V. Petridis, Ath. Kehagias, L. Petrou, A. Bakirtzis, N. Maslaris, S. Kiartzis and H. Panagiotou.
"A Bayesian Multiple Models Combination Method for Time Series Prediction".

This paper has appeared in the journal:
Journal of Intelligent and Robotic Systems, vol. 31, pp.69-89, 2001.

A Bayesian Multiple Models Combination Method for Time Series Prediction

V. Petridis, A. Kehagias, L. Petrou, A. Bakirtzis, N. Maslaris, S. Kiartzis, H. Panagiotou

October 11, 2001

Abstract

In this paper we present the Bayesian Combined Predictor (BCP), a probabilistically motivated predictor for time series prediction. BCP utilizes local predictors of several types (e.g. linear predictors, artificial neural network predictors, polynomial predictors etc.) and produces a final prediction which is a weighted combination of the local predictions; the weights can be interpreted as Bayesian posterior probabilities and are computed online. Two examples of the method are given, based on real world data: (a) short term load forecasting for the Greek Public Power Corporation dispatching center of the island of Crete, and (b) prediction of sugar beet yield based on data collected from the Greek Sugar Industry. In both cases, the BCP outperforms conventional predictors.

1 Introduction

The problem addressed in this paper is the development of *modular* time series predictors. This is an example of a *multiple models* methodology applied to time series prediction.

In the last decade there has been great activity in the *machine learning* community for the development of "multiple models" methods. There is special interest in the development of clustering, classification, prediction and parameter estimation algorithms for time series ("dynamic") problems. Some remarkable efforts in this direction include *partition algorithms* [10, 19], *mixtures of experts* [5, 12, 13, 14, 15, 16, 25], *ensembles of neural networks* [3, 7, 26], *trees of neural networks* [17, 32], *threshold models* [35], *Takagi-Sugeno fuzzy models* [34], and much more. For an extensive bibliographical coverage see the books [22, 31].

The predictor architecture proposed in this paper is modular in the sense that it makes concurrent use of several alternative models of the same "process" (hence it is a multiple-models method); any one of the models can be replaced by an alternative model (which performs the same or a similar function) without requiring extensive modification (for instance retraining) of the remaining components (hence the total system is modular).

The models which comprise the proposed modular predictor, are themselves predictors. In fact, the general principle utilized is to build a complex predictor with superior predictive power from simpler component predictors which are easier to train. Each of the component predictors may have good predictive performance for a particular segment of the target time series. It is expected that, by incorporating an appropriate number of such *specialized* predictors, as well as a *combination* module which will activate the appropriate predictor at the apropriate time, one may obtain superior total performance. This approach has been used widely in prediction tasks (see for example [18, 28, 29, 30, 31]).

2 The Bayesian Combined Predictor

2.1 Introduction

We now present the *Bayesian Combined Predictor* (BCP). The BCP is based on probabilistic concepts, in particular on conditional probability and Bayes' rule. The original idea appears in [10, 19]; see also [18, 28, 29, 30] and for a more detailed exposition the book [31], where the original probabilistic formulation is expanded to include nonprobabilistic generalizations in the context of time series classification, prediction and parameter estimation.

The central ideas of BCP are the following.

1. Postulate several alternative predictors of a time series.
2. Obtain a recursive formula for computing the *posterior* probability of each predictor, based on the observable predictor error of the same predictor (as well as the errors of the remaining predictors). These predictors are also called *local models* of the time series, because each one may be valid for a particular section (or operating regime) of the time series.
3. Use the posterior probabilities to combine the predictions of the local models and so obtain a better, *global* predictor.

Let us then consider a time series x_t, t=1, 2, ... ; for simplicity we take x_t to be scalar, but extensions to vector valued time series are immediate. We assume the existence of K predictors of the general form

$$\hat{x}_t^k = f(x_{t-1}, ..., x_{t-M}; w_k), \qquad k = 1, 2, ..., K. \tag{1}$$

It can be seen from the above that the K predictors belong to a general family $f(\cdot, w)$, where w is a parameter vector; the k-th predictor is obtained by setting $w = w_k$. Let us now proceed to obtain a recursive formula for the posterior probability of each predictor.

2.2 Recursive Application of Bayes Rule

It is reasonable to assign higher posterior probability to predictors which are more succesful in predicting the actual observations x_1, x_2, ... This observation can be formalized as follows. Assume that the difference between the actual observation x_t and its k-th prediction $\hat{x}_t^k$ is a random variable e_t^k:

$$e_t^k = x_t - \widehat{x}_t^k.$$

It is reasonable to assume that, *if the k-th predictor is the one actually describing the evolution of the time series*, then the prediction errors e_t^k form a sequence of independent, identically distributed random variables with zero mean. In other words, if we denote mathematical expectation by $E(\cdot)$ then we have

$$E(e_t^k) = 0, \qquad E(e_t^k e_s^k) = \sigma^2 \cdot \delta(t, s),$$

where $\delta(t, s)$ is the Kronecker delta function. Let us also denote the probability density function of e_t^k by $g_k(\cdot)$ (independent of t). Now, for the probability density of $x_t - \hat{x}_t^k$, *conditional on the observations* x_1, x_2, ... , x_{t-1} we have

$$p(x_t - \hat{x}_t^k | x_1, x_2, ..., x_{t-1}) = g(x_t - \hat{x}_t^k) = g(x_t - f(x_{t-1}, ..., x_{t-M}; w_k)).$$

Now define a new stochastic process Z_t as follows: Z_t= k if at time t the correct model of x_t is $\widehat{x}_t^k$. Let us define p_t^k as follows

$$p_t^k = Pr(Z_t = k \ | x_1, ..., x_t).$$

Then, from Bayes' rule we obtain the following recursion (for details see [31])

$$p_t^k = \frac{p_{t-1}^k \cdot g(x_t - f(x_{t-1}, ..., x_{t-M}; w_k))}{\sum_{j=1}^K p_{t-1}^j \cdot g(x_t - f(x_{t-1}, ..., x_{t-M}; w_j))}. \quad (2)$$

Eq.(2) is the required recursion for the posterior probability of Z_t, i.e. for the probability of the k-th predictor being the correct one at time t. The validity of this formula depends on our assumptions, namely that (a) the time series is produced by one of the K models of eq.(1) and (b) the prediction error is white noise. When these assumptions hold, eq.(2) expresses the probability that model k actually generates the observed load data; this probability is conditional, dependent on observations up to time t.

There is an alternative, nonprobabilistic interpretation of the p_t^k's, which will become clearer if we temporarily assume a specific form for the density function $g_k(\cdot)$. Assume then that, for all k, e_t^k is zero mean and Gaussian, i.e.

$$g_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp(-\frac{|x|^2}{2\sigma_k^2}). \quad (3)$$

Substituting eq.(3) in eq.(2) we obtain

$$p_t^k = \frac{p_{t-1}^k \cdot \exp(-\frac{|x_t - f(x_{t-1},...,x_{t-M};w_k)|^2}{2\sigma_k^2})}{\sum_{j=1}^K p_{t-1}^j \cdot \exp(-\frac{|x_t - f(x_{t-1},...,x_{t-M};w_j)|^2}{2\sigma_j^2})}. \quad (4)$$

From eq.(4) becomes clear that three factors determine the value of p_t^k.

1. First, there is the absolute value of the current prediction error $|x_t - f(x_{t-1}, ..., x_{t-M}; w_k)|$. When this is large, then the negative exponential results in decrease of p_t^k.

2. However, a second factor which must be taken in account is the previous predictive performance of the k-th predictor, which is reflected in p_{t-1}^k. If this is large (which means close to unity), a temporarily large prediction error will not have a drastic effect on p_t^k. This corresponds to a certain stability in the computation of p_t^k: temporary stochastic fluctuations will not have a catastrophic effect on p_t^k if the k-th predictor performs well on the average.

3. Finally, and quite importantly, in the computation of p_t^k what counts is not the *absolute* predictive accuracy of the k-th predictor, but the relative one. In other words, even if the product

$$p_{t-1}^k \cdot \exp(-\frac{|x_t - f(x_{t-1}, ..., x_{t-M}; w_k)|^2}{2\sigma_k^2}) \quad (5)$$

   is small, p_t^k can still remain relatively close to one, if the respective products for other predictors are even smaller. In other words, if the k-th predictor performs poorly, but relatively better than the remaining predictors, it will maintain a high posterior probability.

We have illustrated the importance of the three factors by assuming that e_t^k is Gaussian, but, in fact, the same conclusions would hold for any reasonable probability density. So one can consider the BCP algorithm as a heuristic credit assignment scheme: the model that best forecasts the observed load data is the one with highest credit (and so with highest conditional probability, under the Bayesian interpretation).

2.3 Bayesian Combination of Predictors

Having obtained the p_t^k's, there are several ways in which they can be used to obtain an improved prediction of x_t^k. We indicate two possibilities.

1. *Maximum Likelihood Prediction.* This is defined as follows

$$\hat{x}_t^k = \hat{x}_t^{k*}, \qquad \text{where } k^* = \arg \max_{1 \le k \le K} p_t^k.$$

   In other words, at every time step we use the forecast of the model with maximum posterior probability; since this model is most likely to have produced the load time series, it must on the average have smaller forecast error. This is the *maximum likelihood* prediction.

2. *Weighted Prediction.* Here $\widehat{x}_t^k$ is computed as a weighted sum of all available predicitons:

$$\hat{x}_t^k = \sum_{k=1}^{K} \hat{x}_t^k p_t^k.$$

   In fact, $\hat{x}_t^k$ as defined above is the conditional expectation of x_t, and it is well known [4] that this is the prediction with minimum mean square error.

There are some practical issues to be considered in connection to the combination of predictor, which are important for the implementation of both the ML and weighted combination scheme; these are discussed in Section 2.4.

2.4 Implementation Issues

The main ideas of our prediction scheme have been presented in the previous paragraphs. The basic components are:

1. A collection of *local* predictors, each suited to predicting a particular portion (or *operating regime*) of the target time series.

2. A scheme for recursively updating the posterior probabilities of these predictors, based on their predictive accuracy.

Up to this point we have not discussed the form or derivation of the local predictors. Not much needs to be said, actually. The probability update scheme is independent of the predictors, which means that various different predictor types can be used, for instance linear regressors, neural networks, fuzzy systems and so on. In each case, the main issue is identifying appropriate data to be used for training the predictors. The actual training will be performed using the appropriate method for the particular class of predictors; for instance if the predictors are neural networks, then the Back Propagation algorithm is appropriate. Finding the appropriate training data presupposes that a *labeled* data set is available, i.e. our method is a *supervised learning* method.

The practical implementation of predictor combination raises some computational issues. For example, perusal of eq.(4) reveals that, in case p_t^k becomes equal to zero for some value t_0, then we will also get $p_t^k = 0$ for all values $t > t_0$ as well. Now, theoretically, eq.(4) may result in an arbitrarily small value of p_t^k, but never equal to zero. However, as a practical issue, computer underflow may result in $p_t^k = 0$. This problem can be rectified by using a *threshold* h (where h is a number close to zero); whenever p_t^k falls below a specified threshold h, it is reset to h. Then the usual normalization of the

p_t^k's is performed; this ensures that the thresholded p_t^k's remain approximately within the $[h, 1]$ range and add to 1.

In essence, this thresholding is equivalent to introducing a forgetting factor : suppose that several samples of the time series are observed, such that predictor k produces a large error; if this process is continued for several time steps, p_t^k will eventually become zero. If we never let p_t^k go below h, we essentially stop penalizing predictor k for further bad predictions; these are, in effect, "forgotten". If h is small, then p_t^k will also be small and will not essentially alter the classification results, while, if the time series enters a regime of operation which is best described by the k-th predictor, this will still be in the position of becoming active.

In addition to thresholding, an important practical matter is the selection of the probability density $g_k(\cdot)$. This entails choosing a functional form and its parameters. As a practical issue, we usually choose a Gaussian, zero-mean density, so that our posterior probability update equation is eq.(5). The only parameter that remains to be determined is the standard deviation σ_k, k=1,2,...,K. This we compute in a standard manner, taking it equal to the root mean square error of the k-th predictor, which has been computed in the training phase.

3 Example: Short Term Load Forecasting

In this section we present an application of the BCP. Namely we consider the problem of short term load forecasting for the electrical power system of the island of Crete, Greece. In the summer of 1994 this system had a peak load of about 300 MW; power is supplied by the Greek Public Power Corporation (PPC). The data used in this example correspond to the period from 1989 to 1994.

3.1 Description of the Problem

The problem consists in predicting a vector time series. In other words, we are given a sequence y_t, $t = 1, 2, ...$, where for each t y_t has dimensions 24×1; each of the y_t components corresponds to the load of a particular hour of the day on day t. The predictors must have the general form $y_t = f(y_{t-1}, y_{t-2}, ..., y_{t-N})$, in other words one may use data from N days from the past load history. At midnight of day $t-1$ it is required to provide a prediction for the 24 hours of day t. This prediction will have practical implications for scheduling the power generators to be activated in the following working day.

The hourly load time series has several intereting features. Typical load for a winter and a summer day are presented in Figure 1. It can be seen that there is a daily variation in the load, which has a somewhat different structure in winter and summer periods.

Figure 1: Two representative daily loads.

It should be remarked that the formulation of economic, reliable and secure operating strategies for a power system requires accurate *short term load forecasting* (STLF). The principal objective of STLF is to provide load predictions for the basic generation scheduling functions, the security assessment of a power system and for the dispatcher's information.

3.2 Previous Work

A large number of computational techniques have been used for the solution of the STLF problem

Statistical STLF models can be generically separated into regression models [6] and time series models [37]; both can be either static or dynamic. In static models, the load is considered to be a linear combination of time functions, while the coefficients of these functions are estimated through linear regression or exponential smoothing techniques [6]. In dynamic models weather data and random effects are also incorporated since autoregressive moving average (ARMA) models are frequently used. In this approach the load forecast value consists of a deterministic component that represents load curve periodicity and a random component that represents deviations from the periodic behavior due to weather abnormalities or random correlation effects. An overview of different statistical approaches to the STLF problem can be found in [8]. The most common (and arguably the most efficient) statistical predictors apply a linear regression on past load and temperature data to forecast future load. For such predictors, we will use the generic term Linear Regression (LR) predictors.

The application of artificial neural networks to STLF yields encouraging results; a discussion can be found in [23]. The ANN approach does not require explicit adoption of a functional relationship between past load or weather variables and forecasted load. Instead, the functional relationship between system inputs and outputs is learned by the network through a training process. Once training has been completed, current data are input to the ANN, which outputs a forecast of tomorrow's hourly load. One of the first neural-network- based STLF models was a three-layer neural network used to forecast the next hour load [24]. A minimum-distance based identification of the appropriate historical patterns of load and temperature used for the training of the ANN has been proposed in [27], while both linear and non-linear terms were adopted by the ANN structure. Due to load curve periodicity, a non-fully connected ANN consisting of one main and three supporting neural networks has been used to incorporate input variables like the day of the week, the hour of the day and temperature. Various methods were proposed to accelerate the ANN training [11], while the structure of the network has been proved to be system depended [2, 20].

Hybrid neuro-fuzzy systems applications to STLF have appeared recently. Such methods synthesize fuzzy-expert systems and ANN techniques to yield impressive results, as reported in [1, 33].

Each of the methods discussed above has its own advantages and shortcomings. Our own experience is that no single predictor type is universally best. For example, an ANN predictor may give more accurate load forecasts during morning hours, while a LR predictor may be superior for evening hours.

Hence, a method that combines various different types of predictors may outperform any single "pure" predictor of the types discussed above. It is clear that the BCP is just such a combination method, hence it is reasonable to apply the BCP methodology to the task at hand. In Section 3.3 we will present the details of the particular BCP implementation; in Section 3.4 we will present our experimental results which show that BCP has better performance than any of the "pure" predictors.

3.3 BCP Implementation

In this section we present the implementation details for three types of "pure" predictors, namely two linear regression predictors and one neural predictor. Then we present the implenetation details for the combination module.

3.3.1 "Long Past" Linear Regression

This predictor performs a straightforward linear regression on two time series: daily loads (for a given hour of the day) and maximum daily temperature. There are $M+N$ inputs, where M is the number of past loads (for the given hour of the day) and N is the number of past temperatures used. Several values of M, between 21 and 56, have been employed. This means we use data from the last 21 to 56 days; hence the designation "long past". (The best value turned out to be 35.) Output is tomorow's load

for the given hour. Hence, for a complete 24-hour load forecast, we need 24 separate predictors. The regression coefficients are determined by least square error training; this is achieved using a standard matrix inversion routine, which takes less than one sec on a MS Windows PC. The training phase is performed only once, offline. It should also be mentioned that the hourly load data were analysed and "irregular days", such as national and religious holidays, major strikes, election days, etc, were excluded from the training data set and replaced by equivalent regular days; of course this substitution was performed only for the training data. Training utilized load and temperature data for the years 1992 and 1993. Training error (computed as the ratio of forecast error divided by the actual load, averaged over all days and hours of the training set) was 2.30%. It must be mentioned that there was a "ceiling" effect as to the possible reduction of forecast error. While training error could be reduced below 2.30% by the introduction of more regression coefficients, this improvement was not reflected in the test error. This is the familiar "overfitting" effect.

3.3.2 "Short Past" Linear Regression

This is very similar to the previous method. Again, it utilizes straightforward linear regression on the time series of loads; but now loads of all hours of the day are used as input., in addition to maximum and minimum daily temperature. There are $(24M + 2N)$ inputs, where M is the number of past loads (for all hours of the day) and N is the number of past temperatures used. Several values of M, between 1 and 8, have been employed. We have found that the best value of M is 4, which means data from four past days are used. For a given forecast day, we use the two immediately previous days and the same weekday of the previous two weeks.; hence this predictor uses a relatively "short past", as compared to the one of Section 3.3.1. Output is tomorrow's load for every hour of the day. The regression coefficients are determined by least square error training; this is achieved using a standard matrix inversion routine, which takes less than one sec on a MS Windows PC. The remarks of Section 3.3.1 on training and overfitting apply here as well. Training error (computed as the ratio of forecast error divided by the actual load, averaged over all days and hours of the training set) was 2.36%.

3.3.3 Neural Network Prediction

A fully connected three layer feedforward ANN was used in this method. The ANN comprises of 57 input neurons, 24 hidden neurons and 24 output neurons representing next day's 24 hourly forecasted loads. The first 48 inputs represent past hourly load data for today and yesterday. Inputs 49-50 are maximum and minimum daily temperatures for today. The last seven inputs, 51-57, represent the day of the week, e.g. Monday is encoded as 1000000, Tuesday as 0100000 and so on . Other input variables were also tested but they did not improve the performance of our model. The ANN was trained by being presented with a set of input-desired output patterns until the average error between the desired and the actual outputs of the ANN over all training patterns is less than a predefined threshold. The well known back propagation algorithm [9] was used for the ANN training. The hourly load data were carefully analysed and all "irregular days", such as national and religious holidays, major strikes, election days, etc, were excluded from the training data set. Special logic for the treatment of missing data has also been incorporated in the data analysis software. The training data set consists of $90+4\times30=210$ input/output patterns created from the current year and the four past years historical data as follows: 90 patterns are created for the 90 days of the current year prior to the forecast day. For every one of the 4 previous years, another 30 patterns are created around the dates of the previous years that correspond to the current year forecast day. Initial offline training takes a few seconds on a MS Windows PC. The ANN parameters are then updated online, on a daily basis through the following procedure. A new round of ANN training is performed on the most recent input/output patterns;

the ANN parameters are initialized to those of the previous day. Since the training data sets of two consecutive days differ by only a few patterns, daily model parameter updating is very efficient. Online training takes between 1 and 3 secs per day. The network is trained until the average error becomes less than 2.5%. It was observed that further training of the network (to an error 1.5% for example) did not improve the accuracy of the forecasts. Training of the ANN to a very small error may result in data overfitting.

3.3.4 The Combination Module

The implementation of the combination is straightforward. We use a Gaussian probability density function and we set $\sigma_k = \sigma$, i.e. identical for all predictors; σ is computed from the training phase. We also use a threshold $h = 0.01$.

3.4 Results

We applied the BCP described in Section 3.3 to the prediction of loads for the period July 1st, 1994 to September 30th, 1994. In Figure 2 we see a comparison of the prediction error for the local predictors as well as for the BCP. The n-th point of each curve in Figure 2 (with $n = 1, 2, ..., 24$) corresponds to the average (over the entire three month test period) prediction error for the 24-th hour of the day, i.e.

$$E_n = \sum_{t=1}^{91} \frac{|x_{t,n} - \widehat{x}_{t,n}|^2}{|x_{t,n}|^2}$$

where the index n corresponds to the hour in question. The final, 25th point represents the average daily error (i.e. averaged over all 24 hours), i.e.

$$E = \frac{\sum_{n=1}^{24} E_n}{24}.$$

Figure 2: Comparative prediction errors for local and combined predictors.

It can be seen that the BCP predictor not only outperforms all local predictors on the average, but usually also outperforms them on individual hours (with a few exceptions). In this connection, it is quite instructive to observe the evolution of the posterior probabilities of the three local predictors for two different hours. In Figure 3 we plot the evolution of the posteriors for the hour 1am and in Figure 4 for the hour 1pm. The reader will see that in Figure 3 the highest probability is generally assigned to the SP LR predictor, even though over short time intervals one of the other two local predictors may outperform it. Similarly, in Figure 4 the highest probability is generally assigned to the LP LR predictor, even though over short time intervals one of the other two predictors may outperform it. These results are consistent with the general results of Figure 2; the additional information presented in Figures 3 and 4 is that a predictor that generally performs poorly, may still outperform its competitors over short time intervals; in such cases the BCP will take this improved performance into account,as evidenced by the adaptively changing posterior probabilities. This explains why the BCP is generally better than the best pure predictor.

Figure 3: Evolution of posterior probabilities.

Figure 4: Evolution of posterior probabilities.

4 Example: Sugar Beet Yield Prediction

In this section we describe an application of the BPC to sugar beet yield prediction. Accurate yield predictions are required by the Greek Sugar Industry (GSI) for the preparation of an optimal sugar beet harvesting plan.

4.1 Description of the Problem

The time series used in this example are part of a large data set maintained by the GSI. In particular, every year the GSI collects measurements from a large network of pilot farms, covering all of Greece. The quantities measured for each pilot farm include sugar beet characteristics (sugar content, average weight of the various parts of sugar beet plant), soil characteristics (concentration of various elements such as natrium, calcium etc.) as well as weather data (temperature etc.). All of these quantities are measured at more or less regular intervals, approximately once every ten days.

The main quantity of interest is sugar content and the final goal is to harvest the sugar beets at about the time when they achieve maximum sugar concentration. This time must be predicted in advance because harvesting a large number of geographically dispersed farms requires advance planning.

Hence the prediction task is to develop predictors for various time series related to the sugar beet crop. We approach this task by the use of several BCP's, which combine various local predictors. In this case the predictors are literally local, in the sense that each one is developed using data from a single geographical region. In the example presented here we have utilized a relatively small part of the original dataset. Specifically, we have concentrated on the POL (sugar concentration), WPOL (sugar concentration multiplied with plant root weight) and QR (ratio of POL to natrium concentration) time series. These quantities are particularly important for scheculing the sugar beet harvest.

4.2 Previous Work

There is relatively little work in applying time series prediction methods to the problem of sugar beet harvesting. For background material see [21] and [38]. For a method utilizing neural network prediction, see [36]. The methods presented in this paper are currently applied at the SGI and there is an ongoing project for more extensive experimentation.

4.3 BCP Implementation

4.3.1 The Data

We have used data from the years 1989-1991 for training and from the years 1992-1994 for testing. For the 1989-1991 period there is a total of 26 pilot farms, resulting in $26\times3= 78$ training time series. For the 1989-1991 period there is a total of 11 pilot farms, resulting in $11\times3= 33$ testing time series. Every time series corresponds to a period of approximately eight months and sampling takes place at approximately every ten days. The actual (calendar) time of the first and last measurement varies for each pilot farm. In order to simplify the prediction task, we arbitrarily set the time of the first measurement as t=1 and obtain interpolated time series which evolve in discrete time t=1, 2, ... , T, where every time unit corresponds to one day. In this manner we obtain 78 training time series. Each time series has a *different length*, with average length being around 150 time steps. Some representative time series are presented in Figure 5.

Figure 5.a: POL time series in farm 511 for the years 1992-1994.

Figure 5.b: WPOL time series in farm 511 for the years 1992-1994.

Figure 5.c: QR time series in farm 511 for the years 1992-1994.

The main difficulty in developing local predictors for the yield prediction problem is deciding how to group the data. We have used two particularly simple grouping strategies: a "unary" strategy, where one predictor is trained for each pilot farm, and a "geographical" strategy, where one predictor is trained for each geographical region.

4.3.2 The Predictors

In what follows we will denote the POL time series by x_t, the WPOL time series by y_t and the QR time series by u_t. Respective predictors will be denoted by $\widehat{x}_t^k$, $\widehat{y}_t^k$, $\widehat{u}_t^k$. For example, a POL predictor will have the form

$$\widehat{x}_{t+T}^k = f_k(x_t, y_t, z_t, x_{t-1}, y_{t-1}, z_{t-1}, ...).$$

Note the presence of a *prediction horizon* T, which will generally be greater than 1. In fact, T, should be as large as possible, provided that the prediction error remains within reasonable bounds. In this study we have experimented with prediction horizon T equal to 10, 15, 20, 25 days. For each data grouping strategy we have developed a variety of predictors. More specifically, we use predictors of the following types.

Interpolation Predictors. In this case every predictor is an "average" time series, e.g. at time t we obtain $\hat{y}_t$ as the average of the y_t's of the time series belonging at the corresponding data group. In this case "training" a predictor for a particular time series depends only on that particular time series, e.g. the POL predictor does not utilize the WPOL and QR data

Polynomial Predictors. These are polynomials in t, i.e. the time variable. Specifically, we use

$$\widehat{x}_{t+T}^k = a_0^k + a_1^k t + a_2^k t^2 + a_3^k t^3,$$

i.e. third degree polynomials. In this case, too, "training" depends only on the POL time series. For every data group, the a coefficients are obtained by least squares regression.

Linear Regression Predictors These are linear regression models. We have used both single input (autoregression) predictors, e.g.

$$\widehat{x}_{t+T}^k = a_0^k x_t + a_1^k x_{t-1} + a_1^k x_{t-2} + ... + a_M^k x_{t-M}.$$

and multi-input predictors of the form

$$\widehat{x}_{t+T}^k = a_0^k x_t + b_0^k y_t + c_0^k u_t + a_1^k x_{t-1} + b_1^k y_{t-1} + c_1^k u_{t-1} + ... + a_M^k x_{t-M} + b_M^k y_{t-M} + c_M^k u_{t-M}$$

Training of the predictors consists in obtaining the a, b, c coefficients using a least squares approximation method.

Neural predictors This is similar to linear regression, except that we use a nonlinear regression implemented by feedforward, sigmoid neural networks of the form

$$\widehat{x}^k_{t+T} = f(x_t, x_{t-1}, ..., x_{t-M}; w_k)$$

or of the form

$$\widehat{x}^k_{t+T} = f(x_t, x_{t-1}, ..., x_{t-M}, y_t, y_{t-1}, ..., y_{t-M}, u_t, u_{t-1}, ..., u_{t-M}; w_k).$$

Here w_k is a matrix of *weights* , i.e. parameters which are determined by the Back Propagation training algorithm. Of course, $f(\cdot; w)$ is a sigmoid function.

The Combination Module The Bayesian combination module is very similar to the one presented in Section 3.

4.4 Results

By choosing a particular target time series, a data grouping method, predictor types, and a combination method we fully determine a prediction experiment. In what follows we organize the presentation of our results into experiment groups, each group pertaining to a particular target time series, i.e. POL, WPOL and QR. For each experiment group we summarize our results in a three part figure. Figures 6.a, 6.b and 6.c correspond to the POL time series, figures 7.a, 7.b and 7.c correspond to the WPOL time series and Figures 8.a, 8.b and 8.c correspond to the QR time series,

Prediction errors are computed for the test period being the second half of each time series and using two different types of combination (weighted combination and maximum likelihood combination); in addition the prediction error is computed for the 20 *"hot"* days of the harvest period, using the weighted combination method. In every case we present combinations of four prediction horizons (T= 10, 15, 20, 25 days) and several local predictors (interpolation, polynomial, single input linear, multi-input linear, single input neural, multi-input neural); an additional parameter is the data grouping method used (unary or geographical). Finally, it should be mentioned that the prediction error is computed as relative error, i.e. by the formula

$$E_n = \sum_{t=1}^{T_{fin}} \frac{|x_t - \widehat{x}_t|^2}{|x_t|^2}$$

for the POL time series and by similar formulas for the WPOL and QR time series. The index n refers to the particular combination of predictors, prediction horizon and prediction combination method.

Figure 6.a: POL time series: relative prediction errors for various prediciton horizons and different types of local predictors. Time series: second half of the 1992-1994 years. Prediction combination method: weighted.

Figure 6.b: POL time series: relative prediction errors for various prediciton horizons and different types of local predictors. Time series: second half of the 1992-1994 years. Prediction combination method: maximum likelihood.

Figure 6.c: POL time series: relative prediction errors for various prediciton horizons and different types of local predictors. Time series: 20 hot days. Prediction combination method: weighted.

Figure 7.a: WPOL time series: relative prediction errors for various prediction horizons and different types of local predictors. Time series: second half of the 1992-1994 years. Prediction combination method: weighted.

Figure 7.b: WPOL time series: relative prediction errors for various prediciton horizons and different types of local predictors. Time series: second half of the 1992-1994 years. Prediction combination method: maximum likelihood.

Figure 7.c: WPOL time series: relative prediction errors for various prediciton horizons and different types of local predictors. Time series: 20 hot days. Prediction combination method: weighted.

Figure 8.a: QR time series: relative prediction errors for various prediciton horizons and different types of local predictors. Time series: second half of the 1992-1994 years. Prediction combination method: weighted.

Figure 8.b: QR time series: relative prediction errors for various prediciton horizons and different types of local predictors. Time series: second half of the 1992-1994 years. Prediction combination method: maximum likelihood.

Figure 8.c: QR time series: relative prediction errors for various prediciton horizons and different types of local predictors. Time series: 20 hot days. Prediction combination method: weighted.

4.4.1 Discussion

We see that weighted prediction combination generally gives better results than maximum likelihood prediction. Prediction accuracy is very good for the POL time series and quite good for the WPOL time series (which is the series of main interest). Results are not so good for the QR time series. Linear regression prediction gives the best overalll results. We conjecture that the neural predictors would perform better if more data were available.

5 Conclusion

We have presented a probabilistically motivated method of time series prediction, namely the *Bayesian Combined Predictor*. The Bayesian Combined Predictor is a modular architecture consisting of *local* predictors, i.e. each predictor is specialized in predicting a particular portion of the time series to be predicted. As a result, each predictor is quite accurate for a specific regime of the time series, and in particular is more accurate than a "global" predictor. In addition, the local predictors can be of various different types, e.g. linear, neural and fuzzy predictors can be combined. The combination of the various predictors is effected by use of Bayes' rule and produces a final prediction which is a weighted combination of the local predictions; the combination coefficients are the Bayesian posterior probabilities of the various predictors and are computed recursively, making our method suitable for online implementation. We have demonstrated the utility of our approach on two real world problems and we have seen that the combined predictor outperforms the local ones. It should be noted that, while our method is based on probabilistic principles, it can be extended into nonprobabilistic contexts.

References

[1] A. Bakirtzis, J. Theocharis, S. Kiartzis and K. Satsios, "Short Term Load Forecasting Using Fuzzy Neural Networks", paper 95 WM 155-2-PWRS presented at the IEEE/PES 1995 Winter Meeting.

[2] A. Bakirtzis, V. Petridis, S. Kiartzis, M. Alexiadis and A. Maissis, "A Neural Network Short Term Load Forecasting Model for the Greek Power System", presented at the IEEE/PES 1995 Summer Meeting.

[3] Baxt, W.G. "Improving the accuracy of an artificial neural network using multiple differently trained networks". *Neural Computation*, 1992, vol. **4**, pp.135-144.

[4] P. Billingsley, *Probability and Measure*, Wiley, 1986.

[5] T.W. Cacciatore and S.J.Nowlan. "Mixtures of Controllers for Jump Linear and Nonlinear Plants", in *Advances in Neural Information Processing Systems 6 (NIPS 93)*, eds. J.D. Cowen, G. Tesauro and J. Alspector, pp. 719-726, San Francisco, CA, Morgan Kaufmann, 1994.

[6] W.R. Christiaanse, "Short term load forecasting using general exponential smoothing," *IEEE Trans. Power App.*

[7] H. Drucker et al., "Boosting and other ensemble methods", *Neural Computation*, 1994, vol.6, pp.1289-1301.

[8] G. Gross and F.D. Galiana, "Short term load forecasting," *Proc. IEEE*, vol. 75, pp. 1558-1573, 1987.

[9] J. Hertz, A. Krogh and R.G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Redwood City, 1991.

[10] C.G. Hilborn and D.G. Lainiotis, "Optimal Estimation in the Presence of Unknown Parameters", *IEEE Trans. on Systems, Man and Cybernetics*, 1969, vol. 5, pp. 38-43.

[11] K.L. Ho, Y. Y. Hsu, and C. C. Yang, "Short Term Load Forecasting Using a Multilayer Neural Network with an Adaptive Learning Algorithm," *IEEE Trans. on Power Systems*, vol. 7, pp. 141-149, 1992.

[12] R.A. Jacobs, M.I. Jordan, S.J. Nowlan and G.E. Hinton, "Adaptive mixtures of local experts", *Neural Computation,* 1991, vol.3, pp.79-87.

[13] R.A. Jacobs and M.I. Jordan, "Linear piecewise control strategies in modular neural network architectures", *IEEE Trans. on Systems, Man and Cybernetics*, 1993, vol.23, pp.337-345.

[14] M.I. Jordan and R.A. Jacobs, "Hierarchies of Adaptive Experts", in *Neural Information Processing Systems 4*, 1992, J. Moody, S. Hanson and R. Lippman (eds.), San Mateo, CA, Morgan Kaufmann.

[15] M.I. Jordan and R.A. Jacobs, "Hierarchical Mixtures of Experts and the EM Algorithm", *Neural Computation*, vol.6, 1994, pp. 181-214.

[16] M.I. Jordan and L. Xu, "Convergence results for the EM approach to mixtures of experts architectures", *Neural Networks*, 1995, vol.8, pp.1409-1431.

[17] Kadirkamanathan, V. & Niranjan, M. "Application of an architecurally dynamic neural network for speech pattern classification". *Proc. of the Inst. of Acoustics*, 1992, vol. **14**, 343-350.

[18] A. Kehagias and V. Petridis. "Predictive Modular Neural Networks for Time Series Classification", *Neural Networks*, 1997, vol.10, pp.31-49.

[19] D.G. Lainiotis, "Adaptive Estimation and Structure Identification", *IEEE Trans. on Automatic Control*, , 1971, vol. 16, pp. 160-170.

[20] C.N. Lu, H.T. Wu and S. Vemuri, "Neural Network Based Short Term Load Forecasting," *IEEE Trans. on Power Systems, 1993,* vol. 8, pp. 336-342.

[21] N. Maslaris, P. Christodoulou, G. Zountsas, "Quantitative effect of root/leaf growth rate on root quality parameters in sugarbeet", Proceedings of the 20th General Assembly of the C.I.T.S., Munich, pp. 85-93

[22] R. Murray-Smith and T. Johansen, *Multiple model approaches to modeling and control*, Taylor and Francis, 1997.

[23] D. Niebur et al., "Artificial neural networks for Power Systems", CIGRE TF38.06.06 Report, ELECTRA, 1995, pp. 77-101.

[24] D.C. Park, M.A. El-Sharkawi, R. J. Marks, L.E. Atlas, and M.J. Damborg, "Electric Load Forecasting Using an Artificial Neural Network," *IEEE Trans. on Power Systems*, 1991, vol. 6, No. 2, pp. 442-449.

[25] K. Pawelzik, J. Kohlmorgen and K.-R. Muller, "Annealed competition of experts for a segmentation and classification of switching dynamics", *Neural Computation*, 1996, vol.8, pp.340-356.

[26] Perrone, M.P. & Cooper, L.N. (1993). "When networks disagree: ensemble methods for hybrid neural networks". In *Neural Networks for Speech and Image Processing*, ed. R.J. Mammone. New York: Chapman-Hall.

[27] T.M. Peng, N.F. Hubele, and G. G. Karady, "Advancement in the Application of Neural Networks for Short Term Load Forecasting," *IEEE Trans. on Power Systems*, 1992, vol. 7, pp. 250-258.

[28] V. Petridis and A. Kehagias, "A Recurrent Network Implementation of Time Series Classification", *Neural Computation*, 1996, vol. 8, pp. 357-372.

[29] V. Petridis and A. Kehagias, "Modular neural networks for Bayesian classification of time series and the partition algorithm", *IEEE Trans. on Neural Networks*, 1996, vol.7, pp.73-86.

[30] V. Petridis and A. Kehagias, "Predictive modular fuzzy systems for time-series classification", *IEEE Trans. on Fuzzy Systems*, 1997, Vol.5, 381-397.

[31] V. Petridis and Ath. Kehagias, *Predictive Modular Neural Networks: Time Series Applications*, Kluwer, 1998.

[32] R.S. Shadafan and M. Niranjan. "A dynamic neural network architecture by sequential partitioning of the input space", *Neural Computation*, 1994, vol. **6**, 1202-1222.

[33] D. Srinivasan, C.S. Chang and A.C. Liew, "Demand Forecasting Using Fuzzy Neural Computation, with special emphasis on weekend and public holiday forecasting", paper 95 WM 158-6-PWRS presented in IEEE/PES 1995 Winter Meeting.

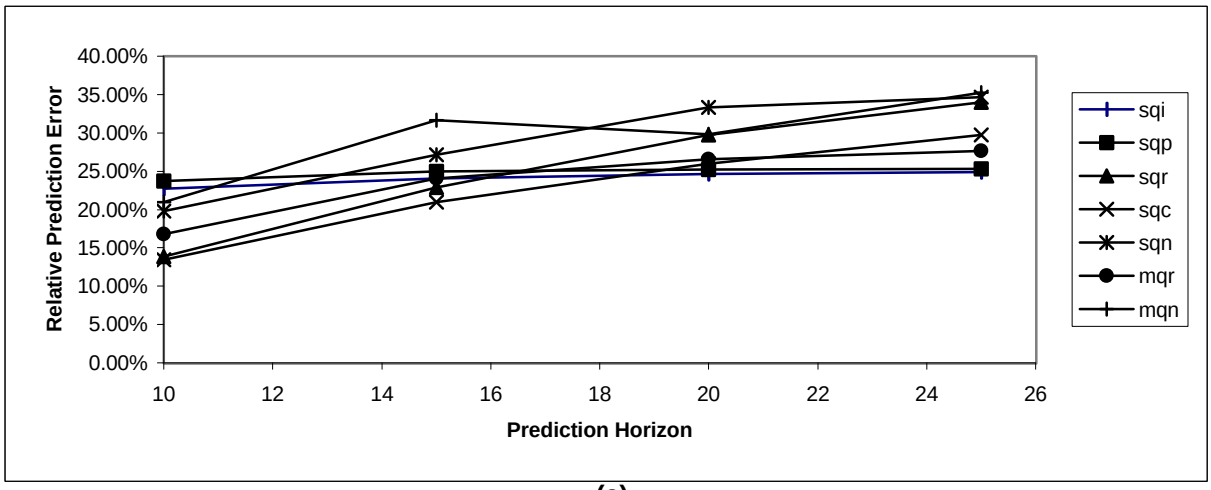
[34] M. Sugeno and T. Yasukawa, "A Fuzzy logic-based approach to qualitative modeling", *IEEE Trans. on Fuzzy Systems*, 1993, vol.1, pp. 7-32.

[35] Tong, H. and K.S.Lim. "Threshold autoregression, limit cycles and cyclical data", *Journal of the Royal Stat. Soc., Part B,* 1980, vol. 42, pp.245–292.

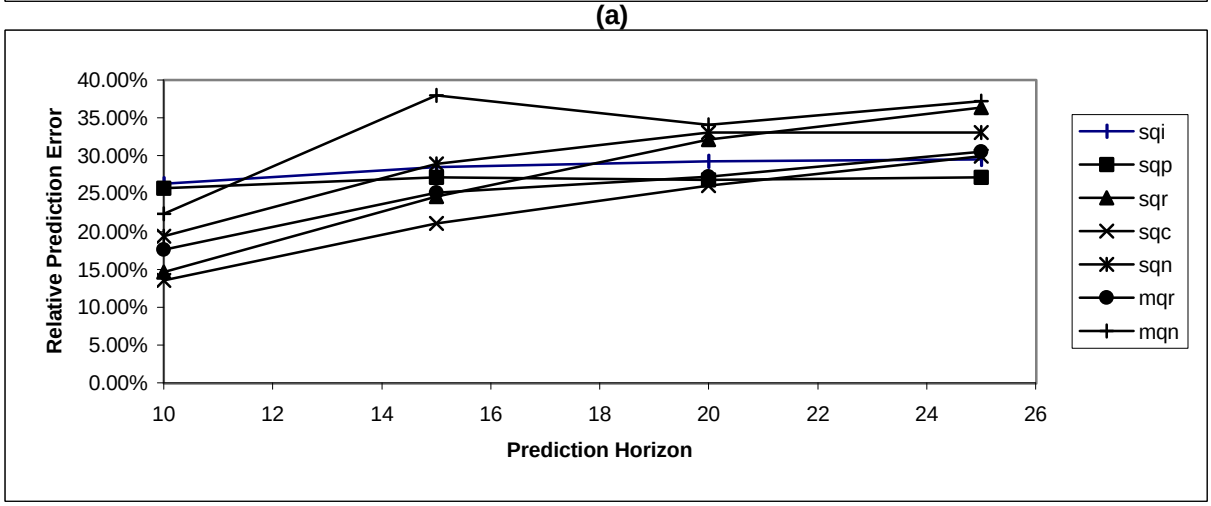
[36] G. Stoicos, Sugar beet yield prediction with artificial neural networks, In *Modern Automatic Control Technologies*, Technical Chamber of Greece, Athens, 1995.

[37] S. Vemuri, W. L. Huang and D. J. Nelson, "On-Line Algorithms for Forecasting Hourly Loads of an Electric Utility," *IEEE Trans. Power App. & Syst.*, 1981, vol. 100, pp. 3775-3784.

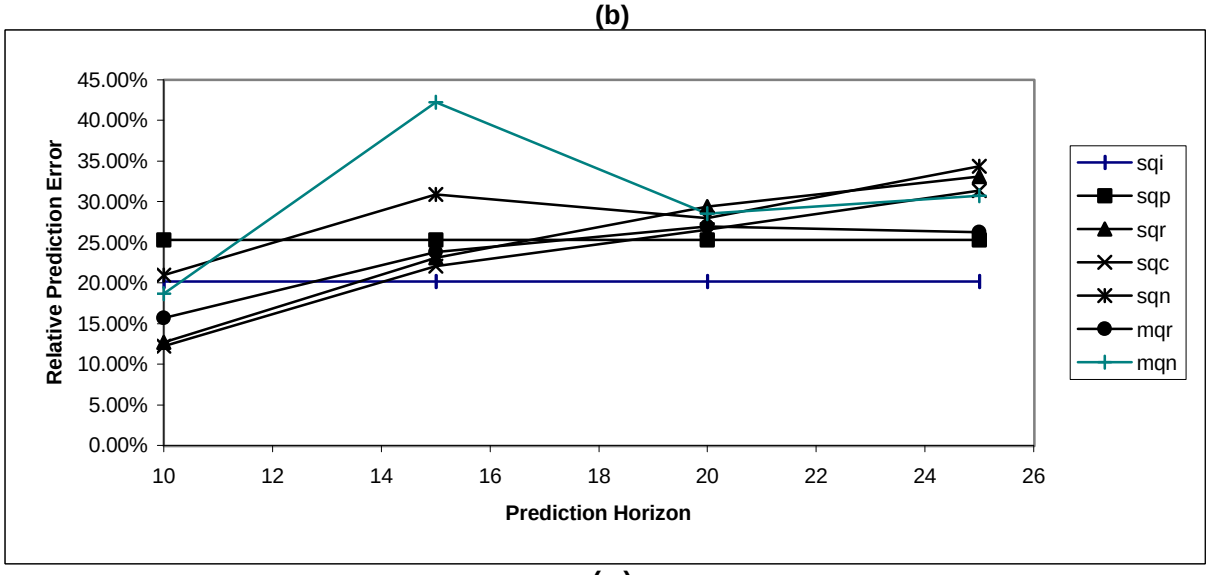
[38] *The Nitro-test Leaf Diagnostic*, Internal Report, Greek Sugar Industry.



(a)



(b)



(c)

Figure 8. (a) Prediction error for the second half of QR time series, using max likelihood prediction.
(b) Prediction error for the second half of the QR time series using weighted prediction.
(c) Prediction error for the 20 hot days of the QR time series, using weighted prediction.

Legend: sqi: single input interpolation; sqp: single input polynomial; sqr: single input regression; sqc: single input regression (geog. grouping); sqn: single input neural; mqr: multiinput regression; mqn: multiinput neural.

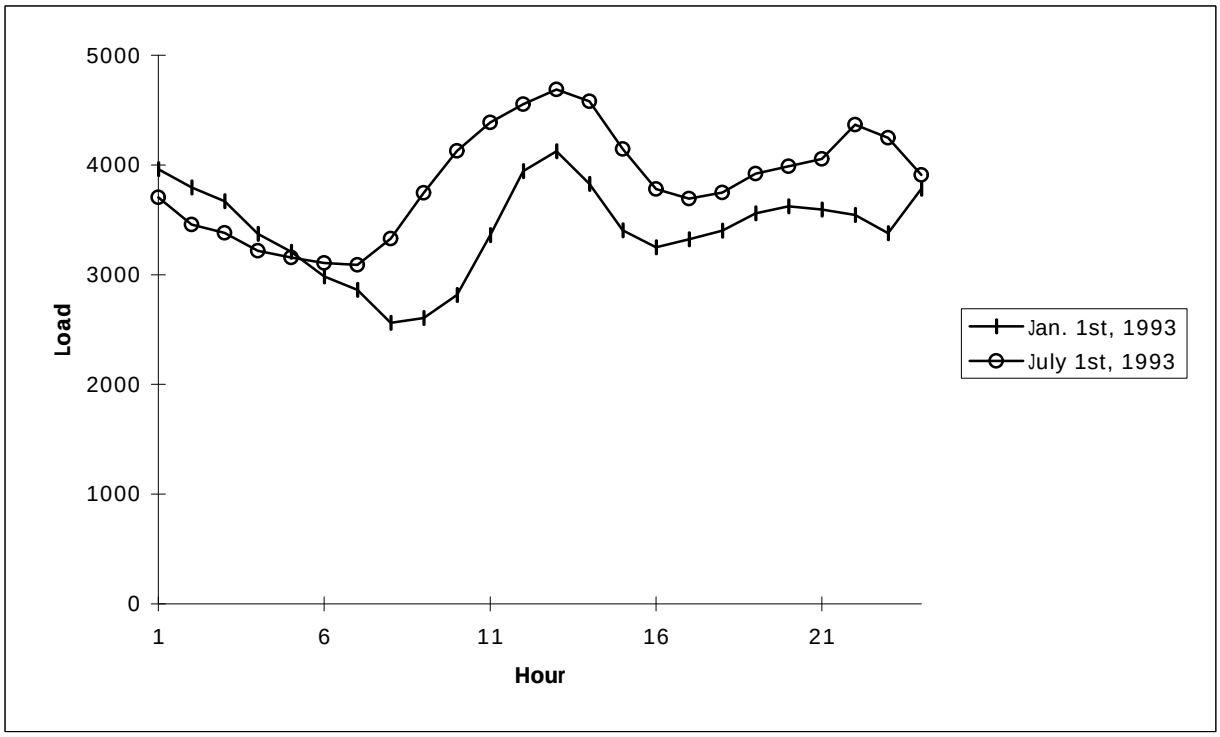


Figure 1 Hourly electric load for the Crete island power network. The load is illustrated for one representative winter day (Jan. 1st, 1993) and for one representative summer day (July 1st, 1993).

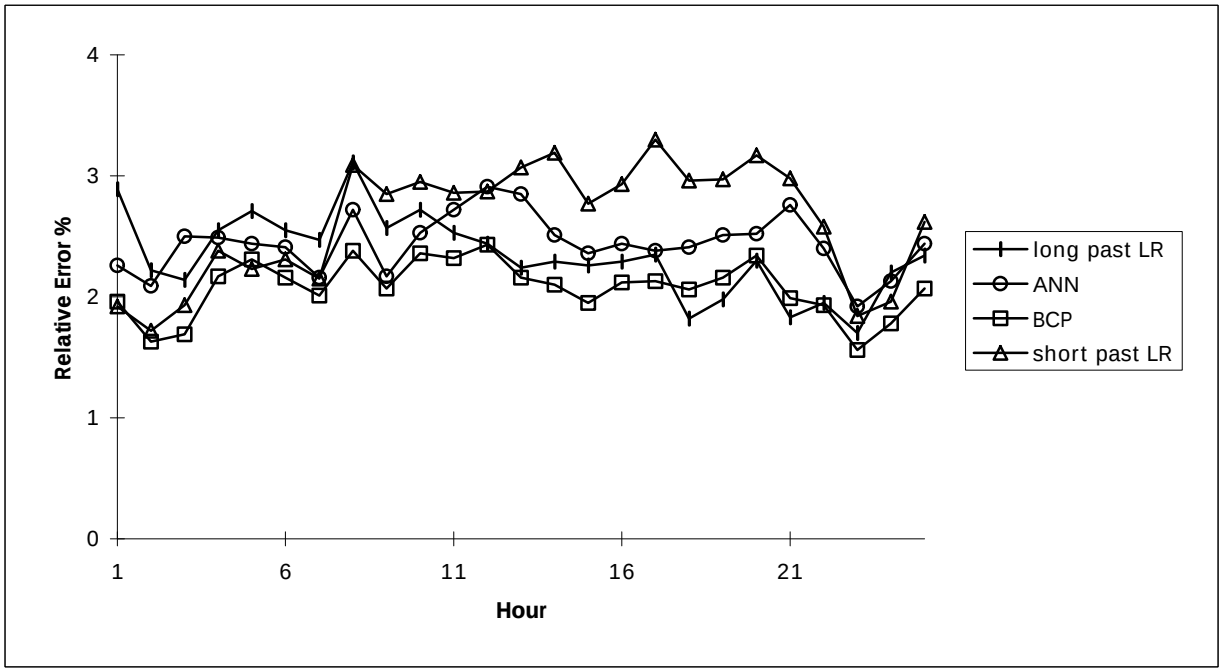


Figure 2. Relative error of the three local predictors and of the BCP, plotted against the 24 hours of the day. The last point of each line is the average error (I.e. averaged over the 24 hours of the day).

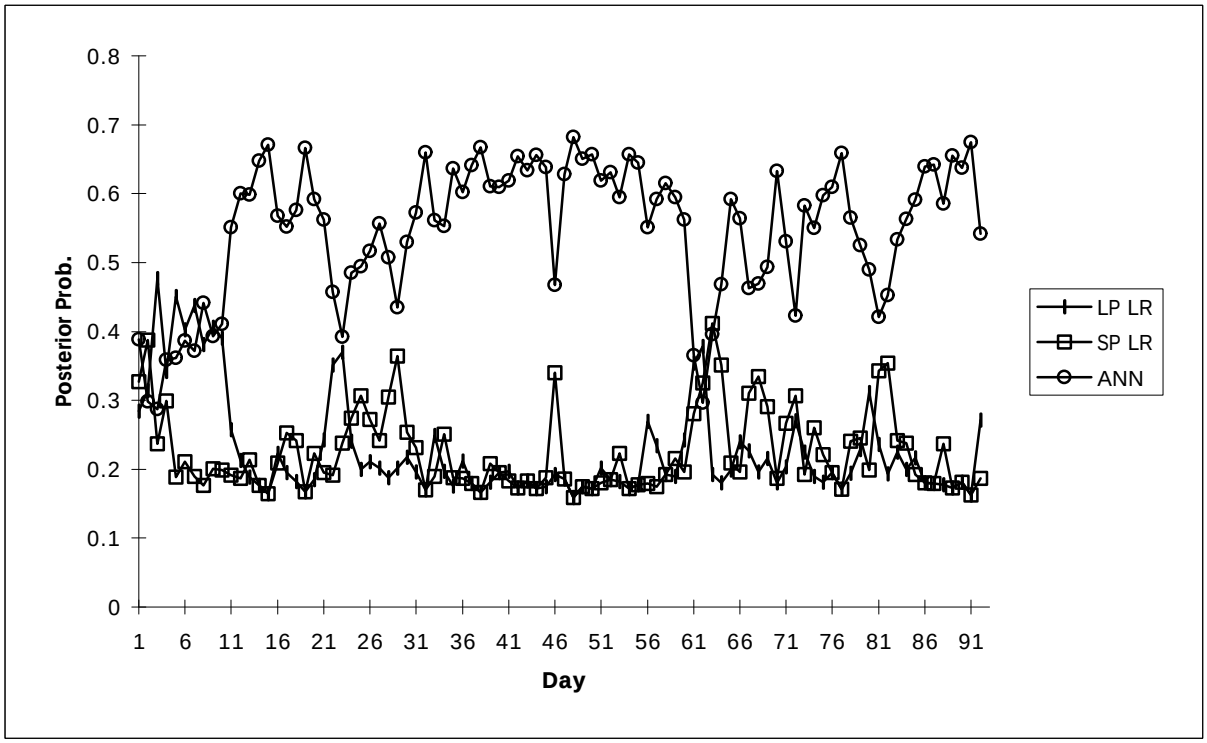


Figure 3 Evolution of posterior probabilities for the predictors of 1am load, over the period July 1st, 1994 to September 30th, 1994 (LP LR: Long Past Lin. Regression, SP LR: Short Past Lin. Regression, ANN: artificial neural network).

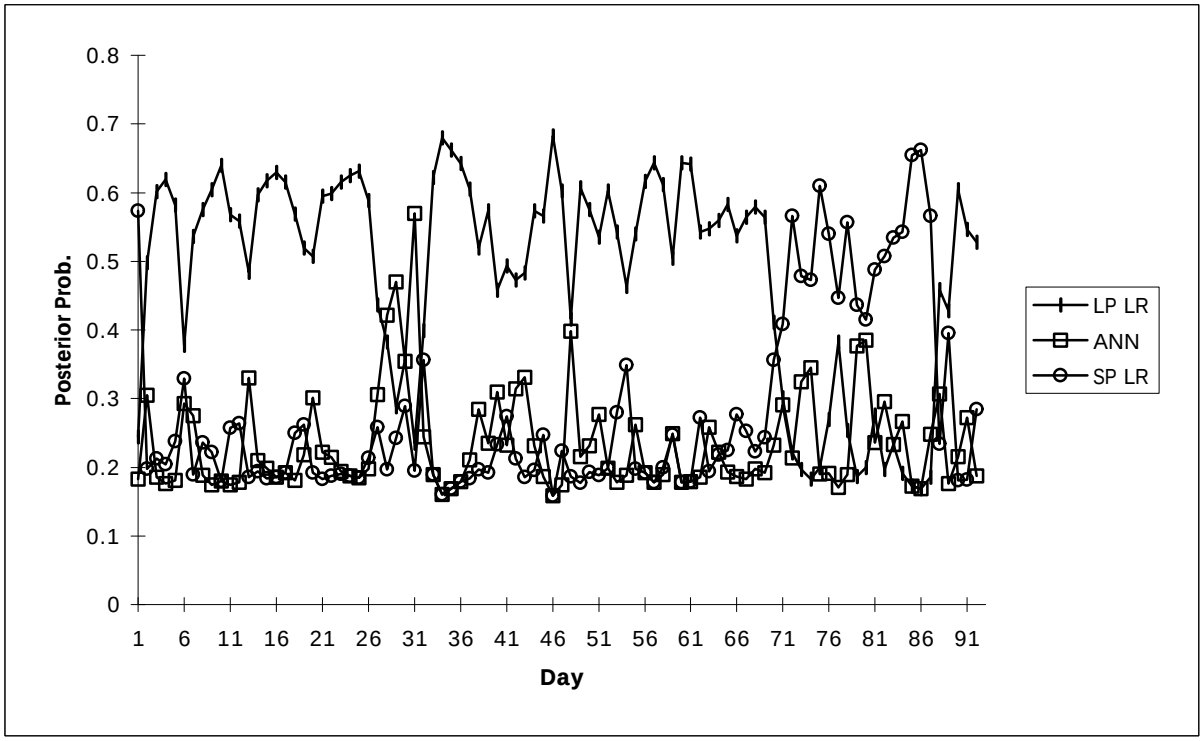
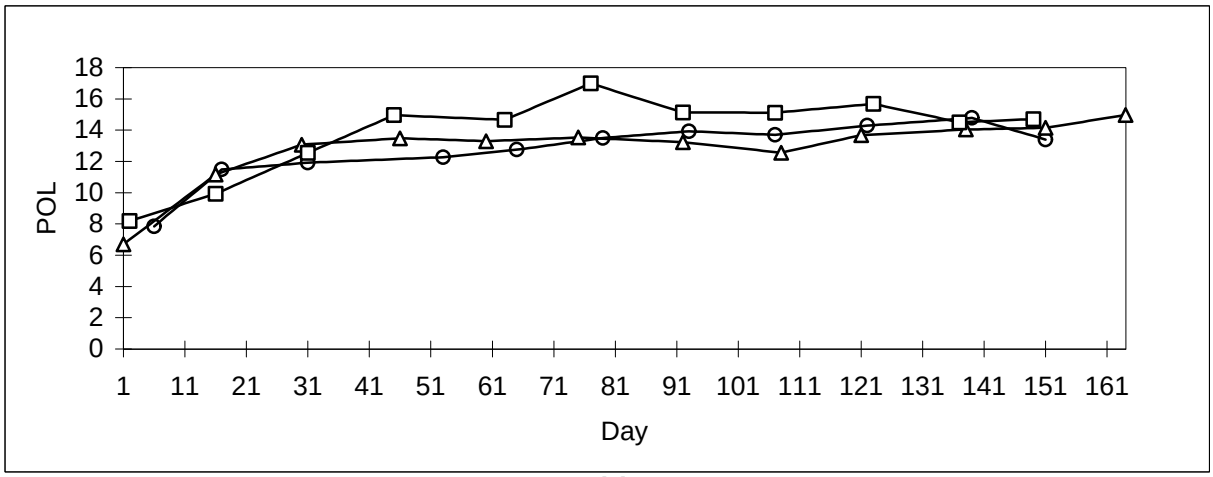
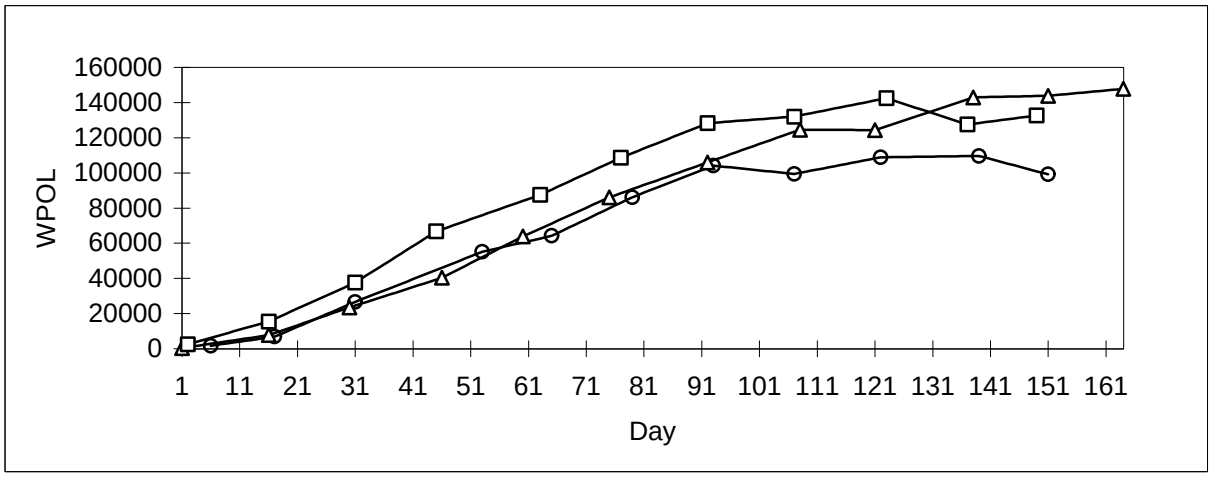


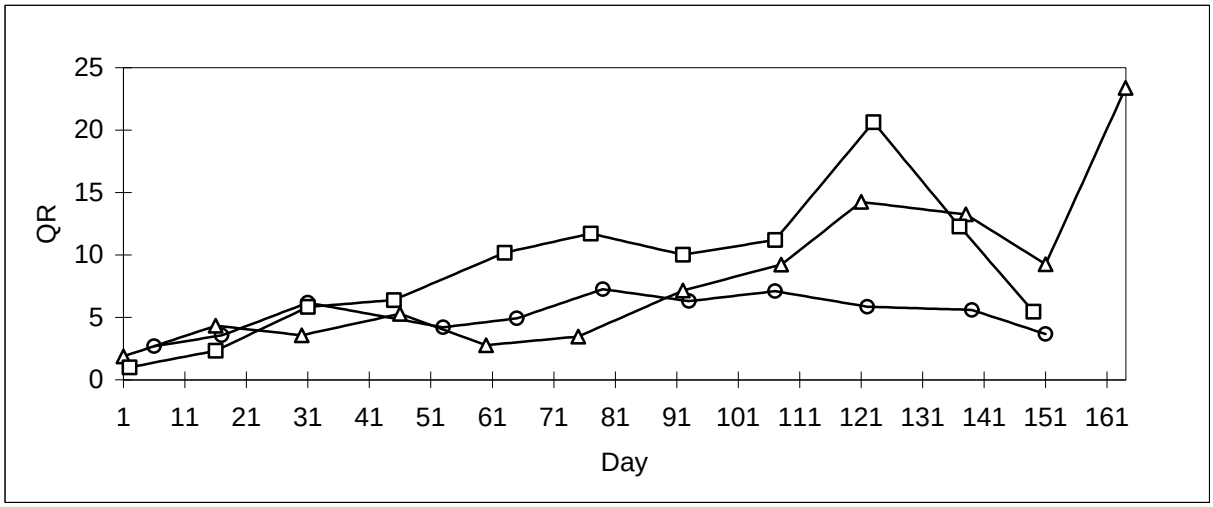
Figure 4 Evolution of posterior probabilities for the predictors of 1pm load, over the period July 1st, 1994 to September 30th, 1994 (LP LR: Long Past Lin. Regression, SP LR: Short Past Lin. Regression, ANN: artificial neural network).



(a)

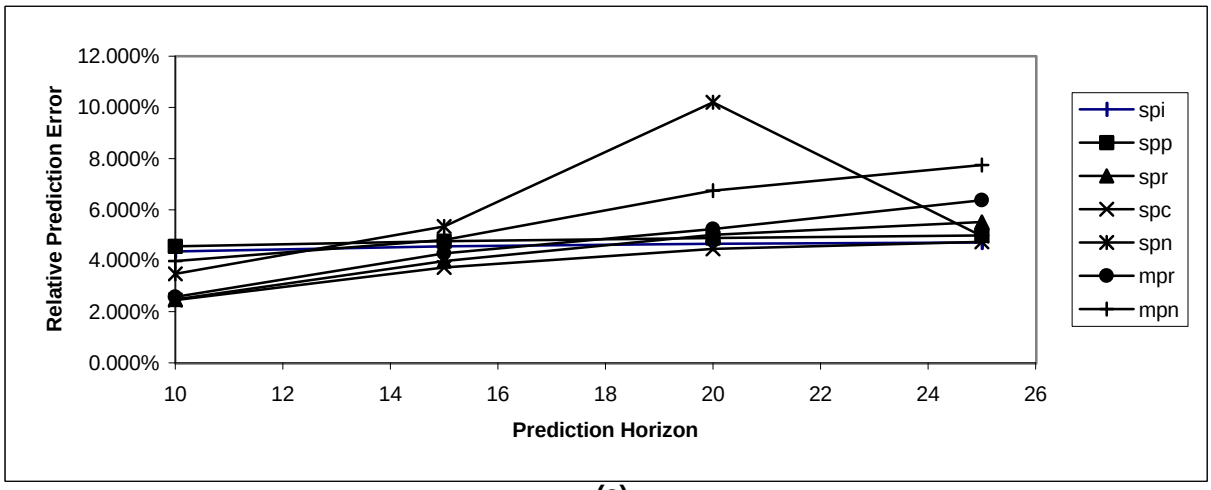


(b)

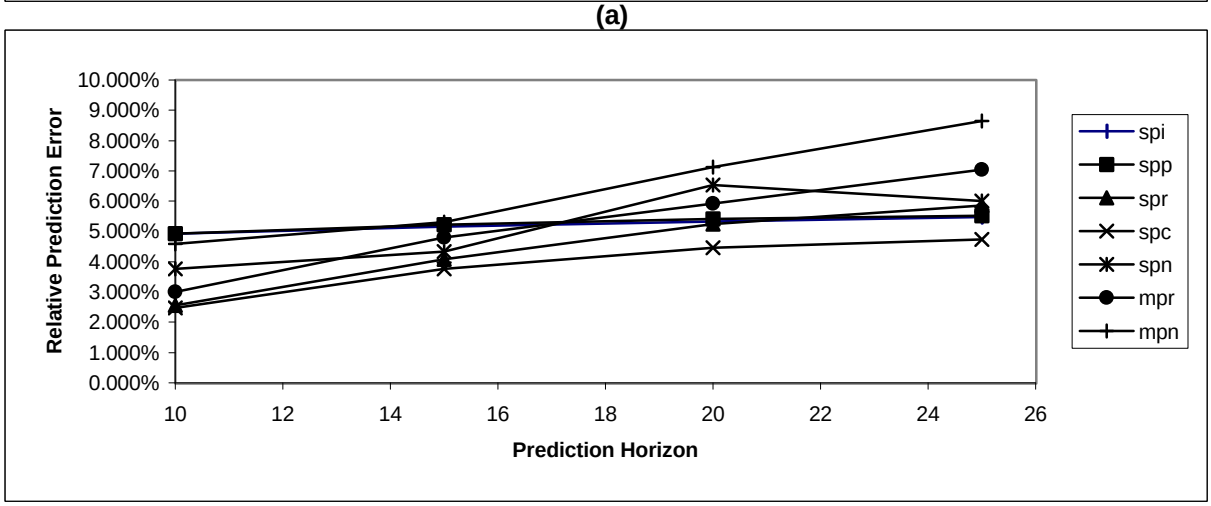


(c)

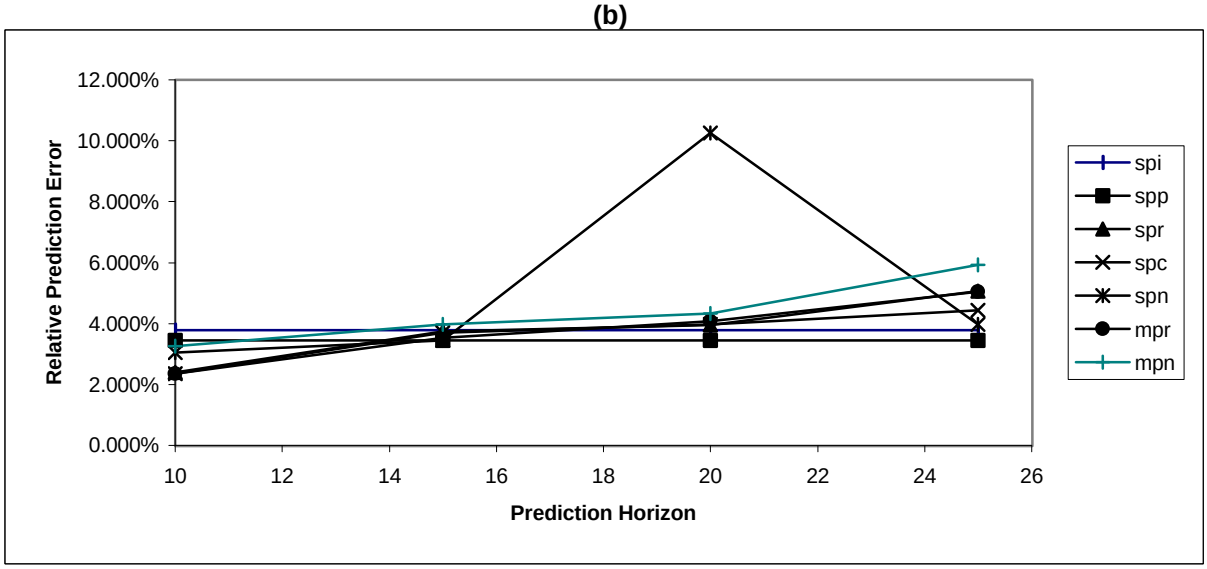
Figure 5 (a) The POL time series in farm 511 for the year 1992 (circle).
(b) The WPOL time series in farm 511 for the year 1993 (triangle).
(c) The QR time series in farm 511 for the year 1994 (square).



(a)



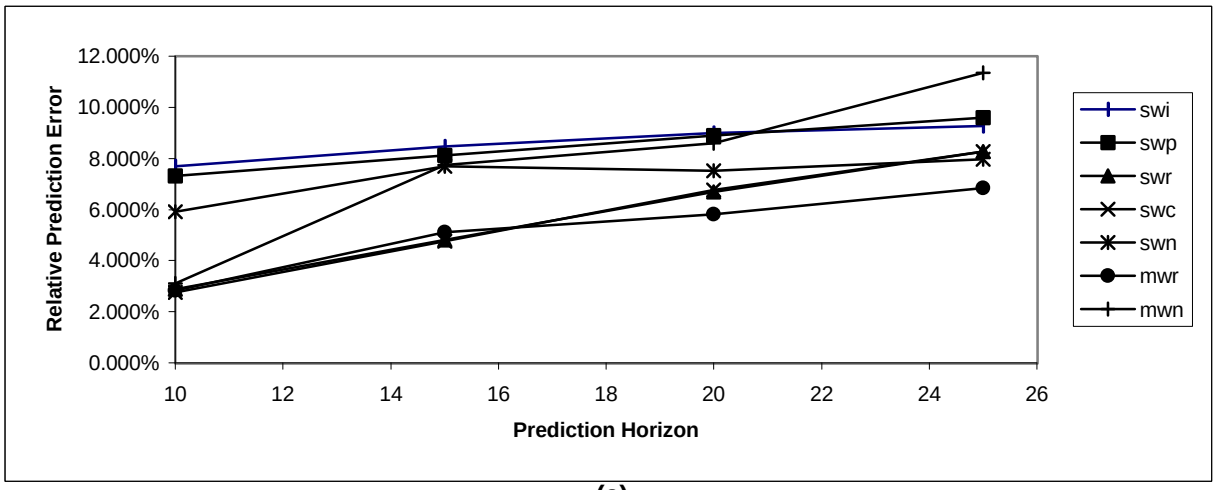
(b)



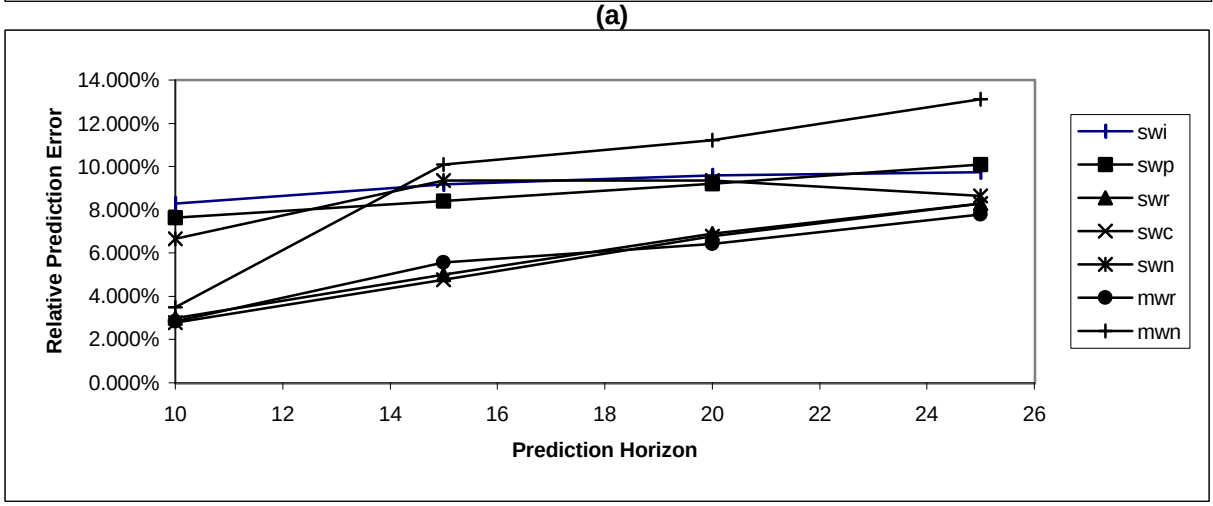
(c)

Figure 6. (a) Prediction error for second half of POL time series, using max likelihood prediction. (b) Prediction error for the second half of the POL time series using weighted prediction. (c) Prediction error for the 20 hot days of the POL time series, using weighted prediction.

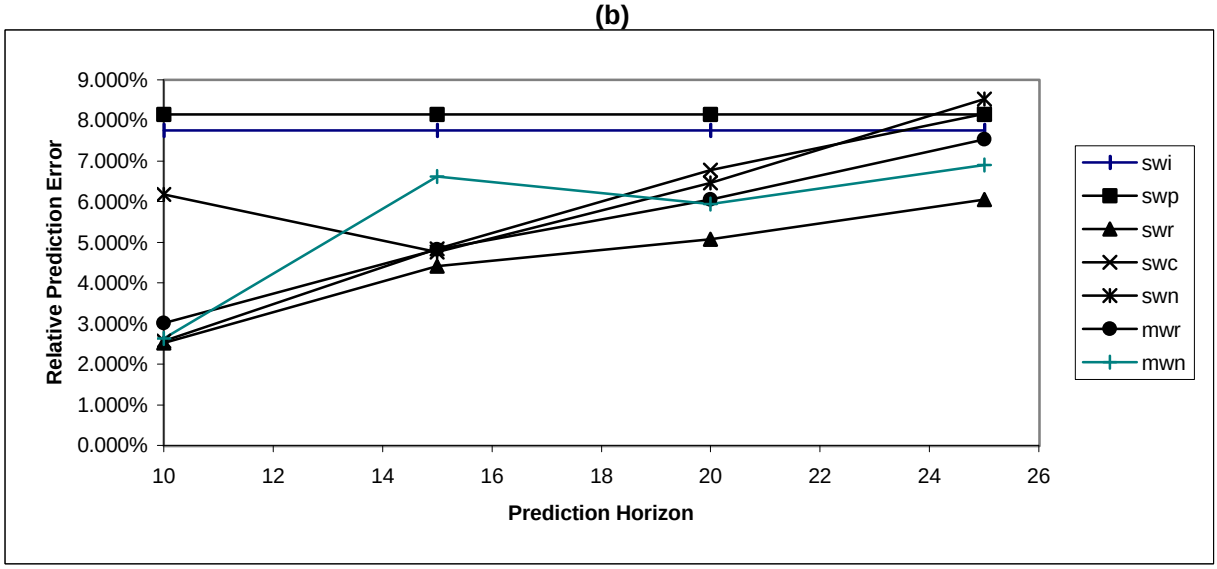
Legend: spi: single input interpolation; spp: single input polynomial; spr: single input regression; spc: single input regression (geog. grouping); spn: single input neural; mpr: multiinput regression; mpn: multiinput neural.



(a)



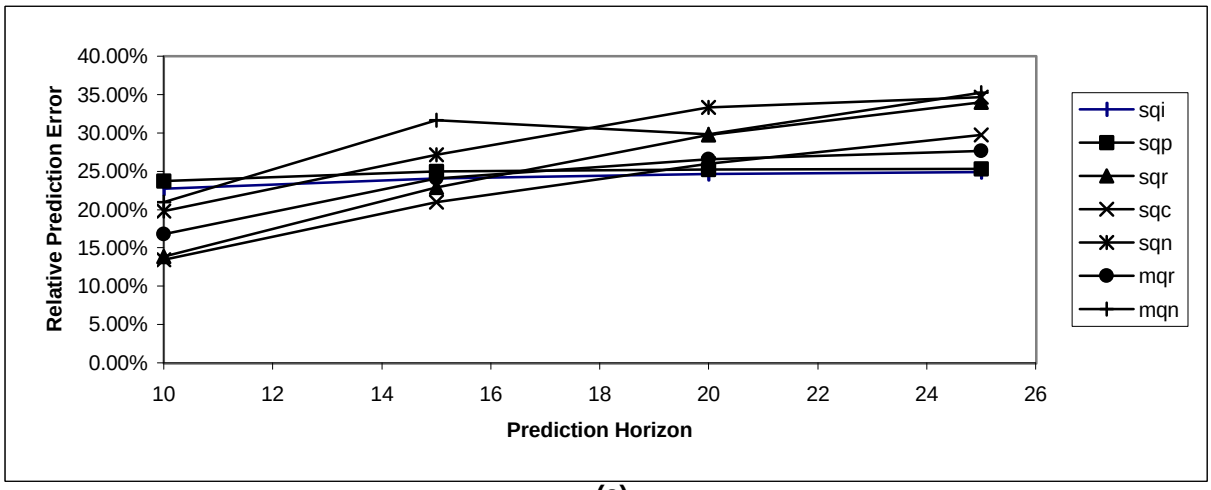
(b)



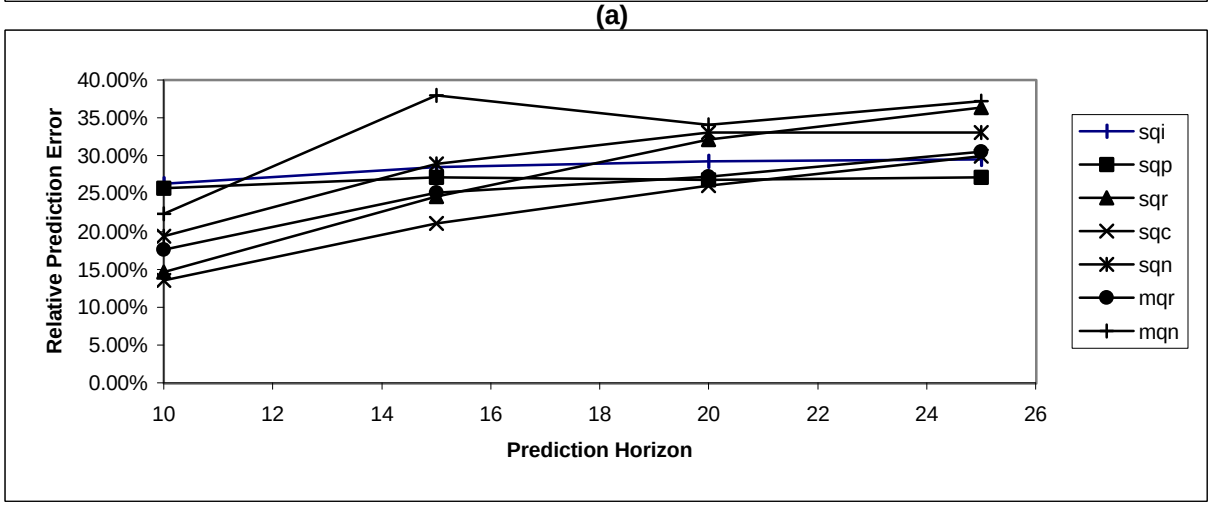
(c)

Figure 7. (a) Prediction error for second half of WPOL time series, using max likelihood prediction. (b) Prediction error for the second half of the WPOL time series using weighted prediction. (c) Prediction error for the 20 hot days of the WPOL time series, using weighted prediction.

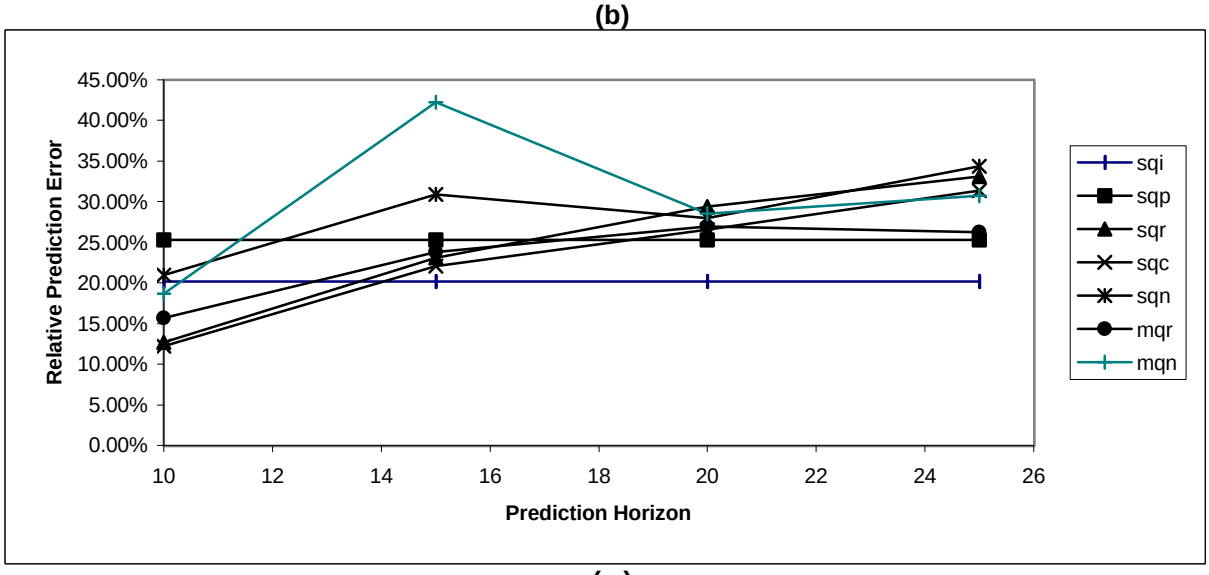
Legend: swi: single input interpolation; swp: single input polynomial; swr: single input regression; swc: single input regression (geog. grouping); swn: single input neural; mwr: multiinput regression; mwn: multiinput neural.



(a)



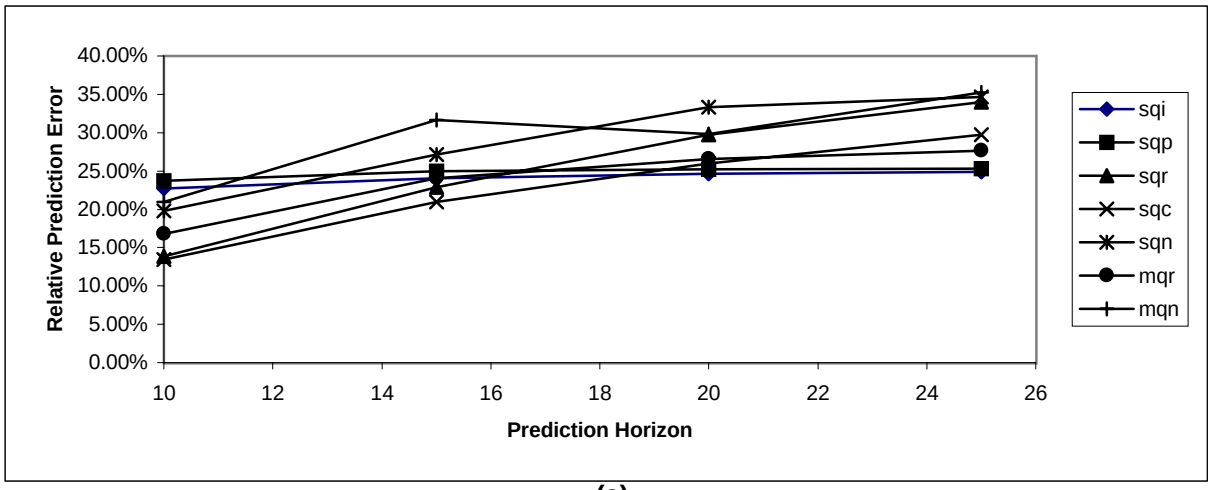
(b)



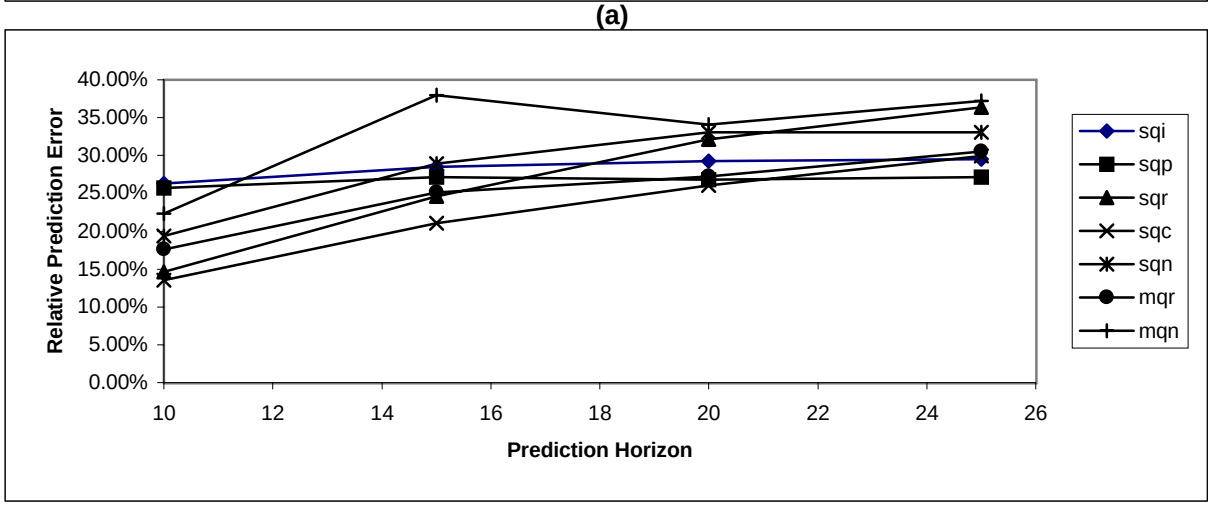
(c)

Figure 8. (a) Prediction error for the second half of QR time series, using max likelihood prediction.
(b) Prediction error for the second half of the QR time series using weighted prediction.
(c) Prediction error for the 20 hot days of the QR time series, using weighted prediction.

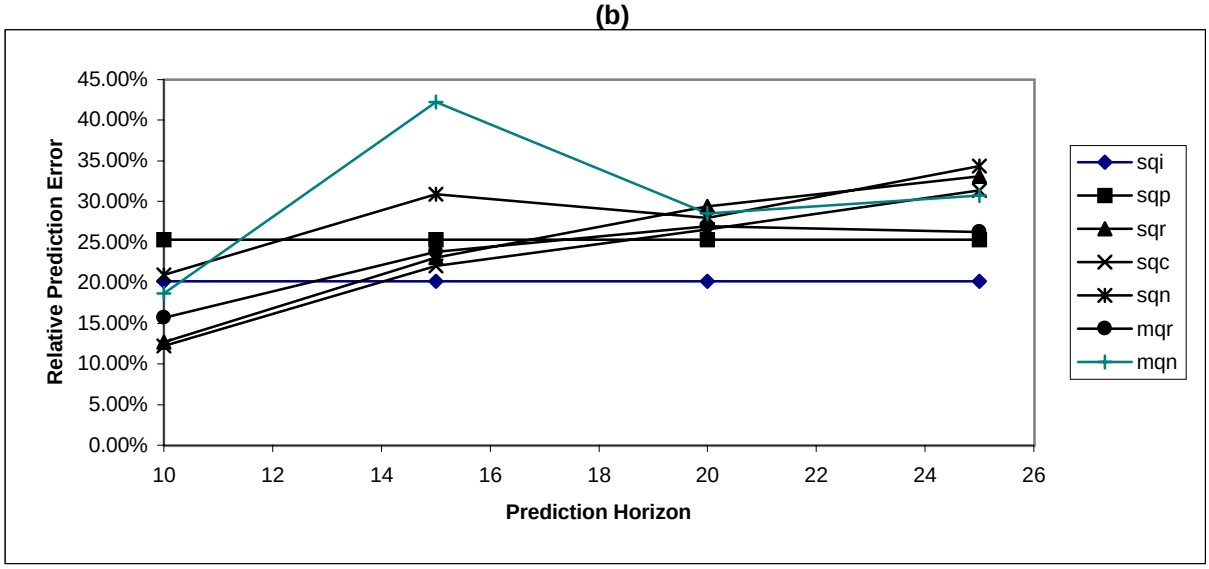
Legend: sqi: single input interpolation; sqp: single input polynomial; sqr: single input regression; sqc: single input regression (geog. grouping); sqn: single input neural; mqr: multiinput regression; mqn: multiinput neural.



(a)



(b)



(c)

Figure 8. (a) Prediction error for the second half of QR time series, using max likelihood prediction.
(b) Prediction error for the second half of the QR time series using weighted prediction.
(c) Prediction error for the 20 hot days of the QR time series, using weighted prediction.

Legend: sqi: single input interpolation; sqp: single input polynomial; sqr: single input regression; sqc: single input regression (geog. grouping); sqn: single input neural; mqr: multiinput regression; mqn: multiinput neural.