Modified Recursive Prediction Error Algorithm
For Training Layered Neural Network

Mohd Yusoff Mashor

Centre for EElectronic Intelligent System (CELIS)
School of Electrical and Electronic Engineering, University Sains Malaysia,
Pulau Pinang, MALAYSIA.
Email: yusof@eng.usm.my

ABSTRACT

Back propagation is a steepest descent type algorithm that normally has slow learning rate and the search for the global minimum often becomes trapped at poor local minima. This paper proposes an algorithm called modified recursive prediction error (MRPE) algorithm for training multilayered perceptron networks. MRPE is a modified version of recursive prediction error (RPE) algorithm. RPE and MRPE are based on Gaussian-Newton type algorithm that generally provides better performance than a steepest type algorithm such as back propagation. The current study investigates the performance of MRPE algorithm to train MLP networks and compares its performance to the famous back propagation algorithm. Three data sets were used for the comparison. It is found that the proposed MRPE is much better than back propagation algorithm.

1. INTRODUCTION

Nowadays, artificial neural networks are studied and applied in various disciplines such as neurobiology, psychology, computer science, cognitive science, engineering, economics, medicine, etc. Tan et al. (1992) used neural networks for forecasting US-Singapore dollar exchange. Linkens and Nie (1993) applied a neuro-fuzzy controller to a problem of multivariable blood pressure control. Yu et al. (1993) used neural networks to solve the travelling salesman problem and the map-colouring problem. Arad et al. (1994) used RBF networks to recognise human facial expressions based on 2D and 3D models of the face. Rosenblum and Davis (1994) have applied RBF networks for a vehicle visual autonomous road following system. Many applications of artificial neural networks are inspired by the ability of the networks to demonstrate brain-like behaviour. Applications of artificial neural networks in these diverse fields have made it possible to tackle some problems that were previously considered to be very difficult or unsolved.

Multilayered perceptron (MLP) network trained using back propagation (BP) algorithm is the most popular choice in neural network applications. It has been shown that the network can provide satisfactory results. However, MLP network and BP algorithm can be considered as the
basic to the neural network studies. For examples RBF and HMLP networks have been proved to provide much better performance than MLP network (Chen, 1992; Mashor, 1999). Back propagation is a steepest descent type algorithm that normally has slow convergence rate and the search for the global minimum often become trapped at poor local minima. This paper proposes an algorithm called modified recursive prediction error (MRPE) algorithm for training multilayered perceptron networks.

MRPE is a modified version of recursive prediction error (RPE) algorithm. RPE and MRPE are based on Gaussian-Newton type algorithm that generally provides better performance than a steepest type algorithm such as back propagation. The current study investigates the performance of MRPE algorithm to train MLP networks and compared its performance to RPE algorithm and the famous back propagation algorithm. The comparison was carried out by using the MLP networks that were trained using the three algorithms to perform non-linear system identification.

\[ v_j^k(t) = F \left( \sum_{i=1}^{n_k} w_{ij}^k v_i^{k-1}(t) + b_j^k \right); \quad \text{for } 1 \leq j \leq n_k \] (1)

and if the \( m \)-th layer is the output layer then the output of the \( l \)-th neuron \( \hat{y}_l \) of the output layer is given by

\[ \hat{y}_l(t) = \sum_{i=1}^{n_o} w_{il}^m v_i^{m-1}(t); \quad \text{for } 1 \leq l \leq n_o \] (2)

where \( n_k, n_o \), \( w \)'s, \( b \)'s and \( F(\cdot) \) are the number of neurons in \( k \)-th layer, number of neurons in output layer, weights, thresholds and activation function respectively.

2. MULTILAYERED PERCEPTRON NETWORKS

MLP network is a feed forward neural network with one or more hidden layers. Cybenko (1989) and Funahashi (1989) have proved that the MLP network is a general function approximator and one hidden layer networks will always be sufficient to approximate any continuous function up to certain accuracy. A MLP network with two hidden layers is shown in Figure 1. The input layer acts as an input data holder that distributes the input to the first hidden layer. The outputs from the first hidden layer then become the inputs to the second layer and so on. The last layer acts as the network output layer.

A hidden neuron performs two functions that are the combining function and the activation function. The output of the \( j \)-th neuron of the \( k \)-th hidden layer, is given by

\[ v_j^k(t) = F \left( \sum_{i=1}^{n_k} w_{ij}^k v_i^{k-1}(t) + b_j^k \right); \quad \text{for } 1 \leq j \leq n_k \] (1)

and if the \( m \)-th layer is the output layer then the output of the \( l \)-th neuron \( \hat{y}_l \) of the output layer is given by

\[ \hat{y}_l(t) = \sum_{i=1}^{n_o} w_{il}^m v_i^{m-1}(t); \quad \text{for } 1 \leq l \leq n_o \] (2)

where \( n_k, n_o \), \( w \)'s, \( b \)'s and \( F(\cdot) \) are the number of neurons in \( k \)-th layer, number of neurons in output layer, weights, thresholds and activation function respectively.

![Figure 1: Multilayered perceptron networks](image-url)
In the current study, the network with a single output node and a single hidden layer was used, i.e \( m = 2 \) and \( n_o = 1 \). With these simplifications the network output is:

\[
y(t) = \sum_{i=1}^{n_i} w_i^2 v_i^1(t) = \sum_{i=1}^{n_i} w_i^2 F \left( \sum_{j=1}^{n_o} w_{ij}^1 v_j(t) + b_j^1 \right)
\]

(3)

where \( n_i \) is the number of nodes in the input layer. The activation function \( F(.) \) is selected to be

\[
F(v(t)) = \frac{1}{1 + e^{-v(t)}}
\]

(4)

The weights \( w_i \) and threshold \( b_j \) are unknown and should be selected to minimized the prediction errors defined as

\[
\varepsilon(t) = y(t) - \hat{y}(t)
\]

(5)

where \( y(t) \) is the actual output and \( \hat{y}(t) \) is the network output.

3. TRAINING ALGORITHMS

This section briefly presents back propagation and the proposed MRPE algorithms. The back propagation algorithm with momentum has been used in this study. It is well known that this version of back propagation has better learning rate compared to the original back propagation.

3.1 Back Propagation Algorithm

Back propagation algorithm was initially introduced by Werbos (1974) and further developed by Rumelhart and McClelland (1986). Back propagation is a steepest decent type algorithm where the weight connection between the \( i \)-th neuron of the \((k-1)\)-th layer and the \( j \)-th neuron of the \( k \)-th layer are respectively updated according to

\[
w_{ij}^k(t) = w_{ij}^k(t-1) + \Delta w_{ij}^k(t)
\]

\[
b_i^k(t) = b_i^k(t-1) + \Delta b_i^k(t)
\]

(6)

with the increment \( \Delta w_{ij}^k(t) \) and \( \Delta b_i^k(t) \) given by

\[
\Delta w_{ij}^k(t) = \eta_w \rho_i^k(t) v_{ji}^{k-1}(t) + \alpha_w \Delta w_{ij}^{k-1}(t-1)
\]

\[
\Delta b_i^k(t) = \eta_b \rho_i^k(t) + \alpha_b \Delta b_i^{k-1}(t-1)
\]

(7)

where the subscripts \( w \) and \( b \) represent the weight and threshold respectively, \( \alpha_w \) and \( \alpha_b \) are momentum constants which determine the influence of the past parameter changes on the current direction of movement in the parameter space, \( \eta_w \) and \( \eta_b \) represent the learning rates and \( \rho_i^k(t) \) is the error signal of the \( i \)-th neuron of the \( k \)-th layer which is back propagated in the network. Since the activation function of the output neuron is linear, the error signal at the output node is

\[
\rho^m(t) = y(t) - \hat{y}(t)
\]

(8)

and for the neurons in the hidden layer

\[
\rho_i^k(t) = F'(v_i^k(t)) \sum_j \rho_j^{k+1}(t) w_{ji}^{k+1}(t-1)
\]

\[
k = m-1, ..., 2, 1
\]

(9)

where \( F'(v_i^k(t)) \) is the first derivative of \( F(v_i^k(t)) \) with respect to \( v_i^k(t) \).

Since back propagation algorithm is a steepest decent type algorithm, the algorithm suffers from a slow convergence rate. The search for the global minima may become trapped at local minima and the algorithm can be sensitive to the user selectable parameters.
3.2 Modified Recursive Prediction Error Algorithm

Recursive prediction error algorithm (RPE) was originally derived by Ljung and Soderstrom (1983) and modified by Chen et al. (1990) to train MLP networks. RPE algorithm is a Gauss-Newton type algorithm that will generally give better performance than a steepest descent type algorithm such as back propagation algorithm. In the present study, the convergence rate of the RPE algorithm is further improved by using the optimised momentum and learning rate. The momentum and learning rate in this research are varied compared to the constant values in Chen et al. (1990).

The RPE algorithm modified by Chen et al. (1990) minimises the following cost function,

$$ J(\hat{\Theta}) = \frac{1}{2N} \sum_{t=1}^{N} e^T(t, \hat{\Theta}) \Lambda^{-1}e(t, \hat{\Theta}) $$

(10)

by updating the estimated parameter vector, \( \hat{\Theta} \) (consists of \( w \)'s and \( b \)'s), recursively using Gauss-Newton algorithm:

$$ \hat{\Theta}(t) = \hat{\Theta}(t-1) + P(t) \Delta(t) $$

(11)

and

$$ \Delta(t) = \alpha_m(t) \Delta(t-1) + \alpha_g(t) \psi(t) e(t) $$

(12)

where \( e(t) \) and \( \Lambda \) are the prediction error and \( m \times m \) symmetric positive definite matrix respectively, and \( m \) is the number of output nodes; and \( \alpha_m(t) \) and \( \alpha_g(t) \) are the momentum and learning rate respectively. \( \alpha_m(t) \) and \( \alpha_g(t) \) can be arbitrarily assigned to some values between 0 and 1 and the typical value of \( \alpha_m(t) \) and \( \alpha_g(t) \) are closed to 1 and 0 respectively. In the present study, \( \alpha_m(t) \) and \( \alpha_g(t) \) are varied to further improve the convergence rate of the RPE algorithm according to:

$$ \alpha_m(t) = \alpha_m(t-1) + a $$

(13)

and

$$ \alpha_g(t) = \alpha_m(t)(1 - \alpha_m(t)) $$

(14)

where \( a \) is a small constant (typically \( a = 0.01 \)); \( \alpha_m(0) \) is normally initialised to \( 0 \leq \alpha_m(0) < 1 \). \( \psi(t) \) represents the gradient of the one step ahead predicted output with respect to the network parameters:

$$ \psi(t, \Theta) = \left[ \frac{d\hat{y}(t, \Theta)}{d\Theta} \right] $$

(15)

\( P(t) \) in equation (11) is updated recursively according to:

$$ P(t) = \frac{1}{\lambda(t)} \left[ P(t-1) - \frac{P(t-1) \psi(t) \psi^T(t) P(t-1)}{\gamma} \right] $$

(16)

where \( \lambda(t) \) is the forgetting factor, \( 0 < \lambda(t) < 1 \), and normally been updated using the following scheme, Ljung and Soderstrom (1983):

$$ \lambda(t) = \lambda(0) \lambda(t-1) + (1 - \lambda(0)) $$

(17)

where \( \lambda_0 \) and the initial forgetting factor \( \lambda(0) \) are the design values. Initial value of \( P(t) \) matrix, \( P(0) \) is normally set to \( \alpha I \) where \( I \) is the identity matrix and \( \alpha \) is a constant, typically between 100 to 10000. Small value of \( \alpha \) will cause slow learning however too large \( \alpha \) may cause the estimated parameters do not converge properly. Hence, it should be selected to compromise between the two points, \( \alpha = 1000 \) is adequate for most cases.
The gradient matrix $\psi(t)$ for one-hidden-layer MLP network can be obtained by differentiating equation (3) with respect to the parameters, $\theta$, to yield:

$$
\psi(t) = \frac{d\hat{y}(t)}{d\theta} = \begin{cases} 
  v_j^t & \text{if } \theta_c = w_{jk}^t, \quad 1 \leq j \leq n_h \\
  v_j^t(1-v_j^t)w_j^2 & \text{if } \theta_c = b_j, \quad 1 \leq j \leq n_h \\
  v_j^t(1-v_j^t)w_{ij} & \text{if } \theta_c = w_{ij}, \quad 1 \leq j \leq n_h, 1 \leq i \leq n \\
  0 & \text{otherwise}
\end{cases}
$$

(18)

The above gradient matrix is derived based on sigmoid function therefore, if other activation functions were used the matrix should be changed accordingly.

The modified recursive prediction error algorithm (MRPE) algorithm for one hidden layer MLP network can be implemented as follows:

i) Initialise weights, thresholds, $P(0), a, b, \alpha_m(0), \lambda_0$ and $\lambda(0)$.

ii) Present inputs to the network and compute the network outputs according to equation (3).

iii) Calculate the prediction error according to equation (5) and compute matrix $\psi(t)$ according to equation (18). Note that, elements of $\psi(t)$ should be calculated from the output layer down to the hidden layer.

iv) Compute matrix $\lambda(t)$ and $P(t)$ according to equation (17) and (16) respectively.

v) If $\alpha_m(t) < b$, update $\alpha_m(t)$ according to equation (13).

vi) Update $\alpha(t)$ and then $\Delta(t)$ according to equation (11) and (12) respectively.

vii) Update parameter vector $\hat{\theta}(t)$ according to equation (11).

viii) Repeat steps (ii) to (vii) for each training data sample.

The design parameter $b$ in step (v) is the upper limit of momentum that has typical value between 0.8 to 0.9. So momentum will be increased for each data sample from a small value (normally close to 0) to this value.

4. MODELLING NON-LINEAR SYSTEMS USING MLP NETWORKS

Modelling using MLP networks can be considered as fitting a surface in a multi-dimensional space to represent the training data set and using the surface to predict over the testing data set. A wide class of non-linear systems can be represented by non-linear auto-regressive moving average with exogenous input (NARMAX) model, Leontaritis and Billings (1985). The NARMAX model can be expressed in terms of a non-linear function expansion of lagged input, output and noise terms as follows:

$$
y(t) = f_x(y(t-1), \ldots, y(t-n_y), u(t-1), \ldots, u(t-n_u), e(t-1), \ldots, e(t-n_e)) + e(t) \quad (19)
$$

where

$$
y(t) = \begin{bmatrix} y_1(t) \\
... \\
y_m(t) \\
u_1(t) \\
... \\
u_n(t) \\
e_1(t) \\
... \\
e_m(t)
\end{bmatrix}
$$

are the system output, input and noise vector respectively; $n_y, n_u$ and $n_e$ are the maximum lags in the output, input and noise vector respectively.
Modified Recursive Prediction Error Algorithm

The non-linear function \( f_s(\bullet) \) is normally very complicated and rarely known a priori for practical systems. If the mechanisms of a system are known the function \( f_s(\bullet) \) can be derived from the functions that govern those mechanisms. In the case of an unknown system, \( f_s(\bullet) \) is normally constructed based on the observation of the input and output data. In the present study, MLP networks will be used to model the input-output relationship. In other words, \( f_s(\bullet) \) will be approximated by using equation (3) where \( F(\bullet) \) is selected to be sigmoid function. The network input vector, \( \upsilon(t) \) is formed from lagged input, output and noise terms, which are denoted as \( u(t-1) \cdots u(t-n_u), y(t-1) \cdots y(t-n_y) \) and \( e(t-1) \cdots e(t-n_e) \) respectively in equation (19).

The final stage in system identification is model validation. There are several ways of testing a model such as one step ahead predictions (OSA), model predicted outputs (MPO), mean squared error (MSE), correlation tests and chi-squares tests. In the present study, only OSA and MSE tests will be used since it is not easy to see the performance different using other tests.

OSA is a common measure of predictive accuracy of a model that has been considered by many researchers. OSA can be expressed as:

\[
\hat{y}(t) = f_s[u(t-1), \cdots, u(t-n_u), y(t-1), \cdots, y(t-n_y), e(t-1), \cdots, e(t-n_e)]
\]

and the residual or prediction error is defined as:

\[
\hat{\varepsilon}(t, \hat{\theta}) = y(t) - \hat{y}(t)
\]

where \( f_s(\bullet) \) is a non-linear function, in this case the MLP network. A good model will normally give a good prediction, however a model that has a good one step ahead prediction might not always be unbiased. The model may be significantly biased and prediction over a different set of data often reveals this problem. Splitting the data into two sets, a training set and a testing set, can normally detect this condition.

MSE is an iterative method of model validation where the model is tested by calculating the mean squared errors after each training step. MSE test will indicate how fast a prediction error or residual converges with the number of training data. The MSE at the \( t \)-th training step, is given by:

\[
E_{\text{MSE}}(t, \Theta(t)) = \frac{1}{n_d} \sum_{i=1}^{n_d} (y(i) - \hat{y}(i, \Theta(t)))^2
\]

where \( E_{\text{MSE}}(t, \Theta(t)) \) and \( \hat{y}(i, \Theta(t)) \) are the MSE and OSA for a given set of estimated parameters \( \Theta(t) \) after \( t \) training steps respectively, and \( n_d \) is the number of data that were used to calculate the MSE.

5. SIMULATION RESULTS

The performance of MLP networks trained using the BP, RPE and MRPE algorithms presented in section 3 were compared. One simulated and two real data sets were used for this comparison. The networks were used to perform system identification and the resulting models were used to produce OSA and MSE tests.
Example 1

The first data set is a simulated system defined by the following difference equation:

\[ y(t) = 0.3y(t-1) + 0.6y(t-2) + u^2(t-1) + 0.3u^2(t-1) - 0.4u(t-1) + e(t) \]

where \( e(t) \) is a Gaussian white noise sequence with zero mean and variance 0.005 and the input, \( u(t) \) is a uniformly random sequence between (-1,+1). This system was used to generate 1000 pairs of data input and output. The first 600 data were used to train the network and the remaining 400 data were used to test the fitted model. The network was trained based on the following configuration:

\[ v(t) = [u(t-1) \quad y(t-1) \quad y(t-2)] \]

All the network models have the same input \( v(t) \) and 7 hidden nodes but different training algorithm. The designing parameters for BP, RPE and MRPE algorithms were set as follows,

BP algorithm:

\[ \eta_w = \eta_b = 0.0002 \text{ and } \alpha_w = \alpha_b = 0.85 \]

RPE algorithm:

\[ P(0) = 1000I, \quad \alpha_m = 0.85, \quad \alpha_g = 0.1, \quad \lambda_0 = 0.99 \text{ and } \lambda(0) = 0.95. \]

MRPE algorithm:

\[ P(t) = 1000I, \quad \alpha_m(0) = 0.2, \quad a = 0.01, \quad b = 0.85, \quad \lambda_0 = 0.99 \text{ and } \lambda(0) = 0.95. \]

The MSE calculated over both the training and testing data sets for the network models trained using BP, RPE and MRPE are shown in Figure (2) and (3) respectively. These figures also indicate that the MRPE and RPE algorithms produced much better MSE than the one trained using BP algorithm. These figures also indicate that the MRPE and RPE algorithms produced about the same performance.

OSA tests over the training and testing data sets for the network models trained using BP, RPE and MRPE are shown in Figure (4), (5) and (6) respectively. These plots again show that RPE and MRPE have similar performance. However, the OSA test produced by the network trained using BP algorithm is much worse than the network models that have been trained using RPE and MRPE algorithms. The OSA test in Figure (4) shows that BP trained-network fail to predict properly.

![Figure 2: MSE calculated over training data set](image)

![Figure 3: MSE calculated over testing data set](image)
Example 2

The second data set was taken from a heat exchanger system and consists of 1000 samples. The first 500 data were used to train the network and the remaining 500 data were used to test the fitted network model. The network has been trained using the following specification:

\[ v(t) = [u(t-1) \ u(t-2) \ y(t-1) \ y(t-4) \ e(t-3) \ e(t-4) \ e(t-5) \text{ and bias input}] \]

All the network models have the same structure but different training algorithm.

The design parameters for BP, RPE and MRPE algorithms were set as follows,

BP algorithm:
\[ \eta_w = \eta_b = 0.0002 \text{ and } \alpha_w = \alpha_b = 0.85 \]

RPE algorithm:
\[ P(0) = 1000I, \ \alpha_m = 0.85, \ \alpha_g = 0.1, \ \lambda_0 = 0.99 \text{ and } \lambda(0) = 0.95. \]

MRPE algorithm:
\[ P(t) = 1000I, \ \alpha_m(0) = 0.6, \ a = 0.01, \ b = 0.85, \ \lambda_0 = 0.99 \text{ and } \lambda(0) = 0.95. \]

The MSE calculated over both the training and testing data sets for the network models trained using BP and MRPE algorithms are shown in Figure (7) and (8) respectively. These figures show that MRPE algorithm produced significantly better MSE than RPE algorithm and much better than BP algorithm.
OSA tests over the training and testing data sets for the network models trained using BP, RPE and MRPE are shown in Figure (9), (10) and (11) respectively. The result in Figure (9) reconfirms that the network trained using BP algorithm cannot predict properly where the prediction over both training and testing data set are not satisfactory. The networks trained using RPE and MRPE algorithms on the other hand predict very well over both the training and testing data sets. Referring to Figure (10) and (11), it can be said that the network trained using MRPE algorithm give significantly better prediction (OSA test) compared to the network trained using RPE algorithm.
Example 3

A data set of 1000 input-output samples were taken from a tension leg platform. Description of the process can be found in Mashor (1995). The data set consists of 1000 data samples where the first 600 data samples were used for training and the next 400 data samples were used for testing. The network has been trained using the following specification:

$v(t) = [u(t-1) \cdots u(t-8) y(t-1) \cdots y(t-4) e(t-3) e(t-5)]$ and bias input

BP algorithm:

$\eta_w = \eta_b = 0.001$ and $\alpha_w = \alpha_b = 0.85$

RPE algorithm:

$P(0) = 1000I$, $\alpha_m = 0.85$, $\alpha_g = 0.07$, $\lambda_0 = 0.99$ and $\lambda(0) = 0.95$.

MRPE algorithm:

$P(t) = 1000I$, $\alpha_m(0) = 0.4$, $a = 0.01$, $b = 0.85$, $\lambda_0 = 0.99$ and $\lambda(0) = 0.95$.

The MSE calculated over both training and testing data sets for the network models trained using BP, RPE and MRPE algorithms are shown in Figure (12) and (13) respectively. In this example the network models trained using MRPE and RPE algorithms produced much better MSE than the one trained using BP algorithm. These figures also indicate that the MRPE is significantly better than RPE algorithm. Figure (12) and (13) also suggesting that the network trained using BP algorithm cannot learn properly where the convergence of its MSE values are not significant.
OSA tests over the training and testing data sets for the network models trained using BP, RPE and MRPE are shown in Figure (14), (15) and (16) respectively. For this example it is quite hard to distinguish the performance advantage between the networks trained using MRPE and RPE algorithms. However, the OSA tests produced by the two networks are much better than the one produced by the network model that have been trained using BP algorithm.

All the three examples show that the MLP network trained using BP algorithm could not learn properly. This is because the learning rate of BP is very slow. Thus further analysis has been carried out to check how good is BP with more training epochs. Figure (17), (18) and (19) show the MSE plots over the training data set produced by BP algorithm after 100 epochs for the system in example 1, 2 and 3 respectively. Each figure has three MSE plots for different learning rates. The network specification for the networks were the same as in previous analysis except for the learning rates that were assign as indicated in the respective figures. Comparing these results with the ones in Figure (2), (7) and (12) it was found that BP algorithm cannot train the MLP network as good as RPE and MRPE even after 100 training epochs. Therefore, it can be deduced that the learning rate of BP algorithm is very slow compare to MRPE and RPE algorithms.
6. CONCLUSION

MRPE algorithm is proposed to train MLP network and its performance was compared to RPE and BP algorithms. Three data sets were used to test the performance of the algorithms. The MSE and OSA tests for the examples indicated that the MRPE has significantly improved the performance of RPE algorithm especially for the two real data sets. The results also proved that both RPE and MRPE algorithms are much better than BP algorithm. The performance of BP algorithm with 100 training epochs still cannot compete with MRPE algorithm with one training epoch. Hence, it can be concluded that MRPE has much faster learning rate than BP algorithm and does not require multiple training epoch.

REFERENCES


