Tuning of Proportional Integral Derivative Controller Using Artificial Neural Network

[1]Gitanjali Chadar, [2]Sudeep K Mohaney

[1]*Department of Electrical Engineering, Jabalpur Engineering College, Madhya Pradesh, India*
[2]*Assistant Professor, Department of Electrical Engineering, Jabalpur Engineering College, Madhya Pradesh, India*

---***---

Abstract - *In this literature an artificial neural network is used in tuning the PID controller, which is being trained over a wide range of inputs and the outcome is a function fitting neural network block. Here a PID controller is used which is tuned by designing an ANN which is trained by LM(Levenberg marquardt) algorithm in attaining the desired inputs of PID which are K_P (proportional gain), K_I (integral gain) K_D (derivative gain) ; so by substituting these inputs the response of output gets improved and the system becomes stable.*

Key Words: power system stability, artificial neural network, proportional integral derivative controller, feedback control.

1. INTRODUCTION

In the present era controllers plays a huge role in maintaining the power system output within the stable limits, the growing scale and complexity of linked electric power networks, along with the industry commitment to optimize security at the lowest possible cost, has resulted in the creation of specialized control devices. These control mechanisms guarantee that the electric power system can function in a wide range of situations without becoming unstable. By lowering the steady-state error, controllers increase steady-state accuracy. The stability improves as the steady-state accuracy improves. Controllers also aid in the reduction of the system's undesired offsets. The maximum overrun of the system can be controlled using controllers which are tuned with the help of an ANN.

2. METHODOLOGY

2.1 PID (proportional integral derivative) controller

PID stands for proportional integral derivative, and it is a type of device used in industrial applications to manage various process variables such as pressure, flow, temperature, and speed. All process variables in this controller are governed by a control loop feedback device.

This form of control is used to steer a system toward a goal point when it is otherwise level. It's utilized practically everywhere for temperature control, as well as in scientific procedures, automation, and a wide range of chemical applications. Closed-loop feedback is used in this controller to keep the true output from a technique like near to the objective if feasible, else output at the fixed point. The PID controller design, as well as the control modes employed in them, such as P, I, and D, are addressed in this article.

A closed-loop system, such as a PID controller, includes a feedback control mechanism. To create an error signal E(s), this system uses a fixed point to assess the feedback variable. Based on this, it changes the system output. This operation will be repeated until the error hits zero, at which time the feedback variable's value will be equal to a fixed point. As shown in fig.2 the error signal is the difference between the input and the output signal $[E(s) = R(s) - C(s)]$.

When compared to the other type of controller, this controller produces excellent performance. It will switch on whenever the process value falls below the specified point. Similarly, if the value exceeds a certain threshold, it will switch off. In this type of controller, the output is not steady, and it swings about a lot around the fixed point. In comparison to the other type of controller, this controller is more stable and accurate. The following diagram depicts the various control signals of PID controller in fig.1.

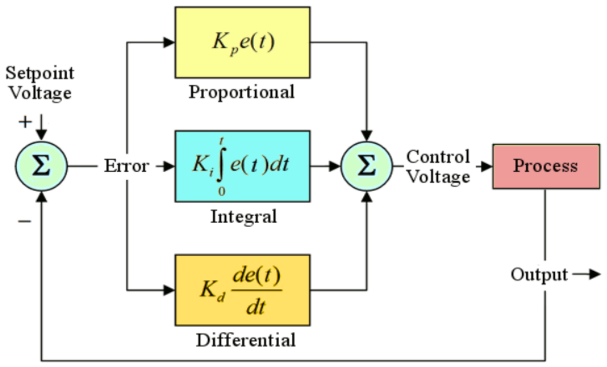

Fig -1: Schematic diagram of PID controller

The PID can be expressed mathematically as:

$$u(t) = K_P e(t) + K_I \int e(t)dt + K_D \frac{de(t)}{dt} \qquad (1)$$

Where u(t)= Input signal.
e(t)= Output signal.
On both sides, apply the Laplace transform -

$$U(s) = \left(K_P + \frac{K_I}{s} + K_D s\right) E(s) \qquad (2)$$

$$\frac{U(s)}{E(s)} = K_P + \frac{K_I}{s} + K_D s \qquad (3)$$

Where, K_P=proportional gain
K_I=integral gain
K_D=derivative gain

As a result, the proportional integral derivative controller's transfer function is $K_P + \frac{K_I}{s} + K_D s$. The following graphic depicts the block diagram of a unity negative feedback closed loop control system using a proportional integral derivative controller. The following diagram depicts the block diagram of PID controller in which R(s) is the reference signal, E(s) is the error signal, U(s) is the actuating signal, G(s) is the open loop gain of the plant and C(s) is the output signal.

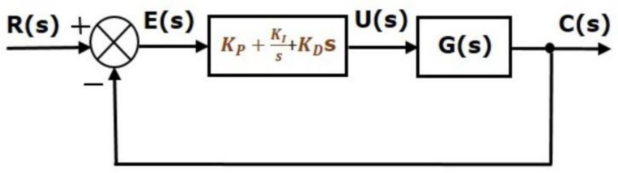

Fig -2: Block diagram of PID controller

2.2 ANN (Artificial neural network)

A neural network (artificial neuron network) is a computer model that simulates the way nerve cells in the brain operate. Artificial neural networks (ANNs) make use of learning algorithms that can make modifications - or learn - on their own when fresh information is received. As a result, they're a great tool for non-linear statistical data modeling. Learning at a deeper level Machine learning (ML) and the larger area of artificial intelligence (AI) technologies rely heavily on ANNs.

The concept behind ANNs is to simulate the functioning of a human brain by creating the appropriate connections. It was forbidden to utilize silicon and wires as living neurons and dendrites.

The human brain is made up of neuronal cells. There were 86 billion nerve cells in all. Axons also link tens of thousands of cells. However, sensory organs provide a variety of inputs.

Electric impulses are produced as a result. This is how you get around the Artificial Neural Network. As a result, each neuron sends a message to another neuron to deal with the various challenges. ANNs are made up of numerous nodes, as the name suggests. That is designed to look like biological neurons in the human brain. We do, however, use connections to connect these neurons. They also engage with one another.

Nodes, on the other hand, are utilized to collect data. Perform simple operations on the data as well. As a result, these tasks are delegated to other neurons. In addition, each node's output is referred to as its activation or node value. Each connection has a weight connected with it. They have the ability to learn as well. This is accomplished by adjusting weight values. As a result, the following diagram depicts a basic ANN:-

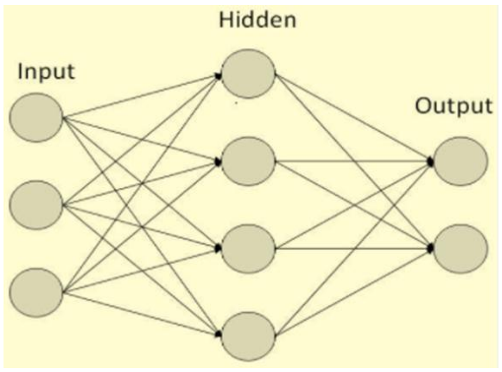

Fig -3: Structure of artificial neural network

Here the ANN is trained for 10 hidden layers in which the validation stops at 15 epoch.

3. SIMULATION & RESULTS

A neural network block is trained to tune the PID so to reduce the transient, below are the step responses of the AVR with tuning the PID through ANN and desirable values are obtained , there is decrease in the overshoot and peak value and settling time. The overshoot is decreased from 27.1% to 7.75% and the peak is decreased from 1.27 to 1.08 and so does the settling time is decreased from 4.37 second to 3.69 seconds. Here the ANN is trained by Trainlm which is a network training function that uses Levenberg-Marquardt optimization to update weight and bias variables. Table 1 depicts the input and output of the ANN block shown in fig.4.and the previous values of PID is given in table 2 when it is not tuned.

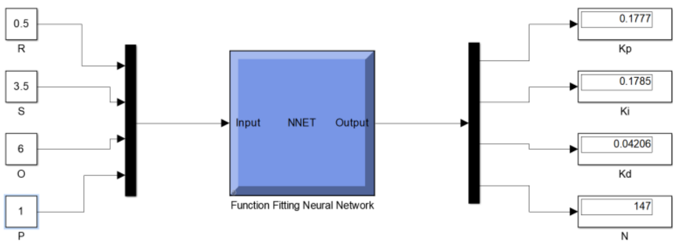

Fig -4: ANN block with the PID gains

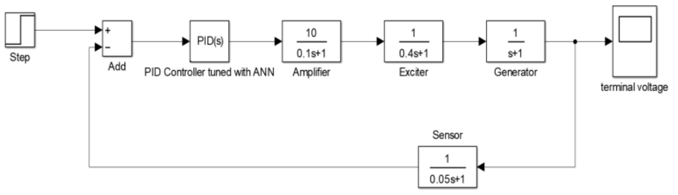

Fig -5: System in which PID is tuned with an ANN

Fig.5. Depicts a system block diagram in which PID controller is tuned by an ANN in which the inputs of the PID are given according to the required output of response.

3.1 The following are the different plots of ANN after training

- Regression plot

The correlation between outputs and objectives is measured using regression R values. An R value of 1 indicates a close link, whereas a value of 0 indicates a random relationship.

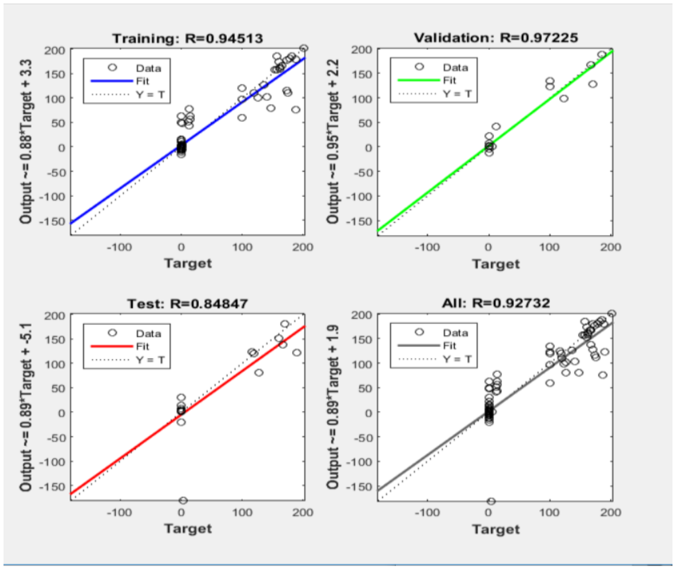

Chart -1: Regression plot of trained ANN

- Plot performance

The graphic depicts the network's mean squared error as it decreases from a big number to a lower value. To put it another way, it demonstrates that the network is growing.

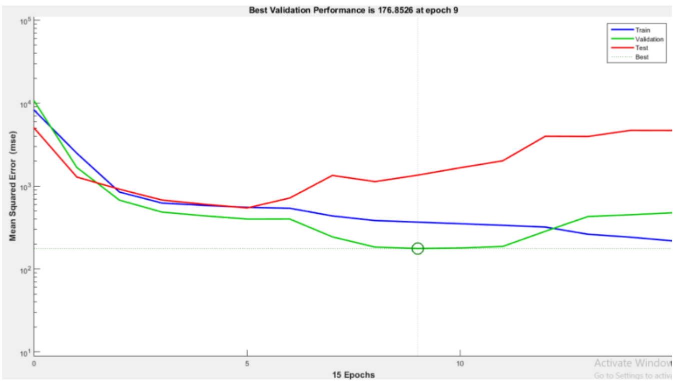

Chart -2: Plot performance of trained ANN

- Training state plot

This graph depicts the neural network's best validation performance as well as the training, validation, and test curves during the training process. At the 15th epoch, the training has accomplished its purpose. The training state throughputs of a neural network are displayed for 15 epochs. During the 15 epochs of neural network training, the gradient and mu (weight changes of neural network) fluctuate, as well as the number of validation check is 6.The gradient is 127.7849 and mu is 10 at epoch 15.

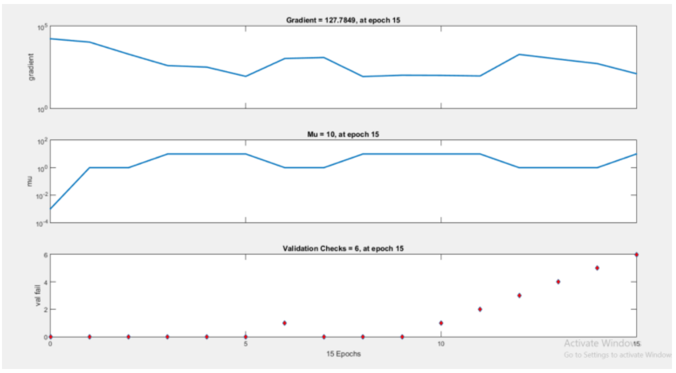

Chart -3: Training state plot of ANN

- Error histogram

The histogram of errors between target and predicted values after training a feed forward neural network is known as the error histogram. These error numbers might be negative since they represent how anticipated values depart from goal values. The number of vertical bars you see on the graph is measured in bins. Here, the total error range is broken down into 20 smaller bins. The number of samples in the dataset is represented on the Y-axis.

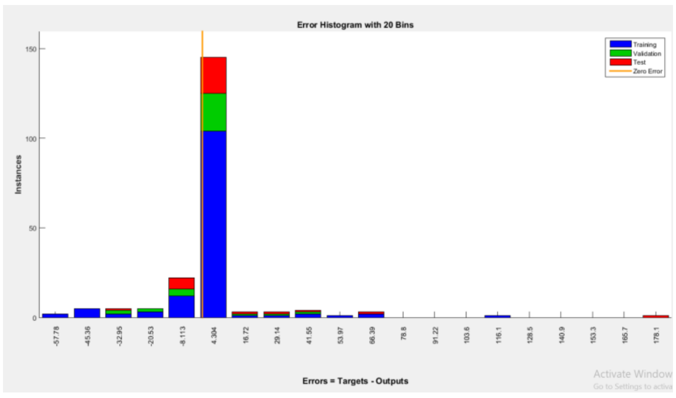

Chart -4: Error histogram of trained ANN

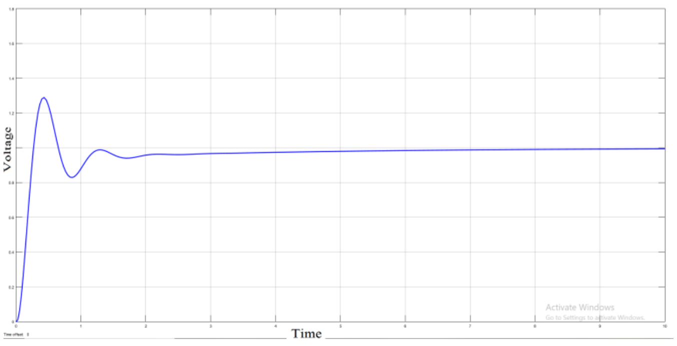

Fig -6: Response of the system in which PID is not tuned

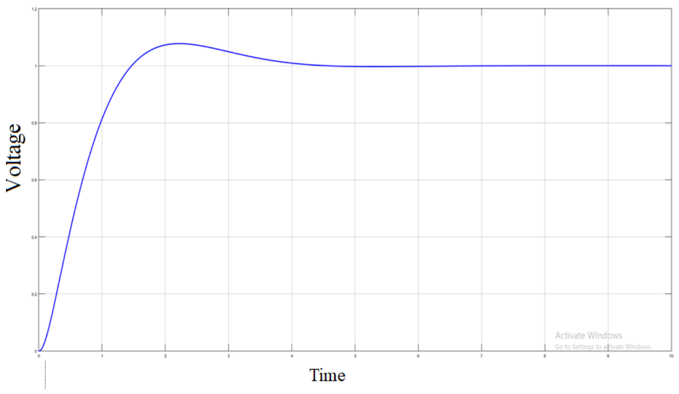

Fig -7: Response of the system in which PID is tuned with ANN

Table -1: Previously inputs and outputs of PID when not tuned

Inputs of PID		Outputs of PID	
K_P	1	Rise time	0.205 sec.
K_I	0.28	Settling time	4.37sec.
K_D	0.25	Overshoot	27.1%
N	100	peak	1.27

Table -2: Inputs and outputs of PID when tuned With ANN

Inputs of PID		Outputs of PID when tuned	
K_P	0.1777	Rise time	1
K_I	0.1785	Settling time	3.69
K_D	0.04206	Overshoot	7.75%
N	147	peak	1.08

4. CONCLUSIONS

In this article the ANN which is trained by Levenberg marquart which uses trainlm network training function is employed to tune PID controller to improve the response of output as well as the stability of the power system, the values of the PID constant obtained from the ANN are:- K_P=0.1777, K_I=0.1785, K_D=0.04206, all these values are shown in fig.4 in simulations and the outcome of the system are mentioned in table.1 and table.2.

REFERENCES

[1] Salah G. Foda, "ANN Based Power System Stabilizers for Large Synchronous Generators" J. King Saud Univ.,Vol. 14, Eng. Sci. (2), pp. 199-209, Riyadh (A.H. 1422/2002).

[2] Modu M. Ibrahim, Jibril D. Jiya and Idakwo O. Harrison, "Modelling and Simulation of Automatic Voltage Regulator System", International Journal of Computer Applications (0975 – 8887) Volume 178 – No.1, November 2017.

[3] Dr. Abdulrahim Thiab Humod and Abdullah sahib abdulsada, "Neural Network-based Robust Automatic Voltage Regulator (AVR) Of Synchronous Generator" published by research gate on 7 july 2017.

[4] Lodh B (2014) "Simulink Based Model for Analyzing the Ziegler – Nichols Tuning Algorithm as Applied on Speed Control of DC Motor" International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering. Vol 3, issue 1, PP 6641 – 6645.

[5] Patonding H. E, Lobo E. T and Sau M (2015) "Modeling Control of Automatic Voltage Regulator with Proportional Integral Derivative" International journal of research in Engineering and Technology. Vol.4, issue 9, PP 241 – 244.