CHAPTER-4

TUNING OF PID CONTROLLER USING ANN AND FLC FOR NON-LINEAR FLANK WEAR MODEL

4.1 INTRODUCTION

The conventional control theory relies on the key assumption of small range of operation for the linear model to be valid. When the operation range is large, a linear controller is likely to perform poorly or to be unstable, because the non-linearities in the system cannot be properly compensated for. For a long time the automatic control of physical process, in spite of the use of the principle of feedback, has been an experimental technique deriving more from art than from scientific basis.

The requirement for more complex and higher-performance control systems have been the impulse for the development of a systematic control theory. Even with the development of such a systematic control theory, there is still usually something else missing when a practical control system must be designed; a good knowledge of the dynamic characteristics of the controlled non-linear plant like opto-electrical applications and Manufacturing process by tool wear.

The non-linearities include Coulomb friction, manufacturing process in metal cutting process etc., are often found in control engineering. These effects cannot be derived from linear model and need a non-linear techniques. In designing linear controllers, it is usually
necessary to assume that the parameters of the system model are
deterministic and time-invariant. However, many control problems
involve many uncertainties in the model parameters that are
un/controllable, un/observable and/or both [45,56]. This may be due to
slow variations of the parameters like fouling of heat exchangers to an
abrupt change in parameters and like inertial parameters in
manufacturing process in metal cutting by flank wear, the rate of
diffusion between work piece and tool materials rises very rapidly as
temperature increases to past the critical, etc.

Controllers, based on inaccurate or absolute values of the model
parameters may exhibit significant degrade of the performance or may
cause instability. Certain non-linearities can be intentionally introduced
into the controller part of a control system so that model uncertainties
can be tolerated [57].

To implement and develop high-performance control systems when
the plant dynamic characteristics are poorly known or when large and
unpredictable variations occur, a new class of control systems called
non-linear advanced control systems have evolved which provide
potential solutions based on tuning of PID controller.

There are two classes of non-linear controller systems for the
proposed are Artificial Neural Network (ANN) and Fuzzy Logic Controller,
a brief overview on both controllers is summarized in this chapter.
4.2 ARTIFICIAL NEURAL NETWORK (ANN)

The study of artificial neural networks has been one of the most interesting topics in the control community because they have the ability to treat many problems that cannot be handled by traditional analytic approaches. In general, feed forward multilayer neural networks are the most prevalent neural network architecture for identification and control applications [57, 58]. A widely used training method for feed forward Multilayer Neural Networks (MNN) is the back propagation (BP) algorithm developed by Rumelhart et al [59]. The standard BP learning algorithm has several limitations. Most of all, a long and unpredictable training process is the most troublesome, for example the rate of convergence is seriously affected by the initial weights and the learning rate of the parameters. Many researchers have proposed the modifications of the classical BP algorithm [60, 61]. Recently another modified algorithm was derived by Scalero and Tepedelenlioglu as an alternative to the BP algorithm. It uses a modified form of the BP algorithm to minimize the mean square error between the desired output and the actual output with respect to the summation output [62]. Even though the performance of this new algorithm overwhelms the BP method, it is not a stable learning algorithm in practical real-life applications. Thus, the faster and more stable learning neural network is required, which is indeed the main purpose of this work. For solving those problems addressed above, the novel error self-recurrent neural networks are presented, and the
desired outputs of the hidden layers are obtained by the nearly optimal learning algorithm. A new neural network is considerably faster than the BP algorithm and has advantages of being less affected by poor initial weights and learning rate. Nonlinear adaptive PID controller based on these neural networks has been derived and tested for the fast tracking problem in a tool wear mechanism manipulator model.

4.2.1 Learning scheme

Back propagation algorithm: This algorithm, which performs a stochastic gradient descent, provides an effective method to train a feed forward neural network to approximate a given continuous function over a compact domain. Derive the stochastic Back propagation algorithm for the general case. The derivation is simply,

\[ x_j = \text{Input vector for unit J (} x_{ij} = \text{ith input to the } j\text{th unit}). \]

\[ w_j = \text{Weight vector for unit } j (w_{ij} = \text{weight on } x_{ij}). \]

\[ Z_{-w_j} = w_j \cdot x_j, \text{ the weighted sum of inputs for unit } j. \] \hspace{1cm} (4.1)

\[ Z_j = \text{output of unit } j \left( O_j = f(Z_j) \right). \]

\[ t_j = \text{target for unit } j. \]

Downstream \((f)\) = set of units whose immediate inputs include the output of \(j\).

Outputs = set of output units in the final layer.
Since we update after each training case study, we can simplify the notation somewhat by imagining that the training set consists of exactly one case study and so the error can simply be denoted by $E$. Consider an arbitrary activation function $f(x)$. The derivation of activation function for Back propagation or generalized delta rule is follows[63]:

$$
y_{-ink} = \sum_{i} z_{i} W_{jk} \tag{4.2}
$$

$$
z_{-inj} = \sum_{i} v_{ij} x_{i} \tag{4.3}
$$

$$
Y_{k} = f(y_{-ink}) \tag{4.4}
$$

The error to be minimized is $E=0.5 \sum_{k} [t_{k} - y_{k}]^{2} \tag{4.5}$

By use of chain rule we have

$$
\frac{\partial E}{\partial w_{ij}} = \frac{\partial}{\partial w_{jk}} \left( 0.5 \sum_{k} [t_{k} - y_{k}]^{2} \right) \tag{4.6}
$$

$$
= [t_{k} - y_{k}] \frac{\partial}{\partial w_{jk}} f(y_{-ink})
$$

$$
= [t_{k} - y_{k}] f^{1}(y_{-ink}) z_{j}
$$

Let us define $\delta_{k} = -[t_{k} - y_{k}] f^{1}(y_{-ink}) \tag{4.7}$

Weights on connections to the hidden unit $z_{j}$

$$
\frac{\partial E}{\partial v_{ij}} = -\sum_{k} [t_{k} - y_{k}] \frac{\partial}{\partial v_{ij}} y_{k} \tag{4.8}
$$

$$
= -\sum_{k} \delta_{k} \frac{\partial}{\partial v_{ij}} y_{-ink}
$$
Rewriting the equation and substituting the value of \( y_{\text{ink}} \)

\[
= -\sum_k \delta_k \frac{\partial}{\partial v_{ij}} (\sum z_j - w_{jk})
\]

\[
= -\sum_k \delta_k w_{jk} \frac{\partial}{\partial v_{ij}} f(z_{inj})
\]

Therefore,

\[
\delta_j = -\sum_k \delta_k w_{jk} f'(z_{inj})
\]  \hspace{1cm} (4.9)

The weight updation for output unit is given by

\[
\Delta w_{jk} = -\alpha \frac{\partial E}{\partial w_{jk}}
\]

\[
= \alpha (t_k - y_k) f'(y_{\text{ink}}) z_j
\]

\[
= \alpha \delta_j x_i
\]  \hspace{1cm} (4.10)

This is the generalized delta rule used in the back propagation network during training.

**4.2.1.1 Application procedure of the algorithm**

Stochastic Back propagation, each training example is of the form \((x_i, t_j)\) where \(x_i\) is the input vector and \(t_j\) is the target vector. \(\eta\) is the learning rate (e.g., .05). \(x_i\), \(z_j\) and \(y_k\) are the number of input, hidden and output nodes respectively. Input from unit \(i\) to unit \(j\) is denoted \(x_{ji}\) and its weight is denoted by \(w_{ji}\). After training, a back propagation neural net is applied by using only the feed-forward phase of the training algorithm. The application procedure is as follows:

Step 0: Initialize weights (from training algorithm).
Step 1: For each input vector, do steps 2-4.

Step 3: for $i=1\ldots\ldots n$:

\[ Z_{\text{in}j} = v_{oj} + \sum_{i=1}^{n} x_i v_{ij}; \]  

\[ Z_j = f(Z_{\text{in}j}). \]

Step 4: for $k=1\ldots\ldots m$:

\[ y_{\text{in}k} = w_{ok} + \sum_{j=1}^{p} z_j w_{jk}; \]  

\[ y_k = f(y_{\text{in}k}). \]

We could stop when the network can recognize all the flank wear model value successfully, but in practice it is usual to let the error fall to a lower value first. This ensures that the flank wear model values are all being well recognized. You can evaluate the total error of the network by adding up all the errors for each individual neuron and then for each pattern in turn to give you a total error as shown in Fig. 4.1[64]. In other words, the network keeps training all the patterns repeatedly until the total error falls to some pre-determined low target value and then it stops. Note that when calculating the final error used to stop the network (which is the sum of all the individual neuron errors for each pattern) you need to make all errors positive so that they add up and do not subtract [65]. Once the network has been trained, it should be able to recognize not just the perfect patterns, but the training may also benefit from applying the patterns in a random order to the network [66].
4.3 DESIGN OF NEURO CONTROLLER

The basic objective of a controller is to provide the desired output for any system. Since neural networks have learning and self-organizing abilities allowing them to adapt changes in data, such networks are used for control of non-linear flank wear. An artificial neural network is defined as

![Flowchart: Total Error Network]

*Fig. 4.1 Total error network*
a data processing system consisting of a large number of simple highly interconnected processing elements (artificial neurons) in an architecture inspired by the structure of the cerebral cortex of the brain. Neural networks learn by example [67]. They can therefore be trained with known examples to acquire knowledge about a problem. Once appropriately trained, the network can be put to effective use in solving unknown or untrained instances of the problem. In supervised learning, a teacher is assumed to be present during the learning process in which the network aims to minimize the error between the target (desired output) and the compared output to achieve better performance.

The input-output data necessary for the off-line training of the neural network have been obtained in the present work using non-linear flank wear model, experimental tool wear mechanism data. The data set is sufficiently rich to ensure stable operation since no additional learning will take place after training by using quasi-Newton back propagation learning algorithm.

Quasi-Newton back-propagation algorithm is employed to update weights in this work in view of the reasons: quasi-Newton method is a one-dimensional minimization related numerical interpolation method which has a powerful (fast) convergence property known as quadratic convergence and hence it exhibits super linear convergence near the target. This algorithm requires less memory space than other training algorithms. It is the most popular supervised learning rule for multi-layer
feed forward networks [68]. With this algorithm, input data is repeatedly presented to the neural network. With each presentation, the output of the neural network is compared with the desired output and an error is computed. This error is then fed back (back-propagated) to the neural network and used to adjust the weights such that the error decreases with each iteration and the neural network output gets closer and closer to the desired output. This process is known as “training”.

For back-propagation training algorithm, the derivative of the activation function is needed. The logistic or sigmoid function satisfies this requirement and it is the commonly used soft-limiting activation function. It is also quite common to use linear output nodes to make learning easier and using linear activation function in the output layer does not ‘squash’ (compress) the range of output. Hence bipolar sigmoid type activation function and linear activation function are used for hidden and output layers respectively.

Using hyperbolic tangent sigmoid (tansig) activation function for hidden layer is better than log sigmoid (logsig) activation function. Since logsig takes approximately 10 epochs more than that taken by tansig. There is no general procedure to determine the exact size of the neural network. However, the size of the network developed in this work showed itself satisfactory as far as the output of flank wear regulation is concerned. Trials have been carried out to obtain maximum accuracy
with minimum number of neurons per layer[69]. The feed forward neural network controller developed consists of three layers with one neuron in the input layer, four neurons in the hidden layer and one neuron in the output layer (Fig. 4.2-4.5).

![Fig.4.2 SIMULINK block for neuro controller](image1)

**Fig.4.2 SIMULINK block for neuro controller**

![Fig.4.3 SIMULINK model of neural network chosen for flank wear model](image2)

**Fig.4.3 SIMULINK model of neural network chosen for flank wear model**

![Fig.4.4 SIMULINK model of hidden layer](image3)

**Fig.4.4 SIMULINK model of hidden layer**
The optimum number of neurons for hidden layer is chosen as around four since with this optimum number of hidden neurons, number of epochs for training the neural network reduced considerably. And satisfactory dynamical performances of the converters have been obtained. It has also been found that the use of more than four hidden nodes cannot produce significant improvement and there is need for additional computation time.

Fig. 4.6 Block diagram of neuro control flank wear model

Hence the choice of hyperbolic tangent sigmoid (tansig) activation function for hidden layer with four hidden neurons and linear activation
function for output layer trains the network in lesser number of epochs with better performance criteria and also yields better control.

The block diagram of neuro controller as Neuro-PID controller for a non-linear metal cutting process by flank wear model is shown in Fig 4.6. The neural network developed in this work is off-line trained using the input data (error of cutting force) and target (output value of the surface roughness) pairs. The neuro-controller has to map correctly the input patterns to the desired output within the required accuracy. Neuro-controller begins to mature after training and the target is reached in this work [70].

Once the neuro-controller matures, it can respond correctly with any other input-output patterns. The output of the flank wear regulates, this is compared with reference values. The error in output surface roughness of flank wear is fed as input to the developed neuro controller which outputs the corrected to the surface roughness of tool wear mechanism.

4.4 FUZZY LOGIC CONTROLLER (FLC)

This section introduces the concept of fuzzy logic and deals with different fuzzification and defuzzification methods. A detailed discussion of center of gravity method for defuzzification is also presented here. Fuzzy logic is a super set of conventional (Boolean) logic that has been extended to
handle the concept of partial truth, i.e. truth values between “completely true and completely false”.

Expert system principles are fundamentally formulated based on Boolean logic. It has been argued that human thinking does not always follow crisp “Yes” or “No” logic, but is often vague, qualitative, uncertain imprecise, or fuzzy in nature [71].

4.4.1 Structure of Fuzzy Knowledge Basis Controller (FKBC)

The functional block diagram of fuzzy logic control system is illustrated in Fig.4.7[71].

*Fig. 4.7 Functional Block diagram of Fuzzy Logic Controller System*
It includes four major blocks. They are Fuzzification, Knowledge Base, Inference Mechanism and defuzzification.

### 4.4.1.1 Crisp set

A crisp set is defined by crisp boundary that is there no uncertainty in the prescription or location of the boundaries of the set.

### 4.4.1.2 Fuzzification

The fuzzification module performance:

(i) A scale transformation, which maps the physical values of the current process state variables into a normalized universe of discourse. It also maps the normalized value of the control output variable in its physical domain.

(ii) This converts a point wise (crisp set) current value of a process state variable into a fuzzy set in order to make it compatible with the fuzzy set representation of the process state variable in the rule antecedent.

### 4.4.1.3 Knowledge Base

The knowledge base consists of database and a rule base. The function of a database is to provide the necessary information for the proper functioning of the fuzzification module, the rule base and the defuzzification module.
4.4.1.4 Rule Base

The basic function of the rule base is to represent in a structural way the control policy of an experienced process operator in the form of a set of production rules such as

\[
\text{IF(process state) THEN ( control action)}
\]

The IF part of such a rule is called the rule-antecedent and it's a description of a process state in terms of logic combination of fuzzy propositions. The THEN part of rule is called the value consequent and is again a description of the control output in terms of logical combinations of fuzzy propositions.

4.4.1.5 Inference Mechanism

The basic function of the inference engine is to compute the overall value of the control output variable based on the individual contribution of each rule in the rule base. The output of the fuzzification module representing the current crisp values of the process state variables is matched to each rule antecedent and the degree of match for each rule is established.

4.4.1.6 Defuzzification

The function of defuzzification is scale mapping which converts the range of values of output variable into corresponding universe of discourse and it yields a non-fuzzy (crisp) action. They are height method, mean of
maxima method, centroid method and center of sums method. However, out of the four methods the centroid method is commonly used as defuzzification method because it provides the area of centre occupied by the fuzzy set.

4.4.2 Design of Fuzzy Logic Controller

In fuzzy logic control, linguistic descriptions of human expertise in controlling a process is represented as fuzzy rules and relations. This knowledge base is used by an inference mechanism in conjunction with some knowledge of the states of the process in order to determine control action. Although they do not have an apparent structure of PI controller, the parameters are determined on-line, based on the error signal and their difference [72, 74].

In this work, a rule-based scheme for fuzzy logic controllers is presented. This scheme utilizes fuzzy rules and reasoning to determine the controller settings. The human expertise on PI controller can be presented in fuzzy rules [73].

4.4.2.1 PI Controller

The transfer function of a PI controller has the following form

\[ G_c = K_p e(s) + \frac{K_i}{s} e(s) + bias \]  

(4.13)

Where \( K_p, K_i \) are proportional and integral gains respectively.
The discrete-time equivalent egression for PI controller is

\[
U(k) = K_p e(k) + K_i T_i \sum_{i=1}^{n} e(i) \quad (4.14)
\]

Where \( U(k) \) is control signal.

The parameters of the PI controller \( K_p, K_i \), and or \( K_p, T_i \) can be manipulated to produce various responses from a given process. In the following section PI controller based fuzzy rule is introduced [46].

### 4.4.2.2 Fuzzy Membership Function

Fig.4.8 shows the like closed-loop control system. This is used to know the error and change in error range to form the fuzzy rule table4.1[73].
4.4.2.2.1 Triangular membership function

Error range for the triangular membership function is obtained as in Fig.4.9 by using MATLAB software. The error range is between -1 to 1 with maximum membership grade value 1, and this error range is named as input-1 for one of the input of the FLC controller of flank wear model.
Like wise the range of change in error and output range are defined the same range for the triangular membership function shown in Fig.4.10 and Fig.4.11 respectively.

**4.4.2.2.2 Rule table for triangular membership function**

The fuzzy rule table for the triangular member function is shown as Table 4.1. Which is developed based on IF then Else rule of PI controller parameters manipulation, a detail explanation is in the following chapter for the evaluation of FLC by flank wear model.

![Input variable “input-2”](image)

*Fig. 4.10 Change in error for triangular membership Function*

The output range for the triangular membership function is shown in Fig.4.11.
Output variable “output1”

Fig. 4.11 Output range for triangular membership Function

Table 4.1 Fuzzy rule table for triangular membership function

<table>
<thead>
<tr>
<th>(e(k))</th>
<th>NB</th>
<th>NM</th>
<th>NS</th>
<th>ZE</th>
<th>PS</th>
<th>PM</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\Delta e(k))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>NS</td>
<td>ZE</td>
</tr>
<tr>
<td>NM</td>
<td>NB</td>
<td>NM</td>
<td>NM</td>
<td>NM</td>
<td>NS</td>
<td>ZE</td>
<td>PS</td>
</tr>
<tr>
<td>NS</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>NS</td>
<td>ZE</td>
<td>PS</td>
<td>PM</td>
</tr>
<tr>
<td>ZE</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>ZE</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
</tr>
<tr>
<td>PS</td>
<td>NM</td>
<td>NS</td>
<td>ZE</td>
<td>PS</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
</tr>
<tr>
<td>PM</td>
<td>NS</td>
<td>ZE</td>
<td>ZE</td>
<td>PS</td>
<td>PM</td>
<td>PM</td>
<td>PB</td>
</tr>
<tr>
<td>PB</td>
<td>ZE</td>
<td>PS</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
</tr>
</tbody>
</table>
4.4.2.2.3 Membership Function Value Calculation

A triangular membership function is used which can be given by the expression. The Fig.4.12 is used to define the membership grade value with the x-variations in terms of a, b, and c constants which follows with expression.

\[
\mu(x) = \begin{cases} 
0 & x \leq a \\
\frac{(x-a)}{(x-b)} & a \leq x \leq b \\
\frac{(c-x)}{(c-b)} & b \leq x \leq c \\
0 & c \leq x 
\end{cases}
\]

*Fig. 4.12 triangular Membership function*
4.4.2.2.4 Defuzzification Method

Centroid method is used as the defuzzification method in this work. The algebraic expression for the centroid method is

\[ Z^* = \frac{\int \mu_c(z) zdz}{\int \mu_c(z) dz} \]  \hspace{1cm} (4.15)

Where \( \int \) denotes as algebraic integration.

After forming the rules the next step is to generate the control action with the application of the inference mechanism to combine these rules. The procedure for defuzzification is presented as follows[74].

The error can be a member of more than one fuzzy measure and hence the grade of membership of error in every fuzzy measure is determined by the use of the membership function. This forms the premise of the rule. The action of each rule is calculated from the corresponding consequent part, the action is the midpoint of the corresponding fuzzy measures [48]. The results of fuzzy and neuro-PID controller has been explained in the following chapter.