PID TUNING OF FOPDT SYSTEM USING MULTIVARIATE LINEAR REGRESSION WITH GRADIENT DESCENT

Thaker Maharsh Kalpeshbhai
ME Student - Applied Instrumentation
Instrumentation and Control Department
L. D. College of Engineering, Ahmedabad, Gujarat, India

Patel Vinod Purushottamdas
Associate Professor
Instrumentation and Control Department
L. D. College of Engineering, Ahmedabad, Gujarat, India

Abstract—PID controllers are popular in industries for controlling process of a system or a plant. There are many different tuning techniques available to tune a PID controller. In this paper machine learning and optimization approach is used to tune the controller. Multiple Linear Regression algorithm of Machine Learning and Gradient Descent based optimization algorithm is used to obtain PID parameters. PID tuning is performed on a PID controller of Blending Control System. This paper presents a method of obtaining FOPDT model from system’s transient specifications, generating data-set of FOPDT model and then applying Multivariate Linear Regression with Gradient Descent to obtain optimal tuned values of $K_p$, $K_i$ and $K_d$ parameters of PID controller.

Keywords—Multivariate Linear Regression, Gradient Descent, PID, FOPDT, Blending Control, Data-set Generation, Machine Learning, Optimization

I. INTRODUCTION

After the advent of PID controllers in the early 20th century, control theory has developed to a great extent. From the age of Pneumatic Controllers to this age of Artificial Intelligence and Automation, PID controllers have constantly evolved and still holds the most prominent place in Industrial Control. Similarly PID tuning techniques have also adopted new technologies to simplify the tuning process and obtain better control results. Starting from classical tuning approaches like Ziegler-Nicolas tuning[1], Cohen-Coon Tuning[2] to modern tuning techniques like IMC tuning, Lambda Tuning[3] and Meta-heuristic tuning techniques like Genetic Algorithm[4], Particle Swarm Optimization[5], TLBO[6], etc are different tuning techniques suggested by different researchers. Current trend in PID algorithm is towards Machine Learning, Deep Learning and Fuzzy Logic[7] implementation on PID controller. Machine Learning approach consists of different supervised and unsupervised techniques like Regression,
Input to controller is error signal and controller output is given to system input. The mathematical form of PID controller is given by this equation:

\[ u(t) = K_p e(t) + K_i \int_0^t e(t) \, dt + K_d \frac{de(t)}{dt} \quad (1) \]

### III. DEVELOPMENT OF FOPDT MODEL

A mixer tank has one inlet stream and one outlet stream. No reaction terms are considered for this application. It is assumed that mixing in the tank is perfect, so we can assume the concentration and temperature in the tank to be constant. The volume of the mixer tank is also assumed to be constant. By species balance we can derive the following equation of change in outlet concentration per unit time:

\[ \frac{dC_a}{dt} = \frac{q}{V} (C_{af} - C_a) \quad (2) \]

Where \( C_a \) is outlet concentration, \( C_{af} \) is feed concentration (inlet concentration). \( q \) is volumetric flow rate and \( V \) is the volume of tank. Units of all specifications are described in Table 1 [12]. The figure (2) below shows step response of mixing tank. Giving a step input to the Blender we obtain the as response shown in Figure 2. Table 2 shows the data of Time and Concentration values after providing step input to the system.

#### Table -1 Blender Specifications

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Volumetric Flow-rate</td>
<td>100 m3/hr.</td>
</tr>
<tr>
<td>2.</td>
<td>CSTR Volume</td>
<td>100 m3</td>
</tr>
<tr>
<td>3.</td>
<td>Feed Temperature</td>
<td>300 K</td>
</tr>
<tr>
<td>4.</td>
<td>Feed Concentration</td>
<td>1 mole/L</td>
</tr>
</tbody>
</table>

Using Smith’s Method of identification of FOPDT model, the dead time (L) value of -0.333 and tau value of 0.222 is obtained. FOPDT identification is obtained by finding 63.2% and 28.3% values of time constants \( t_1 \) and \( t_2 \) respectively. Process dead time (L) and process time constant (tau) values are given by equations (3) and (4).

\[ L = t_2 - t_1 \quad (3) \]
\[ \tau = 1.5 \left( t_2 - t_1 \right) \quad (4) \]

After performing step test we obtain the transfer function of FOPDT model shown in equation (5).

\[ H = e^{-0.333s} \frac{1}{0.222s+1} \quad (5) \]

Other popular FOPDT identification method is Sundaresan and Krishnaswamy [13] method to find values of dead time and tau. For this study Smith’s FOPDT method is considered. As compared to Nishikawa’s method [14] of FOPDT identification, Smith’s method [15] and Sundaresan-Krishnaswamy method has lower computational cost.

#### Table -2 Time versus Concentration Data

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Time</th>
<th>Concentration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>0.101</td>
<td>0.0960</td>
</tr>
<tr>
<td>2.</td>
<td>0.202</td>
<td>0.1829</td>
</tr>
<tr>
<td>3.</td>
<td>0.303</td>
<td>0.2614</td>
</tr>
<tr>
<td>4.</td>
<td>0.404</td>
<td>0.3323</td>
</tr>
</tbody>
</table>

### IV. MULTIVARIATE REGRESSION WITH GRADIENT DESCENT

The Linear Regression is a mathematical approach which is based on modeling a system using linear relationship between input and output variables. Multivariate Linear Regression contains one output variable and multiple input variables. Linear Regression is a type of supervised learning technique. For this technique to function properly there
should be linear relationship between input and output variable.

First finding closed loop transfer function. On this transfer function we apply Linear Regression that relates three input variables with one output variable. Transient specifications of system like Rise Time, Settling Time and Overshoot are used as input variables. Mathematical equation of Linear Regression is given by (6).

\[ y = \sum_{n=0}^{j} \theta_{(j)} X_{(j)} \]  

(6)

\[ \theta_{(j)} \] are weights associated with input variables \( X_{(j)} \), \( j \) is the number of input variables. Gradient Descent algorithm minimizes the error between the predicted output variable and the actual output variable [6]. The error can be calculated using cost function given by equation (7).

\[ J(\theta) = \frac{1}{2m} \sum_{j=0}^{m} (h(\theta)X_{(j)} - y_{(j)})^2 \]  

(7)

Where \( m \) is the number of iterations. After performing the partial derivation of cost function we get the update rule as given by equation (8).

\[ \theta_{(i)} = \theta_{(i)} - \alpha \frac{1}{m} \sum_{j=0}^{m} (h(\theta)X_{(j)} - y_{(j)})X_{(j)}^{i} \]  

(8)

Where \( \alpha \) (\( \alpha \)) is the Learning Rate. Learning rate being small, algorithm convergence will be slow and if the learning rate is very high then algorithm may miss the optimal point by converging faster than expected.

V. PROBLEM FORMULATION

Transfer function of closed loop system is created using FOPDT system and controller. This closed loop system comprises of controller, system (FOPDT) and feedback element in general. For training the model one of the parameter \( K_p \), \( K_i \) and \( K_d \) is varied in a certain range and other two parameters are kept constant. For example, we vary \( K_d \) from 0.1 to 10 at steps of 0.1, keeping value of \( K_p \) at 100 and value of \( K_i \) at 7. Output is noted after giving a step input. For this study \( K_p \), \( K_i \) and \( K_d \) are taken as output variables and Rise Time, Settling Time and Overshoot as considered as input variables. Transient parameters of rise time (\( t_{rise} \)), settling time (\( t_{set} \)) and overshoot (\( O_v \)) are obtained by using ‘stepinfo’ function of MATLAB. Similarly \( K_i \) is varied, keeping \( K_p \) and \( K_d \) constant. Then \( K_p \) is varied, keeping \( K_i \) and \( K_d \) constant. We obtain the optimal PID parameters by the given equations (9), (10) and (11) respectively.

\[ K_p = t_{rise} \theta_1 + t_{set} \theta_2 + O_v \theta_3 \]  

(9)

\[ K_i = t_{rise} \theta_4 + t_{set} \theta_5 + O_v \theta_6 \]  

(10)

\[ K_d = t_{rise} \theta_7 + t_{set} \theta_8 + O_v \theta_9 \]  

(11)

Theta are the optimal weights associated to the input variables. Other transient input variables like Peak and Peak time can be added to work with five input variables.

VI. SIMULATION

Sample Dataset for \( K_p \): Table 2 shows sample data-set for \( K_p \) obtained by keeping \( K_i \) and \( K_d \) constant at 2.67 and 0.13 respectively. \( K_p \) is varied from 0 to 3.25 with the steps of 0.01 and transient response is recorded in a ‘.csv’ file.

<table>
<thead>
<tr>
<th>( K_p )</th>
<th>Rise Time</th>
<th>Settling Time</th>
<th>Overshoot</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.662</td>
<td>5.72</td>
<td>35.99</td>
</tr>
<tr>
<td>0.01</td>
<td>0.66</td>
<td>5.69</td>
<td>35.4</td>
</tr>
<tr>
<td>0.02</td>
<td>0.659</td>
<td>5.67</td>
<td>34.81</td>
</tr>
<tr>
<td>0.03</td>
<td>0.657</td>
<td>5.63</td>
<td>34.23</td>
</tr>
</tbody>
</table>

Sample Dataset for \( K_i \): Table 4 shows sample data-set for \( K_i \) obtained by keeping \( K_p \) and \( K_d \) constant at 1.62 and 0.13 respectively. \( K_i \) is varied from 0 to 5 with the steps of 0.05 and transient response is recorded in a ‘.csv’ file.

<table>
<thead>
<tr>
<th>( K_i )</th>
<th>Rise Time</th>
<th>Settling Time</th>
<th>Overshoot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1719</td>
<td>6.55</td>
<td>0</td>
</tr>
<tr>
<td>1.05</td>
<td>0.1712</td>
<td>6.20</td>
<td>0</td>
</tr>
<tr>
<td>1.1</td>
<td>0.1691</td>
<td>5.87</td>
<td>0</td>
</tr>
<tr>
<td>1.15</td>
<td>0.1685</td>
<td>5.58</td>
<td>0</td>
</tr>
</tbody>
</table>

Sample dataset for \( K_d \): Table 5 shows sample data-set for \( K_d \) obtained by keeping \( K_i \) and \( K_p \) constant at 2.49 and 1.26 respectively. \( K_d \) is varied from 0 to 0.25 with the steps of 0.001 and transient response is recorded in a ‘.csv’ file.

<table>
<thead>
<tr>
<th>( K_d )</th>
<th>Rise Time</th>
<th>Settling Time</th>
<th>Overshoot</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.1844</td>
<td>1.93</td>
<td>19.30</td>
</tr>
</tbody>
</table>

186
After generating the data-sets for Kp, Ki and Kd, model training is performed using multivariate linear regression with gradient descent algorithm. The result of the training is shown in Table 6 and compared with the results of MATLAB Auto-Tuner.

Comparing the PID response obtained by linear regression with gradient descent algorithm versus the response obtained by MATLAB Auto-Tuner we find that linear regression with gradient descent works better for this application. PID response obtained by linear regression with gradient descent has less overshoot as compared to that of the response obtained by MATLAB Auto-Tuner. Figure 4 shows the response of system to PID controller parameters tuned by linear regression with gradient descent.

Convergence curve of gradient descent algorithm shows that algorithm correctly converged after 2000 iterations. Figure 3 shows the convergence curve of gradient descent. The learning rate affects the convergence of the curve. Table 7 shows the effect of hyper-parameter (learning rate and number of iterations) changes on the $K_p$ value.

If we change number of input variable it does not affect much on the PID tuning performance. Till now we were using three input parameters now after adding two more input parameters (peak and peak time) the result obtained is almost same as with three input parameters.

<table>
<thead>
<tr>
<th>Alpha Value</th>
<th>$K_p$ at 500 Iterations</th>
<th>$K_p$ at 3000 Iterations</th>
<th>$K_p$ at 10000 Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>1.614</td>
<td>1.624</td>
<td>1.625</td>
</tr>
<tr>
<td>0.005</td>
<td>1.492</td>
<td>1.624</td>
<td>1.625</td>
</tr>
<tr>
<td>0.001</td>
<td>0.639</td>
<td>1.54</td>
<td>1.624</td>
</tr>
<tr>
<td>0.0001</td>
<td>0.079</td>
<td>0.421</td>
<td>1.072</td>
</tr>
</tbody>
</table>

From the data of learning rate (Alpha) and number of iterations of gradient descent it is clear that small values of learning rate requires more number of iterations to achieve proper convergence. If the value of alpha is large and number of iterations are less than algorithm training will result in incomplete convergence and tuning will not take place as expected.

Figure 4 shows how the PID controller response changes with change in number of iterations. For this response the value of learning rate selected is 0.005. Vale of n equal to 500 and Alpha equal to 0.005 gives less overshoot as compared to n = 3000.
with $\alpha = 0.005$ and $n = 10000$ with $\alpha = 0.001$. Using optimal value to alpha and number of iterations we can perform fine tuning of PID controller.

![Step Response](image)

Fig. 5. PID Controller Response on Variation in Iterations

VII. CONCLUSION

Very less research work is done till now on tuning PID parameters with linear regression techniques so there is vast scope of research in the field of PID controller tuning. This study shows the result of PID tuning of FOPDT system using multivariate linear regression with gradient descent algorithm, this tuning technique can be applied to variety of systems including first order, second order and FOPDT systems. Selection of optimal hyper-parameters results in fine tuning of the model. For tuning of Non-Linear systems using the technique presented in the paper, we can linearize the system at any one parameter and perform training of the model. Feasibility check of PID tuning of Non-Linear system can also be performed.

VIII. REFERENCE


