# An Adaptive Learning Radial Basis Function Neural Network for Online Time Series Forecasting

**Mazlina Mamat, Rosalyn R. Porle, Norfarariyanti Parimon and Md. Nazrul Islam**

**Abstract** Most of the neural network based forecaster operated in offline mode, in which the neural network is trained by using the same training data repeatedly. After the neural network reaches its optimized condition, the training process stop and the neural network is ready for real forecasting. Different from this, an online time series forecasting by using an adaptive learning Radial Basis Function neural network is presented in this paper. The parameters of the Radial Basis Function neural network are updated continuously with the latest data while conducting the desired forecasting. The adaptive learning was achieved using the Exponential Weighted Recursive Least Square and Adaptive Fuzzy C-Means Clustering algorithms. The results show that the online Radial Basis Function forecaster was able to produce reliable forecasting results up to several steps ahead with high accuracy to compare with the offline Radial Basis Function forecaster.

## 1 Introduction

Forecasting has become an important research area and is applied in many fields such as in sciences, economy, meteorology, politic and to any system if there, exist uncertainty on that system in the future. Before the emergence of mathematical and

M. Mamat (✉)
Artificial Intelligence Research Unit (AIRU), Universiti Malaysia Sabah, 88450 Kota Kinabalu, Sabah, Malaysia
e-mail: mazlina@ums.edu.my

R.R. Porle · N. Parimon · Md.N. Islam
Faculty of Engineering, Universiti Malaysia Sabah, 88450 Kota Kinabalu Sabah, Malaysia
e-mail: rlyn39@ums.edu.my

N. Parimon
e-mail: fara2012@ums.edu.my

Md.N. Islam
e-mail: nazrul@ums.edu.my



P.J. Soh et al. (eds.), *Advances in Machine Learning and Signal Processing*, Lecture Notes in Electrical Engineering 387, DOI 10.1007/978-3-319-32213-1_3

computer models so called machine learning algorithms, forecasting was carried out by human experts. In this approach, all parameters that are possibly give effect to the system to be forecasted are considered and judged by the experts before producing the forecasting output. Unfortunately, forecasting using human experts is very vague and sometimes arguable since it is totally depends on the expert's knowledge, experiences an interest. Other than using human experts, there exists other prediction approach: Statistical Prediction Rules which is more reliable and robust [1]. One of the popular methods in Statistical Prediction Rules is Time Series Prediction where it uses a model to forecast future events based on known past events: to forecast future data points before they are measured.

A time series consists of sequence of numbers which explained the status of an activity versus time. In more detail, a time series is a sequence of data points, measured typically at successive times, spaced at (often uniform) time intervals. A time series has features that are easily understood. For instance a stock price time series has a long term trend, seasonal and random variations while a cereal crops price time series contains only seasonal components [2]. There exist many approaches to perform time series forecasting. Among the approaches are Box-Jenkins Approach (ARMA/ARIMA) [3], Regression analysis [4], Artificial Neural Networks (ANN) [5], Fuzzy Logic (FL) [6] and Genetic Algorithms [GA] [7]. However among them, the computational intelligence technique such as ANN, FL and GA are getting more attention in time series forecasting because they are non-linear in nature and able to approximate easily complex dynamically system [8–11].

In its typical implementation, ANN will be trained by using the existing set of previous data. After the ANN reaches its optimized performance, the training process is stopped. Then the optimized ANN will be used to estimate the forthcoming output based on the current received inputs. This form of implementation is called as offline mode and is implemented in real-world, especially in power generator station [12, 13]. However, studies shown forecasting in the offline mode has several disadvantages. The major disadvantage is that the ANN parameters must be updated from time to time to suite with the various changing of the incoming data. This requires the ANN to be trained again by including the latest available data for the training. If the updating process is neglected, the ANN will generate incorrect forecasting whenever they receive unseen input data beyond the training data set. In situation where the data are non-stationary, the offline forecaster will be under performance, unless it is continuously being updated to track non-stationarities. This calls for online forecasting technique, where the parameters of the ANN will be adaptive to the latest available data.

## 2 Materials and Methods

### 2.1 Radial Basis Function

The Radial basis function (RBF) neural networks typically have three layers: an input layer, a hidden layer with a non-linear RBF activation function and a linear output layer. Each layer consists of one or more nodes depending on the design. The nodes in input layer are connected to the nodes in hidden layer while the nodes in hidden layer are connected to the nodes in the output layer via linear weights [14]. The output from each node in the hidden layer is given as:

$$Z_j = \Phi\big(||v(t) - c_j(t)||\big) \quad j = 1, 2, 3. \ldots \ldots \ldots n_h. \tag{1}$$

where $c_j(t)$ and $n_h$ representing the RBF centre and number of hidden nodes, $v(t)$ is the input vector to the RBF and $\Phi(\bullet)$ representing the temporary activation function while $\|\bullet\|$ represents the Euclidean distance between the input vector and RBF centre. The initial value of the RBF centre, $c_j(t)$ is given by taking the first data of the series as the RBF centre. The activation function $\Phi(\bullet)$ used is Thin Plate Spline given by $\Phi(a) = a^2 \log(a)$, where $a = a = ||v(t) - c_j(t)||$ is Euclidean distance. The Euclidean distance for each hidden node is given by:

Euclidean distance,

$$a_j = \sqrt{\sum_{i=1}^{n} \big(c_{ij}(t) - v_j(t)\big)^2}. \tag{2}$$

where $c_{ij}(t)$ = RBF centre for the $j$-th hidden node and $i$-th input, and $v_i(t)$ = the $i$-th input. The RBF output is given by:

$$y_k(t) = w_{k0} + \sum_{j=1}^{n_h} w_{kj} \Phi\big(||v(t) - c_j(t)||\big); \quad k = 1, 2, 3, \ldots \ldots m. \tag{3}$$

where $w_{kj}$, is the weight between hidden node and output node and $m$ is the number of output node.

### 2.2 Adaptive Learning Algorithms

Two parameters, namely, the RBF centre in hidden nodes and weights between the hidden nodes and the output nodes were updated using the *Adaptive Fuzzy C-Means Clustering* (*AFCMC*) and the *Exponential Weighted Recursive Least Square* (*e-WRLS*) algorithms respectively [15, 16].

**Adaptive Fuzzy C-Means Clustering**. Give initial value to $c_j(0)$, $\mu(0)$ and $q$, where $0 \leq \mu(0) \leq 1.0$ and $0 \leq q \leq 1.0$ (the typical value is between 0.30 to 0.95 respectively). Then compute the Euclidean distance $d_j(t)$ between input $v(t)$ and centre $c_j(t)$. Obtain the shortest distance, $d_s(t)$ longest distance $d_l(t)$, nearest centret $c_s(t)$ and distant centre $c_l(t)$.

For $j = 1$ to $j = n_c$, where $n_c$ = number of RBF centre,

(a) Updates the square distance between centre and input $v(t)$ using:

$$\gamma(t) = \frac{1}{n_c} \sum_{k=1}^{n_c} [||v(t) - c_k(t)||]^2. \tag{4}$$

(b) if $j \neq s$ that is if that centre is not $c_s$ centre, updates the centre by referring to:

$$\Delta c_j = \mu(t)\vartheta(t)\big[v(t) - c_j(t-1)\big]. \tag{5}$$

where

$$\vartheta(t) = \begin{cases} D_l(t)D_j(t)\exp[-D_a(t)] & \text{if } d_j > 0 \\ D_l(t)\exp[-D_a(t)] & \text{if } d_j = 0 \end{cases}. \tag{6}$$

and

$D_l(t) = \frac{\gamma(t)}{d_l^2}$,$D_j(t) = \frac{d_a^2(t)}{d_j^2(t)}$ and $D_a(t) = \frac{d_a^2(t)}{\gamma(t)}$ with $a = s$ if $d_s(t) > 0$ and $a = z$ if $d_s(t) = 0$, ($d_z(t)$ = smallest nonzero distance between $v(t)$ and $c_j(t)$).

(c) updates $c_s(t)$ using

$$\Delta c_s(t) = \mu(t)\varphi(t)[v(t) - c_s(t-1)]. \tag{7}$$

Where

$$\varphi(t) = \begin{cases} \exp\left(-\frac{d_s^2(t)}{\gamma(t)}\right) & \text{if } d_s(t) > 0. \\ 0 & \text{if } d_s(t) = 0. \end{cases} \tag{8}$$

Measure the distance between $c_s(t)$ with all centres, $h_k(t) = (\|c_s(t) - c_k(t)\|)$, $k = 1, 2, 3, \ldots n_c$ and $k \neq s$. If the shortest distance $h_c(t)$, is $h_c(t) < \mu(t)d_a(t)$, move the nearest distance, $c_c(t)$ to new location based on:

$$\Delta c_c(t) = -\frac{\mu(t)d_a^2(t)}{d_l^2(t)}(c_c(t) - c_s(t)). \tag{9}$$

Set $t = t + 1$, and repeat the above for each data sample. The diffusion coefficient $\mu(t)$, is updates by using:

$$\mu(t) = \mu(0)\exp\left(-\frac{qt^2}{n_c^2}\right) + \frac{\exp(-q\mu(t-1))}{n_c}. \tag{10}$$

**Exponential Weighted Recursive Least Square**. Set $\hat{\beta}_0^j = 0$ and construct matrix $P_0 = \alpha I$. The typical value for $\alpha$ is $10^4 \leq \alpha \leq 10^6$ and $I$ is Identity matrix of $n_h$ (number of hidden nodes). Read the output from the hidden nodes, $X_k^T$, and calculate $K_k$ and $P_{k+1}$ using:

$$K_k = \frac{P_k X_{k+1}}{\lambda + X_{k+1}^T P_k X_{k+1}}, \tag{11}$$

and

$$P_{k+1} = \frac{P_k}{\lambda}\left\{I - K_k X_{k+1}^T\right\}. \tag{12}$$

where $\lambda$ is a forgetting factor with its typical value of $0.95 \leq \lambda \leq 0.99$. The $\lambda$ can also be computed by:

$$\lambda(t) = \lambda_0\lambda(t-1) + (1-\lambda_0). \tag{13}$$

where $\lambda_0 = 0.99$. Estimates $\hat{\beta}_{k+1}^j$ by using:

$$\hat{\beta}_{k+1}^j = \hat{\beta}_k^j + K_k\left[y_{k+1} - X_{k+1}^T\hat{\beta}_k^j\right]. \tag{14}$$

Set $k = k + 1$, $k = 1, 2, 3, \ldots N$ where $N$ is the number of data. Repeat steps 2–4 till converges. Because $\xi(k)$ cannot be computed, $\hat{\varepsilon}(k)$ is used to replace $\xi(k)$ where $\hat{\varepsilon}(k)$, is measurement error and can be computed by $\hat{\varepsilon}(k) = y(k) - \hat{y}(k)$ where $\hat{y} = X_k^T\hat{\beta}_{k-1}$.

## 3 Results and Discussion

### *3.1 Data*

The forecasting performance is evaluated by using two simulated data and one real data. The simulated data are the Mackey-Glass nonlinear time series and Set A from Santa Fe Competition (SantaFe-A), while the real data is the IBM Stock Price data. The forecasting based on time series produced by the Mackey-Glass equation is regarded as a criterion for comparing the ability of different predicting method and

is used in many time series forecasting researches [17, 18]. The SantaFe-A data were recorded from a Far-Infrared-Laser in a chaotic state. These data were chosen because they are a good example of the complicated behavior that can be seen in a clean, stationary, low-dimensional non-trivial physical system for which the underlying governing equations dynamics are well understood. The IBM data are the daily closing price of IBM stock from January 1, 1980 to October 8, 1992 [19].

## 3.2 Forecaster Optimization

The selection of input lag and the number of hidden node in the RBF have strong impact on the performance of a neural network based forecaster [3]. In parallel to this, the analysis on input selection (input lag) and number of hidden node that produce the optimized forecaster is conducted. The analysis starts by setting the RBF input with the data at elapsed time $(t-1)(t-2)$ … $(t-8)(t-9)(t-10)$ and increasing the hidden nodes one by one until it reaches 50. For each number of hidden nodes, the $R^2$ value for one step ahead forecasting were recorded. The same process is repeated for the other input lag as tabulated in Table 1. The $R^2$ values obtained from the analysis were plotted and the number of hidden node which gives the highest $R^2$ values for all five input lags was used in the analysis to obtain the correct number of input lag. This analysis was conducted by setting the hidden node to the value obtained and varies the input lag from $(t-1)$ to $(t-1)(t-2)$ … $(t-98)(t-99)(t-100)$. Table 2 shows the input lag and the number of hidden node which produce the best multiple steps ahead forecasting performance for the three data.

**Table 1** List of input lags used to determine the correct number of hidden node

| Name | Input lag |
|---|---|
| Input lag 1 | $(t-1)(t-2)(t-3)$ … $(t-9)(t-10)$ |
| Input lag 2 | $(t-1)(t-2)(t-3)$ … $(t-19)(t-20)$ |
| Input lag 3 | $(t-1)(t-2)(t-3)$ … $(t-29)(t-30)$ |
| Input lag 4 | $(t-1)(t-2)(t-3)$ … $(t-39)(t-40)$ |
| Input lag 5 | $(t-1)(t-2)(t-3)$ … $(t-49)(t-50)$ |

**Table 2** The best input lag and number of hidden node for the three data

| Data | Input lag | Hidden node |
|---|---|---|
| Mackey-Glass | $(t-1)(t-2)(t-3)$ … $(t-36)(t-37)$ | 32 |
| SantaFe-A | $(t-1)(t-2)(t-3)$ … $(t-29)$ $(t-30)$ | 25 |
| IBM | $(t-1)(t-2)(t-3)$ … $(t-40)$ $(t-41)$ | *39* |

### 3.3 *Forecasting Performance*

The findings obtained from the analyses in Sect. 3.2 were used to construct a universal online RBF forecaster for Mackey Glass data, SantaFe-A data and IBM Stock Price data. The performance of the forecaster to forecast the three data were tested. Figure 1 presented one to four steps ahead forecasting for the last 500 Mackey Glass data. For SantaFe-A data, due to the nature of data which is too fluctuating, only the last 100 actual and forecasted data are displayed in Fig. 2. Figure 3 presented the forecasting on 500 IBM Stock Price data. From plots in Figs. 1, 2, and 3, it can be observed that the online RBF forecaster is able to produce reliable and close forecasted values in most of the time. For both data, the input lags and number of hidden nodes which produce the best forecasting performance and the $R^2$ values for one to four steps ahead forecasting are presented in Table 3.

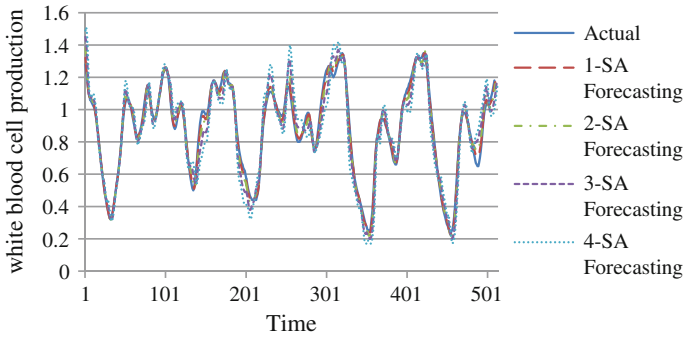


**Fig. 1** One to four steps ahead forecasting for the last 500 Mackey Glass data

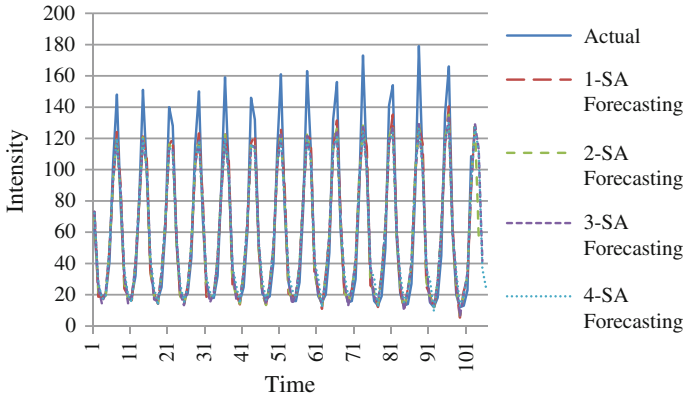


**Fig. 2** One to four steps ahead forecasting for the last 100 SantaFe-A data

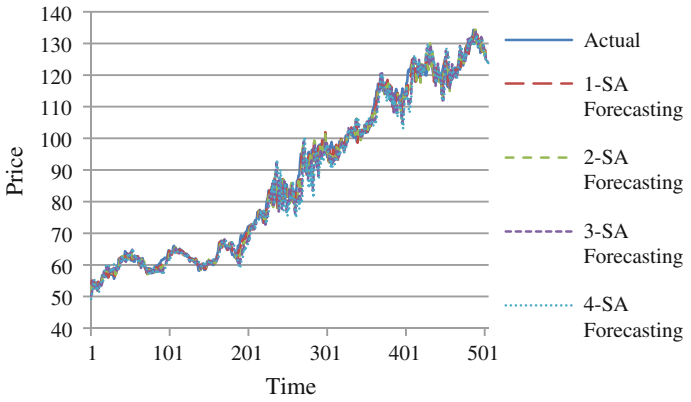


**Fig. 3** One to four steps ahead forecasting for the last 500 IBM data

**Table 3** RMSE and $R^2$ values for one to four steps ahead forecasting obtained by the online RBF forecaster

| Data | Mackey-Glass | | SantaFe-A | | IBM | |
|---|---|---|---|---|---|---|
| Input lag | $(t-1)(t-2)(t-3)\ldots(t-36)(t-37)$ | | $(t-1)(t-2)(t-3)\ldots(t-29)(t-30)$ | | $(t-1)(t-2)(t-3)\ldots(t-40)(t-41)$ | |
| Hidden node | 32 | | 25 | | 39 | |
| Indicator | RMSE | $R^2$ | RMSE | $R^2$ | RMSE | $R^2$ |
| 1-step ahead | 0.019 | 0.995 | 25.477 | 0.653 | 1.872 | 0.994 |
| 2-steps ahead | 0.064 | 0.947 | 27.436 | 0.597 | 6.211 | 0.939 |
| 3-steps ahead | 0.099 | 0.872 | 28.527 | 0.565 | 8.634 | 0.882 |
| 4-steps ahead | 0.137 | 0.758 | 28.510 | 0.565 | 10.512 | 0.825 |

**Table 4** RMSE and $R^2$ values for one to four steps ahead forecasting obtained by the offline RBF forecaster

| Data | Mackey-Glass | | SantaFe-A | | IBM | |
|---|---|---|---|---|---|---|
| Input Lag | $(t-1)(t-2)(t-3)\ldots(t-36)(t-37)$ | | $(t-1)(t-2)(t-3)\ldots(t-29)(t-30)$ | | $(t-1)(t-2)(t-3)\ldots(t-40)(t-41)$ | |
| Hidden node | 32 | | 25 | | 39 | |
| Indicator | RMSE | $R^2$ | RMSE | $R^2$ | RMSE | $R^2$ |
| 1-step ahead | 0.030 | 0.988 | 23.733 | 0.699 | 7.215 | 0.918 |
| 2-steps ahead | 0.083 | 0.911 | 25.317 | 0.657 | 16.424 | 0.573 |
| 3-steps ahead | 0.140 | 0.746 | 26.617 | 0.621 | 28.026 | −0.243 |
| 4-steps ahead | 0.214 | 0.412 | 26.687 | 0.619 | 42.152 | −1.812 |

The forecasting performance of the online RBF forecaster versus the offline RBF forecaster was also evaluated. Table 4 presents the RMSE and $R^2$ values for four steps ahead forecasting achieved by the offline RBF forecaster.

For Mackey Glass data, the performance of the offline RBF for one step ahead forecasting are slightly lower to compare with the online RBF. However for multiple steps ahead forecasting, the performance of offline RBF was absolutely poorer where significant deviations were recorded as the forecasting distance increases. The analysis on IBM stock price data also favors online RBF over offline RBF. It can be noted that the offline RBF is able to produce good one step ahead forecasting with 91 % accuracy. However it is considered low to compare with the accuracy obtained by online RBF which is 99 %. For other forecasting distance, namely two to four steps ahead, the offline RBF was failed to generate the acceptable forecasting performance. The forecasting performance obtained by the two, three and four steps ahead was lower than 0.6 and can be regarded as poor to compare with online RBF.

The superiority of online RBF over offline RBF on the Mackey Glass and IBM stock price data can be explained by the nature of the data themselves. It can be observed that both data display different patterns over times especially in the first half and second half of the data. Therefore by using the first half of the data for training are insufficient for the offline RBF to cover all patterns that were exhibited by the data in the next second half. This finding shows that the offline RBF is unable to generate good forecasting for any system which shows chaotic and non-stationary patterns over time.

However, different observation was obtained from the analysis on SantaFe-A data. For this data, the offline RBF produces higher performance in one to four steps ahead forecasting to compare with online RBF. This is again can be explained by the data itself where it can be noted that SantaFe-A data exhibit almost consistent patterns throughout the time. In brief, it can be said that the data is repeating themselves over times. Therefore by training the offline RBF using the first 500 data repeatedly was enough to cover the next 500 testing data. While for online RBF, continuous learning contributes to over fitting which degraded its forecasting ability.

## 4 Conclusion

More and more fields including science, financial, economy and meteorological adapt time series forecasting to solve uncertainty situation or outcomes in their respective fields. Due to the nature that problems to be solved are affected much by other parameters which change over time, the requirement of the online forecasting model is practical in real-world applications. This paper presented a tool to perform online multiple steps ahead time series forecasting using Radial Basis Function, which shows reliable and accurate forecasting capability.

## References


1. Bishop MA, Trout JD (2004) Epistemology and the psychology of Human judgment. Oxford Uni. Press, USA
2. Vemuri VR, Rogers RD (1994) Artificial neural networks forecasting time series. IEEE Computer Society Press, California
3. Montanes E, Quevedo JR, Prieto MM, Menendez CO (2002) Forecasting time series combining machine learning and box-jenkins time series. Advances in artificial intelligence—IBERAMIA 2002, vol 2527
4. Lin K, Lin Q, Zhou C, Yao J (2007) Time series prediction on linear regression and SVR. Third international conference on natural computation, Hainan, China, pp 688–691
5. Boznar M, Lesjak M, Mlakar P (1993) A neural network-based method for short-term predictions of ambient SO2 concentrations in highly polluted industrial areas of complex terrain. Atmos Environ 27(2):221–230
6. Chen SM, Chung NY (2006) Forecasting enrollments using high-order fuzzy time series and genetic algorithms. Int J Intell Syst 21:485–501
7. Nunnari G, Bertucco L (2001) Modelling air pollution time-series by using wavelet function and genetic algorithms. International conference on artificial neural network and genetic algorithms, Prague, pp 489–492
8. Ferrari S, Robert F (2005) Smooth function approximation using neural networks. IEEE Trans Neural Netw 16(1):24–38
9. Crippa P, Turchetti C, Pirani M (2004) A stochastic model of neural computing. In: Lecture notes in artificial intelligence (Subseries of lecture notes in computer science), Part II, vol 3214. Springer Verlag, Berlin, Germany, pp 683–690
10. Zhang GP (2012) Neural networks for time-series forecasting. In: Rozenberg G, Kok JN, Back T (eds) Handbook of natural computing. Springer, Berlin, Germany, pp 461–477
11. Leu Y, Lee C, Hung C (2010) A fuzzy time series-based neural network approach to option price forecasting. In: Nguyen NT, Le MT, Swiatek J (eds) Lecture notes in computer science, vol 5990. Springer, Berlin, Germany, pp 360–369
12. Khotanzad A, Hwang RC, Abaye A, Maratukulam DJ (1995) An adaptive modular artificial neural network hourly load forecaster and it's implementation at electric utilities. IEEE PES winter meeting, vol 95. New York, pp 290–297
13. Mohammed O, Park D, Merchant R (1995) Practical experiences with an adaptive neural network short-term load forecasting system. IEEE Trans PWRS 10(1):254–265
14. Haykin S (1994) Neural networks a comprehensive foundation. Prentice Hall, USA
15. Mashor MY (2001) Adaptive fuzzy c-means clustering algorithm for a radial basis function network. Int J Syst Sci 32(1):53–63
16. Karl JA, Bjorm W (1997) Computer controlled systems: theory and design, 3rd edn. Prentice Hall, New Jersey
17. Gao CF, Chen TL, Nan TS (2007) Discussion of some problems about nonlinear time-series prediction using v-support vector machine. Commun Theor Phys 48:117–124
18. Muller KR, Smola AJ, Ratsch G, Scholkopf B, Kohlmorgen J (1999) Using support vector machines for time series prediction. In: Advances in kernel methods: support vector learning. MIT Press, Cambridge, USA, 1999)
19. Hyndman RJ (2010) Time series data library. http://robjhyndman.com/TSDL, 10 Feb 2010