UNIVERSITÀ DI TRENTO | Department of Industrial Engineering

DOCTORAL SCHOOL IN MATERIALS, MECHATRONICS AND SYSTEMS ENGINEERING

Numerical optimization

A MATLAB CLASS OF LEVEMBERG MARQUIRD ALGORITHM FOR DATA FITTING

06 march 2023

Abraham Ayele abrahamayele.gelan@unitn.it

1. Introduction

In mechatronics mathematical model play critical role to know the dynamic characteristics of a system. It is often necessary to measure the discrepancy b/n the model and the experimental data to know how well the model describe the physical system. List square data fitting help in obtaining parameters of candidate model by fitting experimental results. The resulting model with known parameter used to estimate unknow value that the experimental result does not address.

In this project I perform a data fitting to a model of piezo electric actuator of grabbing, positioning and release mechanism (GPRM) of LISA space craft with the data obtained on the first inflight test of the mechanism. The data fitting is used to obtain an unknown parameter of piezo electric model. This model along with other models of the mechanism is used to predict the inflight anomalies which were experienced on the first inflight release. I implement MATLAB class of one of the best algorithms in list square fitting technique called levemberg_marquerdt method.

2. The Levenberg Marquardt algorithm

One of the best algorithms to solve the least square minimization problem is Levenberg Marquardt algorithm [1]. It takes the benefits of two algorithms the steepest decent and Gauss-Newton methods. The steepest descent is best candidate when the solution(x*) is far from starting point (x) but it has poor final convergence. In contrary Gauss-Newton methods have quadratic final convergence. As presented [2], to incorporate the advantages of steepest and Gauss-Newton, the Levenberg Marquardt method use term μ as interchanging parameter to take advantage of these two algorithms. The algorithm is shown bellow as presented in [2].

$x = x_0$; $\mu = \mu_o$

repeat

solve $(J^T J + \mu I)\, h_{lm} = -F'$

$\mathbf{x}_{new} := \mathbf{x} + \boldsymbol{\alpha}\mathbf{h}$

until stop.

When x is too far from x*(the solution) , μ will be large so that the decent direction (μI) $h_{lm} = -F'$ is approximately the steepest descent. If x is close to the solution, μ will be small, and the resulting problem can be written as $(J^T J)\, h_{lm} = -F'$, which is the Gauss-Newton decent direction (**h)**. switching to Gauss-Newton helps to achieve quadratic convergence when x_{new} is near the solution.

3. **Levenberg Marquardt MATLAB class**

The algorithm used to build the MATLAB class is presented below, more elaboration can be found [3]. The levemberg_marq class takes initial value (x_0) which is used to start the iteration and an anonymous function of the list square problem (residual of the function) and its Jacobian. Experimental data is passed as an argument when we call the class. There are five functions in the class their description is given blow.

Levenberg–Marquardt algorithm

Given x0, τ, ε1, ε2, k_{max}
Begin
k: = 0; ν: = 2; x: = x_0
A: = $\mathbf{J(x)}^T \mathbf{J}(x)$; **g:** = $\mathbf{J(x)}^T \mathbf{r(x)}$
found: = ($\|\mathbf{g}\| \infty \leq \varepsilon 1$); μ: = τ ∗ max{$a_{ii}$}
while (**not** found) **and** (k < k_{max})
k: = k+1; Solve (**A** + μ**I**) $\mathbf{h}_{lm}$ = **-g**
if $\|\mathbf{h}_{lm}\| \leq \varepsilon_2(\|\mathbf{x}\| + \varepsilon_2)$
found: = **true**
else
$\mathbf{x}_{new}$:= **x** + $\mathbf{h}_{lm}$
ϱ: = δf/δL
if ϱ > 0
x: = x_{new}
A: = $\mathbf{J(x)}^T \mathbf{J(x)}$; **g**:= $\mathbf{J(x)}^T \mathbf{r(x)}$
found: = ($\|g\| \infty \leq \varepsilon 1$)
μ: = μ ∗ max {1/3, 1 – (2_{ϱ} - 1)^3}; ν: = 2
Else
μ: = μ ∗ ν; ν: = 2 ∗ ν
end
end
end
end

J is Jacobian of the residual function; **R** is residual function**;** $\mathbf{h}_{lm}$ is Levenberg Marquardt descent direction. ε1 = the tolerance of the current error in the iteration ε2 = the tolerance of the value of current state of the residual function.

We can find the value of δL, δf using the following two equations [3].

$$\delta L = 1/2\ h^T{}_{lm}\ (\mu\ h_{lm} - \nabla f))$$

$$\delta f = 1/2\ (r(x) - r(x_{new}))^T\ (r(x) + r(x_{new}))$$

The class include the following functions.

1. **solve_inc.m:** this function solves the initial values of the anonymous function, Jacobian, and meu. The resulting values are used in solve_levemberg.m function along with decent_solve.m to obtain the first iteration value of the update of x.
2. **iter_solver.m:** this function is called multiple times to solve iterative values of anonymous function and its Jacobian.
3. **decent_solve.m:** this function used to solve the decent direction (h) by solving $(J^T(x) J(x) + \mu I) h_{lm} = -J^T(x) r(x)$. Here thing to be noted is the result of μI should be positive semi definite matrix.
4. **solve_levemberg.m:** this solves the iterative value of updates (x_{new}). it also solves values of stopping criteria.
5. **save_result.m:** in this function result of all the update of x, last value of mue and stopping criterial is save in array.

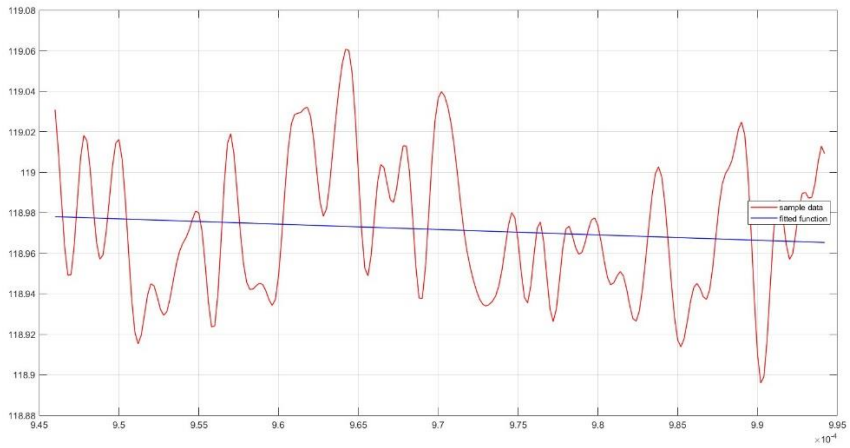

Figure 1: linear fit of the first part of piezoelectric voltage data.

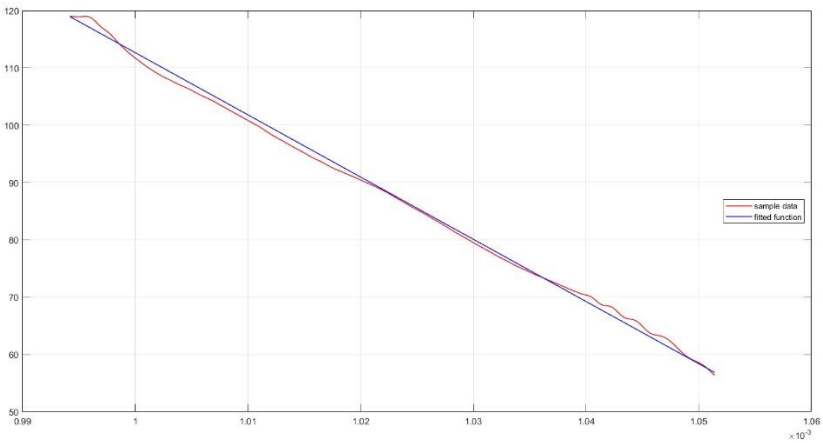

Figure 2: linear fit of the second part of the piezo electric voltage data

4. **Result**

I dived the fitting in to three parts , the first two are fitted to linear model as shown on figure 1 and 2. Finally, the end part of the data is fit to using nonlinear model shown on figure 3. On figure 4 the combination of all fitting is demonstrated. The equation of the fitting is shown below.

$at+b$	$t < t_1$
at_1-ct_1+b+ct	$t_1 < t < t2$
$e^{d(t-t_0)}$	$t2 < t$
$t_0 = t_2 - \ln(at_1-ct_1+b+ct_2)/d$	

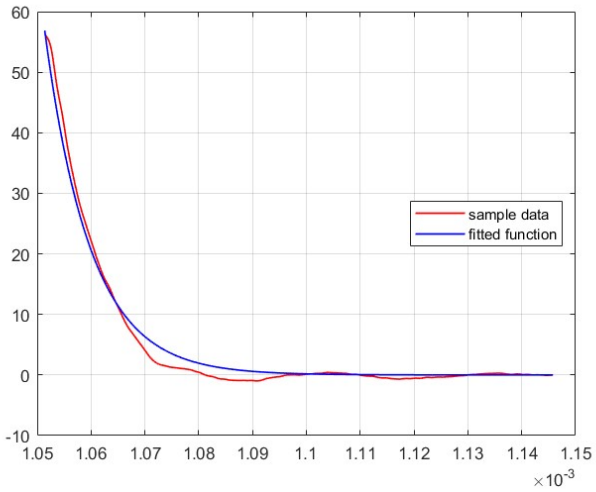

Figure 3: the nonlinear fit of third part of the piezo electric voltage data

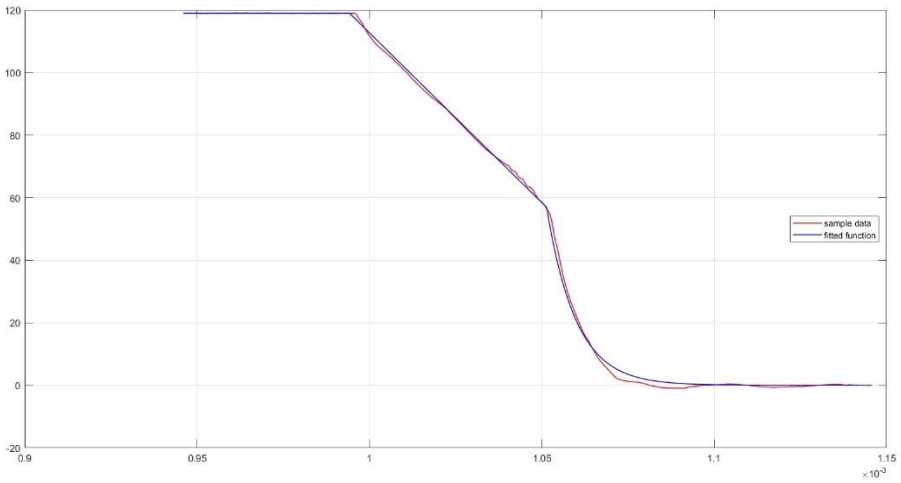

Figure 4: combined fitting of the voltage data.

Reference

1. D. Marquardt: An algorithm for least squares estimation on nonlinear parameters. SIAM J. Appl. Math.11, 431 - 441. 1963
2. Nielsen, Hans. (1999). damping parameter in Marquardt's method
3. Nielsen, Hans. (2010). Introduction to optimization and data fitting.