2019 IEEE 4th Conference on Technology In Electrical and Computer Engineering (ETECH 2019)
Information and Communication Technology (ICT) Tehran, Iran

Calibration Optimization of DC Motor Parameters Using Nonlinear Optimization Techniques under Uncertainty

Hamed Keshmiri Neghab
Faculty of control engineering
Ferdowsi University of Mashhad
Mashhad, Iran
Keshmirineghab.hamed@mail.um.ac.ir
Tel Number: +989125020161

Naser Pariz
Faculty of control engineering
Ferdowsi University of Mashhad
Mashhad, Iran
n-pariz@um.ac.ir
Tel Number: +989155159162

***Abstract*- In this paper we demonstrate how to estimate the parameters of a Nonlinear DC Motor using different Nonlinear Optimization techniques of fitting parameters to model, that called model calibration. Then we identification unknown parameters of the mathematical model with the nonlinear optimization techniques for the fitting routines and model calibration process.**

Keywords— Calibration; Levenberg-Marquardt; Constrained Nonlinear Optimization; Gauss-Newton Optimization; Uncertainty.

I. INTRODUCTION

This study proposes a new parameter calibration method based on optimization techniques to improve the estimation and optimization accuracy of DC motor parameters. This paper demonstrates how to estimate the parameters of a Nonlinear DC motor using different techniques of fitting parameters to model, here called model calibration.

First we specify parameter range and verify maximum and minimum ranges. Also we verify the initial conditions for the fitting routines. Then we identification unknown parameters of the mathematical model with the different nonlinear optimization techniques for the fitting routines and model calibration process. Then we choose the techniques for model calibration. Also, after choosing the technique, notice that some parameters are specific for the technique used. Moreover, we can change the parameters directly by modifying the parameters control (1). When model calibration process is done, we can accept the parameters to transfer the optimum values as initial conditions.

To start we created a doublet for excitation. Also all measured responses is under uncertainty. The electric circuit of the armature and diagram of the rotor are shown in the following figure:

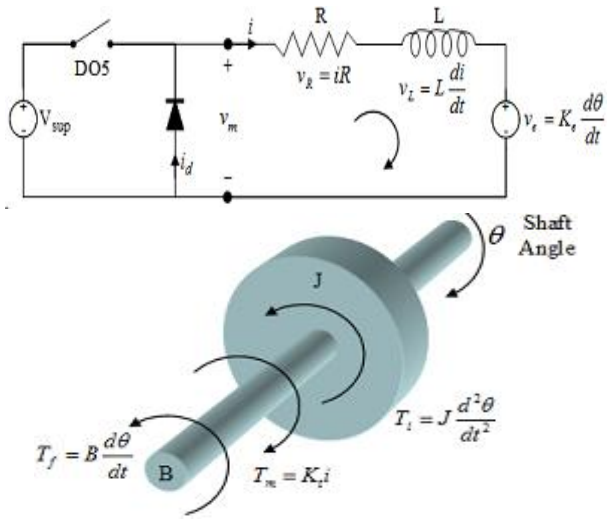


Figure 1: The electric circuit of the armature and diagram of the rotor

DC motor variables are Electric Resistance(R), Inductance(L), Motor Constant(K_e), Armature Constant(K_t), Moment of inertia of the rotor (J) and Damping ratio of the mechanical system (B).

In this study source voltage(V) is Input and the signals of drive current(i), motor position(shaft angle(θ)) and motor velocity(Angular velocity($w = \dot{\theta}$)) are outputs and measured by sensors. Assume that the rotor and shaft are assumed to be rigid. For this study, we will assume the following values for the physical parameters.

Parameters	Real value
R	3 ohm
L	0.4 H
K_e	0.0025 Nm/Amp
K_t	0.0025 Nm/Amp
J	0.005 $kg.m^2/s^2$
B	0.1 Nms

Table 1: Real values physical parameter

A. System Equations

Mathematical model and the dynamic equations in state-space form are the following:

$$T = K_t i$$

$$e = K_e \dot{\theta}$$

The motor torque(T), is related to the armature current(i), by a armature constant factor(K_t). The back electromotive force(e), is related to the Angular velocity($\dot{\theta}$) by a motor constant factor(K_e). From the figure1 we can write the following equations based on Newton's law combined with Kirchhoff's law:

$$J\ddot{\theta} + b\dot{\theta} = Ki$$

$$L\frac{di}{dt} + Ri = V - K\dot{\theta}$$

Using Laplace Transforms the above equations can be expressed in terms of s.

$$s(Js + b)\theta(s) = KI(s)$$

$$(Ls + R)I(s) = V - Ks\theta(s)$$

$$\frac{\theta}{V} = \frac{K}{(Js + b)(Ls + R) + K^2}$$

The dynamic equations in state-space form are the following:

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{I} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & \frac{-b}{J} & \frac{K_t}{J} \\ 0 & \frac{-K_e}{L} & \frac{-R}{L} \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \\ I \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} V$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \\ I \end{bmatrix}$$

II. METHODS

In this section we describe nonlinear optimization techniques to estimate the parameters of a nonlinear equation of fitting parameters to model(3).

A. Constrained Nonlinear Optimization technique

This technique demonstrates fitting a parameterized simulation using the Constrained Nonlinear Optimization. We can solve a general nonlinear optimization problem with nonlinear equality constraint and nonlinear inequality constraint bounds using a sequential quadratic programming method. The connections to the Constrained Nonlinear Optimization technique are shown below:

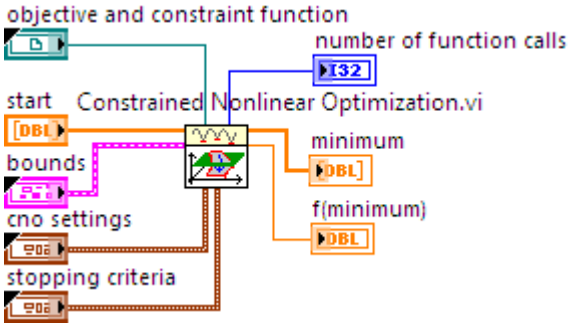


Figure 2: Constrained Nonlinear Optimization technique

- Objective and constraint function is a reference that implements the nonlinear function to minimize.
- Bounds is a cluster that contains the upper and lower numeric limits for the parameters being optimized and the inequality constraints. Minimum value contains the lowest allowed value of the parameters being optimized. Maximum value contains the highest allowed value of the parameters being optimized.
- Number of function calls is the number of times the objective function calls in the optimization process.
- Minimum is the determined local minimum in n dimension.
- f(minimum) is the function value of objective function at the determined minimum.

We calculate the cost function with evaluate position, velocity and current. Also we assume no equality or inequality constraints for this problem.

$$J = \|(\theta - \theta_r)\|w_1 + \|(\dot{\theta} - \dot{\theta}_r)\|w_2 + \|(i - i_r)\|w_3$$

Where θ_r denotes the desired reference and parameter w represent the weighting factor(w=position, velocity and current weight).

B. *Gauss-Newton Optimization Technique*

The purpose of this section is to parameter estimate a user-defined model. We can define a nonlinear model and then use the system identification to estimate user-defined model. This section summarizes the algorithms of the Gauss-Newton optimization method. Also we can use the Gauss-Newton optimization method to refine the estimation of user-defined models. We can estimate a user-defined model by refining the initial estimates using Gauss-Newton optimization routines. The connections to the Gauss-Newton Optimization function are shown below:

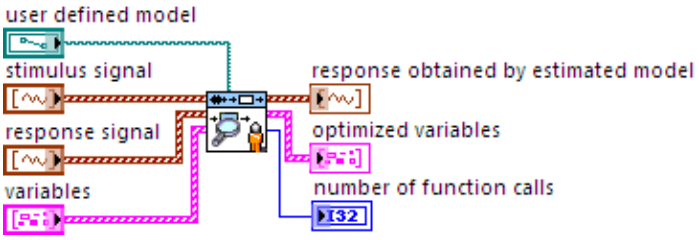


Figure 3: Gauss-Newton Optimization technique

The purpose of Gauss-Newton optimization is to minimize the cost function ($V_N(\theta)$). Also the purpose of model estimation is to identify the model coefficients by minimizing the mean square error ($V_N(\theta)$), which is defined by the following equation:

$$V_N(\theta) = \frac{1}{N}\sum_{k=1}^{N}\frac{1}{2}e^2(k,\theta)$$

$$e(k,\theta) = \hat{y}(k,\theta) - y(k)$$

Where θ is the variable vector, $\hat{y}(k,\theta)$ is the response the user-defined model calculates, and $y(k)$ is the measured response. $e(k,\theta)$ is the error, indicating the difference between the predicted or simulated output of the system($\hat{y}(k,\theta)$), and the measured output($y(k)$). We can use some methods, such as the multi-stage method for polynomial models, to get a coarse estimation of θ. We can use the following iteration to refine θ:

$$\theta^{(i+1)} = \theta^{(i)} + af(\theta^{(i)})$$

Where a is the step size and $f(\theta^{(i)})$ is the search direction. The purpose of iteration is to minimize the mean square error $V_N(\theta)$. The problem to solve lies in how to select and compute the search direction.

The algorithm Gauss-Newton minimization defines the search direction $f(\theta)$ as:

$$f(\theta) = -[V''(\theta)]^{-1}.V'(\theta) \quad (3)$$

$$V'(\theta) = -\frac{1}{N}\sum_{k=1}^{N}\psi(k,\theta).e(k,\theta) \quad (4)$$

$$V''(\theta) \approx \frac{1}{N}\sum_{k=1}^{N}\psi(k,\theta).\psi^T(k,\theta) \quad (5)$$

$$\psi(k,\theta) = \frac{dy(k,\theta)}{dk} \quad (6)$$

T denotes transposing. By inserting Equations 4 and 5 into 3, we obtain the following equation:

$$f(\theta) = \frac{\sum_{k=1}^{N}\psi(k,\theta)e(k,\theta)}{\sum_{k=1}^{N}\psi(k,\theta)\psi^T(k,\theta)}$$

$$= \frac{[\psi(1,\theta) \quad \psi(2,\theta) \quad \dots \quad \psi(N,\theta)]\begin{bmatrix}e(1,\theta)\\e(2,\theta)\\\dots\\e(N,\theta)\end{bmatrix}}{[\psi(1,\theta) \quad \psi(2,\theta) \quad \dots \quad \psi(N,\theta)]\begin{bmatrix}\psi^T(1,\theta)\\\psi^T(2,\theta)\\\dots\\\psi^T(N,\theta)\end{bmatrix}}$$

$$= \begin{bmatrix}\psi^T(1,\theta)\\\psi^T(2,\theta)\\\dots\\\psi^T(N,\theta)\end{bmatrix}^{-1}\begin{bmatrix}e(1,\theta)\\e(2,\theta)\\\dots\\e(N,\theta)\end{bmatrix}$$

we can evaluate $f(\theta)$ by solving the following linear equation:

$$\begin{bmatrix}\psi^T(1,\theta)\\\psi^T(2,\theta)\\\dots\\\psi^T(N,\theta)\end{bmatrix}f(\theta) = \begin{bmatrix}e(1,\theta)\\e(2,\theta)\\\dots\\e(N,\theta)\end{bmatrix}$$

The following equations define $\psi(k,\theta)$ to calculate the gradients.

$$\psi(k,\theta) = \frac{y(k,\theta+d\theta) - y(k,\theta-d\theta)}{2.d\theta}$$

$$d\theta = \theta * DIFF\_2$$

$$where:\ DIFF\_2 = 3.2057501263E-6$$

C. *Nonlinear Levenberg-Marquardt Fit Theory*

The Nonlinear Lev-Mar Fit Theory used to calculate the best fit parameters that minimize the weighted mean square error between the observations in curve and the best nonlinear fit. The Nonlinear Lev-Mar Fit Theory determines the set of coefficients $(a_1, a_2, \dots, a_M)$ that best fits the observations. The best fit coefficients minimize the following equation (chi-square quantity), which describes the distance between the curve and the fitted model.

$$\sum_{i=0}^{N-1} w_i \left(y_i - f(x_i; a_1, a_2, \dots, a_M)\right)^2$$

Where N is the length of Y and yi, xi and wi are the i-th element of Y, X and Weight, respectively. Where X is the independent variable and A = $\{a_1, a_2, \dots, a_M\}$ are unknown coefficients.

In this equation, (x_i, y_i) are the input data points, and $f(x_i; a_1, a_2, \dots, a_M) = f(X, A)$ is the nonlinear function. If the measurement errors are independent and normally distributed with constant standard deviation, this is also the least-square estimation.

However, when a nonlinear relationship exists, we can use the Nonlinear Lev-Mar Fit function to determine the coefficients. We can use the Lev-Mar method, which is very robust, to find the coefficients of the nonlinear relationship between A and $y[i]$.

As a preliminary step, we need to specify the nonlinear function $y = f(X, A)$ where the set of coefficients(A), is determined by the Lev-Mar algorithm. The connections to the Nonlinear Lev-Mar Fit technique are shown below:

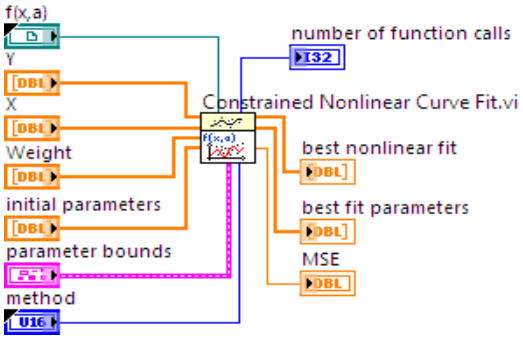


Figure 4: Nonlinear Lev-Mar Fit

- X and Y are the input data points $x[i]$ and $y[i]$.
- Best Fit Coefficients the values of the coefficients A that best fit the model of the experimental data.
- Number of function calls returns the number of times LabVIEW called f(x,a) during the fitting process.
- Initial parameters specifies the initial guess for a solution. The success of the nonlinear curve fit depends on how close the initial parameters are to the solution.

In mathematics and computing, the Lev–Mar algorithm, also known as the least-squares method, is used to solve nonlinear least squares problems. These minimization problems arise especially in least squares curve fitting. The following equation defines the curve model:

$$y[i] = f(x[i], a_0, a_1, a_2, \dots)$$

The purpose of curve fitting is to find a nonlinear function $f(X, A)$ for the data (xi, yi) where $i = 0, 1, 2, \dots, n-1$. The function $f(X, A)$ minimizes the residual (MSE) under the weight W. The residual is the distance between the data samples and $f(X, A)$. A smaller residual (MSE) means a better fit. In geometry, curve fitting is a curve $Y = f(X, A)$ that fits the data(xi, yi). In the Nonlinear Curve Fit assumes that we have prior knowledge of the nonlinear relationship between the independent variable x and dependent variable y.

The Lev-Mar algorithm accepts X and the set of coefficients $(a_1, a_2, \dots, a_M)$ as inputs and returns $f(X; a_1, a_2, \dots, a_M)$ and a 2D array of the partial derivatives with respect to the coefficients. The following equations describes the layout of the partial derivative matrix of a function with M coefficients and n X values.

$$f'(X, A) = \begin{bmatrix} \frac{\partial f(x_0, A)}{\partial a_1} & \frac{\partial f(x_0, A)}{\partial a_2} & \dots & \frac{\partial f(x_0, A)}{\partial a_M} \\ \frac{\partial f(x_1, A)}{\partial a_1} & \frac{\partial f(x_1, A)}{\partial a_2} & \dots & \frac{\partial f(x_1, A)}{\partial a_M} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f(x_{n-1}, A)}{\partial a_1} & \frac{\partial f(x_{n-1}, A)}{\partial a_2} & \dots & \frac{\partial f(x_{n-1}, A)}{\partial a_M} \end{bmatrix}$$

The number of columns(M) in the matrix is equal to the number of unknown coefficients, while the number of rows in the matrix is equal to the number of X values.

III. RESULT AND DISSCUTION

A. *Results of Levenberg-Marquardt Method*

Method	Least Square
Cost Function	2.08801
Number of function calls	91
MSE	8.82137E-5

Parameters	Identified Parameters	Error
R	4.98834	1.98834
L	0.499045	0.09900451
K_e	0.00234385	-0.00015615
K_t	0.00258264	8.26434E-5
J	0.00495403	-4.59719E-5
B	0.0996606	-0.00033971

Table 2: Results of Lev-Mar(Least Square) Method

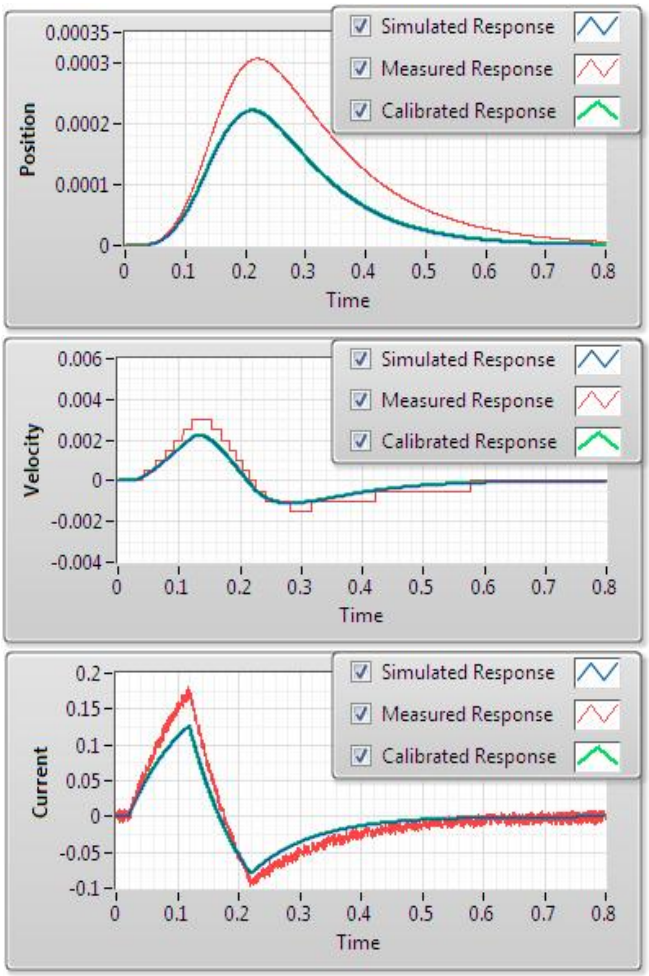


Figure 5: Results of Lev-Mar(Least Square) Method

Method	Bisquare
Cost Function	2.10319
Number of function calls	494
MSE	1.57821E-12

Parameters	Identified Parameters	Error
R	4.99999	1.99999
L	0.500114	0.100114
K_e	0.0024995	-4.98257E-7
K_t	0.00383366	0.00133366
J	0.00657493	0.00157493
B	0.0998273	-0.000173012

Table 3: Results of Lev-Mar(Bisquare) Method

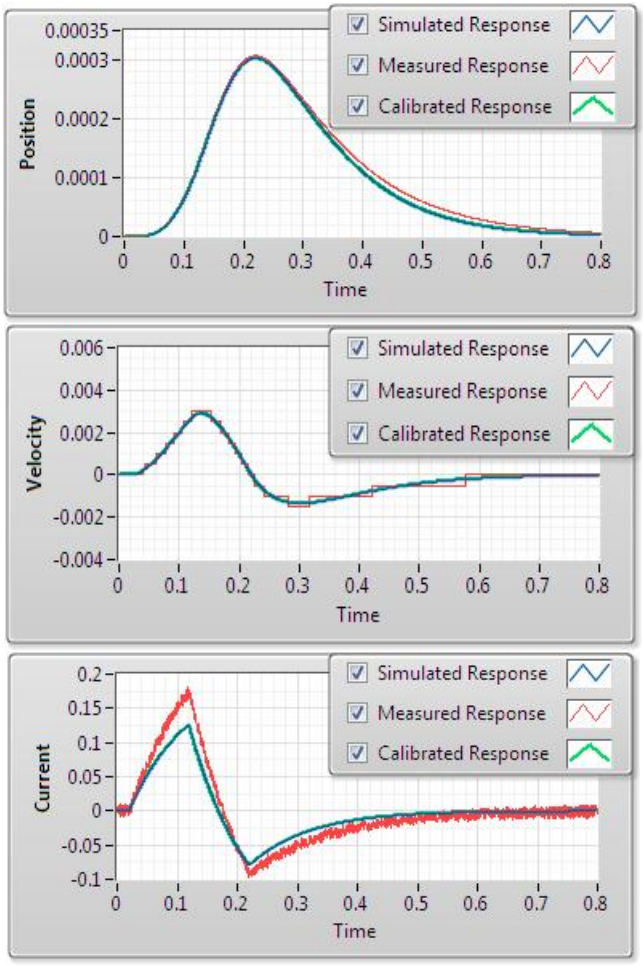


Figure 6: Results of Lev-Mar(Bisquare) Method

Method	Least Absolute Residual
Cost Function	0.0748517
Number of function calls	325
MSE	0.00136936

Parameters	Identified Parameters	Error
R	3.06112	0.0616486
L	0.40934	0.0094177
K_e	1E-5	-0.00249
K_t	0.00252899	2.81609E-5
J	0.00487799	-0.000129496
B	0.098828	-0.00113768

Table 4: Results of Lev-Mar(Least Absolute Residual) Method

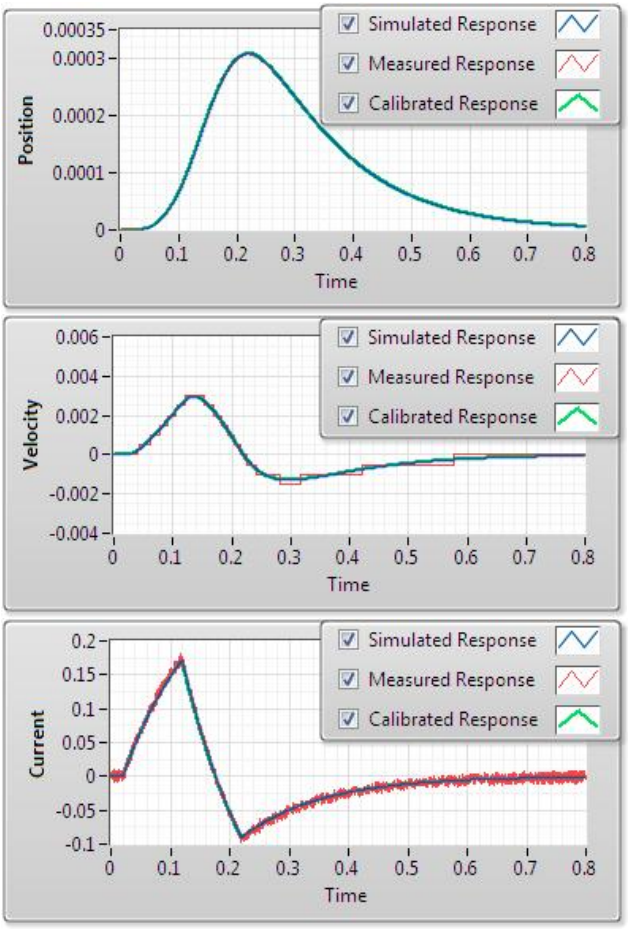


Figure 7: Results of Lev-Mar(Least Absolute Residual) Method

B. Results of Gauss-Newton Method

Method	Gauss-Newton
Cost Function	0.0561348
Number of function calls	67
MSE	0.0013293

Parameters	Identified Parameters	Error
R	2.95541	-0.0445906
L	0.403294	0.00329365
K_e	0.01	0.0075
K_t	0.00252173	2.17299E-5
J	0.00496237	-3.76258E-5
B	0.100691	0.000691178

Table 5: Results of Gauss-Newton Method

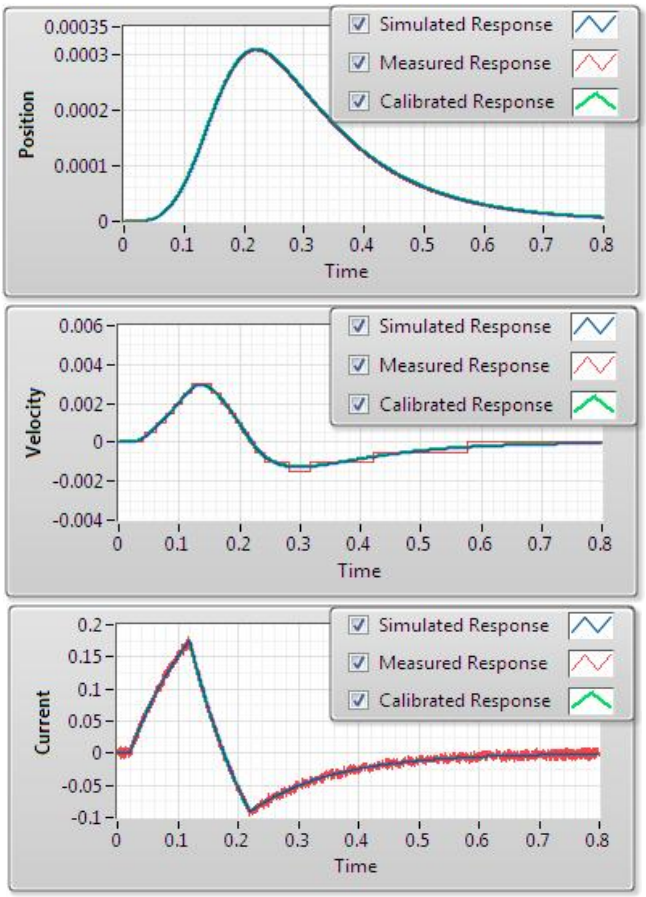


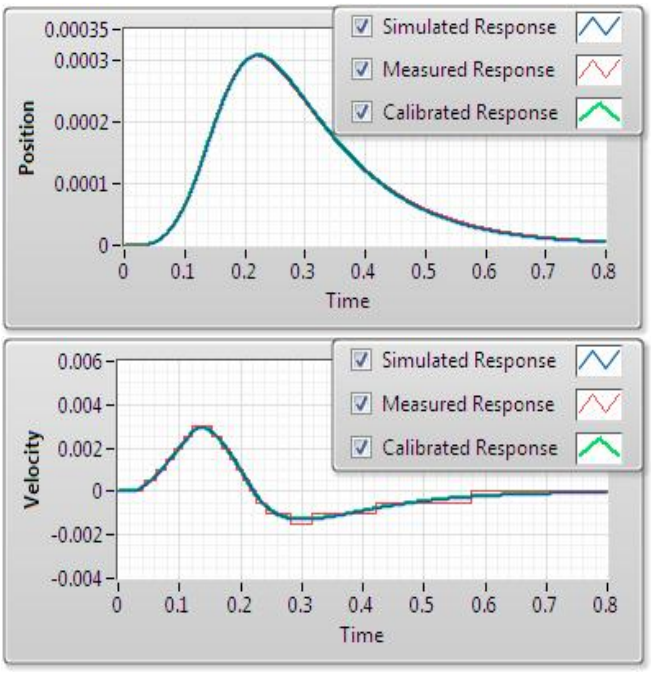
Figure 8: Results of Gauss-Newton Method

C. *Result Constrained Nonlinear Optimization Method*

Method	Constrained Nonlinear Optimization
Cost Function	0.0885244
Number of function calls	275
MSE	0.00111373

Parameters	Identified Parameters	Error
R	3.03953	0.0395296
L	0.397634	-0.00236579
K_e	0.0024586	-4.13982E-5
K_t	0.00361373	0.00111373
J	0.00735488	0.00235488
B	0.143119	0.043119

Table 6: Results of Constrained Nonlinear Optimization Method



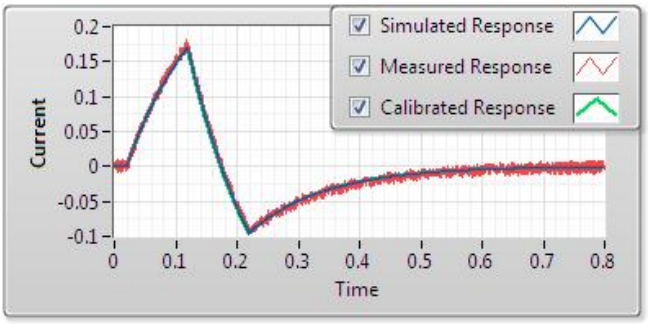


Figure 9: Results of Constrained Nonlinear Optimization Method

These results show that Lev-Mar(Least Square and Bisquare) methods are not appropriate due to low accuracy. But the Lev-Mar(Least Absolute Residual), Nonlinear Optimization and Gauss–Newton technique are remarkably accurate since they present smaller MSE means causing better fit. Furthermore, we can accept the parameters to transfer the optimal values as initial conditions.

I. CONCLUSION

This study presented The Lev-Mar, Nonlinear Optimization and Gauss–Newton algorithm for fitting parameters problems to model calibration. However, as with many fitting algorithms, the Lev-Mar finds only a local minimum, which is not necessarily the global minimum. The Lev-Mar is more robust than the Gauss–Newton algorithm, which means that in many cases it finds a solution even if it starts very far off the final minimum. For well-behaved functions and reasonable starting parameters, the Lev-Mar tends to be a bit slower than the Gauss–Newton algorithm.

REFRENCES

[1] X. Hu, R. Eberhart, "Solving Constrained Nonlinear Optimization Problems with Particle Swarm Optimization", 6th World Multiconference on Systemics, Cybernetics and Informatics, pp. 1-3, 2002.

[2] S. Leyffer and A. Mahajan, "Nonlinear Constrained Optimization: Methods and Software", pp. 1-24, 2010.

[3] M. Abdollahi, D. Abdollahi and A. Isazadeh, "Solving the Constrained Nonlinear Optimization based on Imperialist Competitive Algorithm", Proceedings,