A NEURAL NETWORK PREDICTION MODEL FOR SURFACE FINISH IN TURNING PROCESS

Dr. Bahaa Ibraheem Kazem

Department of Mechatronic Eng., Al-khwarizmi college of Eng. , University of Baghdad , Al- Jadiryah campus , AL- Jadiryah Baghdad , IRAQ .

E. mail : bahaa51170@hotmail.com

ABSTRACT

This paper presents a neural network based surface finish Prediction model in turning operation , Orthogonal cutting tests were performed on mild steel using H.S.S cutting tool with different cutting parameters cutting speed , feed and nose radius of the cutting tool . The collected data was used to train feed forward back propagation neural network. The developed model has been tested to predict surface finish for various cutting conditions. The model was found to be powerful & capable of accurate surface finish prediction for the range it had been trained but the accuracy deteriorated as the cutting conditions were changed significantly.

الخلاصة

يتضمن هذا البحث تصميم نموذج لشبكة عصبية للتنبأ بخواص انهاء السطوح في عمليات القطع حيث تمت دراسة عمليات قطع عمودي على عينات من سبائك الحديد المطاوع وبأستخدام عدة قطع فولاذ السرع العالية ولظروف قطع مختافة (سرعة قطع وتغذية ونصف قطر مقدمة قلم القطع). تم استخدام المعلومات المستخلصة لتعليم شبكة عصبية ذات تغذية امامية مرتدة. تم اختبار النموذج للتنبأ بخواص انهاء السطوح لظروف قطع مختلفة حيث وجد ا ن للنموذج قابلية عالية ودقيقة للتنبأ بانهاء السطوح ضمن المدى الذي تم تدريبه عليه وان الدقة تتلاشى عند تغيير ظروف القطع تغيرا جذريا.

KEY WORDS

Surface finish, neural net work, prediction model

INTRODUCTION

Machining or metal removal processes such turning are widely used in Manufacturing. Productivity and quality in the finish turning of hardened steels can be improved by optimum selection of the cutting conditions, and because of the complicated relationships between the parameters of cutting operation, the machining process is hard to be decomposed or described by classical differential equations due to larger number of variables & their stochastic, nonlinear relations.

The effect of the cutting parameters on the surface roughness in a turning operation has been investigated [M Hasegawa, 1976]. The mathematical prediction models for surface roughness have been obtained for common mild steel.

Neural networks have achieved in recent years a high degree of importance. The availability of process control computers as well as data historians made it easy to develop neural network solutions for process modeling and control. From medical to industrial application, neural networks have been applied in countless number of situations. Lately, mainly due to the increasing pressure from consumption markets a lot of research has been done under the field of fault detection, diagnosis and quality control. The needs produce more and better, informed markets, has had to the investigation of new methodologies in the quality control area [N. Costa, 1998] Neural computing , as one of such techniques , became an attractive approach in this area, since neural networks are adaptable to an involving environment and are able to take a quick decision once they have learned the proper control function . Artificial neural networks (ANN) because they are cost – effective , easy to understand and because of their ability to learn from examples, have found many applications in process modeling and control as intelligent sensors, to estimate variables that usually can be measured on - line in dynamic system identification in fault defection diagnosis and finally . in process control [N.Costa,1998].

Tugral et.al. [Tugral Ozel,2002] Outlined a neural network based tool condition monitoring system, (TCMS), for cutting tool state classification. Orthogonal cutting tests were performed on H13 steel using PCBN inserts and on line cutting forces data was acquired with a piezoelectric force dynamometer. Simultaneously flank wear data was measured using a tool maker's microscope and along with the processed data were fed to aback propagation neural network to be trained. The developed system then was tested to predict flank wear for various cutting condition. The system was found to be capable of accurate tool wear prediction for the range it had been trained but the accuracy deteriorated as the cutting condition were changed significantly.

R.G. Khunchustombham et.al [R.G. Khunchustombham,2001] showed that it can use a neural network approach to on tina monitoring of a turning process emphisis is given to applying neural networks to perform information processing and to recognize the process abnormalities in a machining operation . A neural network monitor based on a feed forward back- propagation algorithm is developed. The monitor is trained by detect cutting force signal and measured surface finish.

Ahmad Ghasempoor [Ahmed Ghasempoor,1997] described a methodology for on – line adjustment of cutting conditions in a turning operation. The system presented consists of on – line wear monitoring and optimal adjustment of cutting conditions. A practical optimization goal has been defined. Simulation results the feasibility of the proposed method.

Sarah. et.al. [Sara S.Y,] discussed the preliminary development of a neural network based process monitor and off – line controller for abrasive flow machining of automotive engine intake manifolds. The process is only observable indirectly, yet the time at which machining achieves the specified air flow rate must be estimated accurately. A neural network model is used to estimate when the process has achieved air flow specification so that machining can be terminated. This model uses surrogate process parameters as inputs because of the inaccessibility of the product parameter of interest, air flow rate through the manifold.

NEURAL NETWORK MODEL

In this study two neural network models are used. The first model is the back propagation training neural network (BPTNN) and the second one is the back propagation prediction neural network (BPPNN)

Neural Network Layers

The network always consists of at lest three layers of processing elements **Fig (1)**. This model has three layers of processing elements: a) The input layer b) The middle layer c) the output layer (as shown in **Fig. (2)**)

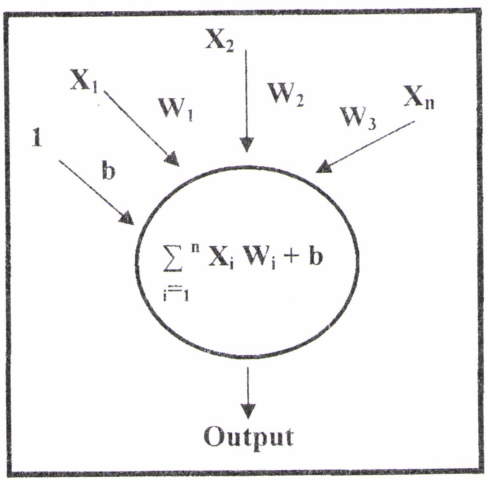

Fig. (1) Illustration of
a processing element

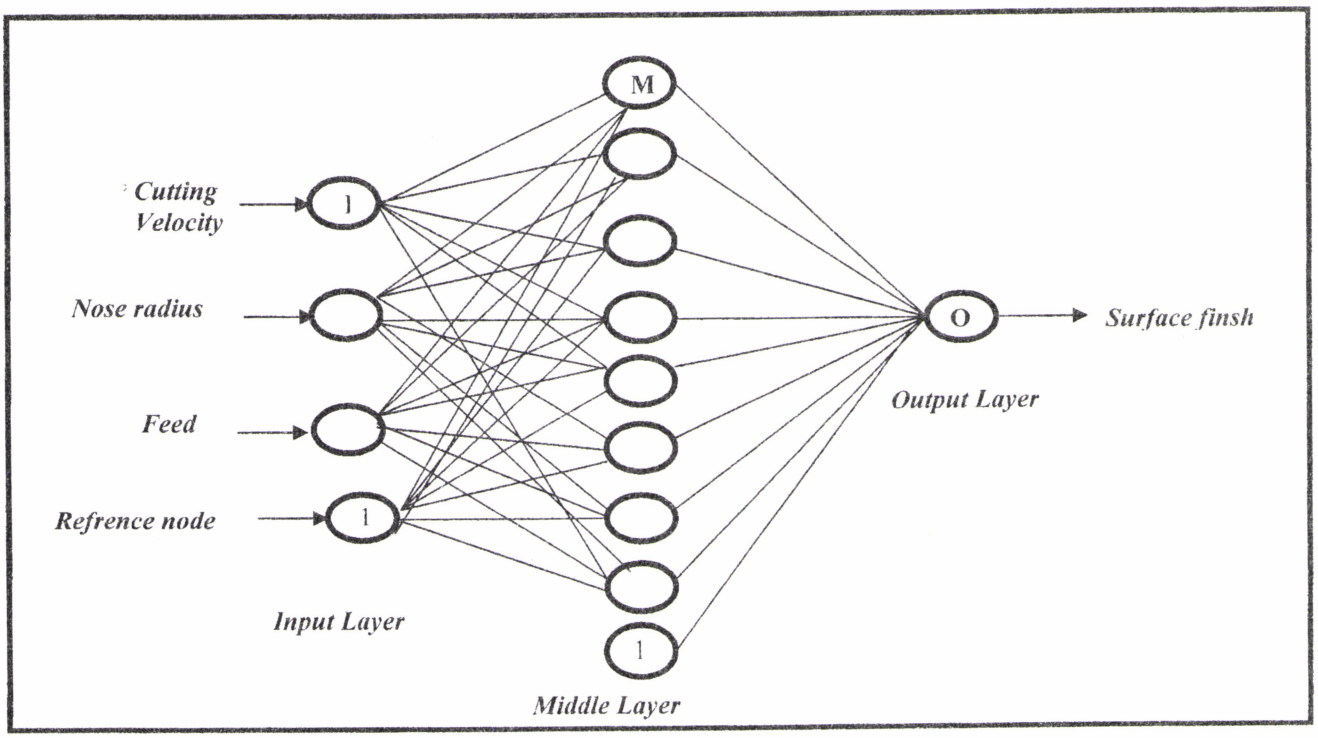

Fig. (2) NETWORK ARCHITCTURE

The input variable is cutting velocity, feed, and nose radius the depth of cut was 0.25 mm. Each input variable is assigned to a single input layer-processing element. The output value at the output layer processing elements is represented as O the surface roughness

The input variable values are carried over to the output of the input layer without any processing. The output values of the input layer processing units are represented by I, (where I=1, 2, 3...) the number of middle layer processing elements is determined by trial and error after testing the model

for 2to 8 processing element at the hidden layer. The output values of the middle layer-processing element are represented by Mj (where j=1,2,3…8). The output layer, which consists of a single processing element, surface roughness

Training Patterns

The training patterns include three input variable and the measured surface roughness value. In this study, 32 different training sets for cutting experiments were used as training patterns as shown in **Table (1).**

Interconnection Weights

AII of the processing elements at input layer are connected to each of the middle layer processing elements via interconnections, which are weighted and represented by W_{ji} (where j=1, 2, 8 and I=1, 2, 3 …).

The output and the middle layer elements also connected similarly and the interconnection weights are represented by W_{kj} (where K = 1, … and j= 1, 2, 8…)

The interconnections weights are unknown to the process. A random number generator is initially used in assigning weights to the inter- layer connections. These weights are trained, using the BPTNN model; to match the input patters to the output surface roughness.

Back Propagation Training Neural Network-Forward Passes

Every input layer-processing element, I, is multiplied by the corresponding weight on the inter-layer connections of a middle layer processing element. All the products of I, and W_{ji} are then summed and form the input to a middle layer-processing element. A sigmoid activation function as given with eq (1) is applied to the input value of the middle layer processing element to get a scaled output at the output of the middle layer processing element M_j

$$f(L) = \tanh(L) \qquad (1)$$

Where L is given as:

$$L = \Sigma W_{ij} I \qquad (2)$$

The same procedure is followed for all middle layer-processing elements. Following through the network these output values from the middle layer are treated as input values to the output layer. The sum of the product of the entire middle layer output values (M_j) and the inter – layer connection weights (w_{kj}) to an output layer processing element form the input value to that output layer processing element. The linear activation function is applied to the output node O_k is computed the output O_k is then compared to the experimentally measured output surface roughness and the difference in the measured and computed outputs is calculated. This difference in the output forms the error E at the output layer. This procedure constites the forward flow of the back propagation model.

Back Propagation Training Neural Network – Backward Pass

The error computed is back propagated through the same network by changing the weights of the interconnections on the output to middle layer processing elements and also the middle to input

layer processing elements the error at each output layer processing element is passed backwards through the derivative of the sigmoid activation function and is computed as:

$$E_k = \frac{df(L)}{dL} E \qquad (3)$$

Where E is the computed error at the output layer and E_k is the error at the input of the output layer-processing element.

The derivative presents a bell- shaped curve when plotted against the input with relatively large values in the midrange of inputs and small values at either end. The derivative thus contributes to the stability of the network; since it assures that as the outputs approach 0 or 1 only very small changes can occur and the error E_k will be proportionate to the original error propagate by the input values of the middle layer processing elements. A rule known as" Delta rule" is applied to determine how to change the weights [T.Munkata, 1998]. By applied Delta rule to determine the error value,the change is determined as:

$$(W_{kj)\ new} - (W_{ki})_{old} = \frac{\beta E_k L}{L^2} \qquad (4)$$

Where difference for the weights is the delta vector, β is a scalar value of the learning constant. E_k is a scalar value of the error at the output layer processing element, and L is the input vector to the output layer processing element.

A momentum term, which results in faster convergence to the ideal weight vector, is used. The term applies a momentum factor to the difference between the latest known and the previously known weights. The momentum term is calculated as follows:

$$M = \alpha (\Delta W_{kj}) \qquad (5)$$

Where M is the momentum term. α is the momentum factor, and ΔW_{kj} is the difference in the latest known weight and the previously known weight. The values of β, the learning constant and the α. the momentum factor M are optimized between the range of 0.1 and 0.9 by trial and error. The new interconnection weights on the output – middle layer are calculated by adding the delta vector and the momentum term to the old weights.

The middle – input layer interconnection weights also share a part of the error and the error is calculated as follows:

$$E_i = \frac{df(L)}{dL} [\sum W_{kj} E_k] \qquad (6)$$

Where E_i is the error of the middle layer processing element, $[\sum W_{kj} E_k]$ is the summation of the product of the weights of each middle to all output layer processing elements and the error at all output layer processing

elements E_k $\frac{df(L)}{dL}$ is the derivative of the activation function of the

middle – layer processing element for the net input it received.

The error E_i computed is now used to change the weights on the interconnection of the middle – input layer. Delta rule and the momentum term are calculated for every middle layer processing element the summed inputs to the middle layer processing element, error E_i and the original and previous weights, form the inputs error E_i, to the Delta rule and the momentum term.

The remaining input patterns are then presented to the BPTNN sequentially and following the above procedure, The inter – connection weights on the middle – input layer are constantly changed [T.Munkata,1998].

A mean square error (MSE) is calculated based on the error between the computed and desired output at the output layer processing elements. The mean square errors for all the output layer processing elements and for all patterns presented to the BPTNN are added to get the total error through one passes. The patterns are presented to the BPTNN and the weights are constantly changed until the MSE reaches a fixed value of 0.1. The program is terminated after 307 iterations in case it does not reach the fixed MSE value. Trained weights corresponding to this error are then stored. Which are used in the BPPNN model.

Collection Training Data

In order to train the (BPTNN) , training sets have been measured experimentally by Bahaa et .al [Bahaa. I. K.,2002], and given in **Table (1)** The training set consists of three inputs to the input layer cutting speed in m/min , feed in mm/rev and nose radius in (mm) . The depth of cut was kept constant (0.25 mm). The surface roughness corresponding to the cutting condition has been measured.

The discussion here and the tables produced show that for a typical cut for the purpose of generating the training samples consists of the following. The work piece is turned using specified cutting conditions. Surface roughness component is observed using two different methods, [Bahaa. I. K., 2002].

Normalization And Scaling Of Inputs

The input patterns are presented to the BPTNN as a normalized array and are scaled in a range of -1 to 1. The original values are normalized for efficient processing by the net work. The normalization is carried out using a linear mapping given as

$$X = (X_r - X_{min}) \frac{X_{N\,max} - X_{N\,min}}{X_{max} - X_{min}} + X_{N\,min} \quad (7)$$

Where X is the normalized variable. X_r is the real value of the variable before normalization. X_{Nmax} and X_{Nmin} are maximum and minimum values of the variable after normalization.

Back Propagation Prediction Neural Network Model (BPPNN)

BPPNN architecture is the same as the BPTNN; with exactly the same number of input, middle and output layer processing elements. The trained set of weights is assigned to the interconnections of the middle – input and the output – middle – layers. New testing input patterns other then that used for training the BPTNN. Without the surface roughness, are presented to the BPPNN model. The BPPNN model works exactly like the forward pass of the BPTNN model.

All of the input variables are normalized in the same range that was used for the BPTNN model. The input to the middle layer processing elements is the weighted sum of the values from the input layer processing elements .The weights on the interconnections are obtained from the recorded weights at MSE = $5.73* 10^{-20}$for the BPTNN model . The same sigmoid function as in the BPTNN model gives as output at the middle layer processing elements. These outputs and the weight on the interconnection of the output middle layer processing elements. This input when passed through the linear function gives an output at the output layer processing elements. The calculated output value is the predicted surface roughness value for the new testing set or unknown condition.

RESULTS AND DISCUSSION

It has been know that the surface finish is related to the cutting parameters such as the feed, depth of cut, spindle speed and cutting tool nose radius selected during machining. For the purpose of predicting a surface finish through the cutting parameters a mapping function between the detected surface finish values and the other cutting parameters must be valid. In this work only three cutting parameter feed, spindle speed and nose radius of the cutting tool have been considered in the analysis These parameters have a major effect on the surface roughness especially when using H.S.S cutting tool to machined low carbon steel (0.2 % carbon).

To perform this information processing from the measured surface finish values , multilayer feed forward neural network is implemented as a prediction model .

The main characteristics of the predictor was chosen from several testing models of neural network are (1) layered architectures (2) strictly feed -forward connection between neurons and (3) no lateral or back connections.

In the performed model the three nodes in the input layer represent feed, cutting speed and nose radius, in hidden layer different number of hidden nodes was tested as shown in **Table (2.)** Eight nodes where chosen to be used in hidden layer to minimize the performance function

The trained network is used to predict the surface roughness during the orthogonal cutting of hardened steel work pieces.

Fig (3) shows the training graph of the developed network with eight hidden nodes.

Fig (4-a) and **(4-b)** show reasonable agreement between the predicted and measured surface roughness. **figures (5)** and **(6)** show similar results.

Surface roughness is also predicted for the cutting conditions other than the patterns for which the neural network algorithm is trained, fairly large error was observed at those predictions.

In conclusion, predicted surface roughness was found significantly sensitive of the measured cutting conditions. The major advantage of neural network predictions is that the algorithms can estimate Surface roughness progress quite accurately once cutting conditions are known.

REFERENCES

Ahmad Ghasempoor (1997), Automatic Adjustment of cutting conditions in turning, univ. Of polytechnic Department Of Mach; Aero space and Industrial Eng.

Bahaa. Ibraheem. K and Nabeil .k .AL – Sahib (2002), Surface Finish Characteristics in turning processes, J. of science and Eng. AL – Anbar univ. Iraq , Vol_2 NO . 2.

M. Hasegawa A. Seirey and R.A.Lindberg (1976), surface roughness model for turning, Tribology Int., December , p285 – 289 .

N. Costa. B. Ribeiro (1998), A neural Prediction Model for monitoring and Fault Diagnosis of a plastic Injection Molding Process, CISUC – Department of Eng. Informatics, combra, Portugal, 1998.

N. Costa, A Tuna, B. Ribeiro, Monitoring an Industrial plastic Injection Moulding Machine using Neural networks, CISUC – Department of Eng. Informatics, combra, Portugal,

R.G. Khunchustombham and G.M. Zhang (2001), A neural Approach to on- line Monitoring of a turning process, The mechanical Research Report . Univ of Maryland , system Research center.

Sarah. S.Y, Alice E smith (1996), Process Monitoring of Abrasive Flow Machining using A neural Network predictive model, university of Pittsburgh, dept of Industrial Eng.

Tugral Ozel; Abhijit Nodgir (2002), Prediction of Flank Wear by using back propagation neural network modeling when cutting hardened H-13 steel with chamfered and honed CBN tool, Int.J .Of Machine tools and manufacture, 42, 287- 297.

T. Munakata. (1998), Fundamentals Of The New Artificial Intelligence. Beyond Tradition a paradigs, Springer – Verlag New York Inc..

Table (1). Training data
Measured surface Roughness for different cutting conditions and depth of cut 0.25 mm
[Bahaa. I. K.,2002]

Speed M/min	Nose radius mm	Feed mm/ rev	Surface finish µm
7.35	0.25	0.1	38
7.35	0.25	0.2	69.5
7.35	0.25	0.3	66.5
7.35	0.25	0.6	16.4
7.35	0.5	0.1	35.3
7.35	0.5	0.2	33.2
7.35	0.5	0.3	48.7
7.35	0.5	0.6	66.8
28.27	0.25	0.1	40.65
28.27	0.25	0.2	72.2
28.27	0.25	0.3	54.9
28.27	0.25	0.6	19.3
28.27	0.5	0.1	40
28.27	0.5	0.2	36
28.27	0.5	0.3	56.3
28.27	0.5	0.6	57
65.97	0.25	0.1	15.09
65.97	0.25	0.2	47.5
65.97	0.25	0.3	30.5
65.97	0.25	0.6	19
65.97	0.5	0.1	14.2
65.97	0.5	0.2	18
65.97	0.5	0.3	18.3
65.97	0.5	0.6	38.2
98.96	0.25	0.1	24.2
98.96	0.25	0.2	24.2
98.96	0.25	0.3	27.12
98.96	0.25	0.6	8.87
98.96	0.5	0.1	19.3
98.96	0.5	0.2	10.5
98.96	0.5	0.3	15.2
98.96	0.5	0.6	28.3

Table (2) network performance for different architecture

No. Of hidden nods	Error (sse)	Correlation coefficient	Epochs
2	0.0522532	0.9195	45
3	0.0190485	0.9714	489
4	0.0139469	0.9792	756
5	0.0181098	0.973	2000
6	0.00981311	0.985	2000
7	0.00115506	0.998	2000
8	$5.72*10^{-20}$	1	307

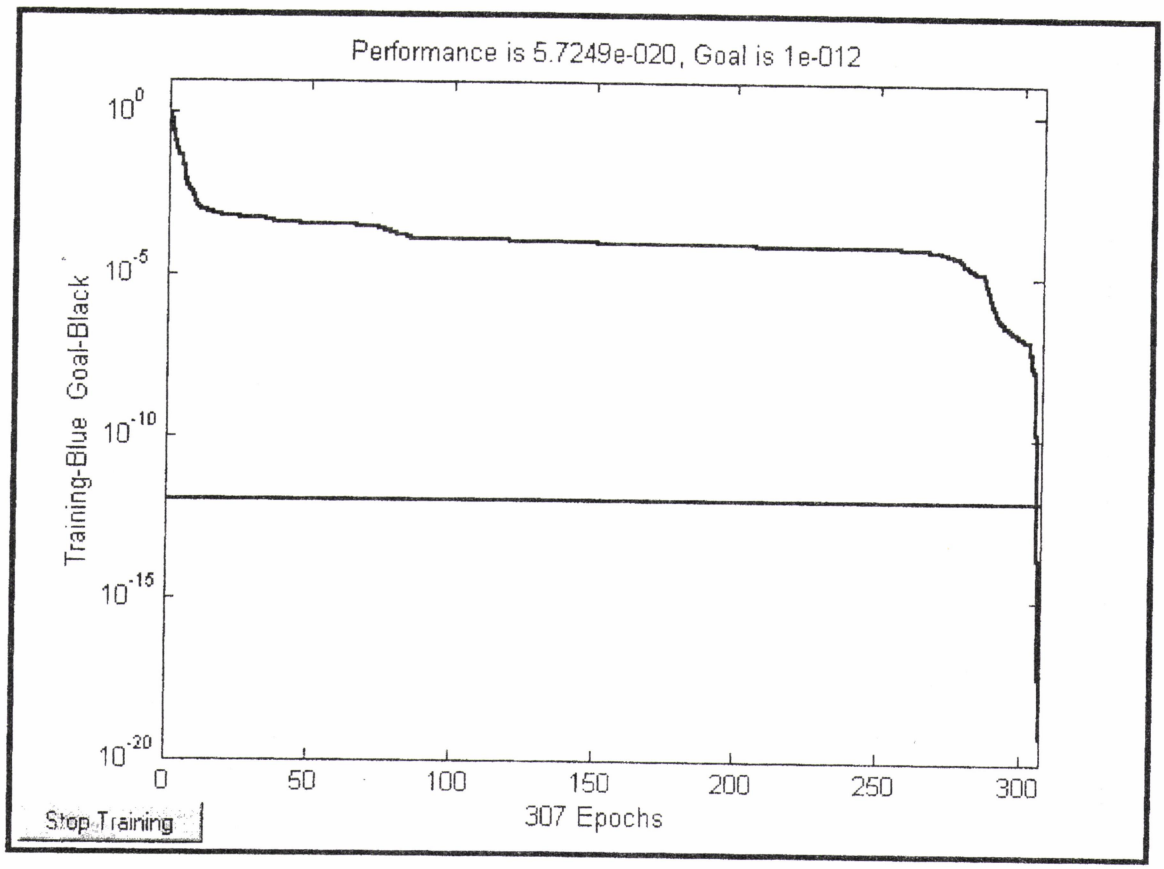

Fig. (3) Network Training Graph

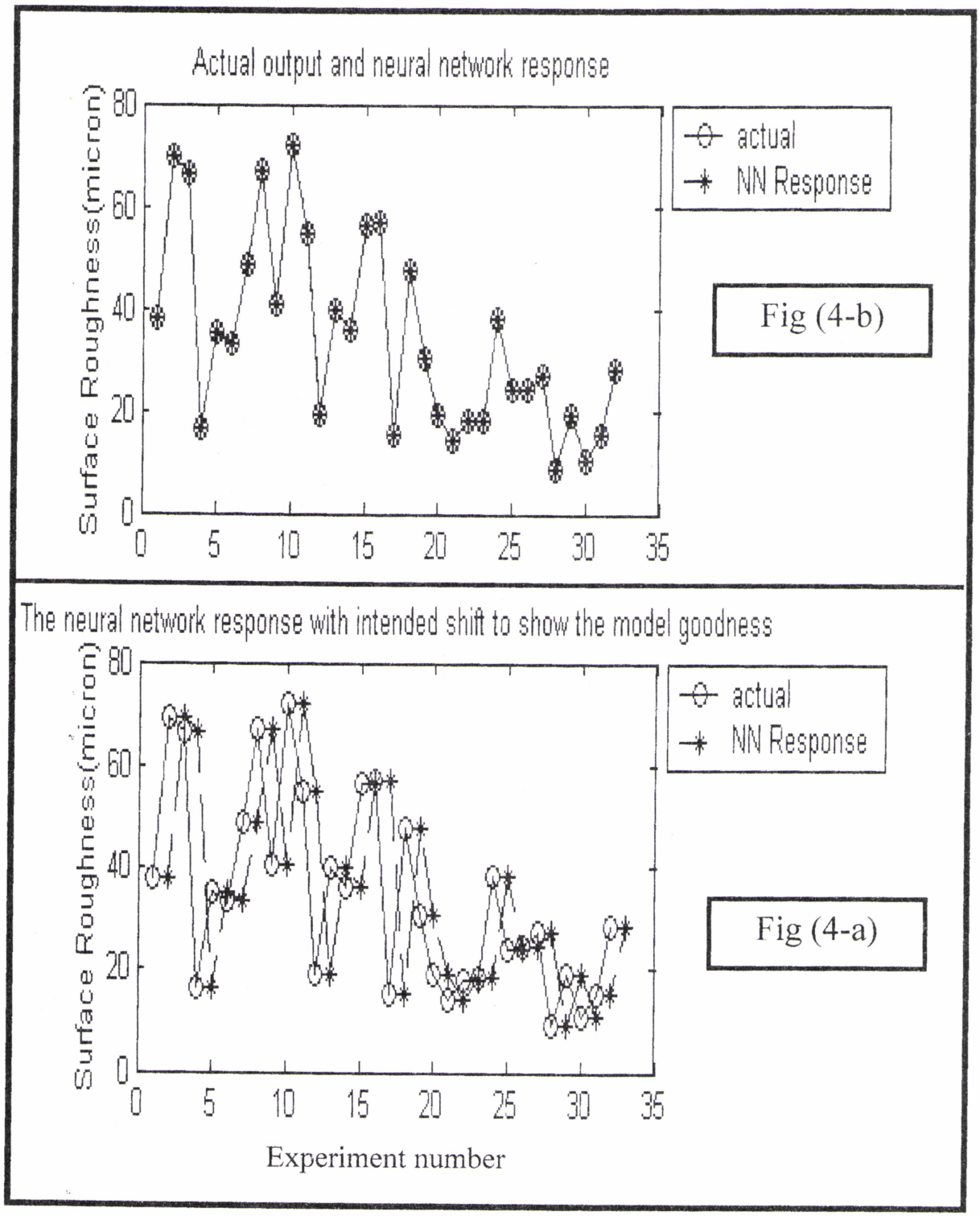

Fig. (4) Neural Network performance

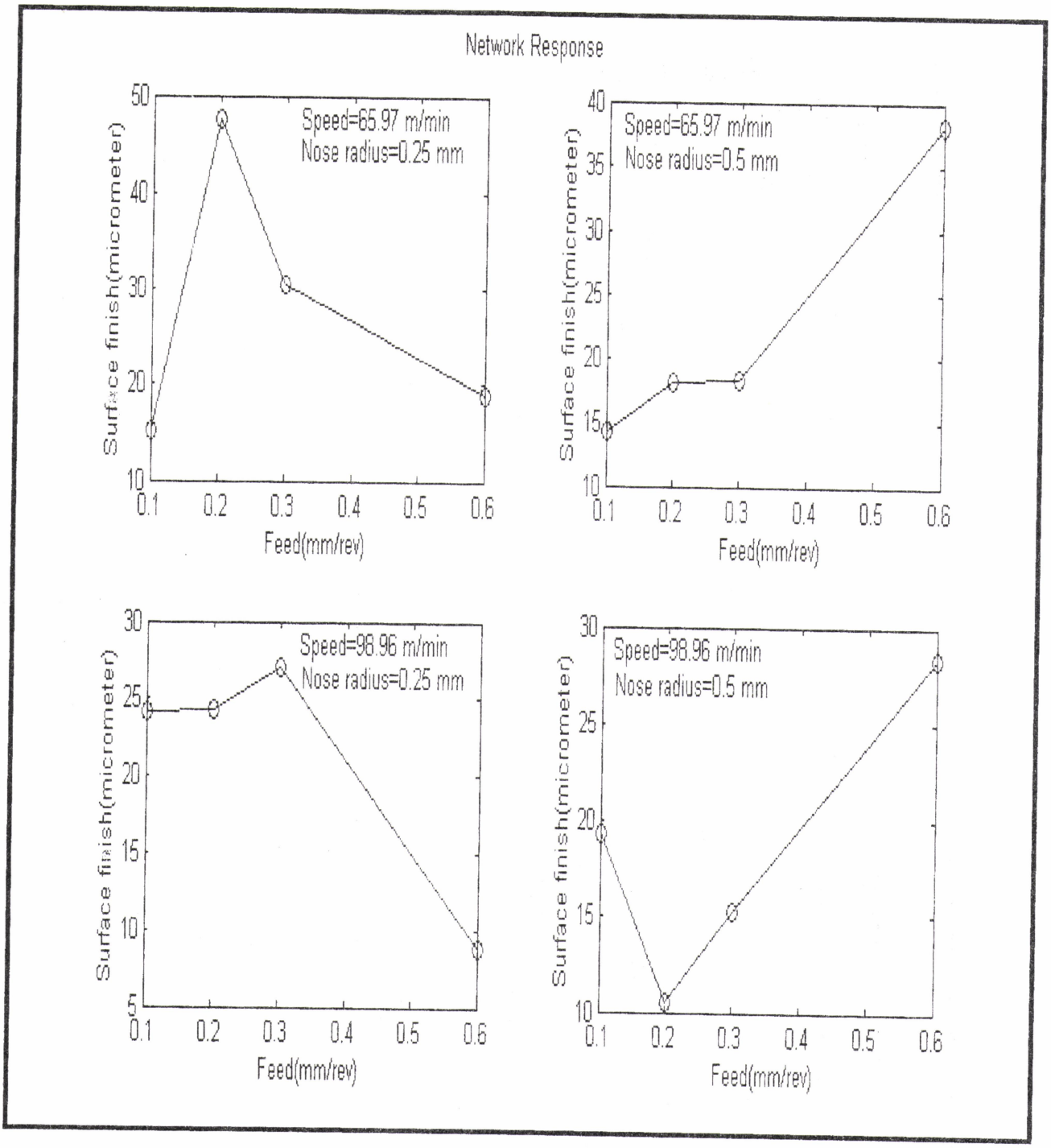

Fig. (5) Neural Network response

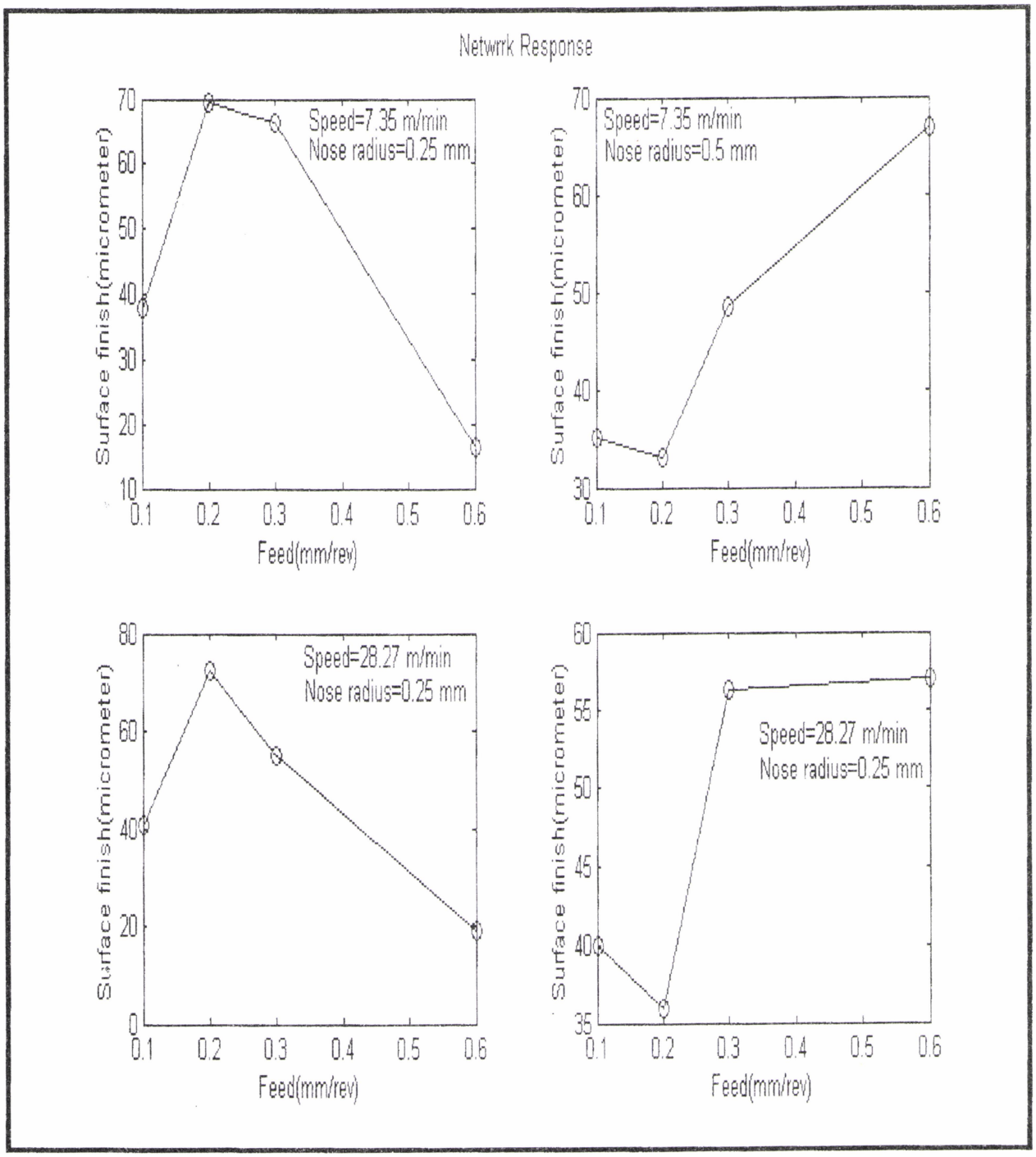

Fig. (6) Neural Network response