International Journal of Engineering Sciences Paradigms and Researches (IJESPR)
Volume 54, Issue 02 and Publishing Date: 22nd June 2025
An Indexed, Referred and Peer Reviewed Journal
ISSN (Online): 2319-6564
www.ijesonline.com

Investigation into Multilayer Perceptron (MLP) Architecture for Optimal PID Parameter Prediction in DC Motor Speed Control

Hatim Osman Elhassan[1], PhD Student, M.I. Shukri[2], Professor, Alsalama College of Science & Technology Khartoum, Sudan

Abstract
This paper investigates the application of a Multilayer Perceptron (MLP) architecture for enhancing DC motor speed control by predicting optimal PID parameters. As a feed forward neural network, the MLP receives dynamic features such as error, integral term, and derivative term to output the predicted Kp, Ki, and Kd values. The architecture includes hidden layers utilizing ReLU activation and an output layer with softplus activation to ensure non-negative parameters. Trained using a synthesized dataset derived from simulating the DC motor model and optimizing PID parameters via a Genetic Algorithm, the network minimizes Mean Squared Error with the Adam optimizer. Simulation results demonstrate the MLP provides fast rise and settling times and exhibits computational efficiency. However, its reliance on static input features limits its adaptability and robustness in highly dynamic environments compared to architectures capable of temporal modeling.

Keywords: DC Motor Control, Artificial Neural Network (ANN), Multilayer Perceptron (MLP), PID Control, Parameter Prediction, Speed Regulation

1. Introduction

DC motors have long stood as a cornerstone of electromechanical systems since their invention in the 19th century, owing to their pivotal role in converting electrical energy into precise mechanical movements [1]. These motors are integral to numerous industrial applications such as robotics, automotive systems, aerospace, and industrial automation, where controlled speed, torque, and position are paramount. For instance, robotic arms rely on DC motors for fine, accurate movements [1], while electric vehicles utilize them for efficient propulsion [2]. The widespread adoption of DC motors stems from their simplicity, reliability, and versatility, enabling them to remain relevant despite competition from AC motors [3]. Effective control strategies are essential for optimizing the performance of DC motors. Traditional methods, such as Proportional-Integral-Derivative (PID) controllers, are widely used to regulate motor speed and torque by adjusting input voltage or current [4]. However, these techniques often assume linear system behavior with fixed parameters, which fails to account for real-world complexities like load variations, friction, and thermal effects [5]. As industries demand higher precision and adaptability, the limitations of conventional PID control become increasingly evident, necessitating advanced solutions capable of handling nonlinear dynamics and uncertain environments.

In response to these challenges, adaptive control systems—such as Model Reference Adaptive Control (MRAC)—have emerged as promising alternatives by modifying controller parameters based on system feedback [6]. More recently, Artificial Neural Networks (ANNs) have shown potential in addressing nonlinear control problems through learning-based approaches. Despite this progress, there remains a significant research gap in applying ANNs, particularly Multi-Layer Perceptron (MLP) networks, to dynamic and nonlinear DC motor speed control tasks. While MLP has been explored in static motor applications, its use for time-varying scenarios remains underexplored, especially when compared to traditional methods like PID and MRAC [7]. This study aims to bridge that gap by implementing an intelligent ANN strategy—specifically, an MLP-based controller—to enhance the precision and adaptability of DC motor speed regulation. Leveraging the MLP's ability to learn complex mappings between inputs and outputs, this research will focus on predicting optimal PID parameters in real-time under varying operating conditions. The contribution lies in developing a novel framework that surpasses the limitations of conventional control methodologies, offering a more robust solution for nonlinear and uncertain motor dynamics. By evaluating the performance of this MLP-PID hybrid against key metrics such as rise time, settling time, and overshoot, this work seeks to demonstrate the superiority of ANN-based control in meeting modern industrial demands.

2. Background

2.1 DC Motor

A Direct Current (DC) motor is an electromechanical device that converts electrical energy into mechanical motion [8]. Its construction consists of two primary components: the stator and the rotor. The stator, the stationary part, contains poles excited by a direct current to produce a magnetic field. The rotor, the rotating component, features a laminated iron core with slots housing coils. These coils are connected in series and interact with the magnetic field to generate motion [9].

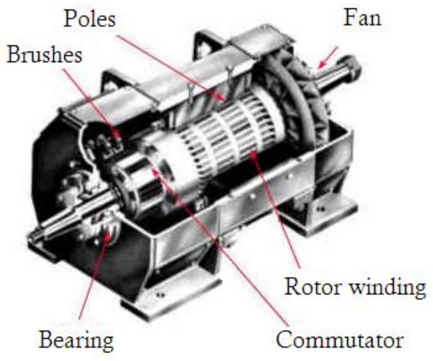

Fig 1. DC motor rotor construction

The principle of operation of a DC motor is rooted in electromagnetic induction and torque generation. Electromagnetic induction occurs when the rotor rotates within the stator's magnetic field, causing the armature coils to cut through magnetic flux lines. This induces an electromotive force (EMF), known as back EMF:

$$E_b = K_e \omega \#(1)$$

Where E_b is the back EMF, K_e is the motor's back EMF constant, and ω is the angular velocity of the rotor.

Torque generation arises from the interaction between the magnetic field and current flowing through the armature windings. The resulting torque drives the rotor and is expressed as:

$$T = K_t I_a \#(2)$$

where T is the torque, K_t is the torque constant, and I_a is the armature current.

The voltage equation for the armature circuit is given by:

$$V_a = E_b + I_a R_a + L_a \frac{dI_a}{dt} \#(3)$$

In steady-state conditions ($\frac{dI_a}{dt} = 0$), this simplifies to:

$$V_a = K_e \omega + I_a R_a \#(4)$$

These equations collectively describe how applied voltage and current influence motor speed and torque, underpinning its operational dynamics [10].

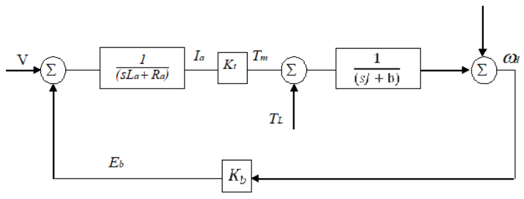

Fig 2. DC Motor Block Diagram

2.2 PID Controller

The Proportional-Integral-Derivative (PID) controller is an important of control engineering, widely used for its simplicity and effectiveness in regulating dynamic systems such as DC motors. A PID controller minimizes the error between the desired setpoint (ω_{set}) and the actual output $(\omega(t))$ by applying a weighted sum of three terms: proportional, integral, and derivative. The control signal $(V(t))$, which corresponds to the applied voltage in the case of a DC motor [11], is computed as:

$$V(t) = K_p \cdot e(t) + K_i \cdot \int e(t)\, dt + K_d \cdot \frac{de(t)}{dt} \#(5)$$

Where:

- $e(t) = \omega_{\text{set}} - \omega(t)$: Error between the set speed and the current speed.
- K_p: Proportional gain, which directly reduces the error.
- K_i: Integral gain, which eliminates steady-state error by accumulating past errors over time.
- K_d: Derivative gain, which minimizes overshoot and oscillations by predicting future trends based on the rate of change of the error [11].

2.3 Multi-Layer Perceptron (MLP)

The Multilayer Perceptron (MLP) is a fundamental class of artificial neural networks that has demonstrated remarkable versatility in modeling complex, nonlinear relationships within data. Its architecture consists of an input layer, one or more hidden layers, and an output layer, with each layer comprising multiple interconnected neurons [12]. This layered structure enables the MLP to function as a universal approximator, capable of capturing intricate patterns in both classification and regression tasks.

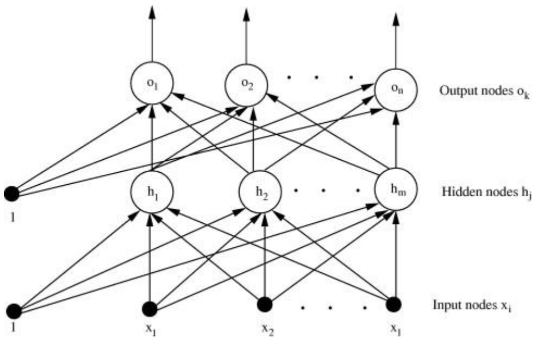

Fig 3. Two Layer Perceptron.

An MLP's architecture is characterized by its fully connected feedforward structure, where each neuron in a layer connects to all neurons in the subsequent layer. The input layer receives raw data features x=[x1,x2,...,xl]∈Rl, which are then propagated through the network. In the hidden layers, neurons compute weighted sums of their inputs and apply a nonlinear activation function f(·). A commonly used activation function is the sigmoid function [12]:

$$(x) = \frac{1}{1 + \exp(-x)} \#(6)$$

3. Methodology

3.1 DC Motor Model assumptions

While the DC motor model presented here is comprehensive, certain assumptions are made to simplify the analysis:

- Linearity: The relationships between voltage, current, torque, and speed are assumed to be linear. Nonlinear effects such as magnetic saturation are neglected.
- Constant Parameters: Parameters such as $(R_a), (L_a), (K_e), (K_t), (J), and (b)$ are assumed to remain constant. Variations due to temperature, wear, or other factors are ignored.
- Ideal Conditions: Effects such as eddy currents, hysteresis, and bearing friction are not explicitly modeled.

These assumptions ensure that the model remains tractable for simulation and control design while still capturing the essential dynamics of the motor.

3.2 DC Motor Model Parameters

To ensure the simulation reflects real-world conditions, we define the parameters of a 0.5 HP, 6000 RPM DC motor. These parameters are chosen based on typical values for small industrial motors, which are commonly used in robotics, automation, and precision control applications. The selected parameters are as follows:

- Armature Resistance (Ra): 0.027 Ω
- Armature Inductance (La): 0.002 H
- Back EMF Constant (Ke): 0.0382 V·s/rad
- Torque Constant (Kt): 0.0382 N·m/A
- Moment of Inertia (J): 5×10^{-5} kg·m²
- Viscous Friction Coefficient (b): 5×10^{-3} N·m·s/rad

3.3 Simulating PID-Controlled DC Motor

To simulate the behavior of a PID-controlled DC motor, the mathematical model of the motor described in Section 2.2 is integrated with the PID control law. The simulation involves numerically solving the coupled electrical and mechanical equations while dynamically updating the control signal based on the error, integral term, and derivative term. Below is step by step instruction on the simulation is achieved using python programming language.

3.4 Key Steps in DC Motor Model Simulation

1. Initialization:

- Define initial conditions, including initial speed ω_0, set speed ω_{set}, load torque T_L, and PID gains K_p K_i K_d.
- Specify simulation parameters time step = $1*10^{-4}$ seconds and maximum simulation time of 1.5 seconds.

2. Dynamic Calculation of Error Terms:

- At each time step, compute the error e(t), integral term $\int \mathrm{e(t)}$, dt, and derivative term $\frac{\mathrm{de(t)}}{\mathrm{dt}}$ using numerical methods.

3. Control Signal Computation:

- Apply the PID control law to calculate the applied voltage V(t).

4. Motor Dynamics Simulation:

- Update the motor's speed and current using the electrical and mechanical equations:

$$V(t) = R_a \cdot I(t) + L_a \cdot \frac{dI(t)}{dt} + E_b(t) \#(7)$$

$$J \cdot \frac{d\omega(t)}{dt} = T_m(t) - T_L(t) - b \cdot \omega(t) \#(8)$$

Where Eq. (7) and Eq. (8) are electrical and mechanical equations consecutively.

5. Performance Evaluation:

- Analyze the motor's response by calculating key performance metrics, as described in the next section.

3.5 Performance Metrics

The performance of the PID-controlled DC motor is evaluated using several key metrics derived from the step response of the system. These metrics provide quantitative insights into the controller's ability to regulate the motor's speed effectively. The equations for each metric are as follows:

1. **Rise Time (tr):** The time taken for the motor speed to reach 90% of the set speed.
2. **Settling Time (*ts*):** The time required for the motor speed to stabilize within ±2% of the set speed.
3. **Overshoot (*Mp*):** The maximum percentage deviation above the set speed during the transient phase.

$$M_p = \frac{\max\big(\omega(t)\big) - \omega_{\text{set}}}{\omega_{\text{set}}} \cdot 100\backslash\%\#(9)$$

4. **Steady-State Error (*ess*):** The difference between the final speed and the set speed after the transient phase.

$$e_{ss} = \lim_{t\to\infty} |\omega(t) - \omega_{\text{set}}| \#(10)$$

5. **Peak Time (*tp*):** The time at which the motor speed reaches its first peak.
6. **Damping Ratio (*ζ*):** A measure of how oscillatory the system is. It is derived from the second-order system approximation.

$$\zeta = \frac{-\ln\big(M_p/100\big)}{\sqrt{\pi^2 + \big[\ln\big(M_p/100\big)\big]^2}} \#(11)$$

7. **Natural Frequency (*ωn*)**: The frequency at which the system oscillates in the absence of damping.

$$\omega_n = \frac{\pi}{t_p\sqrt{1-\zeta^2}} \#(12)$$

8. **Down Time:** The duration during which the motor speed falls below the set speed before stabilizing.

$$\text{Down Time} = \int_{t_1}^{t_2} 1\ dt \quad \text{where } \omega(t) < \omega_{\text{set}} \#(13)$$

These metrics are calculated from the simulated data and used to assess the effectiveness of the PID controller under different operating conditions and gain settings.

3.6 Synthesis of Training Dataset

The synthesis of a high-quality training dataset is a cornerstone in developing machine learning models for controlling DC motors. This dataset serves as the foundation for training MLP model, enabling it to learn the intricate relationships between dynamic features (e.g., error, integral term, derivative term) and optimal PID parameters $\big(K_p\big), (K_i), (K_d)$.

A well-synthesized dataset ensures that the trained models generalize effectively to unseen scenarios, making them robust to variations in operating conditions. For a DC motor control system, the dataset must capture the relationship between:

Input Features: Error e(t), integral term $\int e(t)\ dt$, and derivative term $\frac{de(t)}{dt}$.

Output Targets: Optimal PID parameters $(K_p), (K_i), (K_d)$.

By synthesizing the dataset, we can simulate a wide range of realistic operating conditions, ensuring that the models are exposed to diverse scenarios during training. This approach enables the development of adaptive controllers capable of handling dynamic and uncertain environments and to ensure the dataset is representative of real-world applications, operating conditions are randomly sampled within predefined ranges. These conditions include:

1. **Initial Speed ω_0:**
   - Randomly sampled within a range (500 – 3000 rpm).
   - Represents the motor's starting speed before control is applied.
2. **Set speed ω_{set}:**
   - Randomly sampled within a range (0 – 3000 rpm).
   - Represents the desired speed that the controller aims to achieve.
3. **Load Torque T_L:**
   - Randomly sampled within a range (0 – 0.5 N·m).
   - Represents external forces opposing the motor's motion, such as friction or mechanical loads.

These random samples ensure that the dataset captures a variety of scenarios, including low-speed, high-speed, no-load, and loaded conditions. The diversity of these operating conditions enhances the generalization capability of the trained models.

3.7 Optimization of PID Parameters Using Genetic Algorithm (GA)

For each set of operating conditions, the optimal PID parameters (K_p, K_I, K_D) are determined using a Genetic Algorithm (GA). GA is a heuristic optimization technique inspired by the process of natural selection. It mimics biological evolution by iteratively evolving a population of candidate solutions through processes such as selection, crossover [1], and mutation. In this context, each individual represents a set of PID parameters (K_p, K_I, K_D), and the fitness function evaluates the control performance based on metrics such as rise time, overshoot, and settling time. The process involves the following steps [13]:

1. **Encoding of Candidate Solutions:**
   Each candidate solution represents a set of PID parameters (K_p, K_I, K_D), encoded as a chromosome. For example, a chromosome might be represented as:
   $$\text{Chromosome} = [K_p, K_i, K_d]\#(13)$$
2. **Initialization of Population:**
   A population of candidate solutions is initialized randomly within predefined bounds:
   $$K_p \in [K_{p,\min}, K_{p,\max}], \quad K_i \in [K_{i,\min}, K_{i,\max}], \quad K_d \in [K_{d,\min}, K_{d,\max}]\#(14)$$
   In this context the bound are chosen between (0, 10) for (K_p, K_I, K_D).
3. **Fitness Function:**
   The fitness of each candidate solution is evaluated using an objective function that minimizes a weighted combination of performance metrics. Common metrics include rise time, settling time, overshoot, and steady-state error. For example:
   $$\text{Objective Function} = w_1 \cdot \text{Rise Time} + w_2 \cdot \text{Settling Time} + w_3 \cdot |\text{Overshoot}| + w_4 \cdot |\text{Steady-State Error}|\#(15)$$
   Where w_1, w_2, w_3, w_4 are weighting factors that prioritize specific metrics.
4. **Selection:**
   Individuals with higher fitness scores (better performance) are selected for reproduction. Techniques such as roulette-wheel selection or tournament selection are commonly used.
5. **Crossover:**
   Pairs of selected individuals exchange genetic material to produce offspring. For example, single-point or two-point crossover can be applied to combine portions of two parent chromosomes.
6. **Mutation:**
   Random changes are introduced into the offspring to maintain genetic diversity and prevent premature convergence. Mutation ensures that the algorithm explores new regions of the search space.
7. **Termination Criteria:**
   The algorithm terminates when a predefined number of generations is reached or when the fitness score converges to an acceptable level.

The output of the GA is the set of optimal PID parameters (K_p, K_I, K_D), for each set of operating conditions. These parameters are then paired with the corresponding dynamic features to form input-output pairs.

3.8 Construction of the Dataset

The synthesized dataset is constructed by pairing the extracted dynamic features (error, integral term, derivative term) with the optimized PID parameters (K_p, K_I, K_D), Since these features are time-series data, they are repeated for each time step during the simulation. The dataset is structured as follows:

1. Inputs (Features):
   - Error e(t)
   - Integral Term ($\int e(t)\ dt$)
   - Derivative Term ($\frac{de(t)}{dt}$)
2. Outputs (Targets):
   - Optimal PID parameters (K_p, K_I, K_D).

The dataset is normalized to improve the performance of machine learning models using standardization technique that scales features to have zero mean and unit variance as follows:

$$x_{\text{normalized}} = \frac{x - \mu}{\sigma} \#(16)$$

Where (μ) and (σ) are the mean and standard deviation of the feature.
Finally, the dataset is split into training and testing sets to evaluate the model's generalization performance. A common split ratio is 80% for training and 20% for testing.

3.9 MLP Model Design

The MLP architecture consists of multiple fully connected layers, each followed by an activation function. For this study, the following configuration was adopted:

1. **Input Layer:** Accepts three features which are error $(e(t))$, integral term ($\int e(t)\ dt$) and derivative term ($\frac{de(t)}{dt}$).
2. **Hidden Layer:** Two dense layers with 64 and 32 neurons, respectively. The Rectified Linear Unit (ReLU) activation function is applied to introduce nonlinearity.
3. **Output Layer:** Produces three outputs (K_p, K_I, K_D) and using the softplus activation function to ensure non-negative values.

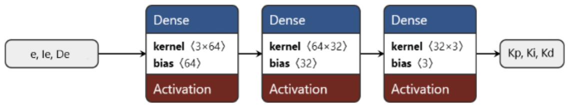

Fig 4. MLP Architecture

3.10 Training Process

The MLP is trained using the synthesized dataset described in Section 3.8. Key aspects of the training process include using Mean Squared Error (MSE) as a Loss function to measure the difference between predicted and actual PID parameters [14], as for the optimizer, Adam optimizer is employed for efficient gradient-based updates [15].
A batch size of 32 is selected to balance computational efficiency and convergence speed and the model is trained for 200 epochs, with early stopping implemented to prevent overfitting.

4. Simulation Results

The simulation is based on a high-speed DC motor with parameters carefully selected to represent a small industrial motor suitable for dynamic applications. The key specifications of the motor include an armature resistance $(R_a) of\, (0.027\, \Omega)$, armature inductance $(L_a)\ of\ 0.002\, H$, back EMF constant K_e and torque constant (K_t) of $0.0382\, V \cdot s/rad$, $and\ 0.0382\, N \cdot m/A$ respectively, a moment of inertia $(J)\ of\ 5 \times 10^{-5}\, kg \cdot m^2$, and a viscous friction coefficient $(b)\ of\ 5 \times 10^{-3}\, N \cdot m \cdot s/rad$. These parameters define the motor's electrical and mechanical behavior during operation, The simulation time step (dt) was set to $(1 \times 10^{-4}\, s)$, with a maximum simulation duration of $(1.5\ s)$ PID parameters were updated at regular intervals $(0.01\ \ s)$ to ensure real-time adaptability.
assess the robustness and adaptability of the control strategies, three distinct trailer scenarios were simulated, each with varying target speeds, initial speeds, and load torques. The results are presented in tabular form for each trial, followed by step response figures that illustrate the performance of MLP model.

1. **Trail 1 (Low Speed Operation):** initial speed is set to 500 RPM, Target Speed of 1500 RPM and Load Torque of 0.2 N.m.

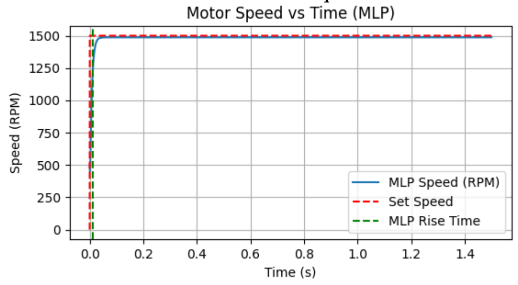

Fig. 5 Step Response of MLP model for Trial 1 (Low-Speed Operation)

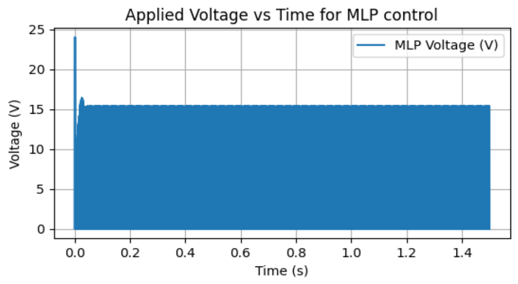

Fig. 6 Voltage output of MLP model for trail 1

2. **Trail 2 (High Speed Operation):** initial speed is set to 1000 RPM, Target Speed of 2000 RPM and Load Torque of 0.05 N.m.

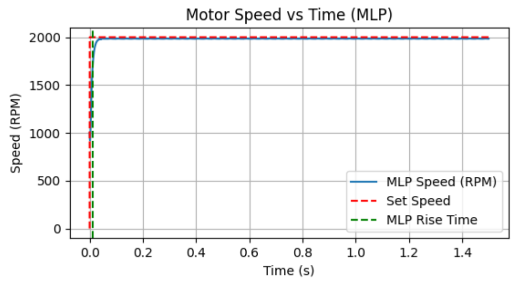

Fig. 7 Step Response of MLP model for Trial 2 (High-Speed Operation)

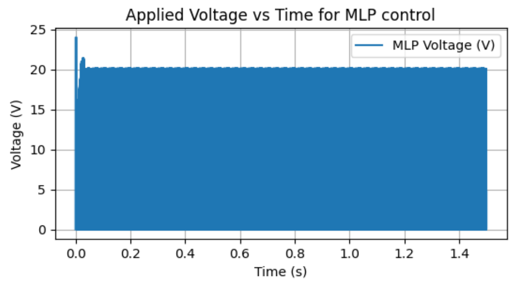

Fig. 8 Voltage output of MLP model for trail 2

3. **Trail 3 (Variable Speed Operation):** initial speed is set to 500 RPM, Target Speed start from 1500 *RPM* with periodic increment of 500 RPM until the set speed reaches 2500 RPM for every 1 second, then decreases speed by 400 RPM for every 1 second also, until it reaches a set speed of 1700 RPM, the entire simulation duration is 5 seconds and finally a variable load torque which is randomly set with every change in set speed between 0.05 and 0.5 N.m.

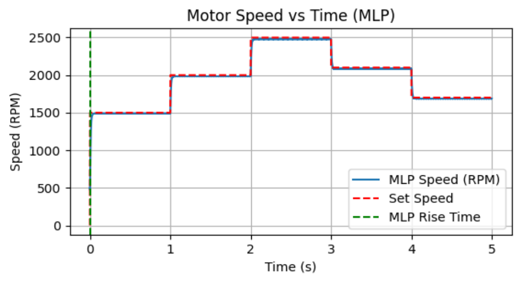

Fig. 9 Step Response of MLP model for Trial 3 (Variable-Speed Variable-Torque Operation)

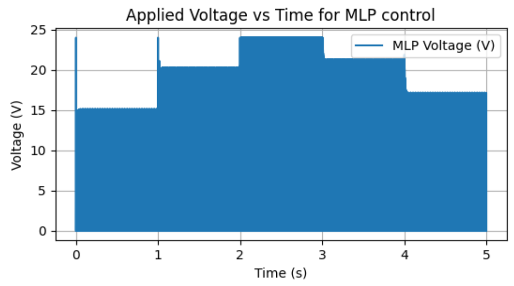

Fig.10 Voltage output of MLP model for trail 3

Table 1 Performance metrics for Trail 1 to 3

MLP Metric	Trail 1	Trail 2	Trail 1
Rise Time (s):	0.0122	0.0103	0.0072
Settling Time (s):	0.0296	0.0283	0.0198
Down Time (s):	inf	inf	0.0231
Overshoot (%):	-0.7591	-0.6971	-0.7583
Steady-State Error:	11.5865	15.2061	15.1447
Peak Time (s):	0.3496	0.4395	0.2397
Damping Ratio:	0.8409	0.8451	0.841
Natural Frequency (rad/s):	16.6032	13.3704	25.2575

5. Conclusion

This study evaluated artificial neural network strategies for DC motor speed control, specifically focusing on the Multilayer Perceptron (MLP) as a predictor for PID parameters based on simulation results and discussion. Simulation analysis indicated that MLP demonstrated rapid response characteristics, exhibiting the shortest rise times and competitive settling times compared to other methods. MLP also proved to be the most computationally efficient among the strategies investigated, making it suitable for systems with limited resources. However, its reliance on static input features meant that MLP struggled to match the desired adaptability and robustness in dynamic environments, and it showed higher steady-state errors in simulation trials. Consequently, MLP is best suited for resource-constrained systems or applications operating under relatively stable conditions, where computational efficiency and rapid initial response are priorities and significant dynamic variations are minimal.

References

[1] K. J. Åström and B. Wittenmark, Adaptive Control, 2nd ed. Reading, MA: Addison-Wesley, 1995, pp. 1-20.

[2] N. Mohan, T. M. Undeland, and W. P. Robbins, Power Electronics: Converters, Applications, and Design, 3rd ed. Hoboken, NJ: Wiley, 2003, ch. 8.

[3] P. C. Sen, Principles of Electric Machines and Power Electronics, 3rd ed. Hoboken, NJ: Wiley, 2013, pp. 45-60.

[4] R. C. Dorf and R. H. Bishop, Modern Control Systems, 13th ed. Upper Saddle River, NJ: Pearson, 2017, ch. 5.

[5] J. J. E. Slotine and W. Li, Applied Nonlinear Control. Englewood Cliffs, NJ: Prentice Hall, 1991, pp. 123-145.

[6] K. S. Narendra and A. M. Annaswamy, Stable Adaptive Systems. Englewood Cliffs, NJ: Prentice Hall, 1989, ch. 3.

[7] B. K. Bose, “Neural network applications in power electronics and motor drives—An introduction and perspective,” IEEE Trans. Ind. Electron., vol. 54, no. 1, pp. 14-33, Feb. 2007.

[8] M. Azri and B. A. Mutalib, “Speed Control of Dc Motor Using Pi Controller Mohd Azri Bin Abd Mutalib,” 2008. Accessed: Mar. 15, 2025. [Online]. Available: https://www.semanticscholar.org/paper/Speed-Control-of-Dc-Motor-Using-Pi-Controller-Mohd-Azri-Mutalib/223931aa76bdd865a54ae28ccc8f54af28c5ce8c

[9] Tan Kiong Howe, “Evaluation of the Transient Response of a DC motor using MATLAB/SIMULINK,” May 2003.

[10] Muhammed Nasiruddin Bin Mahyddin, “Direct model reference adaptive control of coupled tank liquid level control system,” Nov. 2005.

[11] K. J. Åström, T. Hägglund, and K. J. Åström, PID controllers, 2nd ed. Research Triangle Park, N.C: International Society for Measurement and Control, 1995.

[12] A. Meyer-Baese and V. J. Schmid, Pattern Recognition and Signal Analysis in Medical Imaging. Academic Press, 2016.

[13] T. A. Faris Ku Yusoff, M. F. Atan, N. Abdul Rahman, S. F. Salleh, and N. A. Wahab, “Optimization of PID Tuning Using Genetic

Algorithm," J. Appl. Sci. Process Eng., vol. 2, no. 2, Jan. 1970, doi: 10.33736/jaspe.168.2015.

[14] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. in Adaptive computation and machine learning. Cambridge, Massachusetts: The MIT Press, 2016.

[15] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," 2014, arXiv. doi: 10.48550/ARXIV.1412.6980.