

# **IMPLEMENTATION OF MACHINE LEARNING IN CONTROL SYSTEM**

*Dissertation Submitted for*

**PARTIAL FULFILLMENT OF THE REQUIREMENT  
FOR THE AWARD OF THE DEGREE OF  
BACHELOR OF TECHNOLOGY IN  
ELECTRICAL ENGINEERING**

*Submitted by*

**KANNEKANTI ASISH VENKATA SAI (16JE001921)**

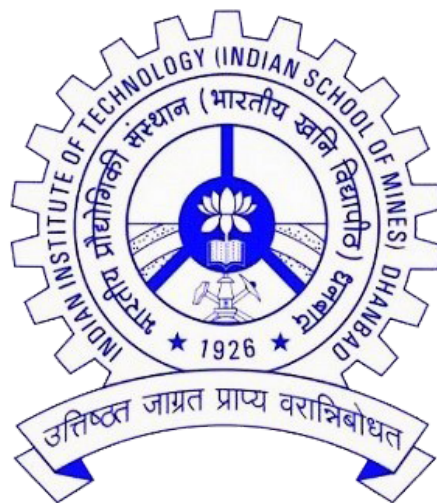
**BOKKA KOWSHIK (16JE002196)**

**PRADEEP (16JE002330)**

**Under the guidance of**

**Mr.Arijit Baral**

**Assistant Professor**



**DEPARTMENT OF ELECTRICAL ENGINEERING**

**INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES)**

**DHANBAD-826004**

01/05/2020

## **CERTIFICATE**

This is to certify that the dissertation titled “Implementation Of Machine learning in Control System” is being submitted to Indian Institute of Technology (Indian School of Mines) Dhanbad , by Mr. KANNEKANTI ASISH VENKATA SAI(Admission no. 16JE001921) in partial fulfilment of his Bachelor of Technology degree in Electrical Engineering of the same institution incorporates the results of his own work, carried out under my supervision and guidance. This dissertation has not been submitted for any other degree elsewhere to the best of my knowledge.

**Mr.Arijit Baral**

Project Guide

Assistant Professor

Department of Electrical Engg

IIT (ISM) Dhanbad

Jharkhand-826004

**Dr. Kalyan Chatterjee**

Head of the Department

Department of Electrical Engg

IIT (ISM) Dhanbad

Jharkhand-826004

## **ACKNOWLEDGEMENT**

I would take this opportunity to express my deep sense of gratitude to my project guide, **Mr.Arijit Baral**, Assistant Professor Department of Electrical Engineering, Indian Institute of Technology (Indian School of Mines), Dhanbad for his excellence guidance. Without his help, it would not have been possible for me to carry out the work smoothly. I am thankful for his valuable suggestions, supervision and constant inspiration throughout the course of this work without it would not have been possible to pursue this work and I am deeply indebted to him.

I would also like to express deep gratitude to **Dr. Kalyan Chatterjee**, Head of Department of Electrical Engineering for his periodic suggestions and cooperation during the course work. I was beneficiary of essential advice and assistance of the faculty and staff and Indian Institute of Technology (Indian School of Mines), Dhanbad. I also owe thanks to all persons who directly and indirectly helped me during the work till now.

**KANNEKANTI ASISH VENKATA SAI**

**B.Tech (Electrical Engineering)**

**Admission no. 16JE001921**

**DATE : 01/05/2020**

**Place: Dhanbad**

# **DECLARATION**

I, hereby declare that the thesis submitted is my own work and that I have exercised reasonable care to ensure that the work is original and to the best of my knowledge it does not breach any law of copyright and has not been accepted for the award of any degree or diploma of any other universities or other institutes of higher learning. To the extent, if any work from other papers or thesis is related, such work has been cited and acknowledged within the text of my work.

**KANNEKANTI ASISH VENKATA SAI**

**B.Tech (Electrical Engineering)**

**Admission no. 16JE001921**

Department of Electrical Engineering

IIT (ISM) Dhanbad

Dhanbad-826004

**Date:** 01/05/2020

**Place:** Dhanbad

## **ABSTRACT**

This paper proposes use of machine learning in control systems. In this paper we used Neural Networks and Genetic Algorithms as tools of machine learning for creating a neural network controller and tuning of PID parameters respectively. We used Matlab-Simulink platform for implementation.

# Contents

Acknowledgement

Declaration

Abstract

Table of contents

**Chapter 1: INTRODUCTION**

**Chapter 2: TUNING PID PARAMETERS USING GENETIC ALGORITHM**

**Chapter 3: CREATING A NEURAL NETWORK CONTROLLER**

**Chapter 4: RESULTS**

**REFERENCES**

## **CHAPTER 1**

### **INTRODUCTION**

#### **1.1 Motive:**

We started this project as a stimulator of our interests and as a base for our futures endeavors. The project uses the concepts of machine learning combined with control systems. Control systems are in general based on the same structure, building blocks and models of the dynamic system regardless of application, and can be mathematically analyzed w.r.t. stability, etc. Machine learning methods (ML), on the other hand, are highly flexible and adaptable methods but are not subject to physics-based models and therefore lack mathematical analysis. This paper presents the state of the art results using ML in the control system.

Before moving ahead we would like to provide a brief introduction to machine learning and control systems.

#### **1.2 Machine Learning:**

Machine learning according to Arthur Samuel: "The field of study that gives computers the ability to learn without being explicitly programmed."

Machine learning according to Tom Mitchell: "A computer program is said to learn from experience  $E$  with respect to some class of tasks  $T$  and to learn from experience  $E$  with respect to some class of tasks  $T$  and performance measure  $P$ , if its performance at tasks in  $T$ , as measured by  $P$ , improves with experience  $E$ ."

Machine learning is a collection of algorithms that allows the system to develop to become more accurate. The basic premise of machine learning is to build algorithms that can receive input data and use statistical analysis to predict an output while updating outputs will be available as new data. which is basically similar to the closed-loop control system.

Types of machine learning

- 1)Supervised Learning
- 2)Unsupervised Learning

## Supervised Learning

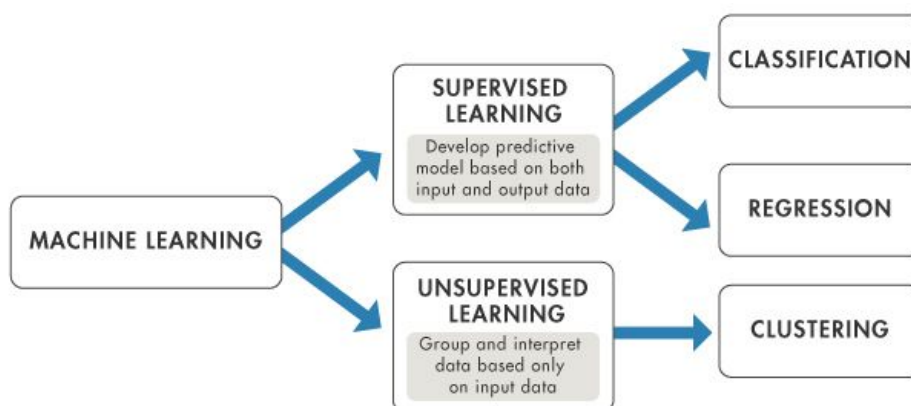
In supervised learning, we are given a data set and already know what our correct output should look like, having the idea that there is a relationship between the input and the output.

Supervised learning problems are divided into “regression” and “classification” problems. In a regression problem, supervised learning tries to predict results for continuous output, meaning it is trying to map input variables to create a continuous function. In a classification problem, it tries to predict results in the form of a discrete output.

## Unsupervised Learning

Unsupervised learning is used when there is little or no idea about what our results should look like. Unsupervised learning can create a structure with data where the effect of the variables doesn’t need to be known.

The structure can be derived by clustering the data based on relationships among the variables in the data. In unsupervised learning, the prediction results don’t have any feedback.



In this paper, we used a neural network algorithm which is a supervised learning algorithm



### 1.3 Control Systems:

The control system which provides the desired response by controlling the output is called a control system. The below diagram represents a simplified block diagram of a control system.



In the above figure control system is represented in a single block. Since the output can be controlled with the help of varying input thus control system has gained its name. The input can be varied with the help of some mechanism. With the development of industry, the control system became more popular, because of the complexity of the system by unknown parameters with time-varying, large delays, and nonlinear complex systems.

PID(Proportional-Integral-Derivative) controller is the best control algorithm among the industry.

Which is highly used in industries like robotics, chemical industries, etc.

PID controllers are best for reasons such as robust performance, low cost, and easy maintenance, a wide range of operating conditions, and most importantly simplicity of function.

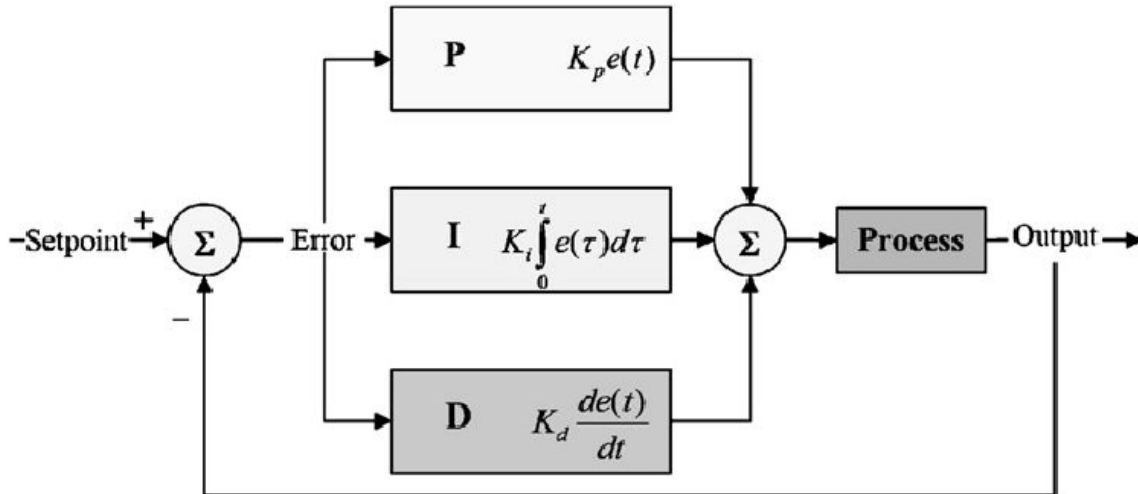
But there are still some disadvantages of the PID controller: changing PID parameters in real-time is difficult. To compensate this effect certain various methods are included such as

Fuzzy controller, genetic algorithms, and artificial neural networks. Where the major advantage is it doesn't depend upon a defined/precise mathematical model.

This paper consists of only a basic neural network controller as a replacement of a PID controller.

As mentioned above PID controller is one of the best control methods for a system.

PID controller



$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$

Choosing appropriate PID parameters is also difficult. There are certain mathematically computing methods like the Iterative method and Ziegler-Nichols method etc. But these methods had some limitations like the effect of delay instability of systems. Since there are certain cases where we can not use the above tuning methods. In these types of systems, there is the development of soft computing ways of parameterizing PID parameters with real-time cases.

In this paper, we have worked on Implementing PID parameters using the genetic algorithm by considering Integral of time multiplied by Absolute Error (ITAE) as the objective function.

#### 1.4 Objective of the paper:

Tuning PID (proportional integral derivative) parameters with genetic algorithms, Creating a neural network controller block. The paper provides a detailed explanation of the procedures and tools used.

## **CHAPTER 2**

### **TUNING PID PARAMETERS USING GENETIC ALGORITHM**

#### **2.1 Genetic Algorithm Theory:**

A genetic algorithm is a search algorithm that learns from itself which is inspired by Charles Darwin's theory of natural evolution. This algorithm is the reflection of the process of natural selection where the fittest individuals are selected among all selections for the reproduction of the new generation. Genetic algorithms are search algorithms that act on a population of possible solutions. They are based upon the mechanics of population generation and selection. The potential solutions are stated as 'genes' – strings of characters. New solutions are produced by mutating members of the current population, and by 'mating' two solutions together which generate a new solution. The better solutions are selected to breed and mutate to form a new generation and the worse ones are discarded from the group. They are also called probabilistic search methods; this means that the states which they explore are not determined only by the properties of the problems. A random process helps in guiding the search. Genetic algorithms are used in artificial intelligence like to search for potential solutions to find one which solves the problem

**The following statements outlines how the genetic algorithm works:**

- a) It creates a random initial population.
- b) Then it creates a sequence of new populations. At each step, the algorithm creates a new next population using the current generation. he algorithm performs the following steps for creating a new population:
  1. Scores of each member of the current population are calculated according to their fitness value.
  2. Scaling is done to convert them into a more useful range of values.
  3. Parents are the selected members who are selected, based on their fitness.
  4. The individuals that have lower fitness than the current The population is chosen as elite. Which are passed as the next

- population.
5. The children are produced from the parents by either making some random changes to a single parent (mutation) or by combining the vector entire to a pair of parents (Crossover).
  6. The current population is replaced with their children for forming the next generation.

c) when the stopping criteria are met, The algorithm stops.

### **Stopping Conditions for the Genetic Algorithm**

The genetic algorithm uses the following as stopping conditions

**Generations** — The algorithm stops when no of generations reaches the value of set for Generations.

**Time limit** — The algorithm stops after running a set amount of time (seconds).

**Fitness limit** — The algorithm stops when the value of the fitness function for the best point in the current population is equal to or less than the Fitness limit.

**Stall generations** — The algorithm stops if the average relative change in the fitness function value over Stall generations is less than Function tolerance.

**Stall time limit** — The algorithm stops if there is no improvement in the objective function over an interval of time (seconds).

**Stall test** — The stall condition is either geometric weighted or average change, The weighting function is  $1/2n$ , For geometric weighted, where  $n$  is the number of generations. Both conditions are applied to the relative change in the fitness function over Stall generations.

**Tolerance** — The algorithm stops after the average relative change in the fitness function value over Stall generations is greater than or equal to Function tolerance.

**Nonlinear constraint tolerance** — Nonlinear constraint tolerance is not a stopping criterion rather it is used to determine the feasibility with respect to nonlinear constraints. Also, If the linear constraints violation is below the square root of Nonlinear constraint tolerance. then the point is feasible.

When any one of the conditions is met the algorithm stops. In the Optimization app, the values of stopping criteria can be adjusted accordingly or The default

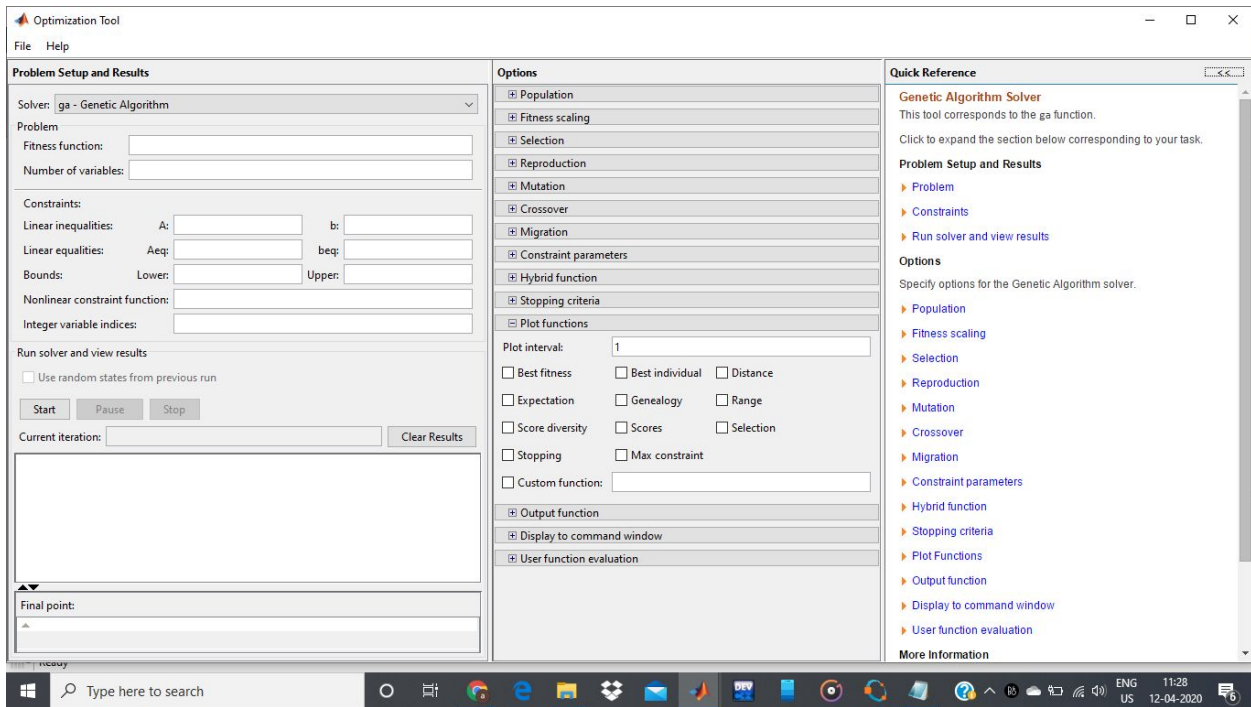
values are set. When the algorithm stops, the result panel displays the criteria that caused the algorithm to stop. The Stall time limit and Time limit prevent the algorithm from running too long. If the algorithm is stopped by any of the above reasons, where you can improve the Stall time limit and Time limit for better results.

## **2.2 Implementation Of Genetic Algorithm In Matlab using Optimization Toolbox**

Optimization Toolbox is a function that provides parameters that minimize or maximize objective function while satisfying the constraints. The toolbox includes solvers for linear programming, mixed-integer linear programming, quadratic programming, nonlinear optimization, and nonlinear least squares. These solvers can be used to find optimal solutions to discrete and continuous problems, also perform some tradeoff analyses, and can incorporate optimization methods into algorithms as well as applications.

The steps for implementation optimization toolbox

- Choose an optimization solver.
- Create an objective function.
- Choose the constraints if required.
- Set another option, or can use the default options.
- Finally call the appropriate solver.



In this paper, we considered Solver as a ga-Genetic Algorithm and other functions are mentioned in the procedure.

## 2.3 Procedure

### Assumptions about plant:

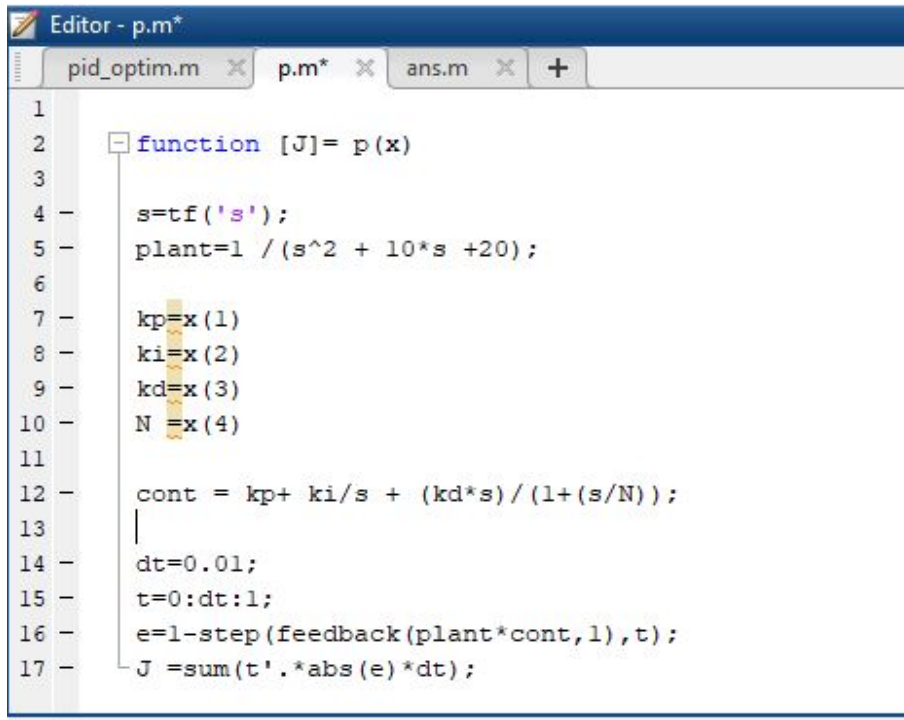
- 1) considering the plant model as  

$$\text{plant} = 1/(s^2 + 10s + 20);$$
- 2) cost function is considered as the difference between the required output and required output only
- 3) considering parameters range between 0-500

### Steps:

- 1) Launch MATLAB

## 2) Code the following



```
1
2 function [J]= p(x)
3
4 s=tf('s');
5 plant=1/(s^2 + 10*s +20);
6
7 kp=x(1)
8 ki=x(2)
9 kd=x(3)
10 N =x(4)
11
12 cont = kp+ ki/s + (kd*s)/(1+(s/N));
13
14 dt=0.01;
15 t=0:dt:1;
16 e=1-step(feedback(plant*cont,1),t);
17 J =sum(t'.*abs(e)*dt);
```

## 3) open apps -> optimization in Matlab

Solver	genetic algorithm
fitness function	@(x)function_name(x)
no of variables	Four
Variables	Kp, Ki, Kd, N
lower bound	[0 0 0 0]
upper bound	[500 500 500 500]

here @ implies the input x varies are stored

Upper and Lower Bound are according to assumptions

Kp - Proportional Term

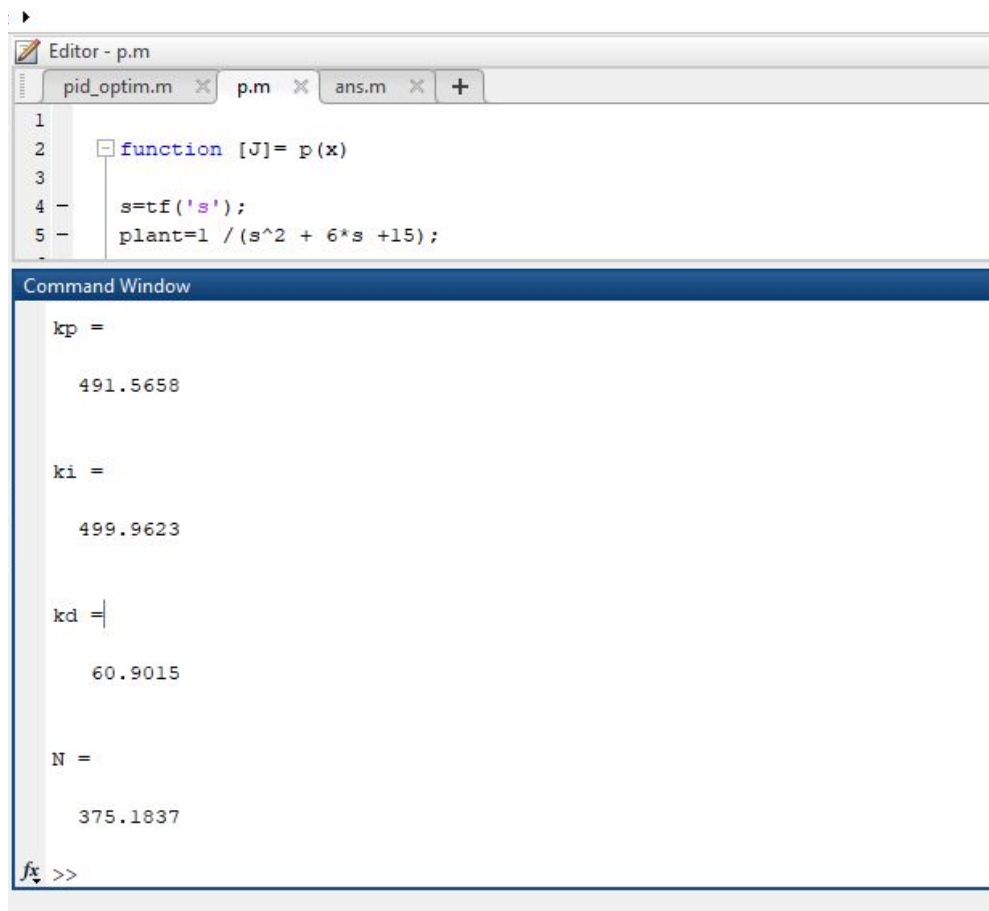
Kd - Differential Term

Ki - Integral Term

N - Filter coefficient

## 4) Start the simulation

## 2.4 Output:



The image shows a MATLAB environment with an Editor window and a Command Window. The Editor window displays a function `p(x)` that defines a transfer function `s` and a plant model `plant`. The Command Window shows the results of an optimization process, displaying the values of `kp`, `ki`, `kd`, and `N`.

```
Editor - p.m
pid_optim.m x p.m x ans.m x +
1
2 function [J]= p(x)
3
4 s=tf('s');
5 plant=1/(s^2 + 6*s +15);

Command Window

kp =

    491.5658

ki =

    499.9623

kd =

    60.9015

N =

    375.1837

fx >>
```



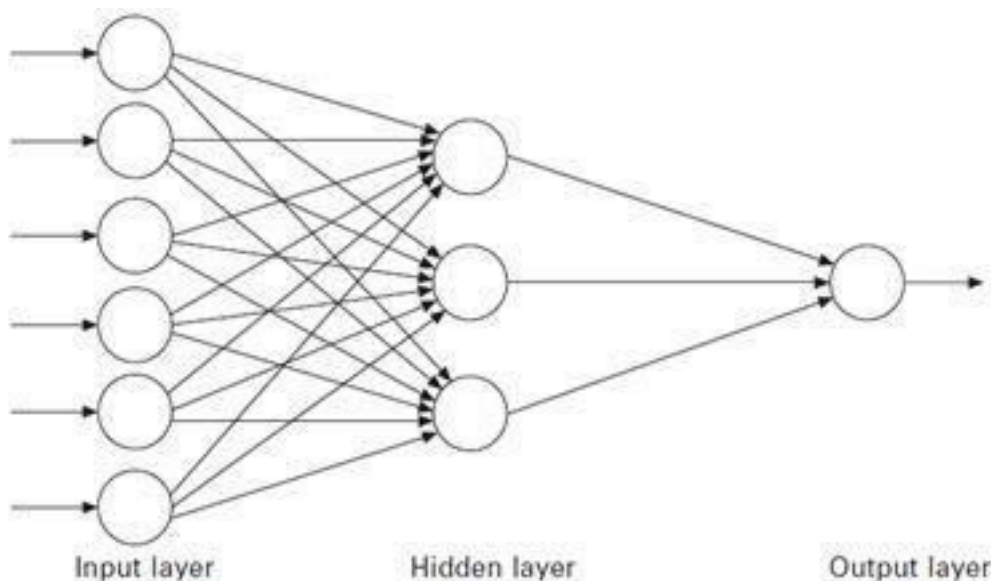
## CHAPTER 3

### CREATING A NEURAL NETWORK CONTROLLER

#### 3.1.1 Neural Network

As described above Supervised learning is one of the branches of machine learning. The major objective of supervised learning is to predict the target variable with help of sample target data, in other words, it is stated as Supervised learning helps in calculating the target value with considering several data or sample input and output pairs.

Where Artificial Neural Networks(ANN) is one of the supervised learning models. ANN was replicated as the way of neurons in our brain work. It is created of multiple layers consisting of multiple numbers of artificial neurons. These layers are called neural layers and the neurons are called activation functions.



Structure of an artificial neural network

## Neural Layers

As shown in the above figure basic neural layers are of three types according to their activity

- 1) The input layer consists of the raw information which will feed into the network.
- 2) The activity of the hidden layer is determined by the activities of the input units and the weights on the connections between the input and the hidden units.
- 3) The output layer behaves upon the activity of the hidden units and the weights between the hidden and output units.

## Activation function

The activation function is simply a mathematical function which simply generates an output according to the input.

Mathematically, Let each layer has several inputs  $X$ . Each input has a weight  $W$  which is added to a bias  $B$  forming a sum equation  $XW + B$ . The sum equation is fed into an activation

The function will compute the output.

### 3.1.2 Back Propagation

A neural network is trained with the data by adjusting the weight and bias parameters of each neuron with the help of the backpropagation algorithm.

Initially, weights are initialized with some random values those random weights which produce a certain output for input data which is way different from the required output i.e. the error value is high. The model needs to change the weight parameters to reduce the error. This work is done by the backpropagation algorithm.

The backpropagation works with the help of gradient descent or delta rule which minimizes the error in the weight space. The weights which produce the minimum error is considered as a solution to a learning problem.

Neural network training has got basically 3 steps

## 2)Forward Propagation

Initially, the output is calculated according to the initialized weights/previously changed weights and the Mean Squared Error(E)is calculated.

## 2)Back Propagation

Now, we will propagate backward trying to reduce the error with the help of changing the weights according to gradient descent rule.

## 3)Changing the weights according to the error

Weights are updated accordingly.

This procedure is calculated for every data set in training data, weights are updated accordingly after each pair of data, thus the model optimizes for better prediction of the future model.

In this paper, we are trying to implement a neural network controller in place of the PID controller for a DC Motor.

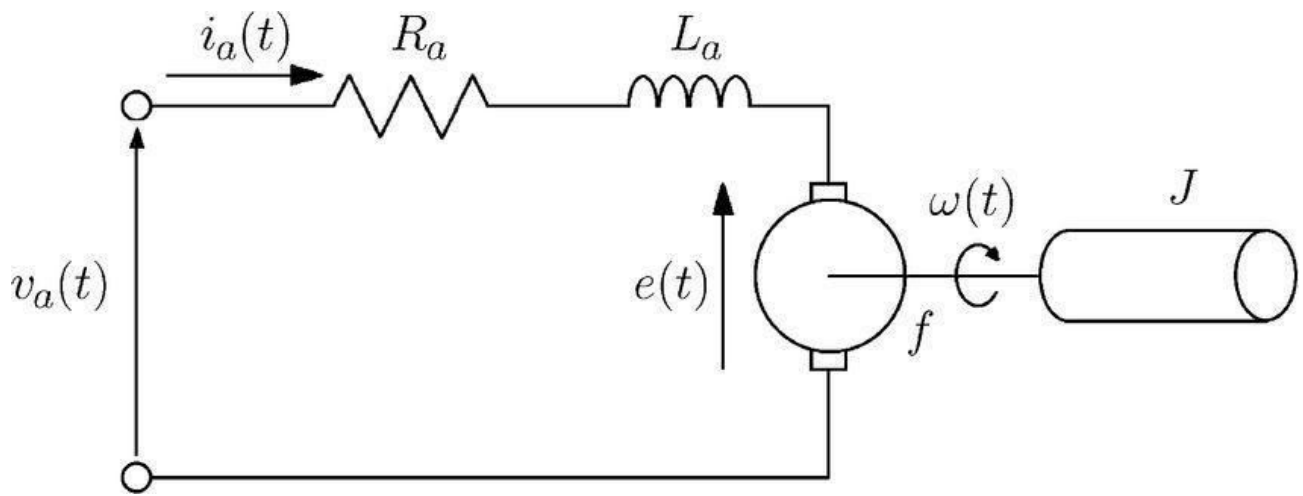
## **3.2 PID control DC Motor**

DC motors are often used in various industrial applications where a wide range of responses is required to follow a predetermined trajectory of speed or position under variable load for implementation. Where the PID controller is a common control application for DC Motor.

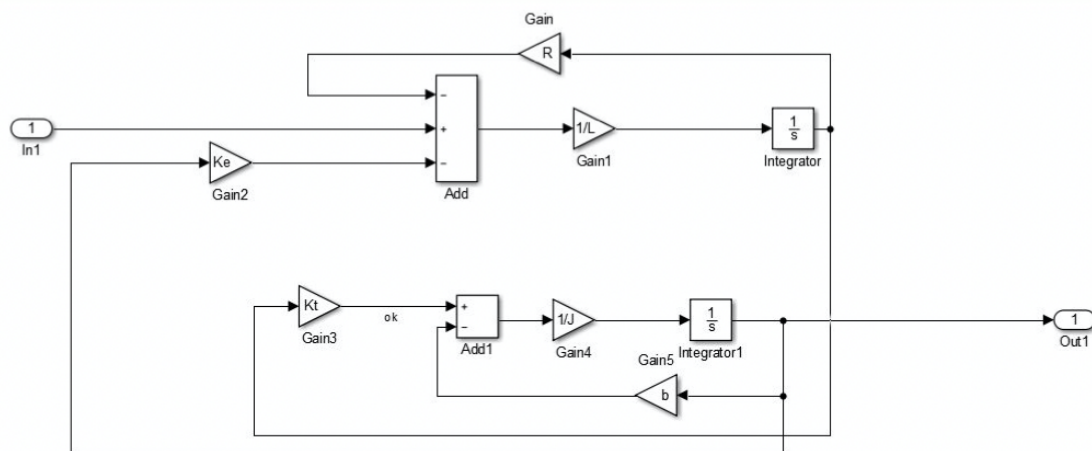
### **3.2.1 Creating a DC Motor Simulink block**

Considering a basic DC motor model

From the above equations, we created as DC SIMULINK Model and masked it as DC motor



$$\begin{cases} \frac{di(t)}{dt} = \frac{e(t) - R_a i(t) - K_b \omega(t)}{L_a} \\ \frac{d\omega(t)}{dt} = \frac{K_t i(t) - B \omega(t) - A_r - \tau_d}{J} \end{cases}$$



### 3.2.2 Implementing a PID controller based DC Motor in SIMULINK

Created a PID controller based DC motor circuit in Simulink and simulated  
Assuming DC motor as

Block Parameters: DC MOTOR

Subsystem (mask)

Parameters

R

1

L

0.5

J

0.01

b

0.1

Ke

0.01

Kt

0.01

PID controller as

Block Parameters: PID Controller

PID Controller

This block implements continuous- and discrete-time PID control algorithms and includes advanced features such as anti-windup, external reset, and signal tracking. You can tune the PID gains automatically using the 'Tune...' button (requires Simulink Control Design).

Controller: PID Form: Parallel

Time domain:

☒ Continuous-time

☐ Discrete-time

Main PID Advanced Data Types State Attributes

Controller parameters

Source: internal [Compensator formula](#)

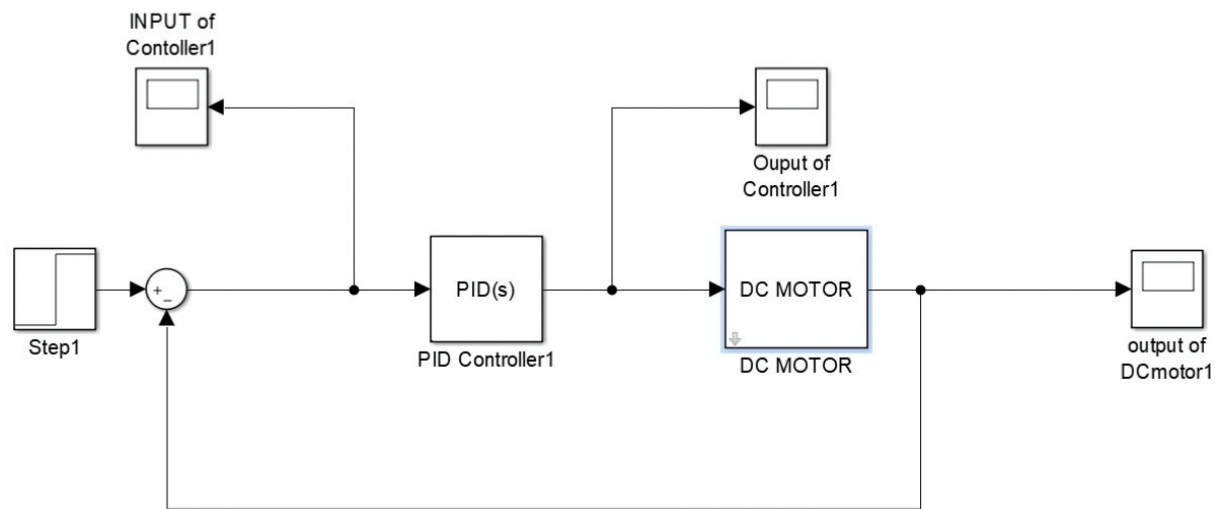
Proportional (P): 20

Integral (I): 50

Derivative (D): 2

Filter coefficient (N): 100

$$P + I \frac{1}{s} + D \frac{N}{1 + N \frac{1}{s}}$$



Collected data from scope and stored

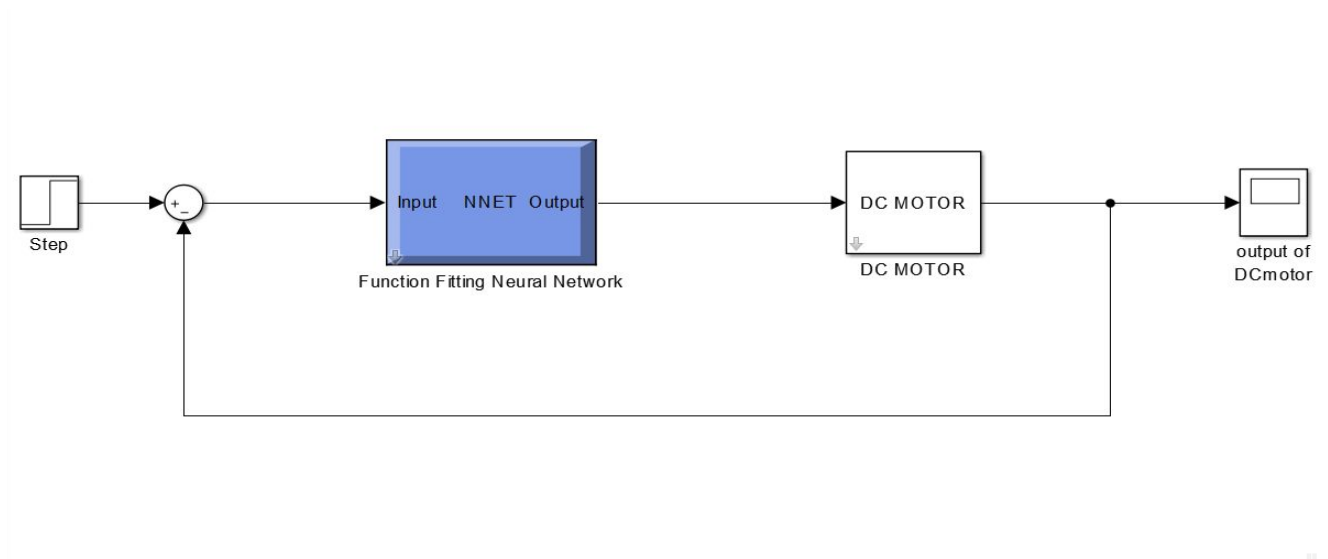
([https://docs.google.com/spreadsheets/d/18\\_G9SsRwERKAvl8Kqq-RwLUHScq7icQcLDcowd63fZU/edit?usp=sharing](https://docs.google.com/spreadsheets/d/18_G9SsRwERKAvl8Kqq-RwLUHScq7icQcLDcowd63fZU/edit?usp=sharing))

### 3.3 Creating a neural network controller using nf tool

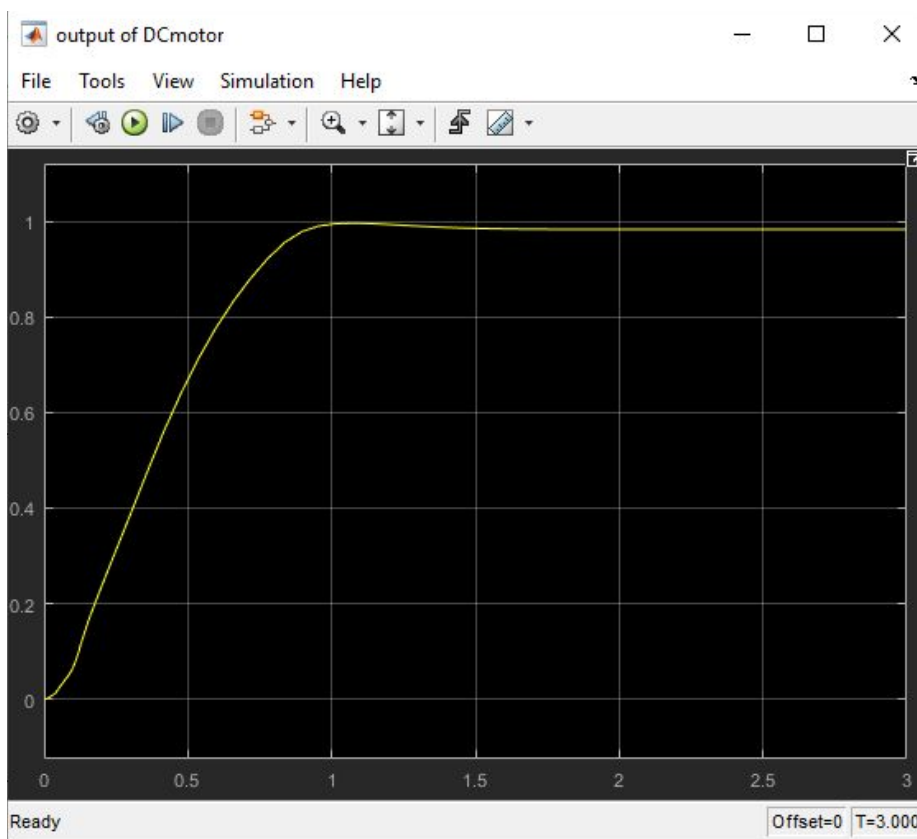
In, this paper we used the collected data as training data and trained the model using nftool kit of (single layer 12 nodes neural network) which generated a code

([https://docs.google.com/document/d/1QPkPt2mGOaPL9ObRwggw\\_K\\_jftlIDyozOw53YiyO\\_D0/edit?usp=sharing](https://docs.google.com/document/d/1QPkPt2mGOaPL9ObRwggw_K_jftlIDyozOw53YiyO_D0/edit?usp=sharing))

From the above code, we created a Neural network controller Simulink block and created the following Simulink model and tested for a step input.



Which generated the following output



## CHAPTER 4

### RESULTS

#### 4.1 PID TUNING RESULTS

Comparing results according to the cost function

Kp	19.32
Ki	42.54
Kd	1.97
N	385.3

a) PID tuned values by Matlab are

b)PID values generated by the given code is

Kp	491.5658
Ki	499.9623
Kd	60.9015
N	375.1837

And cost generated a)-- > 0.0816

b)-- > 0.0054

We can observe from the above result tuning PID parameters with genetic algorithms provides better results but here we had considered only error but not the settling time which might also come in the picture in actual cases.

Future Vision

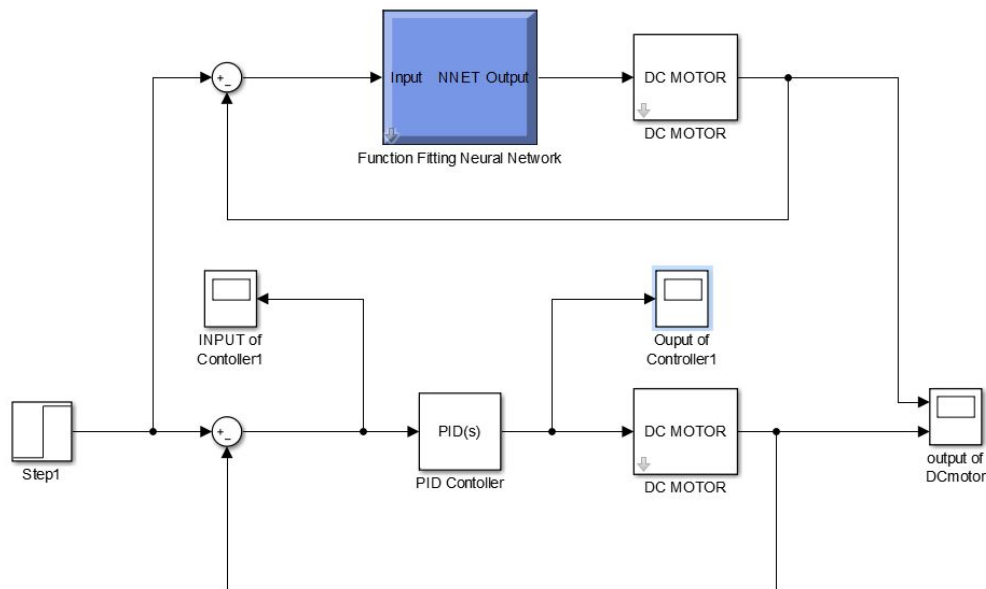
-----> We must generate a cost function which includes all the required parameters(like settling time, max peak, etc)



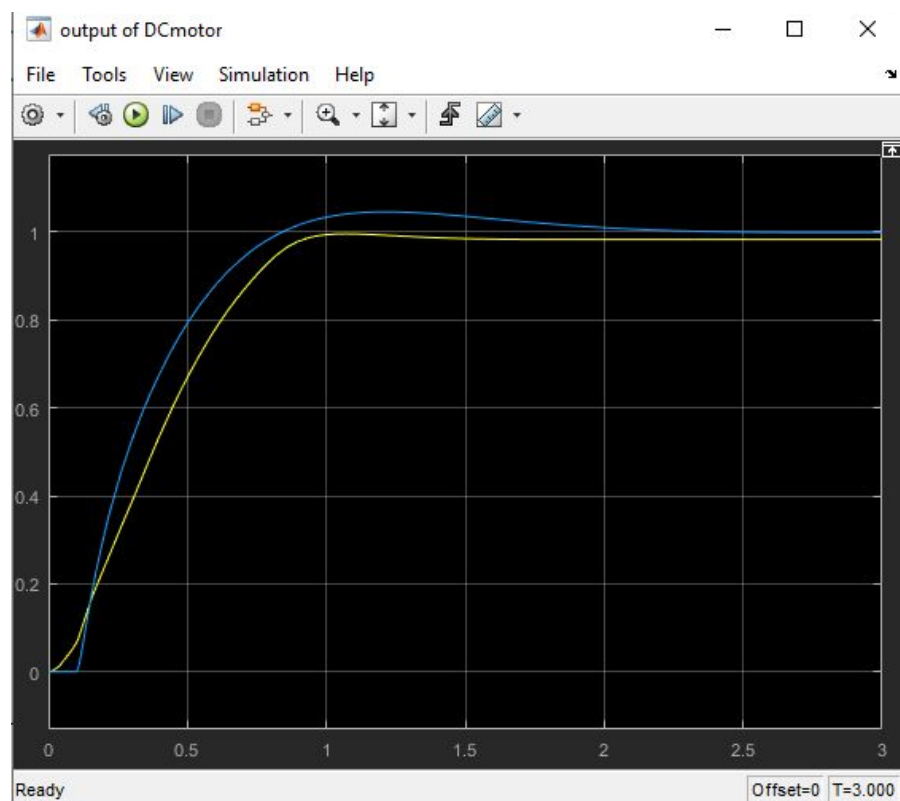
## 4.2 NEURAL NETWORK CONTROLLER RESULTS

Comparing neural network controller with PID controller

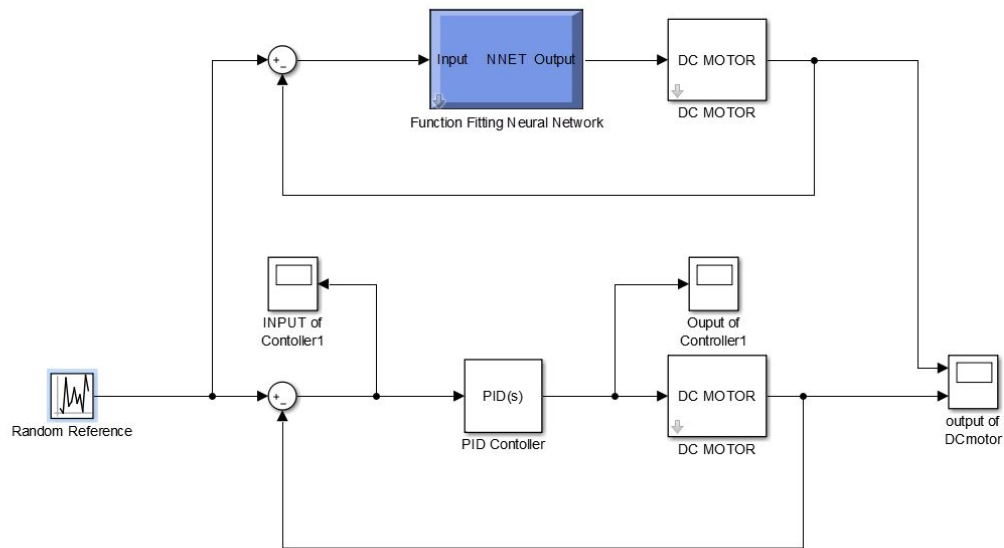
1) For step input Circuit



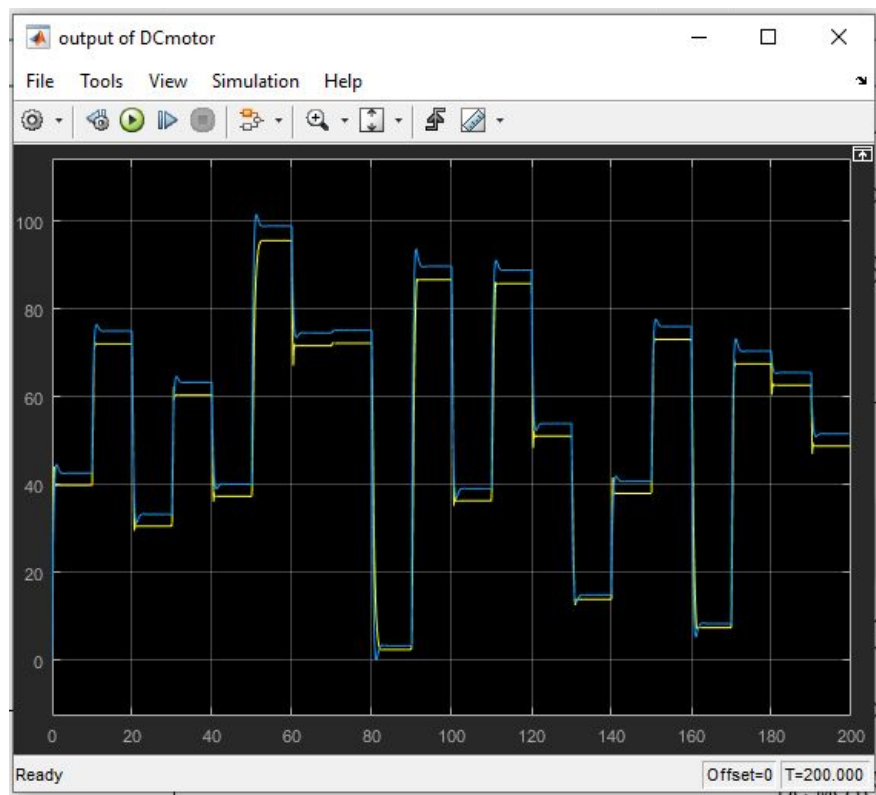
Scope output -> blue color is PID controller output, Yellow color is created controller output



## 2) For random input Circuit Diagram



Scope output -> blue color is PID controller output, Yellow color is created controller output



From the above results

a)

we can observe that our neural network controller is producing a better result than the PID controller

It has produced a better result in settling time, max peak, etc.

b)

We can observe that the neural network controller output curve was smooth along the edges or sudden changes compared to the PID controller

From the above results, we can say that the neural network is performing better than the PID controller.

Future vision-->

we can train the neural network controller according to our requirement with the specified input data which can produce a better result than the above controller.

## REFERENCES

--- Introduction to Machine Learning by Alex Smola and S.V.N. Vishwanathan

--- Machine learning Yearning by ANDREW NG

---. Neural network fundamentals by Bose, N. K.; & Liang P. McGraw-Hill.

---Neural Networks by 3Blue1Brown

[https://www.youtube.com/watch?v=aircAruvnKk&list=PLZHQbOWTQDNU6RI\\_6700DX\\_ZCJB-3pi](https://www.youtube.com/watch?v=aircAruvnKk&list=PLZHQbOWTQDNU6RI_6700DX_ZCJB-3pi)

---Genetic algorithm by David E. Goldberg.

--- Speed control of DC motor using optimization techniques based PID Controller By Santosh Kumar Suman, Vinod Kumar Giri by IEEE.

Department of Electrical Engineering, Madan Mohan Malviya University of Technology, Gorakhpur, a

---Handbook of PI and PID controller tuning by Aidan O'Dwyer