The ALCs are, thus, locally adapted.

Nadine is very modular and easily extendible. Ahmed et. al. claim that the high order neural networks and polynomial discriminant based methods are special cases of Nadine which can, now, be implemented modularly without requiring preprocessing [24].

When used for time series analysis the input to a neural network is usually a time lagged sequence of the present observation. The architecture of NADINE realizes a nonlinear transformation that is amenable to a Volterra series representation.

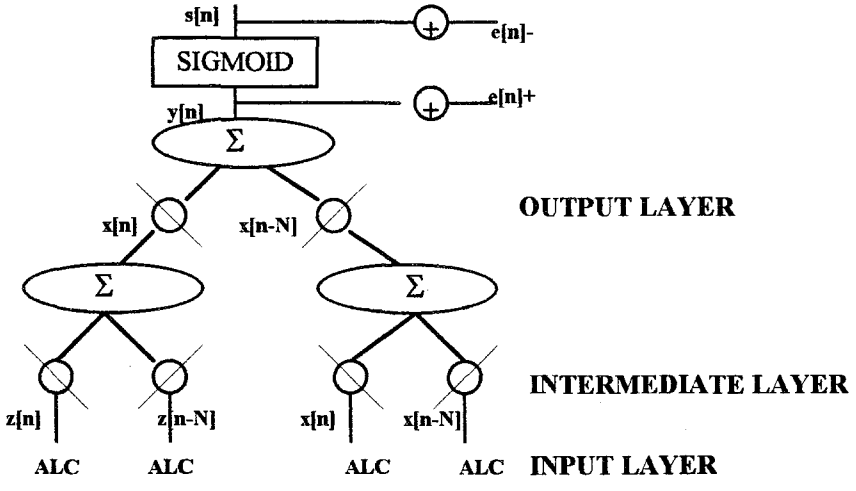

Figure 3.3 NADINE- Feedforward Neural Network

The architecture of NADINE, as can be seen from the figure, implements a constrained Volterra sysytem in which the Volterra kernels do not admit arbitrary shapes.

NADINE can be realized by layering ALCs. The output of a layer is used to drive the weight rather than the activity of the next layer. Each layer obtains the same sensory inputs but one layer provides a dynamic function, or weight, to the next. This structural difference allows NADINE to model nonlinearities with memory that Madalines cannot.

Unlike most multilayer networks, each ALC within the net can be adapted locally, instead of using an overall error measure. It will be shown in the following discussion that local adaptation minimizes the overall error measure.

Consider the figure given above, the output of the linear combiner is:

$$y(n) = \sum_{i=0}^{N-1} w_i z(n-i) \tag{3.3.19}$$

Assume that the fixed nonlinearity *f(t)* is differentiable and thus admits a Taylor series expansion of the form:

$$f(t) = \sum_{j=0}^{\infty} \kappa_j t^j \tag{3.3.20}$$

Substituting *y(n)* for *t* in the above equation, the output, *s(n)*, of the network is found.

$$\begin{aligned} s(n) &= \sum_{j=0}^{\infty} \kappa_j \left[\sum_{i=0}^{N-1} w_i z(n-i) \right]^j \\ &= \kappa_0 + \sum_{i=0}^{N-1} \kappa_1 w_i z(n-i) + \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} b_{i,j} z(n-i) z(n-j) + \ldots \end{aligned} \tag{3.3.21}$$

This is a Volterra series with only N degrees of freedom provided by N weights w_i. The choice of the fixed nonlinearity fixes κ_j, only w_j can be adjusted to learn the function. In an arbitrary Volterra system all the weights are variable. An arbitrary nonlinear system with finite memory is :

$$\begin{aligned} s[n] = w_0 + \sum_{j=0}^{N-1} w_i z(n-i + \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} w_{i,j} z(n-i)z(n-j) \\ + \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\sum_{k=0}^{N-1} w_{i,j,k} z(n-i)z(n-j)z(n-k)+\cdots \end{aligned} \tag{3.3.22}$$

The drawback induced by having only N degrees of freedom is that it can only handle linearly seperable constraints. Madalines overcome this limitation by piecewise linearization, by implementing the nonlinear constraint through multiple linear constraints. The approach chosen for NADINE is to implement the nonlinear constraints by working on succesive tangent spaces of the nonlinear constraints.

The architecture for only the first four terms of the Volterra series is derived here for the sake of computational ease, generalization to higher orders is trivial. Consider a truncated version of the arbitrary Volterra system presented above with only the first four terms. Such an equation is easily factorized to the following form:

$$y(n) = w_0 + \sum_{i=0}^{N-1}\left[w_i + \sum_{j=0}^{N-1}\left[w_{i,j} + \sum_{k=0}^{N-1} w_{i,j,k} z(n-k) \right] z(n-j) \right] z(n-i) \tag{3.3.23}$$

Thus the content of each of the patranthesis is linear and can be implemented using ALCs. The output of each ALC becomes the weight in the next layer. Rewrite the above equation as:

$$y(n) = w_0 + \sum_{i=0}^{N-1} \omega_i(Z) z(n-i) \tag{3.3.24}$$

where Z is the set of N past observations of the time series and $\omega_i(Z)$ is the output of the ALC and is denoted by:

$$\omega_i(Z) = w_i + \sum_{j=0}^{N-1} \delta_{i,j}(Z) z(n-j) \tag{3.3.25}$$

A can be seen the weights of both $y(n)$ and $\omega(Z)$ are state dependent. $\delta(n)$ are the outputs of the input layer of ALCs whose weights are state independent.

$$\delta_{i,j}(Z) = w_{i,j} + \sum_{k=0}^{N-1} w_{i,j,k} z(n-k) \tag{3.3.26}$$

The state dependent weights change because of two reasons, the lower level ALCs are adapted and because of adaptation, whereas the state independent weights change only because of adaptation.

The weight update equation of the LMS learning rule is applicable to all ALCs in all layers. Each ALC in the network uses a localized adaptation scheme to learn the mapping. The output layer consisting of a single ALC minimizes the error between the desired result and the network output. The rest of the structure minimizes the change in the respective weight in the next upper level.

The update for the topmost layer, second layer and input layer are as follows, in the same order:

$$
\begin{aligned}
w_i(n+1) &= w_i(n) + \eta_i z_{n-i}(n) e(n) \\
w_{i,j}(n+1) &= w_{i,j}(n) + \eta_j z_{n-j}(n) e_i(n) \\
w_{i,j,k}(n+1) &= w_{i,j,k}(n) + \eta_k z_{n-k}(n) e_{i,j}(n)
\end{aligned}
\tag{3.3.27}
$$

where *e(n)* is the network error and the errors for intermediate layer and the input layer are defined as:

$$
\begin{aligned}
e_i(n) &= w_i(n+1) - w_i(n) = \eta_i z_{n-i}(n) e(n) \\
e_{i,j}(n) &= w_{i,j}(n+1) - w_{i,j}(n) = \eta_j z_{n-j}(n) e_i(n)
\end{aligned}
\tag{3.3.28}
$$

Local adaptation of ALCs in NADINE is a result of the separability of the layers. LMS based adaptation rules tend to be dependent on the eigenvalue spread of the autocorrelation matrix. The larger the eigenvalue spread the slower will be the learning. In the case of nonlinear constraints the eigenvalue spread increases with the order of the nonlinearity. Nadine overcomes this by using local errors.

Nadine can also be considered as a volterra based nonlinear adaptive filter[26]. This kind of filters are actually linear filters with the input sequence to the filter consisting of a volterra series expansion of the original signal. This introduces the nonlinearity to the filter. The system is defined as follows:

H_0=zero order term
$H_1[z_n]$=first order term=$\Sigma_i h_i z_i$
$H_2[z_n]$=second order term=$\Sigma_i \Sigma_j h_{ij} z_i z_j$
$H_3[z_n]$=third order term=$\Sigma_i \Sigma_j \Sigma_k h_{ijk} z_i z_j z_k$.

The order can be increased according to the requirements of the process. The nonlinear filter then consists of two parts:

1. A nonlinear volterra series expander that actually performs the multiplication of the input signal to account for the $z_i z_j$ and $z_i z_j z_k$.
2. A linear filter that operates on the inputs and adapts the weights of the system.

3.3.4 Chaotic Neural Network

Experiments as well as numerical modelling by Hodgkin-Huxley equations have shown that a single biological neuron is capable of exhibiting a wide range of behavior including chaotic behavior[27]. This is one of the main deficiencies of the arificial neural units. Not only are they incapable of modelling the biological neurons but they are also incapable of exhibiting more than one type of behavior. For example an artificial neural unit with sigmoidal activation function may compute its output by evaluating the following particular formula,

$$hid = \frac{1}{(1+e^{-\gamma net})} \tag{3.3.29}$$

where *hid* is the output and *net* is the input of the neural unit. Therefore the possible range of behavior of such an artificial neural unit is predictable and too rigid to model a biological neuron. This deficiency of a single artificial neural unit to imitate a biological neuron's wide capabilities have trigerred the proposal of a number of neural units that can actually exhibit chaotic behaviour as well as the usual non-decreasing simple behavior of the widely used sigmoidal neural units.

In this section the chaotic neural unit proposed by Dingle et. al. [28] will be introduced. Dingle et. al. introduced the chaotic neural unit into the self organizing map and found promising results. In this project the same neural unit is used in a multilayer network. The application will be explained in more detailed with the results in section 5.2.

One of the simplest chaotic systems is the Feigenbaum logistic equation

$$y(t+1) = 4gy(t)[1-y(n)] \tag{3.3.30}$$

where g is the bifurcation parameter and t can be time or a discrete step. The long time behavior of (3.3.31) depends on g. The system output dies away to zero if $g<0.25$. If $0.25<g<0.75$, the system converges to a single non zero value. For $0.75<g<0.89$ the sytem output oscillates between an increasing number of values as g increases. for $g>0.89$ the stem is defined as a chaotic system, that is small differences in initial conditions result in huge differences in outputs within a finite time. This behavior can be seen in Fig. 3.4.

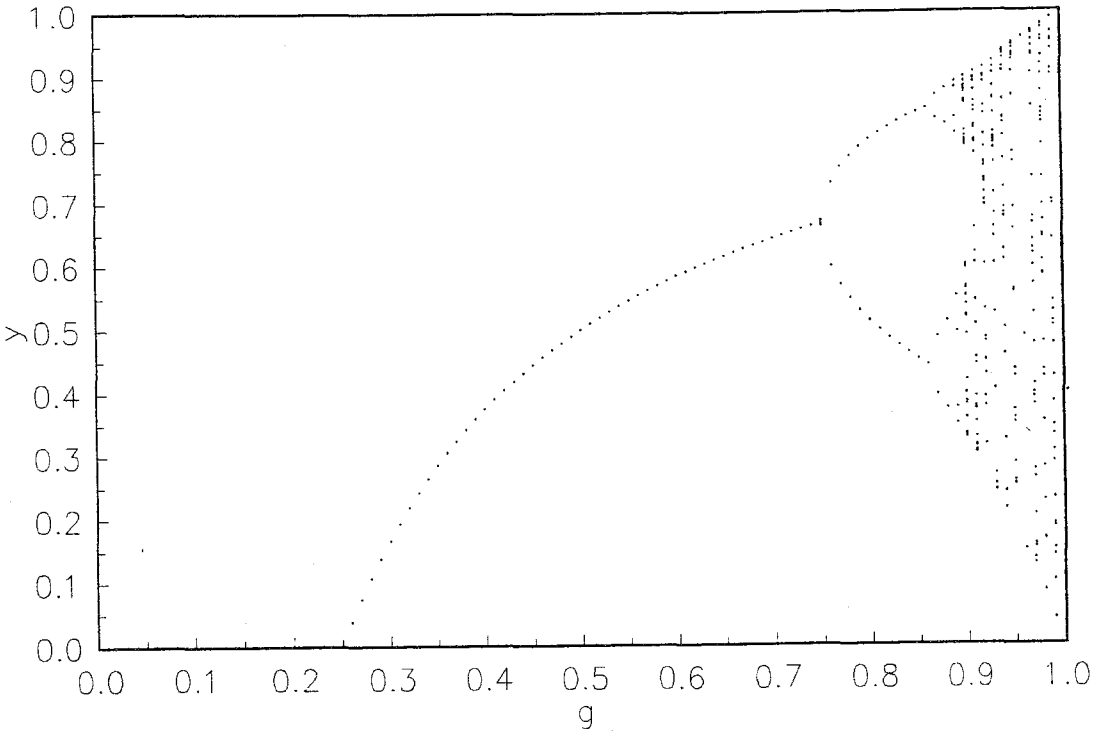

Figure 3.4 Bifurcation Diagram for the Feigenbaum Logistic Equation

In the chaotic neural unit, the total input to the unit determines the type of behavior produced, that is the input to the sytem plays the role of the g parameter. Another novelty is that the previous output of neural unit, is used to calculate the present one. In a conventional artificial neural unit, only the input to the unit is used as a variable parameter while the others are fixed. Note that the output if a neural network are not available until a time T after the inputs are presented. The waiting time T should be long enough for the transients to decay. Then Fig. 3.4 represents the transfer function of the neural unit.

Dingle et. al used an inverted form of the bifurcation diagram in Fig. 3.4 so that it would agree with a set of observations from experiments performed by Freeman. Freeman

has found that the olfactory neurons of a rat appear to have chaotic activity when the rat is subjected to a novel odour, hereas the activation in reaction to a familar odour appeared to me more ordered. This result was immediately transferred to the artificial neural network world by having a large network output when the input is familiar, and allowing for chaos when the input is not familiar. The inverted bifurcation diagram of the Feigenbaum logistic equation can be seen in Fig. 3.5 and is described by the following equation.

$$net = \sum_{i=1}^{M} w_i z_i$$
$$hid(n+1) = 1 - 4(1 - net)hid(n)(1 - hid(n)) \tag{3.3.31}$$

where *net* is the net input into the neural unit and *hid(n)* is the output of the neural unit at step *n*. This terminology was chosen because the neural unit was used only in the hidden layer for this project, but it could also be used in the output layer.

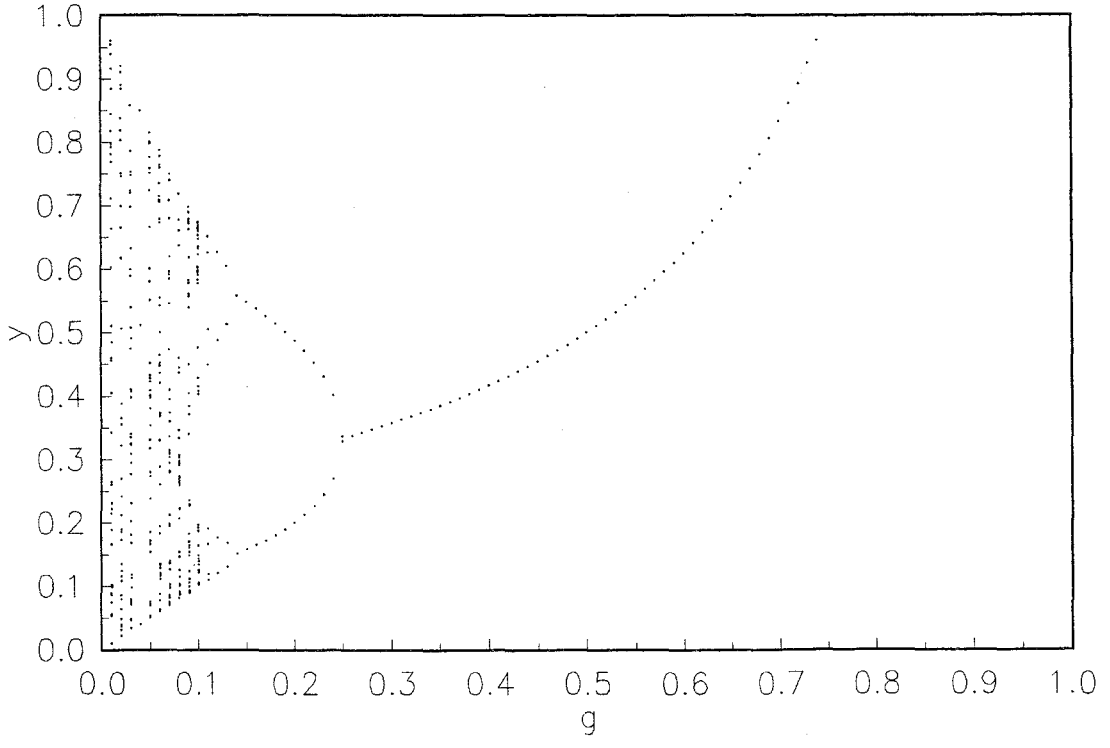

Figure 3.5 Transfer Function of the Chaotic Neural Unit

This neural unit behaves in the same way as a conventional neural unit for $0.25<net<1.0$. But when the net input to the neural unit is less than 0.25, the activity can be either periodic or chaotic.

In [28], it is shown that a network of 10 chaotic neural units connected by pseudo randomly chosen connection weights can exhibit chaotic behavior similar to the electrical activity of brain recorded in electroencephalograms (EEG). It should be noted that the net input to a chaotic neural unit should not be greater than one. In other words the weights should be chosen such that:

$$\left|w_{ji}\right| = \frac{\alpha}{M} \tag{3.3.32}$$

where M is the number of neural units supplying input to the neural unit whose input connection weights are being updated. The parameter α describes the maximum coupling between the neural neural units and the behavior of the network depends on the this parameter. For example the network describedc above for immitating EEG signals exhibit two kinds of behaviour depending on the coupling between the neural units, or in other words the parameter α. A low value of α produces irregular EEG activity whereas a higher value produces EEG alpha rhythms.

Dingle et. al. use the chaotic neural unit in a self organizing map architecture and prove its improved abilities for clustering the input patterns and modelling their probability density function.

4 Earthquake Data Analysis

The area studied in this thesis, bounded by lattitudes 39.50-41.50E, and longitudes 25-28.50N, is roughly the Marmara region of Turkey. A plot of the epicenters of Earthquakes that ocurred during the ten year time span from 1/1/1970 to 1/1/1991, can be seen below in Fig. 4.1. The plot shows all of the 3801 events detected by the local seismographic network.

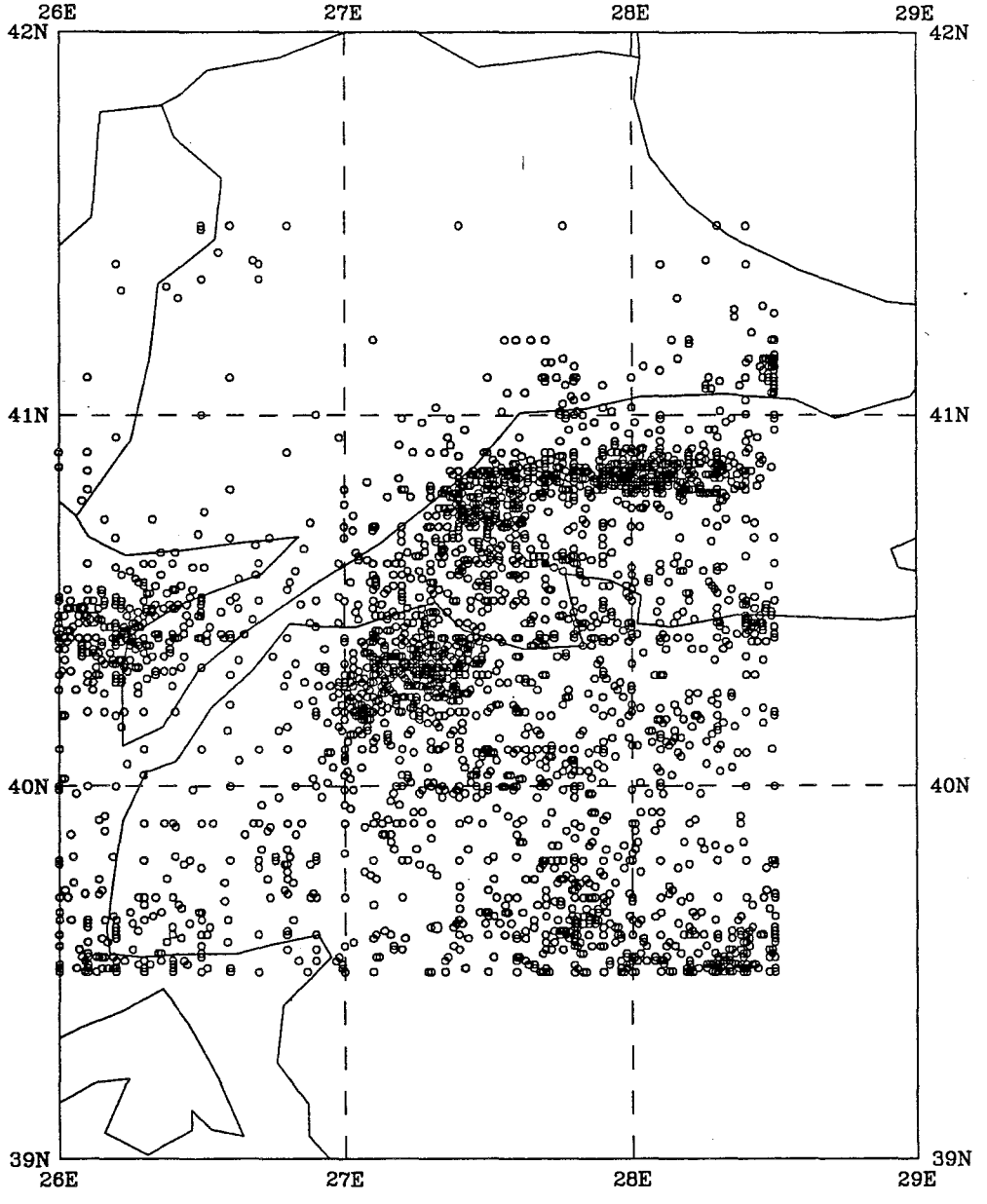

Figure 4.1 Plot of the Earthquake Epicenters in the Region Analyzed

The Agean sea and the surrounding areas, that include the Marmara region, is the most active regions of all Western Eurasia [7,29,33]. The Aegean region is placed between the African and Eurasian continental plates. Three main movements give rise to the active deformation, and high seismic activity in the area [29]. These are:

1. The West-bound movement of the Anatolian continental plate, starting from Bitlis-Zagros region, along the North and East Anatolian transform faults and the Southwest-bound movement of the Aegean sea base, both relative to Eurasia..
2. The subduction in the North and Northeast direction along the Hellenic Arc-Trench system.
3. The continental compression between Northwestern Greece, Albania and the Apulia-Adriatic platform.

In this area the West movement of the Anatolian continental plate and the Southwest movement of the Aegean sea base, are effected, in part, by boundary forces caused by the compression between the continental plates of Eurasia and Africa. Another factor giving rise to these movements are the buoyancy forces caused by the subduction zone, and the graviational potential energy due to the thickenning of the crust in East Anatolia [30].

The geophysical structure of the region is more complex than the above definition. For example the area analyzed had small faults scattered densely in the region. The activity on each of these faults affect a number of others. Thus an earthquake occuring on one fault might trigger an earthquake on another fault.

In the subsequent section the fractal dimension of the temporal distribution of the earthquakes and the completeness analysis are presented as well as the fourier transfom and autocorrelation spectrum of the earthquake data used in the project. Most of the experiments whose results are presented in section 5 were performed on the whole data set. The fractal dimension calculations of the next subsection were also performed on the whole data set in the same spirit. The fractal dimension calculation was performed to show that there is indeed a relationship between the interevent times and magnitudes of two sequential earthquakes. Thus the attempts for predicting the earthquake sequence are validated by the fact that the earthquake sequence posseses a stochastic self similar nature.

The Box-Jenkins algorithm on the other hand was applied to data containing only the main earthquakes. This set contained only the magnitude range that was complete. The completeness analysis showed that the magnitudes less than 2.6 were not recorded reliably by the seismographic network, therefore these were eliminated from the set. Unfortunately

it turned out that high magnitudes especially the range 4.5-5.5 were not complete either. This incompleteness may be because the physical earthquake generation system of the region was not contained fully in the area analyzed. The earthquakes, with high magnitudes, generated by the this system might be occuring outside the study area at least durung the time span studied. Another reason for the incompleteness might be that the time span considered was not long enough to collect all the necessary movements of the particular area. The aftershocks of the main shocks were removed according to the procedure given in [10-12]. All earthquakes occuring within 23 days after a main shock in the area covered by a circle of radius 50km and center as the epicenter of the main shock, are removed if their magnitudes are less than or equal to the magnitude of the main shock. These parameters were chosen by trial and error, but they are also used by [13], other choices did not produce a very different picture.

The Fourier spectrum of the data is presented in section 4.3 in order to give an idea about the frequency content of the data. It is also a good idea to perform a spectral analysis before choosing the activation function of the neural network. The reasoning for this statement will be presented in the same section.

4.1 Fractal Dimension of the Temporal Distribution of Earthquakes

The time clustering of earthquakes have been suspected to posses a chaotic character for a long time. The stochastic self similar nature, or the fractal structure in space and magnitude, and the fractal geometry of active seismogenic faults have been investigated in literature [29-31]. It was found however that the scale invariant behavior of the earthquakes persists only over specified scale lenghts. If the occurence of each earthquake was totally uncorrelated with the others, the earthquake production process would be a random one. It is not expected to find that the seismicity is random as an increase in the state of stress in the region is expected to cause clustering of earth movements. Finding a relation in such a form may help in devising algorithms for earthquake prediction, and in understanding the earthquake generation process of the region considered. Some study has already been done for relating the clustering dimension of seismicity to other measures such as Omori's exponent and b value [31].

In the following an algorithm proposed by Smalley et al [32]. for quantifying the time clustering of the earthquakes on the basis of their temporal distribution is presented. Smalley et al. applied the original algoritihm to New Hebrides. They found that there was a significant deviation from the usually assumed random or poisson behavior and that the fractal dimensions varied from 0.126 to 0.255 for the area they studied. Later on Papadopoulos et. al. [33], applied the same algorithm to the Helleneic Arch-Trench system and found that the fractal dimension varied betweeen 0.137 and 0.251. They have considered the shallow, intermediate depth and all depth shocks seperately. Denoting the fractal dimension of shallow, intermediate depth and all-depth shocks respectively as D_1, D_2,D_3, they determined that fractal dimensions were related by $D_3>D_1>D_2$. Papadopoulos et. al. investigated parts of the area between latitudes 40-34N and longitudes 20-29E. They divided the whole area into seven segments that did not cover the entire region.

Smalley et al. extended the fractal dimension definition of a curve, first proposed by Mandelbrot, to the temporal distribution of earthquakes. The smaller the value of D the more isolated are the clusters. Thus a smaller D value will correspond to concentration of the events being isolated in time from each other. In this method the fraction, v, of the intervals in which an earthquake occurs is related to the time length, τ, considered by:

$$v \cong \tau^{1-D} \tag{4.1.1}$$

then the fractal dimension is D.

Papadopoulos et. al. have examined shallow, $ht<60$ km, intermediate depth, $ht\geq 60$ km, and all depth, $ht\geq 0$ events between the years 1964-1985 for the surface wave cutoff of 4.0 and 1950-1985 for a cutoff of 4.5. The fraction of the time intervals that include an event is plotted as a function of the interval size in logarithmic scale, and D is determined from the slope j of the best fitting line through the data by the relation $j=1-D$. Papadopoulos et. al. used two minutes as the smallest time interval considered and increased the size by factors of two.

The fractal dimension is determined between the upper limit which corresponds to $log(v)=0$, and the lower limit where the distribution deviates from the uniform distribution. Uniform distribution predicts events equally spaced in time, in such a case $v=1$, when an earthquake occurs in every interval, that is the number of events N, is greater than or equal

to the number of time intervals n. If $N<n$, then $v=N/n$. The two limits signify the scale lengths of time over which the clustering of the earthquakes is scale-invariant.

Papadopoulos et. al., found that D_1 ranged between 0.137 and 0.191 in both time intervals and all segments considered expect for two. They attributed the anomalities, high D values, found in these segments to the relatively high number of shallow strong ($M_S \geq 6.0$) earthquakes. In other words the increased number of sequences of dependent events means less isolated clusters in time with respect to other segments examined.

The relation $D_3>D_1>D_2$ has a seismotectonic meaning. Intermediate depth shocks are associated with relatively few number of foreshocks and after shocks. This character of intermediate depth shocks tend to decrease the clustering strength at the time around a specified mainshock with respect to the clustering strength of the foreshock-mainshock-aftershock shallow sequences. This is counteracted by the fact that the intermediate depth earthquakes are more scarce, causing the intermediate depth events to appear in clusters more strongly isolated in time with respect to shallow ones. Thus the relation $D_1>D_2$ is explained. The relation $D_3>D_1$ means that if a sequence of nonshallow earthquake sequence is added to a shallow earthquake sequence both occuring in the same time interval and segment, then the clustering strength decreases. It was shown that the distribution of the shallow earthquakes in this area can be adequately described by a simple poisson process. The probability of occurence of nonshallow earthquakes in the same time position as the shallow ones is relatively low so that when both sequences are considered together the uniformity of the distribution is decreased. This explained $D_3>D_1$.

In this thesis, the fractal dimension of the temporal distiribution of all earthquakes occuring in the area 39.50-41.50N and 25.00-28.50E during the time length 1/1/1970 and 1/1/1991 is calculated. The initial interval considered in the algorithm, τ, covered the complete time span and at each iteration the length of the interval is halved until 1 minute is reached. We counted the number of time intervals of length τ that contained an event where an event is defined as any earthquake that was detected by the network. Unlike Smalley et. al. and Papadopoulos et. al. we did not consider only earthquakes above a certain magnitude. All events registered by the seimological network regardless of their magnitudes were significant for our algorithm and we did not distinguish between shallow and intermediate-depth events. We have considered a total of 3801 events in this study . In the references the authors chose to use a magnitude cutoff to ensure that all earthquakes of the particular magnitude range in the region are registered by the regional seismological network. The graph of the fraction v, and the time interval τ was plot and can be seen in Fig. 4.2. The fractal dimension found was about 0.1892278 which agrees with the results found by Papadopoulos et. al..

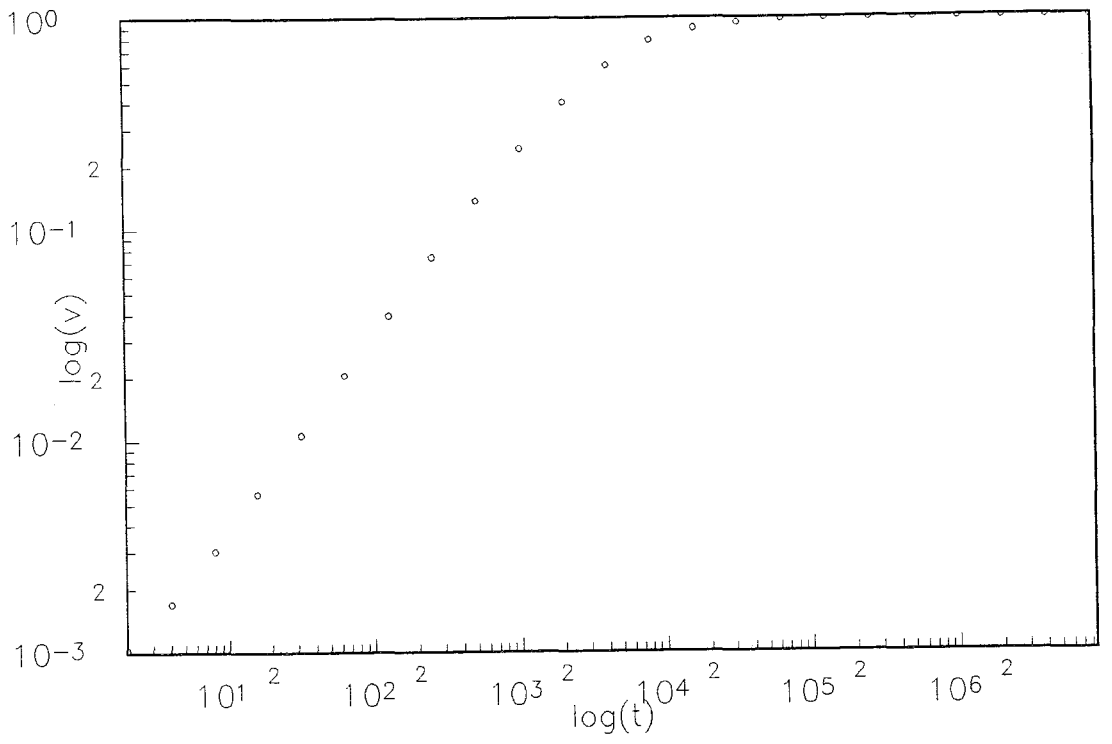

Figure 4.2 Fractal Dimension of Temporal Earthquake Distribution

4.2 Completeness Analysis of Earthquake Data

Completeness Analysis uses an algorithm similar to the one above. In fact this similarity is studied in [29]. The number of earthquakes of a given magnitude is calculated for all magnitudes. It is assumed that the number of earthquakes in log scale will have a linear relationship with the magnitude and *b* will be the slope of this line. The point where the data distribution deviates from such a behavior is taken to be the point where the magnitudes below are incomplete.

This was applied to the earthquake sequence studuied and it was found that the data set was complete for all magnitudes greater than 2.6 according to the Richter scale. At the higher magnitudes the data was scarce and deviated considerable from the linear behavior, and the highest reported was 5.5 according to the Richter scale so that the data was not complete for the high magnitudes.

The plot for completeness analysis can be seen Fig. 4.3.

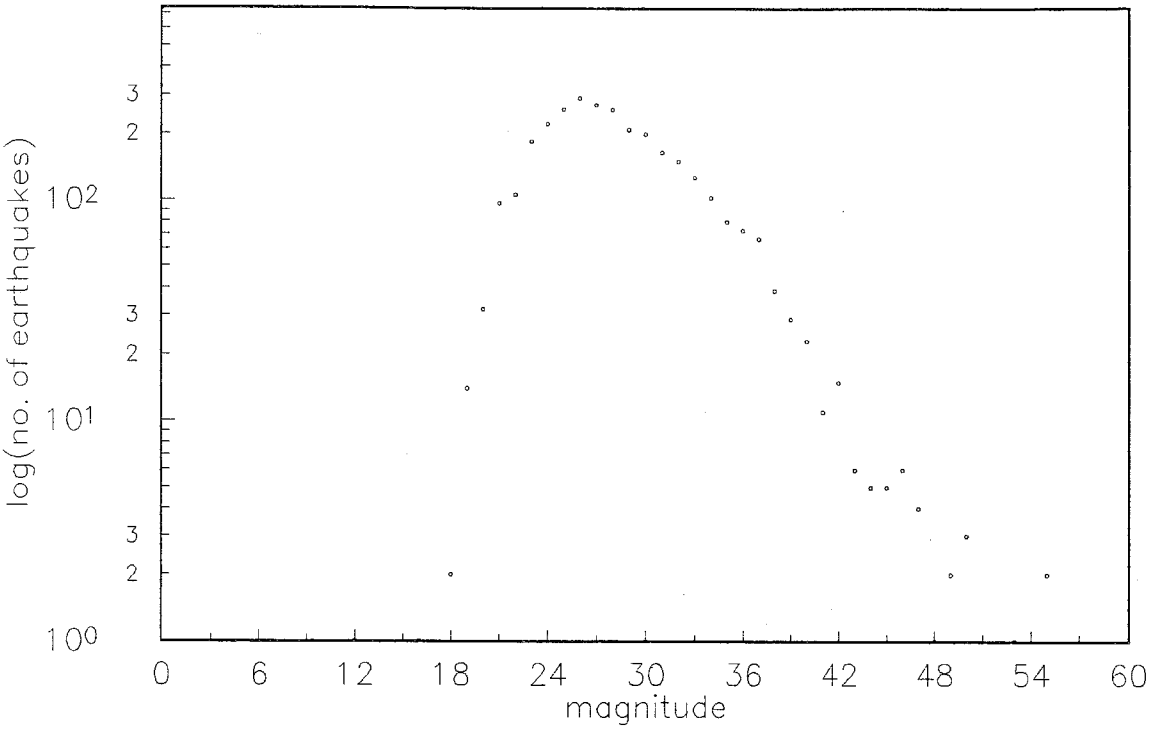

Figure 4.3 Completeness Analysis of Earthquake Magnitudes

4.3 Fourier Spectrum of Earthquake Magnitudes

Bruce E. Segee suggests choosing the basis functions to match the spectral properties of the function to be learned [34]. Segee makes the observation that the network output is a linear superposition of the inputs which may be then passed through a nonlinearity. This makes it possible to use linear superposition to make intelligent choices for the parameters of the Artificial Neural Networks, ANN.

Frequency domain analysis characterizes a signal in terms of the frequency components that it contains. It is usually employed to analyze the effect of a linear system, such as a linear filter, on a given signal. An important aid in such an anlysis is the fact that if a range of frequencies are not present in the input to a linear system they cannot appear in the output. Similarly if the transfer function of the linear system is zero at a certain frequeency range then these frequencies cannot exist in the output, because the output of the linear system in the frequency domain is the product of the transfer function of the filter and the frequency domain representation of the input signal.

The difference between a linear filter as described above and an ANN is that the operation is no longer that of producing a different signal by filtering a signal but that of linearly summing up a set of functions of independent variables. Therefore the input is not a signal but rather a set of independent variables.

An important observation paralleling the fact stated above is that if the desired output function has frequency content in regions where the activation functions have little or no frequency content learning will be very difficult. Since the network is to compute only an approximation of the desired function it is not impossible to learn in such cases. Such a "wrong" choice for the activation function will probably result in slow learning and less reliability.

The activation functions in the network have significant energy in some frequency bands. Learning involves enhancing the desired frequency components that are very small in the activation functions and cancelling out the unwanted ones that are large. The parameters of such a network must be chosen very carefully, and loss any of such parameters may destroy the delicate balance of the network. If on the other hand the network parameters are well matched with the desired output function then the ANN will not devote all or most of its resources to achive a balance and will encode information in a more redundant way giving rise to a more fault tolerant system.

In this light it is desirable for the network designer to have a good idea about the spectral properties of the activation functions that can be used in ANN, as well as the signal that is to be modelled, in order to make intelligent choices.

The fourier analysis was not performed for this project. The main reason for this is that the data used as input was not in any form suitable for this analysis. For most of the applications, an earthquake was represented by its magnitude and the time that elapsed since the previous earthquake. This was a logical alternative to treating the data as a real time series of one dimension, that is the magnitude. As the interevent times between two earthquakes varied from one minute to hundreds of thousands of minutes, the data had to be treated on a minute by minute basis, which could take too long to process.

The fourier spectrum of the earthquake data used in the project are presented in Fig. 4.4.

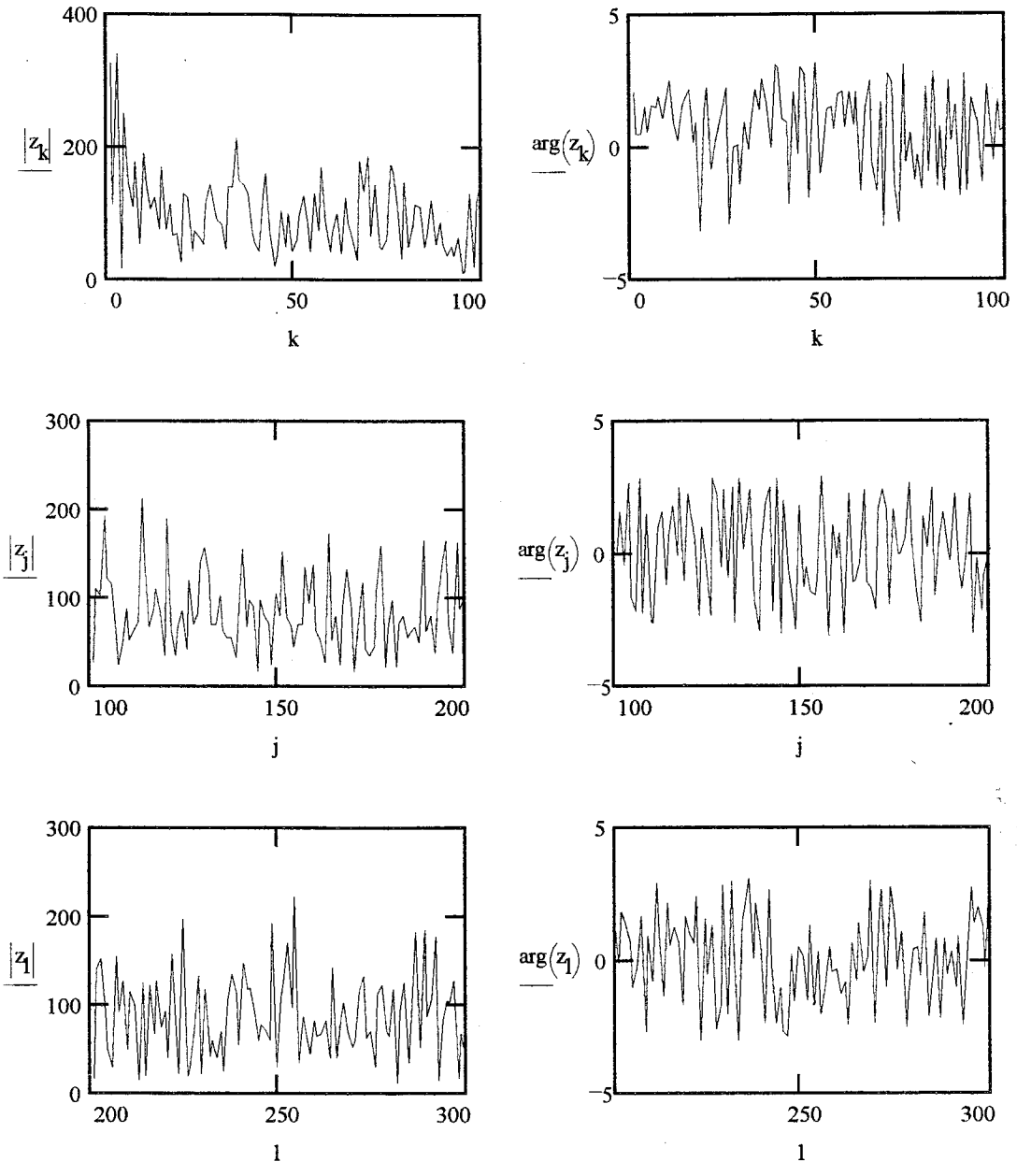

Figure 4.4 Fourier Spectrum of Earthquake Magnitudes

where $|M|$ stands for the magnitude and arg(z) stands for the phase of the complex number z.

4.4 Autocorrelation Functions of the Earthquake Magnitudes

Viewing the autocorrelations and partial autocorrelations of a data set is in itself useful for getting an idea about the function that can give rise to the particular data set. They are also needed for the Box-Jenkins algorithm in order to identify and build a model for prediction. The autocorrelations used for estimation on section 5 are presented here together with a brief account of the formulas used for computation. All the theory and formulas can be found in [14] in great detail. The calculations in this section are very similar to the ones presented in section 3.1. The autocorrelations are computed using the equation

$$r_k = \frac{c_k}{c_0} \tag{4.4.1}$$

where r_k and c_k are estimated autocorrelations and autocovariances, respectively. The estimated autocovariance function is defined as

$$c_k = \frac{1}{n}\sum(w_t - \overline{w})(w_{t+k} - \overline{w}) \tag{4.4.2}$$

where the sequence w is obtained by differencing the original time series an appropriate number of times as explained in section 3.1, and $\overline{w}$ is simply the mean of the sequence w. The plot of the autocorrelation function for the complete set of earthquake data whose incomplete small magnitudes and aftershocks are removed.

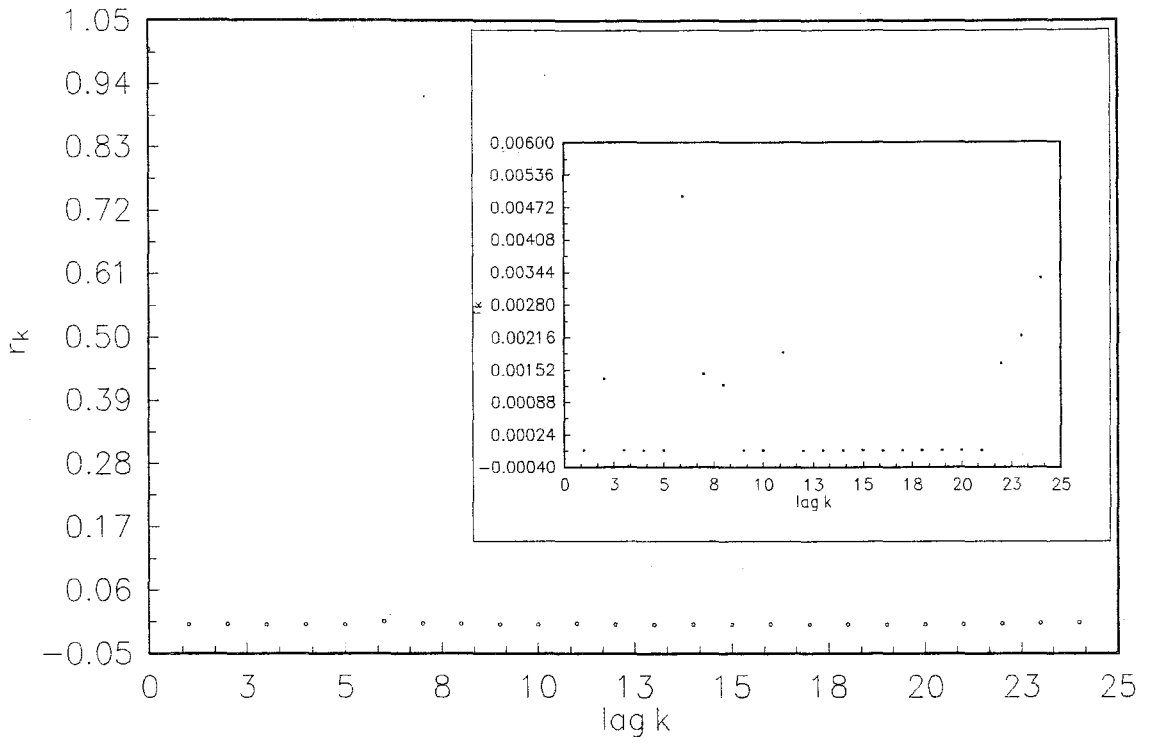

Figure 4.5 Autocorrelations of the Earthquake Magnitudes

The estimated partial autocorrelation is calculated by using the estimated autocorrelation values.

$$\phi_{ll} = \frac{r_l - \sum_{j=1}^{l-1} \hat{\phi}_{l-1} r_{l-j}}{1 - \sum_{j=1}^{l-1} \hat{\phi}_{l-1} r_j} \qquad (4.4.3)$$

where as before the symbol ^ means that the variable is calculated using the partial time series in hand. The equation (4.4.3) is valid only for $l>1$, and $\hat{\phi}_{11} = r_1$. The rest of the unknowns in (4.4.3) are caluculated using (4.4.4) given below.

$$\hat{\phi}_{lj} = \hat{\phi}_{l-1,j} - \hat{\phi}_{ll} \hat{\phi}_{l-1,l-j} \qquad (4.4.4)$$

where j ranges from 1 to l-1. The partial autocorrelations of the earthquake data can be seen in Fig. 4.6 below.

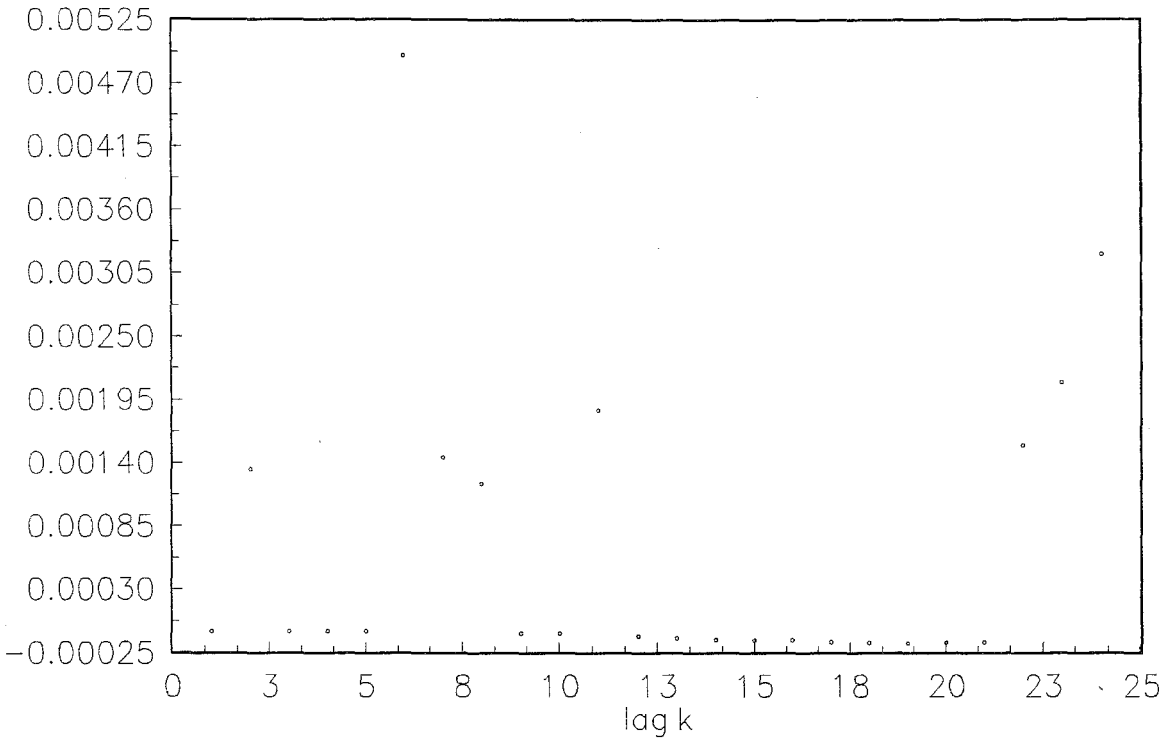

Figure 4.6 The Partial Autocorrelations of the Earthquake Magnitudes.

5. RESULTS OF EARTHQUAKE PREDICTION METHODS

In this thesis a different approach is adopted to the earthquake prediction problem. We treat the earthquake data as a time series that arises from a dynamical system, possibly a low dimensional chaotic system. The initial aim was to use the five dimensional time series that includes the time, magnitude and three spatial coordinates, i.e, lattitude, longitude, and the depth of the earthquakes. This was abondoned for time restriction as well as the complexity of defining such a system in the area considered that is known to have many different small sources acting together. The problem, then was how to account for time as using the absolute time would cause overflow as computations proceed unless some precaution was made and also as the physics of the situation imply that the magnitude or at least the occurence of an event would depend on the interevent time since the previous event. This corresponds to the time needed to gather enough energy in a certain area before releasing it. We have applied the time series consisting of interevent time between the previous and present event, the two coordinates, lattitude and longitude, and the magnitude of the present event. We found that the number of hidden units used and the epochs that the system was trained for was not quite enough and experimentation was not possible due to the time restriction. Also using the lattitude and longitude of the event would imply that there is actually a function that describes the order in which places on earth experience earthquake.. Therefore we have restricted our attention to a small area and worked on the data pertaining only to this region. It was expected that the dynamics of a part of the complete study area would be simpler than the dynamics of the whole region. Another approach, which is probably a more sound one, would be to use the data of the surrounding regions as input as well..

Limiting the region of study should not be a restriction if the system is chaotic, for space distribution, because chaotic systems possess a self-similar nature so that if we can solve the problem for this small area the optimistic view would be that we can solve it for the whole area by appropriately scaling the solution. Otherwise we can just apply a network of prediction networks covering the whole area and overlapping each other in order to compute predictions on a more global scale. Thus we had a two dimensional time series, interevent time and magnitude, of 415 elements. This set contained all events that occured in the ten year time span that was studied. This set is plotted in Fig. 5.1.

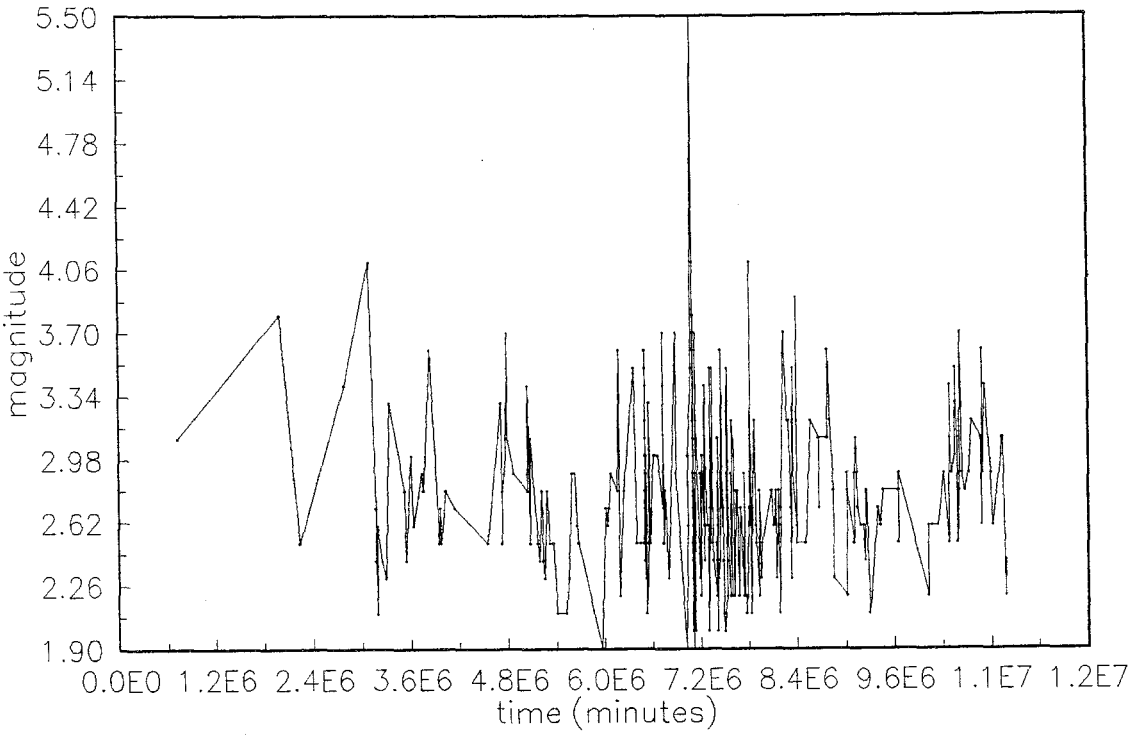

Figure 5.1 Earthquake Magnitudes in the Smaller Area

Unfortunately limiting the region had the drawback that the physical system producing the earthquake sequence was not contained in it entirety in the study area, therefore some of the relevant information on the system was lost. That is since the faulting structure that gave rise to the earthquakes in the region was not contained in the study area, some of the earthquakes occurred outside the study area and thus could not contribute to the model.

The previous approach did not yield encouraging results. Assuming that this was due to loss of information, the earthquake sequence of the complete study region was to be used in order to capture the dynamics of the generation process. The complete data set had 3801 elements. But these elements were cut down to 708 by eliminating small magnitudes that could not be reliably measured or detected by the seismographic network, and then by removing the aftershock sequences produced by major earthquakes. The completeness analysis, in section 4.2, showed that the smallest magnitude detected reliably by the seismographic network was about 2.6-3.00. Using this information, the events with magnitudes less than 3 were eliminated from the data set. Unfortunately, as mentioned before, the data set was not complete for high magnitudes. Either the area analyzed was too small, or the time span considered too short for the dynamics of the process to be portrayed correctly. This situation could not be remedied due to time restriction.

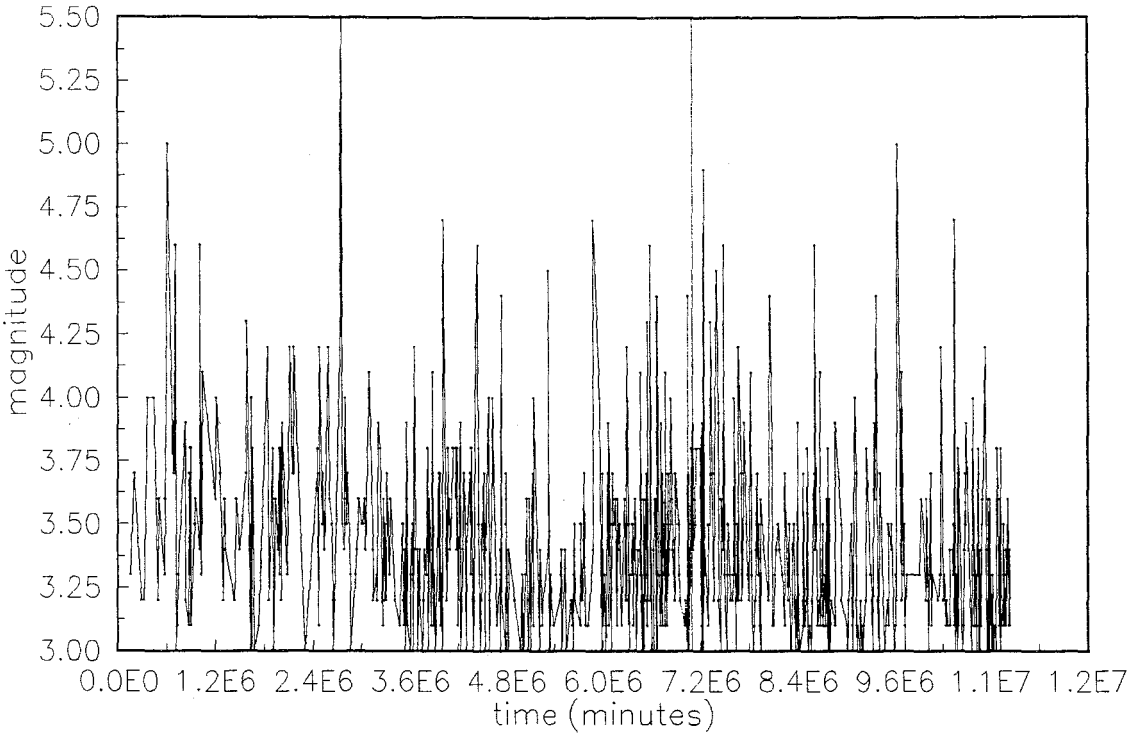

Figure 5.2 Earthquake Magnitudes in the whole region with aftershocks removed

Then the aftershock sequences of the main events were removed. The first event of the data set was assumed to be a main earthquake, and all the events, whose magnitude is less than or equal to the magnitude of the main event, that occured within 23 days after its occurence in the circular area with radius 50 km around its epicenter, were taken to be the aftershocks of the main earthquake. All the aftershocks were removed and the procedure was repeated for the second event in the set and so on. Thus the resulting 708 event set consisted only of the main earthqukes that occured in the entire region. The results of this application are not encouraging either. This is probably due to the loss of information as well as the incompleteness of the data. Even though, it makes sense to analyze only the main events that are complete from a geophysical point of view, the aftershock sequence and all the activity, whether it is measured correctly or not, are also very important. Most of the time, these activities are also used in the analysis as extra parameters. For example CN is applied to a data set containing only the main events. It should be noted, though, that the number of aftershocks and the increase in the seismic activity are used as parameters for detecting a TIP.

The Box-Jenkins algorithm was applied to the 708 element earthquake data containing the main events for magnitudes greater than or equal to three only. The results are presented in this section.

Two different architectures were used for the time series prediction. The input to each of the networks was a time delayed sequence of the two dimensional time series starting with the present values. One of these was a multilayer perceptron with sigmoidal activation function. Different combinations of step size hidden unit number and time delay values were used. The same architecture was used with a different activation function a chaotic function defined as in equation (3.3.32). It was expected that this choice of activation function would be more suited to time series modelling as the sigmoidal nonlinearity turned out to be very restricted and could not learn. Also since the output value of a hidden unit at time t is used to calculate the output of the same unit at the next time step, chaotic activation function involves a short term memory which should be valuable in modelling and predicting continuation of time series.

The second architecture is NADINE, which is described in one of the previous sections. This architecture could calculate the second and third order correlations of the input data, if N, where n is the number inputs, was large enough, so that if there is indeed some correlation between the next event and the preceeding ones it should have been easy for NADINE to pick that up. The fact that N is not large enough for the multiplication of the time series values to be considered as correlations can be seen as a drawback for a conventional system. But if the system is indeed a chaotic one that it is better that N is not large as for a chaotic system nearby trajectories diverge and we do not and cannot have noise-free data with enough precision to stay on the same trajectory for a long time.

We could not apply the Clusnet architecture which actually classifies the input data into a number of distinct classes according to the training. At any time only one clusnet node can be active. The network, then fits a linear equation that was computed for the particular clusnet node according to the input patterns that activate it. The is equivalent to saying that the mappng to be modelled can be expressed as a locally linear function.

Another similar approach can be using a pattern recognition technique based on the k nearest neighbor algorithms. In such an application the distance measure should include the temporal closeness to the pattern as a weighting function. CNLS network could also be used.

5.1 Result of Box and Jenkins

Inspection of the autocorrelation function reveales the fact that the time series does not need differencing and that the order of the AR process can be taken to be zero, as all the autocorrelation values except for the lag zero one are almost zero. The partial autocorrelation function on the other hand exhibits a possibly exponentially damped sine wave decay. This implies that an ARIMA(0,0,1) model can be used for estimation. The preliminary calculations indicate that the model is

$$z_t = (1 - 0.5B)a_t \tag{5.1.1}$$

There are two kinds of noise affecting this estimation system. The first one is the implicit noise in the data due to inaccuracies in reception, measuring and recording of the seismic signal. The other one is the computationally induced one. The fixed point presicion as well as the fact that all through the analysis estimated values from a limited time series are used all give rise to some error.

The results of this section were not any different than the neural network results. The mean square error, MSE, was about 0.000562, which might seem to be a nice number. But it should be kept in mind that there are a lot of zero values in the data set and when the network produces a small number all the time then the total error will be very little. In order to visualize the situation, it is helpful think about the earthquake data set which is weeded from the incomplete magnitudes and the aftershocks. In this data set there are a total of 708 events that have a magnitude other than 0, but the time span is aroung one million minutes. The shortest interevent time is 1 and the longer ones can be on the order of 10000s.

5.2 Neural Network Estimation

As mentioned in the introduction of this section two diffferent neural network architectures were employed with a number of different parameters. Each of the architectures are explained below in their relevant section.

5.2.1 Result of MLP with Sigmoidal Activation Function

In this section and throughout the thesis, a *k*-layerMultilayer Perceptron, MLP, denotes an MLP with *k-1* hidden layers and one output layer. The basic architecture of a two layer MLP can be seen in Fig. 5.1.

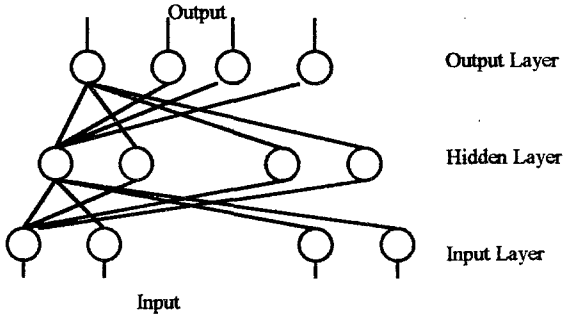

Figure 5.1 Multi Layer Perceptron Architecture

The nodes in the input layer pass the input to the hidden layer with no change. The input layer is static layer. The nodes in all the other layers sum up all the inputs entering the node and either pass this value to the next layer or to the outside as input, or else apply a nonlinearity to it and then pass it forward. The networks applied in this section have sigmoidal activation functions in the hidden layer nodes. The output nodes perform only the addition function. Let *net* denote the total weighted input to a neural unit, and *hid* denote the output of the hidden layer node. Then the output of a hidden layer node is calculated as

$$hid = \frac{1}{1 - e^{-net}} \tag{5.2.1}$$

During the initial stages experimentation with the parameter net in (5.2.1) showed that multiplying it with a constant did not result in any immediate improvement. The initial weights were chosen randomely and adapted according to the backward propagation algorithm.

The input to the network for all of the applications was rescaled so that they would lie in the range [0.2-0.8] as it is proposed that this range facilitates learning better than the acceptable range [0-1] of the sigmoidal function [34].

The results of the application of the MLP to the earthquake prediction task are reported below. Several two and three layer networks are employed for predicting the earthquake data series. Different number of input and hidden nodes and different step sizes were tried.

The first approach to the problem was to treat the data as a four dimensional time series, and try predicting the time of occurence, magnitude, and the two planar coordinates, lattitude and longitude, of an event. The input data set to the network included all the events detected by the seismographic network. The results are presented below.

Table 5.1 Results of 2-Layer MLP applied to four dimensional earthquake sequence

Network Architecture	Step Size	epoch	MSE for Time	MSE for Magnitude	MSE for Lattitude	MSE for Longitude
5-31-4	0.005	10	0.029628	0.264557	0.043281	0.119119
5-31-4	0.5	10	0.027331	0.265913	0.043947	0.116476
5-31-4	0.05	500	0.179791	0.112623	0.066535	0.186016
45-11-4	0.05	5	0.180446	0.280650	0.066613	0.185937
45-11-4	0.5	10	0.181275	0.676863	0.067371	0.187494
45-11-4	0.00005	50	0.092725	0.211001	0.061173	0.179106
45-11-4	1.0	20	0.185565	1.512687	0.094923	0.225698

This was abondoned afterwards as it did not make sense, physically, to predict the location on the crust that will break release some energy in the form of an earthquake. Also the calculations tended to be too involved when the input time series was four dimensional. The next approach was to predict only the magnitude of the event at the next time step. Initally the same input file that was used in the 2-Layer MLPs presented above was intended to be used, except that the zero magnitudes that were recorded by the seismographic network but could not be measured were removed from the data set, as a zero magnitude denoted no event for this application. The interevent times between two succesive earthquakes in this particular data set varied between one minute and order of 100,000 minutes. Therefore processing the input stream on a minute by minute basis took a long time. This is an important point to keep in mind, since even after the aftershock sequences are removed the smallest interval is still on the order of one minute. This implies that the resolution of the input steam should be one minute. But if the input fed is the magnitude recordings at each minute, then there should be about 100,000 input nodes to make sure that at least one of the input nodes is nonzero. Practical limitations restrict the number of inputs of the network, so that usually 10,000's of iterations would be performed to update the weights without having any valid input. That is the input sequence would be all zeros, denoting that there was no events occuring during this time interval. Inspection of the situation revealed the fact that such a scheme cannot be succesfull, especially if the network is trained off line, since it will not be possible to distinguish the time before an earthquke from the time of no-event. In other words the network will have an all zero input but the output can be either a zero magnitude or an earthquake of an arbitrary magnitude. The results of this application are presented in Table (5.2)..

Table 5.2 Results of 2-Layer MLP applied to one dimensional earthquake sequence

Architecture	epoch	Step Size	MSE for Time
11-31-1	20	0.005	0.006120
21-31-1	20	0.005	0.006134
21-31-1	20	0.005	0.005274
21-41--1	50	0.005	0.006838
21-31-1	40	0.05	0.006009
21-31-1	80	0.05	0.018030
21-31-1	40	0.5	0.014885

The major part of this project was dedicated to predicting using a two dimensional earthquake sequence. The coordinates of the earthquake were dropped and a smaller region in the whole area was chosen for prediction purposes. The smaller region was chosen by inspecting the plot of the earthquakes. A region which was dense in the number of earthquakes was chosen somewhat arbitrarily. The data set used for 2-layer MLP's in this section contain 415 data points, 300 were used for training and the rest 115 for prediction.

This turned out to be a bad approach as it did not take into acount the physical system producing the earthquakes. Assuming that the region chosen contained all the relavant physical sources, the two dimensional time series approach seems to be a logical one. The time is replaced by the interevent time, that is the time passed since the last event is used a parameter with the magnitude of the present event. The results of the application of the two dimensional time series to 2-layer MLP can be seen in Table 5.3.

Table 5.3 Results of 2-Layer MLP applied to two dimensional earthquake sequence

Architecture	Step Size	Epoch	MSE for Time	MSE for Magnitude
3-9-2	0.05	1000	0.041959	0.036711
3-19-2	0.05	1000	0.045367	0.039565
3-90-2	0.005	425	0.041370	0.036556
7-90-2	0.005	500	0.044675	0.030602
7-19-2	0.005	1000	0.046267	0.032721
7-19-2	0.05	500	0.042725	0.030221
7-19-1	0.09	100	0.020363	-
41-41-2	0.1	100	0.003397	0.004476
41-41-2	0.3	100	0.000679	0.002342
41-41-2	0.15	300	0.001033	0.002105
41-200-2	0.15	200	0.003130	0.006456

The time passed since the last earthquake is indeed an important factor for determining the magnitude of the present event, as it, in a way, represents the time span that the energy hs been accumulating in the area. In fact in section 2.1 it was shown that they are related. The same data set was applied to a number of 3-layer MLPs also. The set of data on the whole area with aftershocks and incomplete small magnitudes removed was used for the remaining three layer networks. The results are presented in Table 5.5.. For the first 12 entries the input used was taken from the smaller region as described above. The rest of the entries used the main earthquakes only.

Table 5.4 Result of 3-Layer MLP applied to two dimensional earthquake sequence

Network Architecture	step size	epoch	Training MSE for Time	Training MSE for Magnitude
11-11-11-2	0.15	100	0.002795	0.015420
11-11-11-2	0.35	100	0.003498	0.019801
11-11-11-2	0.5	100	0.174811	0.002697
11-11-11-2	0.85	100	0.014125	0.081928
11-21-21-2	0.15	100	0.002719	0.015015
11-21-21-2	0.2	500	0.002651	0.016267
21-11-11-2	0.5	100	0.001651	0.002963
21-21-21-2	0.2	100	0.002817	0.001637
21-21-21-2	0.5	100	0.001617	0.002867
11-21-21-2	0.5	100	0.001693	0.002950
11-31-31-2	0.2	100	0.002936	0.015466
21-21-21-2	0.3	2000	0.051830	0.002122
21-21-21-2	0.5	2000	0.000720	0.002829
21-21-21-2	0.85	2000	0.002393	0.009498
31-31-31-2	0.35	2000	0.000564	0.002238
31-31-31-2	0.5	2000	0.000728	0.002812
31-31-31-2	0.85	2000	0.002424	0.009383
41-41-41-2	0.5	2000	0.000703	0.002846
41-41-41-2	0.85	2000	0.002294	0.009557

Throughout this section only the training MSE will be presented as mentioned above. a network trained for earthquake detection should not be used off line as there are too many factors affecting the generating system and any error will grow very fast.

The data used was always factored to be less than one and greater than zero. This means that if the predictions are small, especially if they are close to zero, the MSE will be small too whether the predictions are good or not. The prediction output of most experiments was very close to straight line usually close to the mean of the data whereas during the training phase the network usually imitated the behavior of the earthquake sequence at the previous step. For example if there was a rise in magnitude at step i to a level M_i, the network would produce an output close to M_i at step $i+1$. The earthquake sequence is not smooth at all, and this breeds the difficulties in predicting the future values.

Table 5.5, below, shows the Prediction error of some of the 3-layer networks that were also presented in Table 5.4. It is not applied off line. That is it is adapted at each step. As mentioned before the earthquake generation process is a dynamic process that is

changing continously because a large number of natural and human induced factors. Therefore it is not a good idea to just build a model and then use it for prediction for a long time. This is also a deficiency of the Box-Jenkins algorithm for the earthquake prediction problem.

Table 5.5 Results of 3-Layer MLP applied one step ahead prediction of two dimensional earthquake sequence

Architecture	Step Size	epoch	MSE Error for Time	MSE Error for Magnitude
21-21-21-2	0.3	2000	12.53441	0.304844
21-21-21-2	0.5	2000	0.082612	0.473277
21-21-21-2	0.85	2000	0.106992	1.037488
31-31-31-2	0.35	2000	0.075214	0.326194
31-31-31-2	0.5	2000	0.787246	0.485573
31-31-31-2	0.85	2000	0.106992	1.037488
41-41-41-2	0.5	2000	0.089847	0.524399
41-41-41-2	0.85	2000	0.106992	1.037488

5.2.2 Result of MLP with Chaotic Activation Function

The network architecture used in this section is similar to the MLP presented above except that the activation function is the chaotic Feigenbaum logistic equation, (3.3.32), instead of the sigmoidal one. The advantage of this activation function over the sigmoidal one is that it actually incorporates a form of memory. The output of a hidden unit with chaotic activation function is calculated by using the output at the previous step. The drawback is that the the hidden unit should perform the calculation in (3.3.32) for at least ten times to make sure that the output has settled. This is similar to the behavior of biological neurons. A biological neuron cannot be automatically turned on and off like the artificial ones. The results of the application of the data from the smaller region are presented below in Table 5.6.

Table 5.6 Result of chaotic MLP applied to two dimensional earthquake sequence

Architecture	Step Size	epoch	MSE Error for Time	MSE Error for Magnitude
65-11-2	0.00005	20	0.000814	0.001981
65-11-2	0.005	50	0.000780	0.001977
11-65-2	0.0015	30	0.000784	0.002261
11-65-2	0.0015	30	0.000774	0.002241
11-65-2	0.0015	30	0.000745	0.002169
11-65-2	0.0015	30	0.000756	0.002071
11-65-2	0.0015	50	0.000766	0.002004
11-65-2	0.0015	30	0.000751	0.001987
11-65-2	0.0015	30	0.000752	0.002016
11-65-2	0.0015	10	0.000724	0.002007
11-65-2	0.0015	30	0.000715	0.002005
11-65-2	0.0015	20	0.000863	0.002042

5.3.3 Result of NADINE

The architecture of NADINE is presented in section 3.3.3, and the training procedures are explained in detail. All the networks presented in Table 5.7 calculate Volterra terms up to order three except the last two which calculate up to the fourth order. The input data used was the same 415 element data set from the smaller region.

The problem with using NADINE for predicting the magnitude and interevent time of earthquake sequence is that a single weight at the output layer is not sufficient model two different variables. It was observed that for most of the time the MSE results of these variables behaved the same way. That is the plot of the MSE versus epochs of the magnitude would be almost a copy of that of the interevent time except for a difference in level.

It can be seen in Table 5.7 that the first three networks have only one output that is the magnitude. Predicting the time is actually the main difficulty in this problem because the very wide range of values it can assume. For this reason, throughout this project atempts were made to predict just the magnitude from the magnitude data. Usually these attempts

seemed to have slightly better results when inspecting the graphs visually. But the good predictions for the magnitudes might be by chance as the network output, failed to take on the wide range of values that the actual data had.

Table 5.7 Result of NADINE applied to two dimensional time series.

No. of Inputs N	Step Size η	No. of Training Epochs	Training MSE for Time	Training MSE for Magnitude
20	0.005	3	-	0.000510
125	0.005	3	-	0.005600
40	0.001	3	-	0.005600
21	0.1	3	0.00099	0.001951
21	0.3	3	0.00110	0.002242
21	0.6	10	0.00156	0.002576
41	0.5	4	0.00321	0.004279
41	0.2	4	0.00122	0.002612
41	0.5	3	0.00182	0.003001
41	0.25	3	0.001312	0.002724
41	0.15	6	0.00106	0.002286
40	0.15	6	0.000781	0.001859
40	0.3	3	0.000993	0.002096
40	0.15	25	0.000729	0.001794
40	0.25	30	0.000810	0.001831
40	0.15	30	0.000713	0.001742
40	0.25	30	0.000749	0.001766
18	0.15	15	0.000718	0.001872
18	0.25	75	0.00916	0.001937

6. CONCLUSION

None of the Prediction schemes applied in this project seems to be able to predict the earthquake sequence. A total of seven different groups of experiments were performed during the project. The epoch column in the tables refers o the epoch number after which they have attained the MSE error specified in the next column. A collection of best performances for predicting the magnitude of the next earthquake of each one of the versions is listed below in Table 6.1.

Table 6.1 Best performances of the neural networks for predicting the magnitude

Description of NN	Input Data	Epoch	MSE for Magnitude
5-31-4 MLP with sigmoidal activation function	4 dimensional	500	0.112623
41-41-2 MLP with sigmoidal activation function	2 dimensional	300	0.002105
21-21-21-2 MLP with sigmoidal activation function	2 dimensional	100	0.001637
65-11-2 MLP with chaotic activation function	2 dimensional	50	0.001977
NADINE with 20 inputs	1 Dimensional	3	0.000510
NADINE with 40 inputs	2 dimensional	30	0.001742

In the above table a four dimensional input means that each earthquake is described by a lattitude, longitude, magnitude and interevent time as described before. When the input is two dimensional, only magnitude and interevent time are used to describe an earthquake and when the input is one dimensional only magnitude is used. In Table 6.1 best performances for time prediction are shown. In this case one dimensional input contains only the interevent time between two earthquakes. The two tables do not coincide, only NADINE listed in the last row of both tables have the minimum for both values.

Table 6.2 Best performances of neural networks for predicting the interevent time

Description of NN	Input Data	Epoch	MSE for Magnitude
5-31-4 MLP with sigmoidal activation function	4 dimensional	10	0.027331
21-31-1 MLP with sigmoidal activation function	1 dimensional	40	0.006009
41-41-2 MLP with sigmoidal activation function	2 dimensional	100	0.000679
31-31-31-2 MLP with sigmoidal activation function	2 dimensional	2000	0.000564
11-65-2 MLP with chaotic activation function	2 dimensional	50	0.000715
NADINE with 40 inputs	2 dimensional	30	0.000713

Even though the mean square error, MSE, is listed throughout this thesis, it is not a good metric. The reason lies in the fact that the numbers involved are very small, after scaling, and the range of allowed values for especially the time data is very large and the movement from one point to the following one is abrupt and almost random.

In the previous sections it was stated that the function is not smooth. This, naturally, impedes the learning. The main problem is deciding on the choice of inputs that will be used. It was mentioned that there is at least one algorithm that is being used by geophysicits. This is the CN algorithm. It is basically a classification scheme that inspects the present time for a number of features and then declares whether there is a possibility of an earthquake or not. This algorithm uses the same data set, namely the earthquake catalog, as the present project. But the data is cleaned from the incomplete magnitude range, and aftershocks first and only the major earthquakes are to be predicted. Yet the information on the aftershocks and the overall activity in the region are not discarded, they are used as features of the classes. The attempt to use only the main shocks and only the complete magnitude range failed in this project, and probably the loss of too much information was one of the reasons for this failure.

In future work the data should be analyzed in more detail before any attempt to prediction or model building is made. The assumption that the data used in this project contains all or most of the necessary information about the eartquake sequence might be wrong. But before this approach is discarded, a complete data set with a large number of elements should be considered. Also ways of making the data more smooth must be explored.

The underlying faulting structure is also very important. Precaution should be exercized so that the physical system releasing the energy is not interrupted by boundaries

erected by the neural network designer. It is important that at this stage an expert provides guidance in the choice of the area.

The failure of the architectures used in this project is caused by the abnormal data nature. As was mentioned many times before the the intervent times vary between one and one houndred thousand minutes. Also the nature of th earthquake sequence is such that after a long time of no events occuring, a number of earth movements very close to each other in time might be experienced. These features are called respectively, quiscence, and swarm, or increase in activity. The pass from a state of quiscence to a state of frequent earth activity is very spontaneous and these are among the features used to detect possibility of an earthquake in the CN algorithm. The architectures that try to fit smooth function to this data, naturally fail. Therefore it might be more logical to apply some localized prediction mechanism such as a k closest neighbor algorithm that matches the present part of the function to past ones using time as a metric too. Events occuring in the nearer past must be weighted more than the ones occuring in distant past.

At this point in the project, it seems as if the pattern classification approach is more suitable to the problem as it is. It is obvious that smoothing the data used in this project willl result in information loss. This is not desirable, even more information than available is needed to solve the problem.

Another advantage of pattern recognotion is that different kinds of data can be used as features for detection rather than restricting the system to only the magnitude and time data. In problems like this as much data as possible should be used.

CITED REFERENCES

1. Falseperla, S., S. Graziani, G.Nunnari, and S. Spampinato,"Automatic Classification of Seismic events by Using Multi-Layered Networks," Preprint, submitted to *Journal of Geophysical Research.* Personal Communication with Dr. Aysin B. Ertuzun,1994.

2. Fortune, L., S. Graziani, M. Lo Presti, G. Nunnaru,"A Neural Network for Seismic Classification," *IEEE International Geoscience and Remote Sensing Symposium,* Helsinki University of Technology, Espoo, Finland, Vol.3, p.163-166, June 3-6 1991.

3. Welstead, Stephen T.,"Multilayer Networks can Learn Strange Attractors," *IEEE International Conference Neural Networks*, Vol. 2, pp.139-144.

4. Sudharsanan, S. I., and M. K. Sundareshan,"Training of a Three Layer Dynamical Recurrent Neural Network for Nonlinear Input-Output Mapping," *IEEE ICNN*, Vol. 2, pp.111-115.

5. Purcaru, G., K. Pawelzik,"A New Tool for Identification of Anomalous Seismicity Patterns", Preprint, Personal Communication with Serif Baris.

6. Radulian, Mircae, Cezar-Ionan Trifu, Florin Octavian Carbunar,"Numeriacal Simulation of the Earthquake Generation Process", *Pure and Applied Geophysics*, Vol. 136, No. 4, pp.499-516, 1991.

7. Papazachos, B. C., "A Time and Magnitude predictable Model for Generation of Shallow Earthquakes in the Aegean Area", *Pure and Applied Geophysics*, Vol. 138, No.2, pp.287-308, 1992.

8. Papadimitrou, E. E., "Long Term Earthquake Prediction along the Western Coast of South and Central America Based on a Time Predictible Model", *Pure and Applied Geophysics*, Vol. 140, No.2, pp.301-316, 1993.

9. Ferraes, Sergio G., "The Bayesian Probabilistic Prediction of the Next Earthquake in the Omatepec Segment in the Mexican Subduction Zone", *Pure and Applied Geophysics*, Vol.139, No.2, pp.309-329, 1992.

10. Keilis-Borok, V. I., I. M. Rotwain, "Diagnosis of Time of Increased Probability of Strong Earthquakes in Different Regions of the World: Algorithm CN", *Physics of the Earth and Planetary Interiors*, Vol.61, pp.57-72, 1990.

11. Keilis-Borok, V. I., V. G. Kossobokov, "Times of Increased Probability of Strong Earthquakes (M≥7.5) Diagnosed by Algorithm M8 in Japan and Adjacent Territories", *Journal of Geophysical Research*, Vol. 95, No.B8, pp.12412-12422, August 10, 1990.

12. Rotwain, I. M.,"Function on Earthquake Flow", *Workshop on Non-linear Dynamics and Earthquake Prediction*, Trieste, Italy,25 November-13 December 1991.

13. Personal Communication with Serif Baris.

14. Box G.P., Jenkins G.M., *Time Series Analysis: Forecasting and Control*, Holden-Day, San Fransisco, CA, 1970.

15. Wolf, A., J. B. Swift, H. L. Swinney, and J. A. Vastano,"Determining Lyapunov exponents from atime series," *Physica D*, Vol. 16, pp.285-317, 1985.

16. Sano, M., and Y. Swada,"Measurement of Lyapunov spectrum from a chaotic Time Series," *Pysical Review Letters*, Vol. 55, No. 10, pp. 1082-1085, September, 2,1985.

17. Rosenstein, M. T., J. J. Collins, and C.. J. De Luca,"A Practical Method for Calculating largest Lyapunov Exponents from Small Data Sets," *Physica D*, Vol. 65, pp. 117-134, 1993.

18. Eckhardt B., and D. Yao,"Local Lyapunov exponents in Chaotic Systems," *Physica D*, Vol. 65, pp.100-108, 1993.

19. Grassberger, P., and I. Procaccia,"Characterization of Strange Attractors," *Physical Review Letters*, Vol. 50, No. 5, pp.346-349, January, 31, 1993.

20. Linsay, Paul S.,"An efficient method for forecasting chaotic time series Using Linear Interpolation," *Physics Letters A*, Vol. 153, No. 6,7, March, 11, 1991.

21. Casdagli, Martin,"Nonlinear Prediction of Chaotic Time Series," *Physica D*, Vol. 35, pp.335-356, 1989.

22. Mead W.C., R.D. Jones, Y.C. Lee, C.W. Barnes, G.W. Flake, L.A. Lee, M.K. O'Rourke, "Using CNLS-Net to Predict the Mackey-Glass Chaotic Time Series", *IEEE ICNN*, Vol. 2, pp.485-490, 1991

23. W.Hsu, L.S. Hsu, and M.F. Tenorio, "A Clusnet Architecture Prediction", *IEEE ICNN*, Vol.1, pp.329-334, 1993.

24. Hassan M. Ahmed and Fawad Rauf, "NADINE - A Feedforward Neural Network for Arbitrary Nonlinear Time Series", *IEEE ICNN*, Vol. 2, pp. 721-726, 1991.

25. Gerald W. Davis and Michael L. Gasperi, "ANN Modelling of Volterra Systems", *IEEE ICNN*, Vol. 2, pp.727-734, 1991.

26. Simon Haykin, *Adaptive Filter Theory*, Englewood Cliffs, N.J., Prentice Hall Inc., 2nd Edition, 1991

27. Dingle, Alison A., John H. Andreae, Richard D. Jones,"A Chaotic Neural Network", *IEEE International Conference on Neural networks*, Vol.1, pp.335-340, 1993.

28. Aihara, K., T. Takabe, and T. Toyoda,"Chaotic Neural Networks", *Physics Letters A*, Vol.144, No.6-7, March, 12 1993.

29. Küleli, H. Sadi, M. Eryilmaz, H. Yüce,"Ege Denizinde Anadolu'nun dogal uzanimi", 1993, Personal communication with Prof. Cemil Gürbüz.

30. McKenzie D., Yilmaz Y.,"Deformation and Volcanism in Western Turkey and the Aegean", *Bull. Tech. Univ. Istanbul*,Vol.44, pp.345-373, 1991.

31. Hirata, Takayuki, "A Correlation Between the b Value and the Fractal Dimension of Earthquakes", *Journal of Geophysical Research*, Vol. 94, No. B6, pp. 7507-7514, June 10, 1989.

32. Smalley, R. F., Jr, J. L. Chatelain, D. L. Turcotte, and R. Prevot, "A Fractal Approach to the Clustering of Earthquakes: Application to the seismicity of the new Hebrides", *Bulletin of the Seismological Society of America*, Vol. 77, No.4, pp.1368-1381, Augus 1987.

33. Papadopoulos, Gerassimos A., Vassilis Dedousis, "Fractal Approach of the Temporal Earthquake Distribution in the Hellenic Arc-Trench System," *Pageoph*, Vol. 139, No.92, pp. 269-276, 1992.

34 Segee, Bruce E., "Using Spectral Techniques for improved Performance in Artificial Neural Networks", International Conference on Neural Networks, IEEE, Vol.1, pp.500-505.

REFERENCES NOT CITED

Tang, Zaiyong, Paul Fishwick, "Feedforward Neural Nets as Models for Time Series Forecasting", *ORSA Journal on Computing*, Vol.5, No. 4, pp374-385, 1993.

Lowe, D., A.R. Webb, "Time Series Prediction by Adaptive Networks: A dynamical Systems Perspective," *IEEE Proceedingd-F*, Vol. 138, No.1, February 1991.

Sudharsanan, S. I., M. K. Sundareshan,"Training of a Three Layer Dynamical Recurrent Neural Network for Nonlinear Input-Output Mapping," *IEEE ICNN*, Vol.2, pp.111-115, 1991

Widrow, Bernard, Michael A. Lehr,"30 Years of Neural Networks: Perceptron, madaline, and Backpropogation," *Proceedings of IEEE*, Vol.78, No.9, September 1990.

Lopez, VIncente, Ramon Huerta, Jose R. Dorronsoro, "Recurrent and Feedforward Polynomial Modelling of Coupled Time Series," *Neural Computation*, No.5, 795-811, 1993.

Mel, Bartlett W., Stephen M. Omohundro, "How Receptive Field Parameters Affect Neuarl Learning,"*Advances in Neural Information Processing Systems*, No.3, pp.757-776, 1993.