

# PERFORMANCE COMPARISON BETWEEN BACK PROPAGATION, RPE AND MRPE ALGORITHMS FOR TRAINING MLP NETWORKS

**Mohd Yusoff Mashor**

School of Electrical and Electronic Engineering,  
University Science Malaysia,  
Perak Branch Campus,  
31750 Tronoh, Perak,  
Malaysia.  
Email: yusof@eng.usm.my

## **ABSTRACT**

*This paper presents the performance comparison between back propagation, recursive prediction error (RPE) and modified recursive prediction error (MRPE) algorithms for training multilayered perceptron networks. Back propagation is a steepest descent type algorithm that normally has slow convergence rate and the search for the global minimum often becomes trapped at poor local minima. RPE and MRPE are based on Gaussian-Newton type algorithm that generally provides better performance. The current study investigates the performance of three algorithms to train MLP networks. Two real data sets were used for the comparison. It was found that the RPE and MRPE are much better than the 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 unsolvable.

Multilayered perceptron (MLP) networks trained using back propagation (BP) algorithm are 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 basis 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, 1999a). The current study compares the performance of BP, RPE and MRPE to train MLP networks. The comparison was carried out by using the MLP networks that were trained using the three algorithms to perform non-linear system identification.

## 2. MULTILAYERED PERCEPTRON NETWORKS

MLP networks are feed forward neural networks with one or more hidden layers. Cybenko (1989) and Funahashi (1989) have proved that the MLP network is a general function approximator and that 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

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

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

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

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 an activation function respectively.

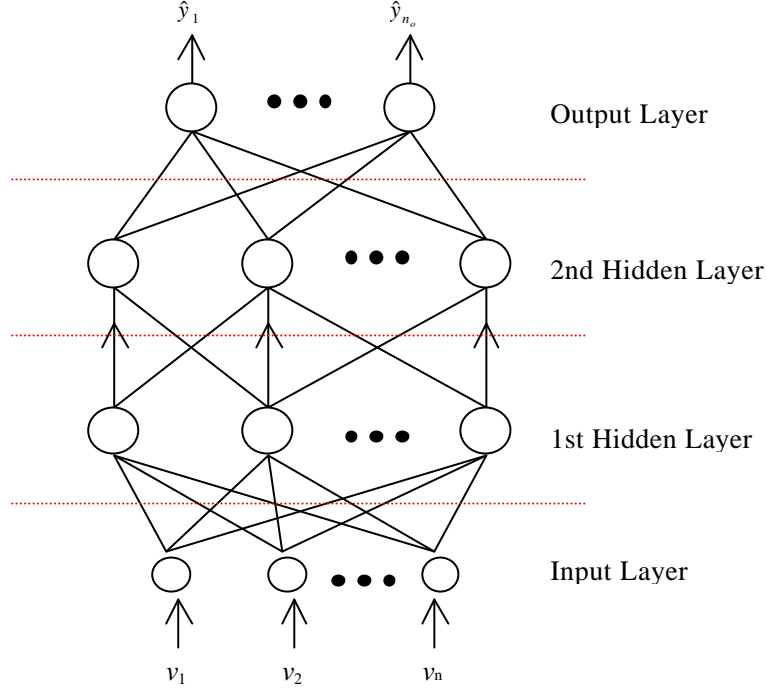


Figure 1: Multilayered perceptron networks

In the current study, the network with a single output node and a single hidden layer is used, i.e  $m = 2$  and  $n_o = 1$ . With these simplifications the network output is:

$$\hat{y}(t) = \sum_{i=1}^{n_1} w_i^2 v_i^1(t) = \sum_{i=1}^{n_1} w_i^2 F\left(\sum_{j=1}^{n_r} w_{ij}^1 v_j(t) + b_j^1\right) \quad (3)$$

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

$$F(v(t)) = \frac{1}{1 + e^{-v(t)}} \quad (4)$$

The weights  $w_i$  and threshold  $b_j$  are unknown and should be selected to minimise the prediction errors defined as

$$\varepsilon(t) = y(t) - \hat{y}(t) \quad (5)$$

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

### 3. TRAINING ALGORITHMS

This section briefly presents back propagation, RPE and MRPE algorithms. Back propagation and RPE algorithms are based on paper by Chen et. al. (1990) and MRPE is based on paper by Mashor (1999b).

#### 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 the steepest decent type algorithm where the weight connection between the  $j$ -th neuron of the  $(k-1)$ -th layer and the  $i$ -th neuron of the  $k$ -th layer are respectively updated according to

$$\begin{aligned} w_{ij}^k(t) &= w_{ij}^k(t-1) + \Delta w_{ij}^k(t) \\ b_i^k(t) &= b_i^k(t-1) + \Delta b_i^k(t) \end{aligned} \quad (6)$$

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

$$\begin{aligned} \Delta w_{ij}^k(t) &= \eta_w \rho_i^k(t) v_j^{k-1}(t) + \alpha_w \Delta w_{ij}^k(t-1) \\ \Delta b_i^k(t) &= \eta_b \rho_i^k(t) + \alpha_b \Delta b_i^k(t-1) \end{aligned} \quad (7)$$

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

$$\rho^m(t) = y(t) - \hat{y}(t) \quad (8)$$

and for the neurons in the hidden layer

$$\rho_i^k(t) = F'(v_i^k(t)) \sum_j \rho_j^{k+1}(t) w_{ji}^{k+1}(t-1) \quad k = m-1, \dots, 2, 1 \quad (9)$$

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 RPE and MRPE Algorithms

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 RPE (Mashor, 1999b). The momentum and learning rate in this research are varied compared to the constant values in Chen et al. (1990). This modified RPE is originally proposed by Mashor (1999b) and referred as *modified recursive prediction error* (MRPE) algorithm.

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

$$J(\hat{\Theta}) = \frac{1}{2N} \sum_{t=1}^N \varepsilon^T(t, \hat{\Theta}) \Lambda^{-1} \varepsilon(t, \hat{\Theta}) \quad (10)$$

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

$$\hat{\Theta}(t) = \hat{\Theta}(t-1) + \mathbf{P}(t)\Delta(t) \quad (11)$$

and

$$\Delta(t) = \alpha_m(t)\Delta(t-1) + \alpha_g(t)\psi(t)\varepsilon(t) \quad (12)$$

where  $\varepsilon(t)$  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 (Mashor, 1999b):

$$\alpha_m(t) = \alpha_m(t-1) + a \quad (13)$$

and

$$\alpha_g(t) = \alpha_m(t)(1 - \alpha_m(t)) \quad (14)$$

where  $a$  is a small constant (typically  $a = 0.01$ );  $\alpha_m(t)$  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:

$$\psi(t, \Theta) = \left[ \frac{d\hat{y}(t, \Theta)}{d\Theta} \right] \quad (15)$$

$\mathbf{P}(t)$  in equation (11) is updated recursively according to:

$$\mathbf{P}(t) = \frac{1}{\lambda(t)} \left[ \mathbf{P}(t-1) - \mathbf{P}(t-1)\psi(t)(\lambda(t)\mathbf{I} + \psi^T(t)\mathbf{P}(t-1)\psi(t))^{-1} \psi^T(t)\mathbf{P}(t-1) \right] \quad (16)$$

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

$$\lambda(t) = \lambda_0 \lambda(t-1) + (1 - \lambda_0) \quad (17)$$

where  $\lambda_0$  and the initial forgetting factor  $\lambda(0)$  are the design values. Initial value of  $\mathbf{P}(t)$  matrix,  $\mathbf{P}(0)$  is normally set to  $\alpha\mathbf{I}$  where  $\mathbf{I}$  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  $\psi(t)$  can be modified to accommodate the extra linear connections for one-hidden-layer HMLP network model by differentiating equation (3) with respect to the parameters,  $\theta_c$ , to yield:

$$\psi_k(t) = \frac{dy_k(t)}{d\theta_c} = \begin{cases} v_j^1 & \text{if } \theta_c = w_{jk}^2 & 1 \leq j \leq n_h \\ v_j^1(1-v_j^1)w_{jk}^2 & \text{if } \theta_c = b_j^1 & 1 \leq j \leq n_h \\ v_j^1(1-v_j^1)w_{jk}^2 v_i^0 & \text{if } \theta_c = w_{ij}^1 & 1 \leq j \leq n_h, 1 \leq i \leq n \\ 0 & \text{otherwise} \end{cases} \quad (18)$$

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 RPE algorithm for one hidden layer MLP network can be implemented as follows (Mashor, 1999b):

- i Initialise weights, thresholds,  $\mathbf{P}(0)$ ,  $a$ ,  $b$ ,  $\alpha_m(0)$ ,  $\lambda_0$  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  $\mathbf{P}(t)$  according to equation (12) and (11) respectively.
- v If  $\alpha_m(t) < b$ , update  $\alpha_m(t)$  according to equation (13).
- vi Update  $\alpha_g(t)$  and then  $\Delta(t)$  according to equation (14) and (12) respectively.
- vii Update parameter vector  $\Theta(t)$  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 HMLP 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. Therefore, MLP networks require all the future data of the system to lie within the domain of the fitted surface to ensure a correct mapping so that good predictions can be achieved. This is normal for the non-linear modelling where the model is only valid over a certain amplitude range.

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:

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

where

$$y(t) = \begin{bmatrix} y_1(t) \\ \vdots \\ y_m(t) \end{bmatrix}, \quad u(t) = \begin{bmatrix} u_1(t) \\ \vdots \\ u_r(t) \end{bmatrix} \quad \text{and} \quad e(t) = \begin{bmatrix} e_1(t) \\ \vdots \\ e_m(t) \end{bmatrix}$$

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.

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,  $v(t)$  is formed from lagged input, output and noise terms, which are denoted as  $u(t-1) \dots u(t-n_u)$ ,  $y(t-1) \dots y(t-n_y)$  and  $e(t-1) \dots e(t-n_e)$  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, OSA,

MPO, MSE and correlation tests were used to justify the performance of the fitted network models. In the present study, only OSA test and MSE test 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:

$$\hat{y}(t) = f_s\left(u(t-1), \dots, u(t-n_u), y(t-1), \dots, y(t-n_y), \varepsilon(t-1, \hat{\theta}), \dots, \varepsilon(t-n_\varepsilon, \hat{\theta})\right) \quad (20)$$

and the residual or prediction error is defined as:

$$\hat{\varepsilon}(t, \hat{\theta}) = y(t) - \hat{y}(t) \quad (21)$$

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 and model predicted output may 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 can test this condition, a training set and a testing set.

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:

$$E_{MSE}(t, \Theta(t)) = \frac{1}{n_d} \sum_{i=1}^{n_d} (y(i) - \hat{y}(i, \Theta(t)))^2 \quad (22)$$

where  $E_{MSE}(t, \Theta(t))$  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. PERFORMANCE COMPARISON

The performance of MLP networks trained using the BP, RPE and MRPE algorithms presented in section 3 were compared. 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 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:

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

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:

$$\eta_w = \eta_b = 0.0005 \text{ and } \alpha_w = \alpha_b = 0.85$$

RPE algorithm:

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

MRPE algorithm:

$$\mathbf{P}(t) = 1000\mathbf{I}, \alpha_m(0) = 0.6, a = 0.01, b = 0.85, \lambda_0 = 0.99 \text{ and } \lambda(0) = 0.95.$$

The MSE calculated over both training and testing data sets for the network models trained using BP, RPE and MRPE algorithms are shown in Figure (2) and (3) respectively. Both figures indicated that MRPE produces MSE that is significantly better than RPE and much better than BP algorithm. These figures also suggest that the network trained using BP algorithm do not have good generalisation since the performance different over the testing data set is larger than the one over the training data set.

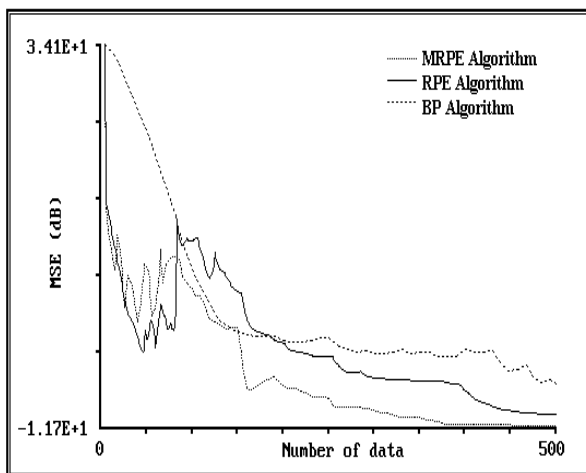


Figure 2: MSE calculated over training data set

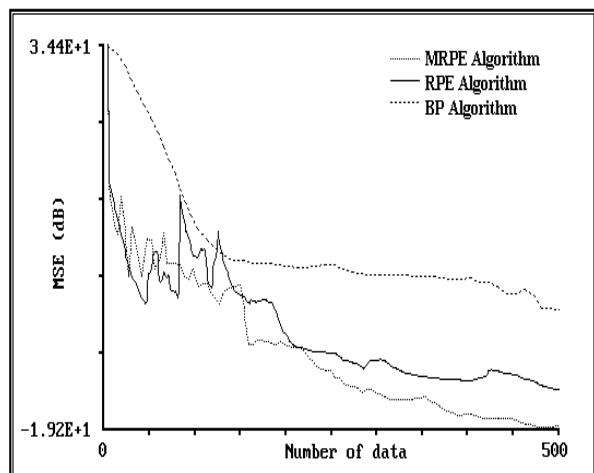


Figure 3: MSE calculated over testing data set

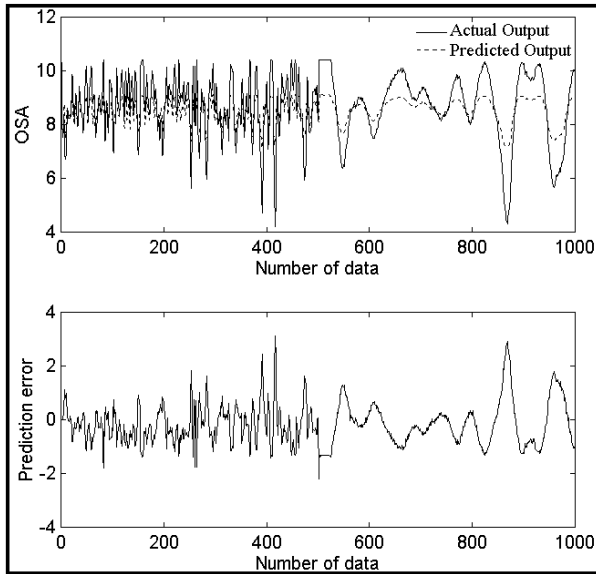


Figure 4: OSA test for MLP network trained using BP algorithm

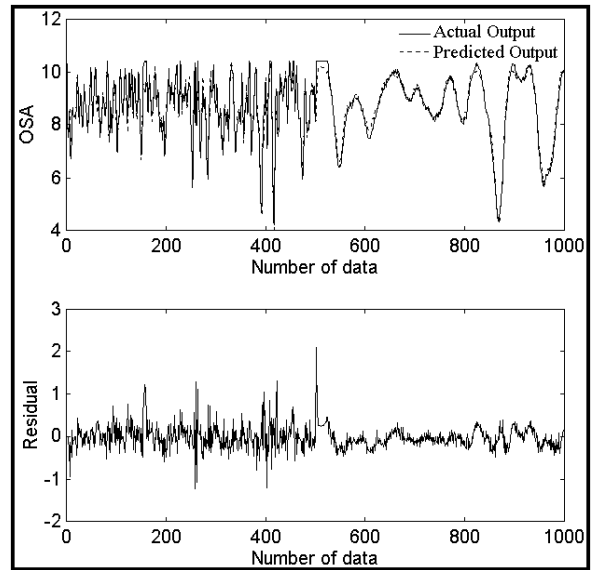


Figure 5: OSA test for MLP network trained using RPE algorithm

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. The result in Figure (4) reconfirms that the network trained using BP algorithm cannot predict properly and do not have good generalisation where the prediction over testing data set is 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 (5) and (6), it can be said that the network trained using MRPE algorithm gives significantly better prediction (OSA test) compared to the network trained using RPE algorithm.

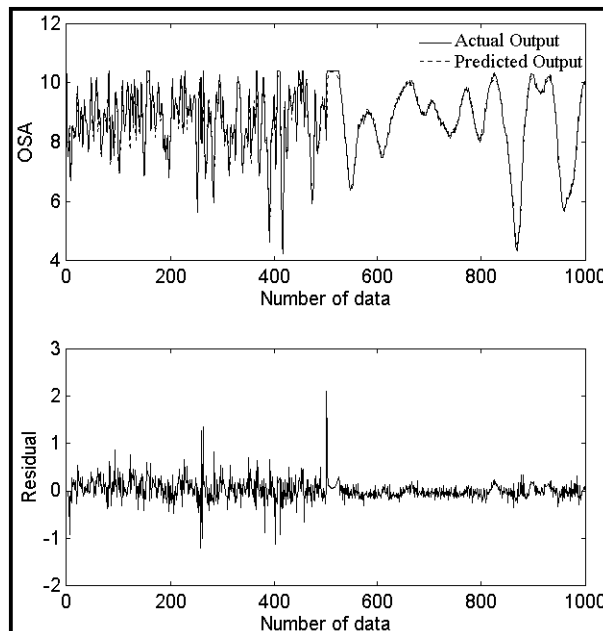


Figure 6: OSA test for MLP network trained using MRPE algorithm

*Example 2*

The second system is a tension legs data consist 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:

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

BP algorithm:

$$\eta_w = \eta_b = 0.001 \text{ and } \alpha_w = \alpha_b = 0.85$$

RPE algorithm:

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

MRPE algorithm:

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

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 (7) and (8) respectively. In this example the network models trained using MRPE and RPE algorithms produced MSE that are much better than the one trained using BP algorithm. These figures also indicate that the MRPE is significantly better than RPE algorithm. Figure (7) and (8) also suggest that the network trained using BP algorithm do not have good generalisation since the performance different over the testing data set is larger than the one over the training data set.

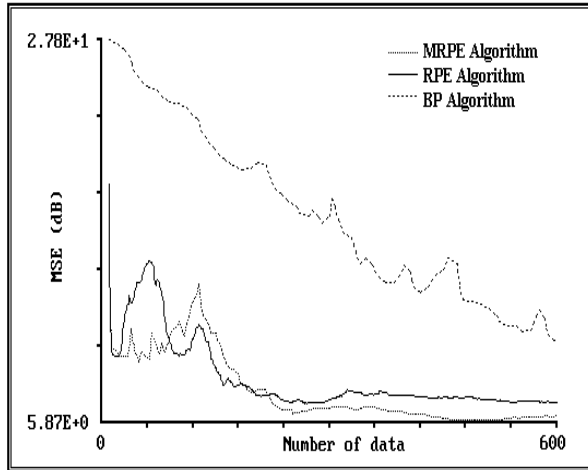


Figure 7: MSE calculated over training data set

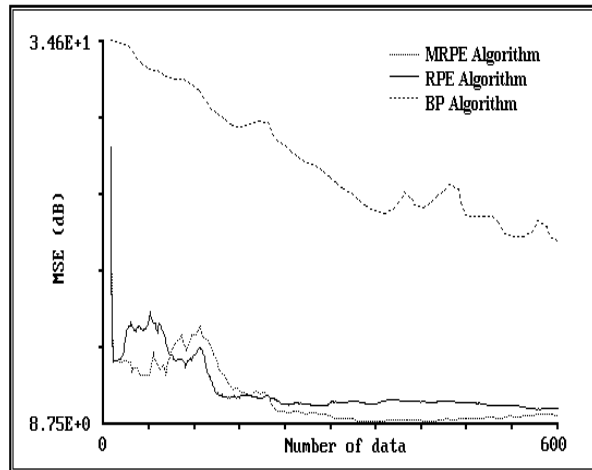


Figure 8: MSE calculated over testing data set

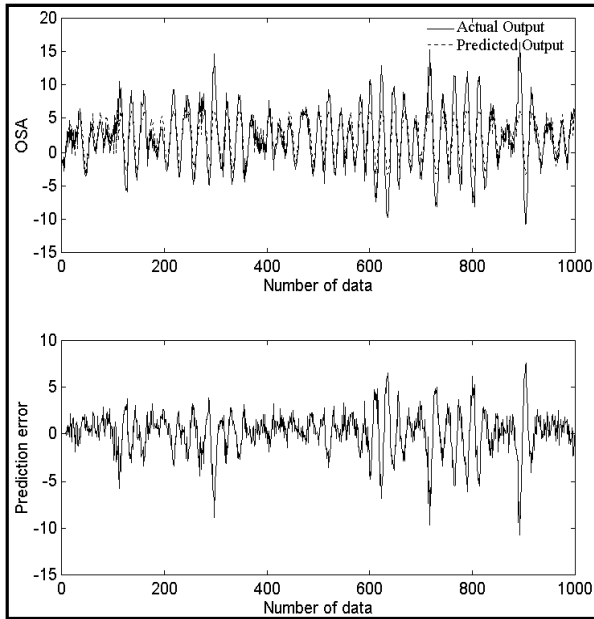


Figure 9: OSA test for MLP network trained using BP algorithm

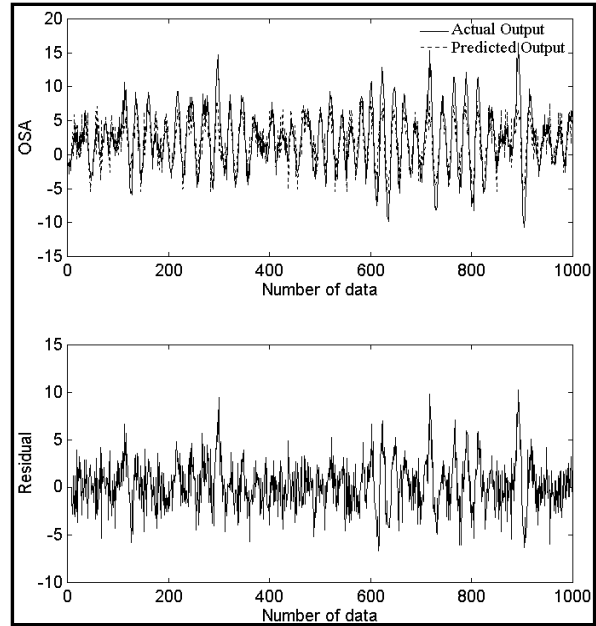


Figure 10: OSA test for MLP network trained using RPE algorithm

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. For this example it is quite hard to distinguish the performance advantage between the networks trained using BP and RPE algorithms. However, the OSA test produced by the network trained using MRPE is significantly better than the network models that have been trained using BP and RPE algorithms.

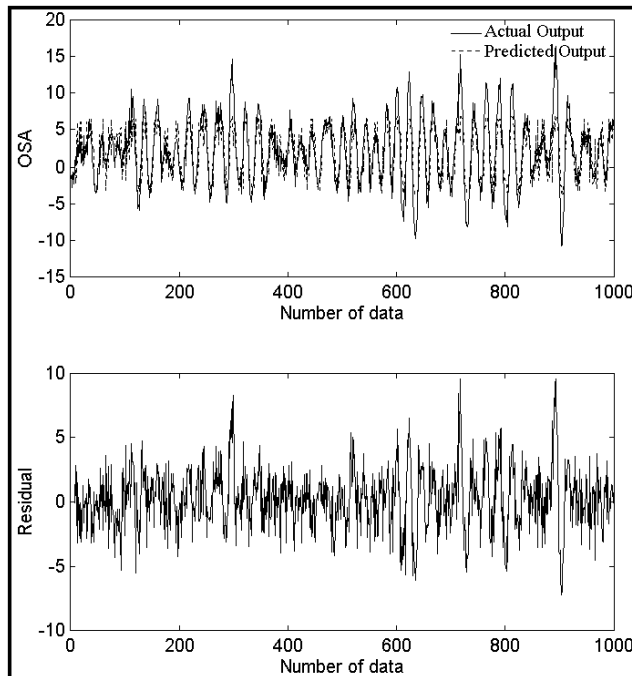


Figure 11: OSA test for MLP network trained using MRPE algorithm

## 6. CONCLUSION

BP, RPE and MRPE algorithms are briefly discussed and used to train MLP networks to perform system identification. Two real data sets were used to test the performance of the algorithms. The MSE and OSA tests for both examples indicated that the MRPE has significantly improved the performance of RPE algorithm and both RPE and MRPE algorithm are much better than BP algorithm. The results also suggest that the network trained using BP algorithm does not normally have good generalisation where the prediction over the testing data set is not as good as over the training data set.

## REFERENCES

- [1] Arad, N., Dyn, N., Reissfeld, D., and Yeshurun, Y., 1994, "Image warping by radial basis functions: application to facial expressions", *CVGIP: Graphical Models and Image Processing*, **56** (2), 161-172.
- [2] Chen, S., Cowan, C.F.N., Billings, S.A., and Grant, P.M., 1990, "A parallel recursive prediction error algorithm for training layered neural networks", *Int. J. Control*, **51** (6), 1215-1228.
- [3] Chen, S., and Billings, S.A., 1992, "Neural networks for non-linear dynamic system modelling and identification", *Int. J. of Control*, **56** (2), 319-346.
- [4] Cybenko, G., 1989, "Approximations by superposition of a sigmoidal function", *Mathematics of Control, Signal and Systems*, **2**, 303-314.
- [5] Funahashi, K., 1989, "On the approximate realisation of continuous mappings by neural networks", *Neural Networks*, **2**, 183-192.
- [6] Leontaritis, I.J., and Billings, S.A., 1985, "Input-output parametric models for non-linear systems. Part I - Deterministic non-linear systems. Part II - Stochastic non-linear systems", *Int. J. Control*, **41**, 303-359
- [7] Linkens, D.A., and Nie, J., 1993, "Fuzzified RBF network-based learning control: Structure and self-construction", *IEEE Int. Conf. on Neural Networks*, **2**, 1016-1021.
- [8] Ljung, L., and Soderstrom, T., 1983, *Theory and Practice of Recursive Identification*, MIT Press, Cambridge.
- [9] Mashor, M.Y., 1999a, "Performance Comparison Between HMLP and MLP Networks", *Int. Conf. on Robotics, Vision & Parallel Processing for Automation (ROVPIA'99)*, pp 123-132.
- [10] Mashor, M.Y., 1999b, "Hybrid multilayered perceptron networks", accepted for publication by *Int. J. of System and Science*.

- [11] Rosenblum, M., and Davis, L.S., 1994, "An improved radial basis function network for visual autonomous road following", *Proc. of the SPIE - The Int. Society for Optical Eng.*, **2103**, 209-225.
- [12] Rumelhart, D.E., and McClelland, J.L., 1986, *Parallel distributed processing: explorations in the microstructure of cognition, I & II*, MIT Press, Cambridge, MA
- [13] Tan, P.Y., Lim, G., Chua, K., Wong, F.S., and Neo, S., 1992, "Comparative studies among neural nets, radial basis functions and regression methods", *ICARCV '92. Second Int. Conf. on Automation, Robotics and Computer Vision*, **1**, NW-3.3/1-6.
- [14] Yu, D.H., Jia, J., and Mori, S., 1993, "A new neural network algorithm with the orthogonal optimised parameters to solve the optimal problems", *IEICE Trans. on Fundamentals of Electronics, Comm. and Computer Sciences*, **E76-A** (9), 1520-1526.
- [15] Werbos, P.J., 1974, "*Beyond Regression: New Tools for Prediction and Analysis in the Behavioural Sciences*", Ph.D. Thesis, Harvard University.
-