Calibration of Nano Composite Sensor using curve fitting techniques

Mahesh Eega   620071

Masters | Embedded systems

5th semester

Rajarajan Ramalingame

Prof. Olfa Kanoun

Agenda

- Introduction
- Theoretical background
- Model generation
- Optimization of Parameters
- Results and Discussion
- Conclusion

Introdution

Ideal Response

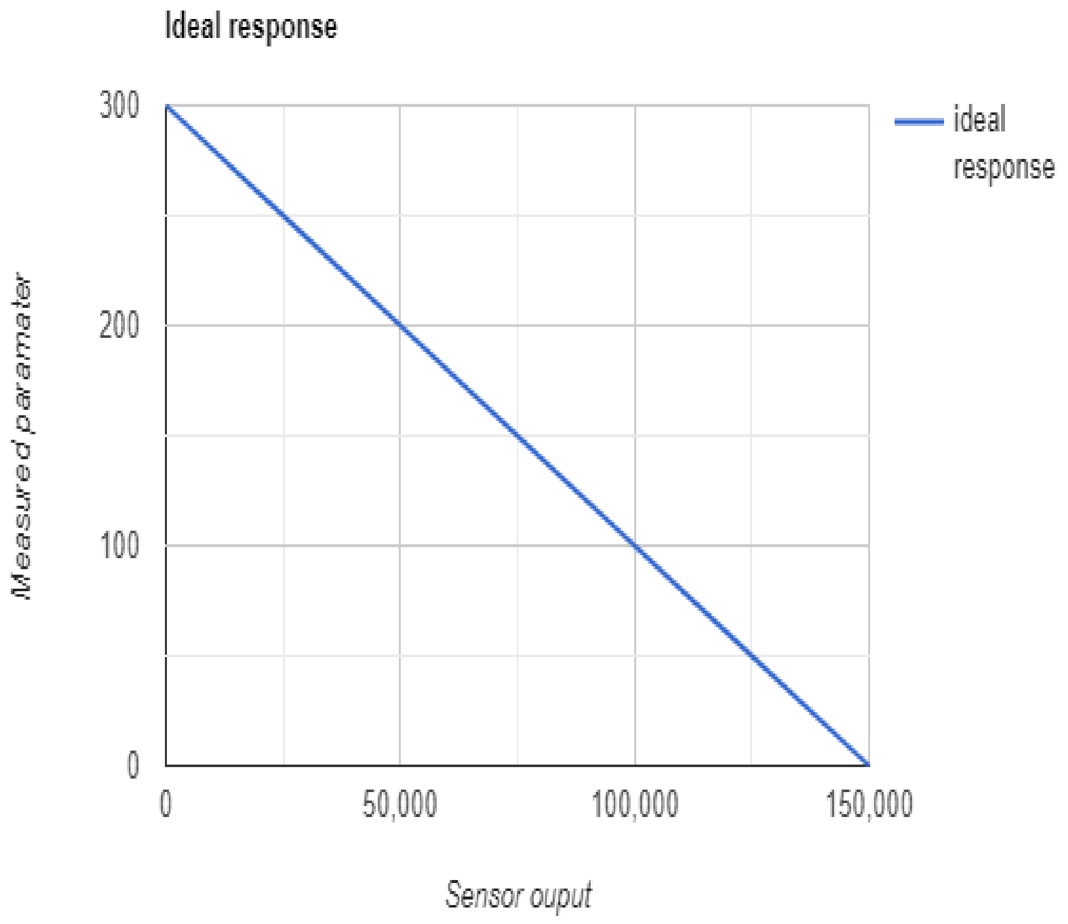


Fig 1: Ideal Response of sensor plot

Actual Response

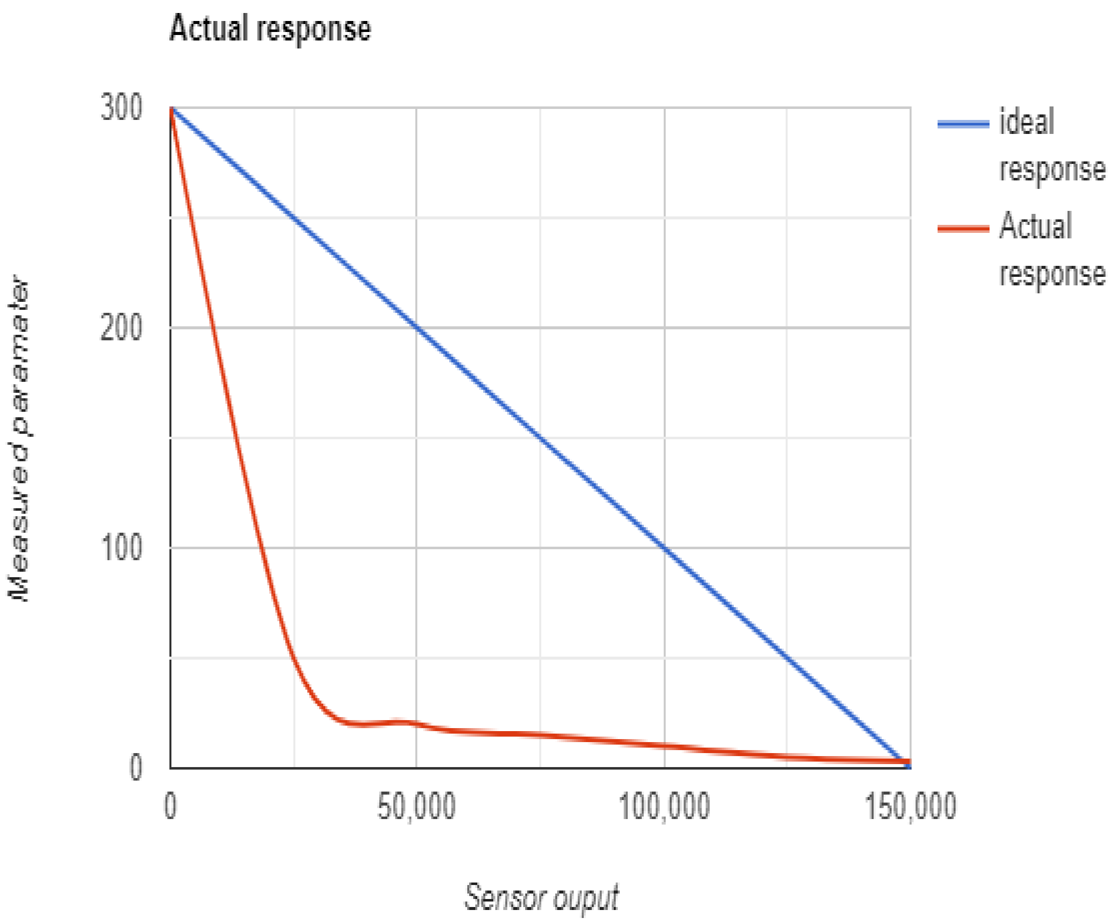


Fig 2: Actual Response of sensor plot

Introdution

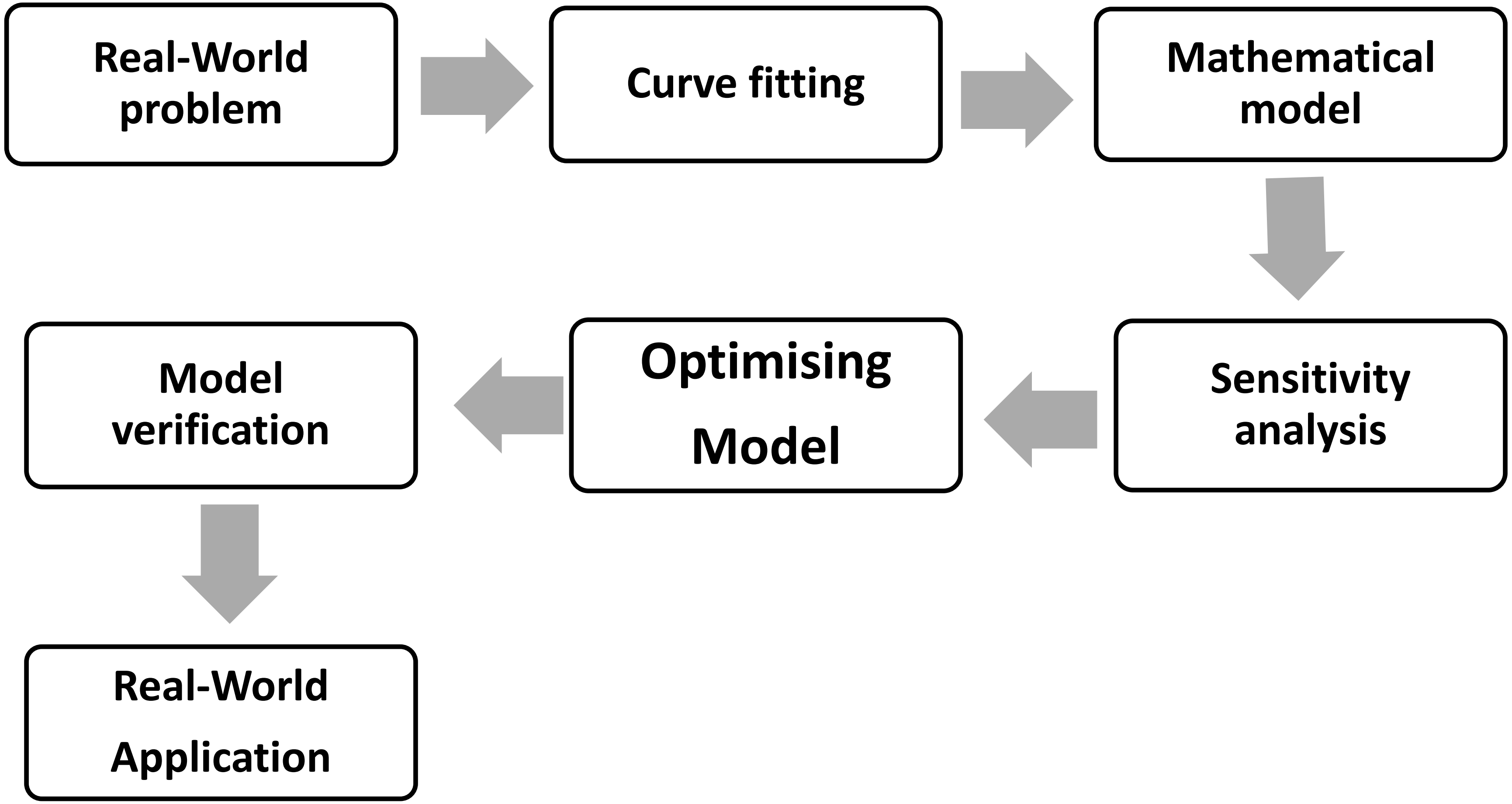
Various modeling techniques:

- Method of analogy
- Difference methods
- Regression analysis
- Curve fitting

Modeling tools :

- MATLAB
- Origin Lab
- Python

Introdution



Data Extraction

Compressive loads are applied on the sensor, when it is preloaded at 45^0 angled in the UTM(universal testing machine)

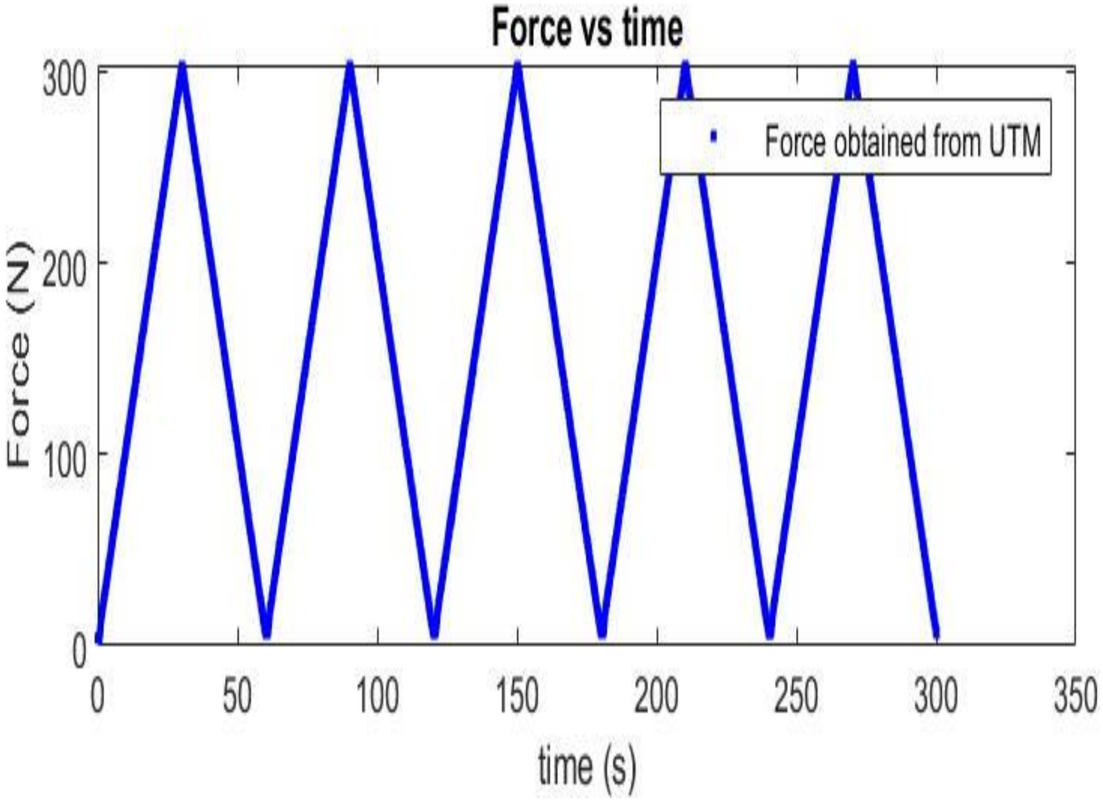


Fig 3: Force vs time plot

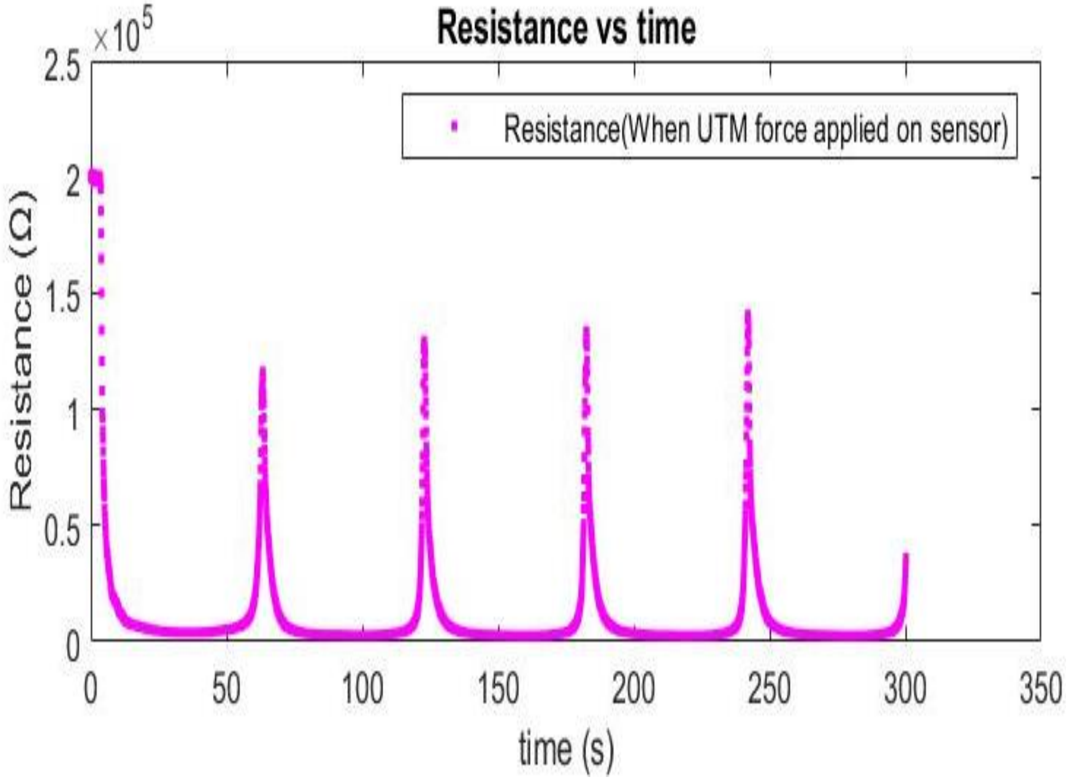


Fig 4: Resistance vs time plot

Data filtering

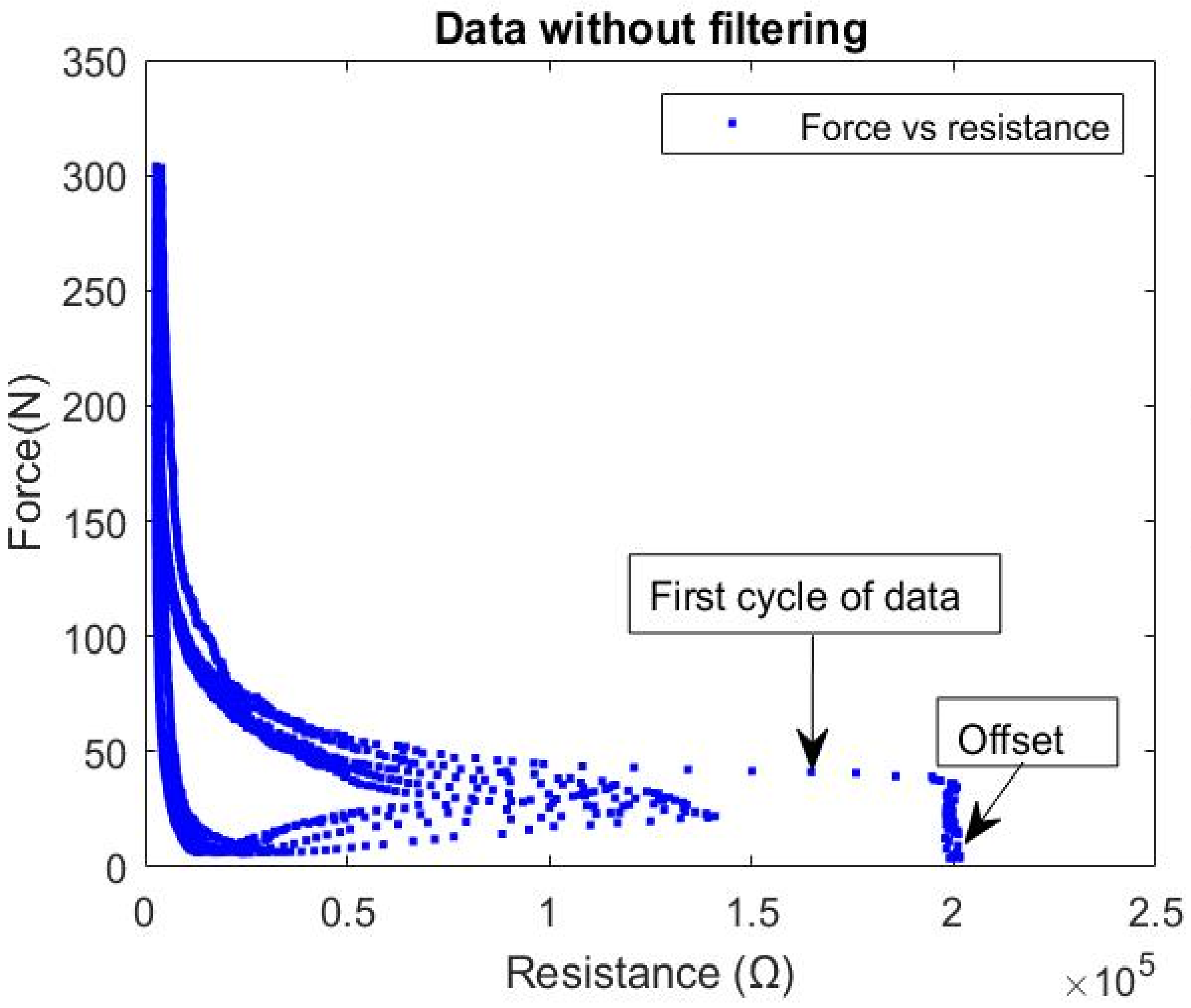


Fig 5: Data before filtering

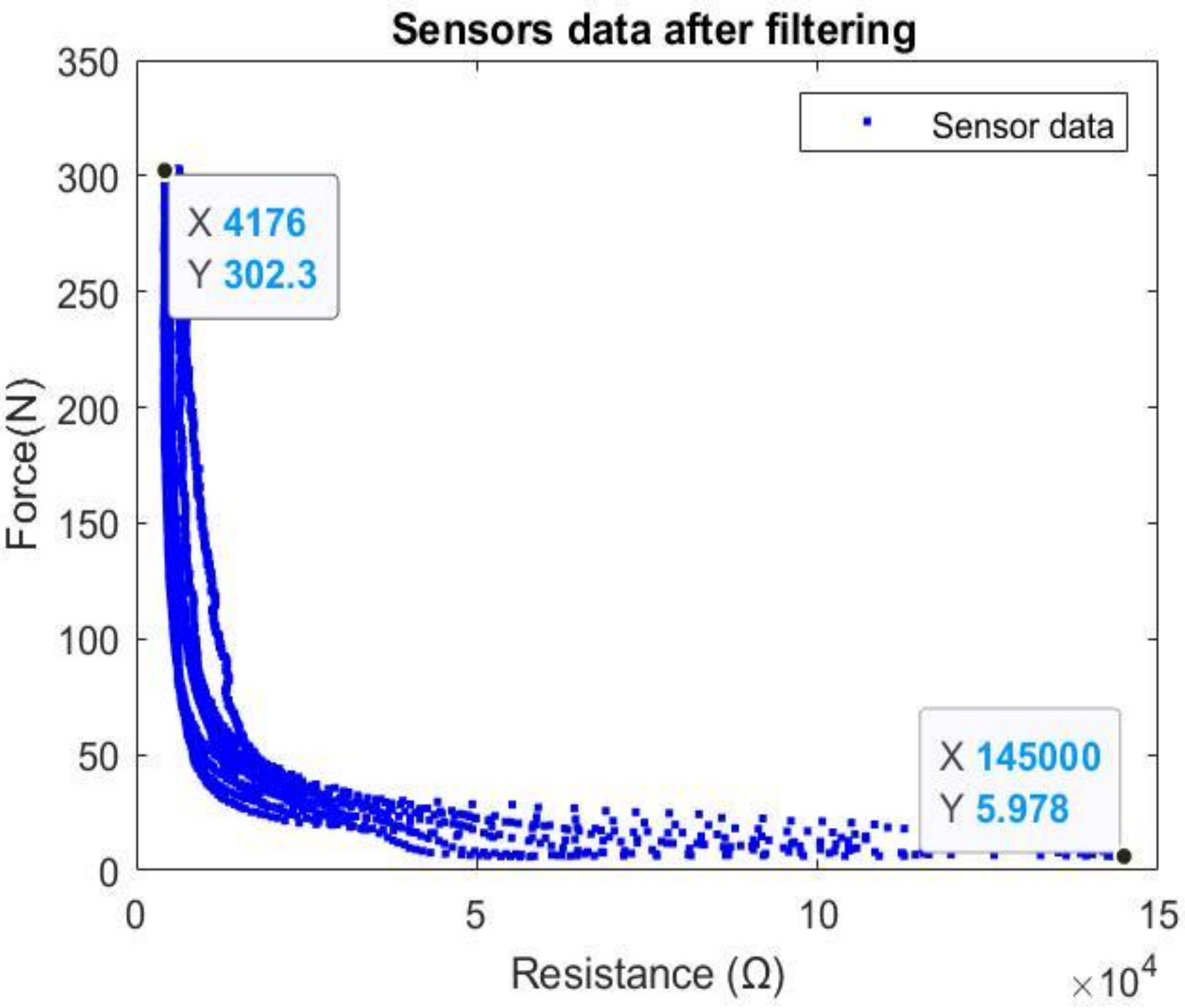


Fig 6 : Data after filtering

Regression

- Import dataset into Google Colab
- Import Matplot, Sklearn, Pandas Libraries
- Define X and Y
- Split Data into Train and Test dataset
- Apply Regression Model
- Selection of degree
- Model Evaluation

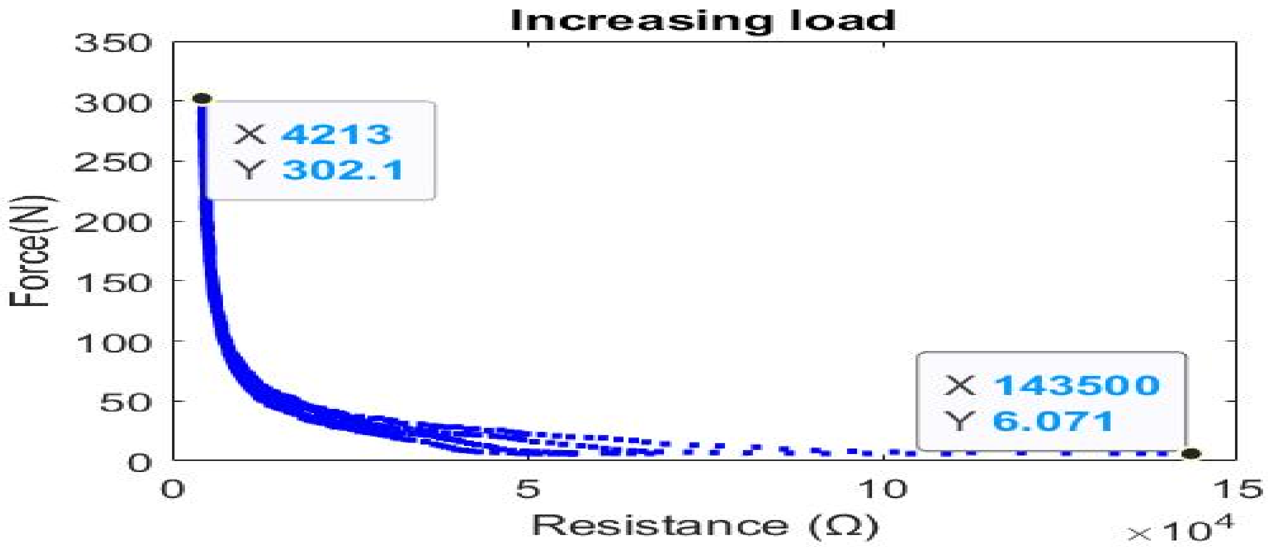


Fig 7: Increasing load

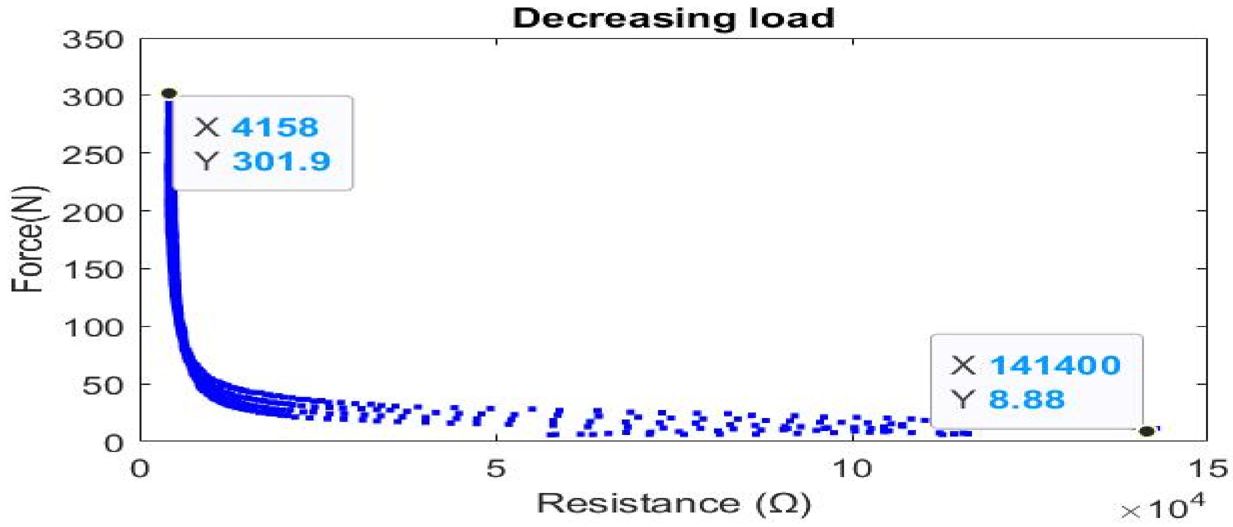


Fig 8: Decreasing load

Regression Models

Simple Linear Model

- Least squares estimation fits the data to linear model
- $y_i = \beta_0 + \beta_1 x_i$
- $\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2$ .... (1)
- $\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x},\ \hat{\beta}_1 = \frac{SXY}{SXX}$ .... (2)
- $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i\ ,\ \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ ....(3)
- SXY = $\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$
- SXX = $\sum_{i=1}^{n}(x_i - \bar{x})^2$ .... (4)
- we estimate the function as follows
- $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$ .... (5)

Polynomial Model

- $y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots + \beta_k x_i^k$ .... *(1)*
- In matrix form as *Y = XB + E*

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix},\ x = \begin{bmatrix} 1 & x_{11} & x_1^2 & \cdots & x_1^k \\ 1 & x_{21} & x_2^2 & \cdots & x_2^k \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_n^2 & \cdots & x_n^k \end{bmatrix},\ B = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix};\ E = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} \quad \text{... (2)}$$

- Using least square estimation the matrix B is obtained
- $\hat{B} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix} = (X'X)^{-1} X'Y$ .... *(3)*
- The estimated function is given as : $\hat{Y} = X\hat{B}$ ....(4)

https://www.originlab.com/doc/Origin-Help/LFit-Theory

Regression Results

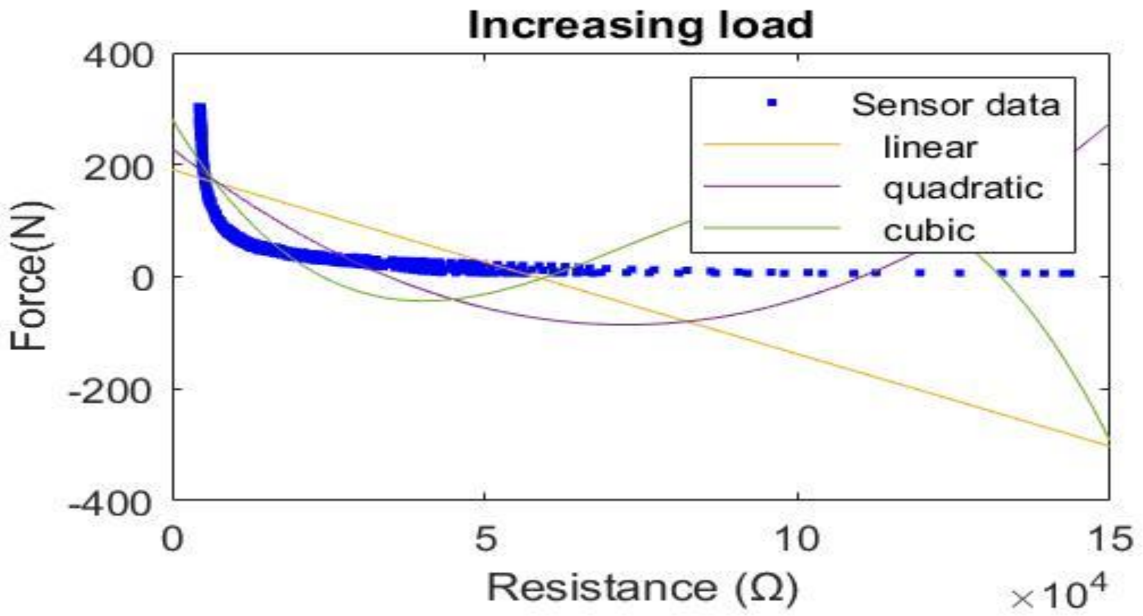


Fig 9: Regression results for Increasing load

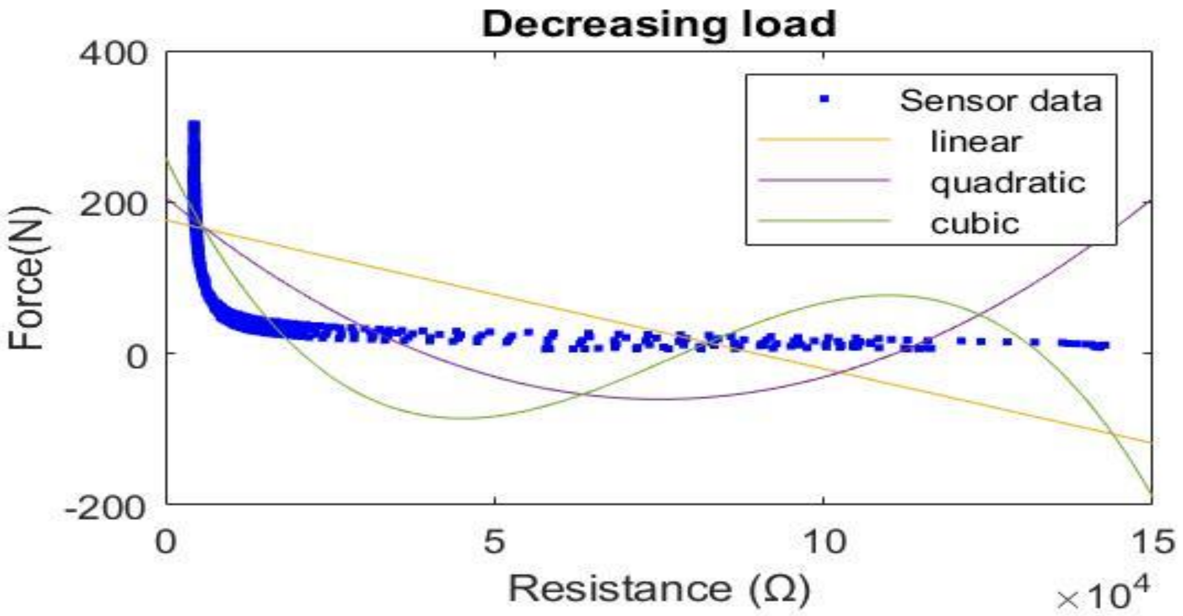


Fig 10: Regression results for Decreasing load

Regression Model	Equation	Increasing (R^2)	Decreasing(R^2)
Degree = 1	$y_i = \beta_0 + \beta_1 x_i$	0.2495	0.3384
Degree = 2	$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2$	0.4471	0.5264
Degree = 3	$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3$	0.54625	0.6614
Degree = 4	$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_4 x_i^4$	0.59392	0.7514
Degree = 5	$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_4 x_i^4 + \beta_5 x_i^5$	0.71203	0.8178
Degree = 6	$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_4 x_i^4 + \beta_5 x_i^5 + \beta_6 x_i^6$	0.8181	0.8648
Degree = 7	$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_4 x_i^4 + \beta_5 x_i^5 + \beta_6 x_i^6 + \beta_7 x_i^7$	0.82965	0.8947

Fig 2: Resistance vs time plot

Model Generation

- Import dataset into Origin Lab
- Define X and Y for the data
- Start the Non-linear Curve fit Tool
- Select Model
- Selection of model function
- Selection Iteration Algorithm
- Model Evaluation

- Non-Linear curve fitting is applied for the models shows behavior like sensor
  1)Logarithmic
  2)Exponential and
  3) Power.

- When a model is non-linear in its parameters, then least squares problem requires iterative solution algorithms

- Iterative Algorithms
  1) Levenberg-Marquardt(L-M) and
  2)Orthogonal Distance(ODR)

https://people.duke.edu/~hpgavin/ce281/lm.pdf

Iteration Algorithms

Levenberg-Marquardt(L-M)

- Basically, it is combination of the Gauss-Newton method and the steepest descent method.
- Good initial parameter values result in fast and reliable convergence.
- And fitting model, for example

$y = ax^b$ .... (1) .

$\ln(y) = \ln(a) + \ln(x^b)$.................(2)

$\ln(y) = \ln(a) + b\ln(x)$ ....(3)

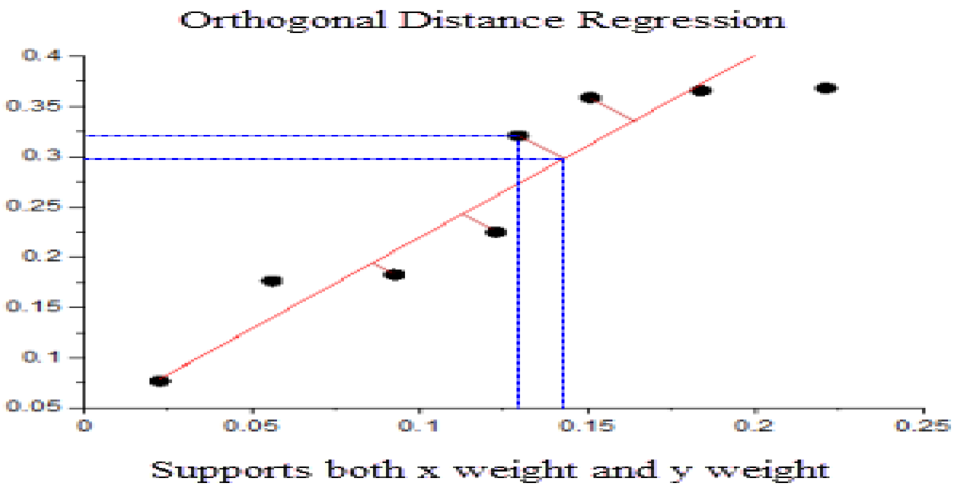
Orthogonal Distance(ODR)

- Residual in ODR is the orthogonal distance from the data to the fitted curve.
- ODR algorithm is based ODRPACK95. [3]
- For explicit function, expressed as $\min(\sum_{i=1}^{n}(\omega_{yi^*}\epsilon^2 + \omega_{xi^*}\delta_i^2))$
- $y_i = f(x_i + \delta_i\beta)$ - ϵ_i , i = 1, 2, ...,n

Were,ω_{x_i} , ω_{y_i} input weights of x_i, y_i
ϵ_i residual corresponding to x_i, y_i, and β is the fitting parameter. *https://www.originlab.com/doc/Origin-Help/NLFit-Theory*

ODR vs L-M

	ODR	L-M
Application	Both Implicit & Explicit	Only explicit functions
Weight	Support both x & y weights	Support only y weights
Residual Source	Orthogonal distance from the data to the fitted curve	Difference between observed & predicted value
Iteration Process	Adjusting fitting parameter values and independent variable	Adjusting the values of fitting parameters



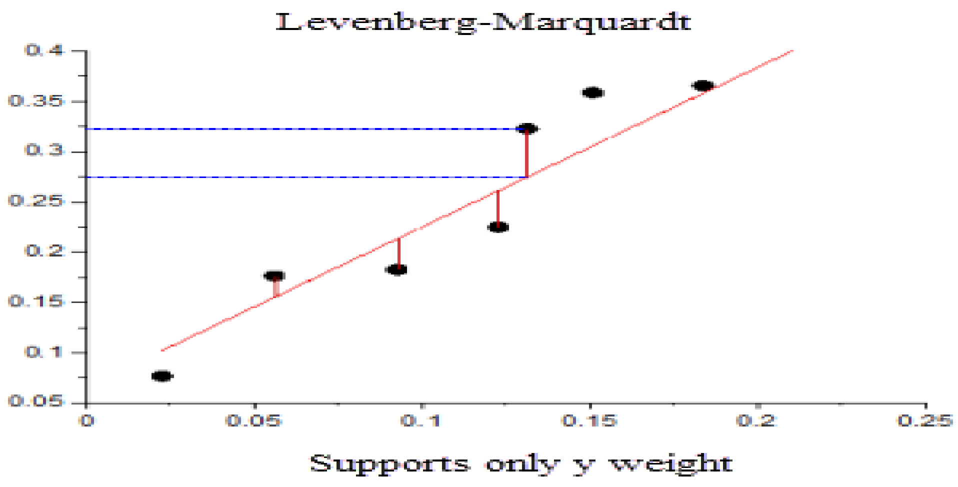


Fig 11 Comparison between L-M and ODR

https://www.originlab.com/doc/origin-help/nlfit-theory

Comparison of Models

- Belehardek function of power category showing good fitting.

- Unlike other models it shows good variation along with sensor data

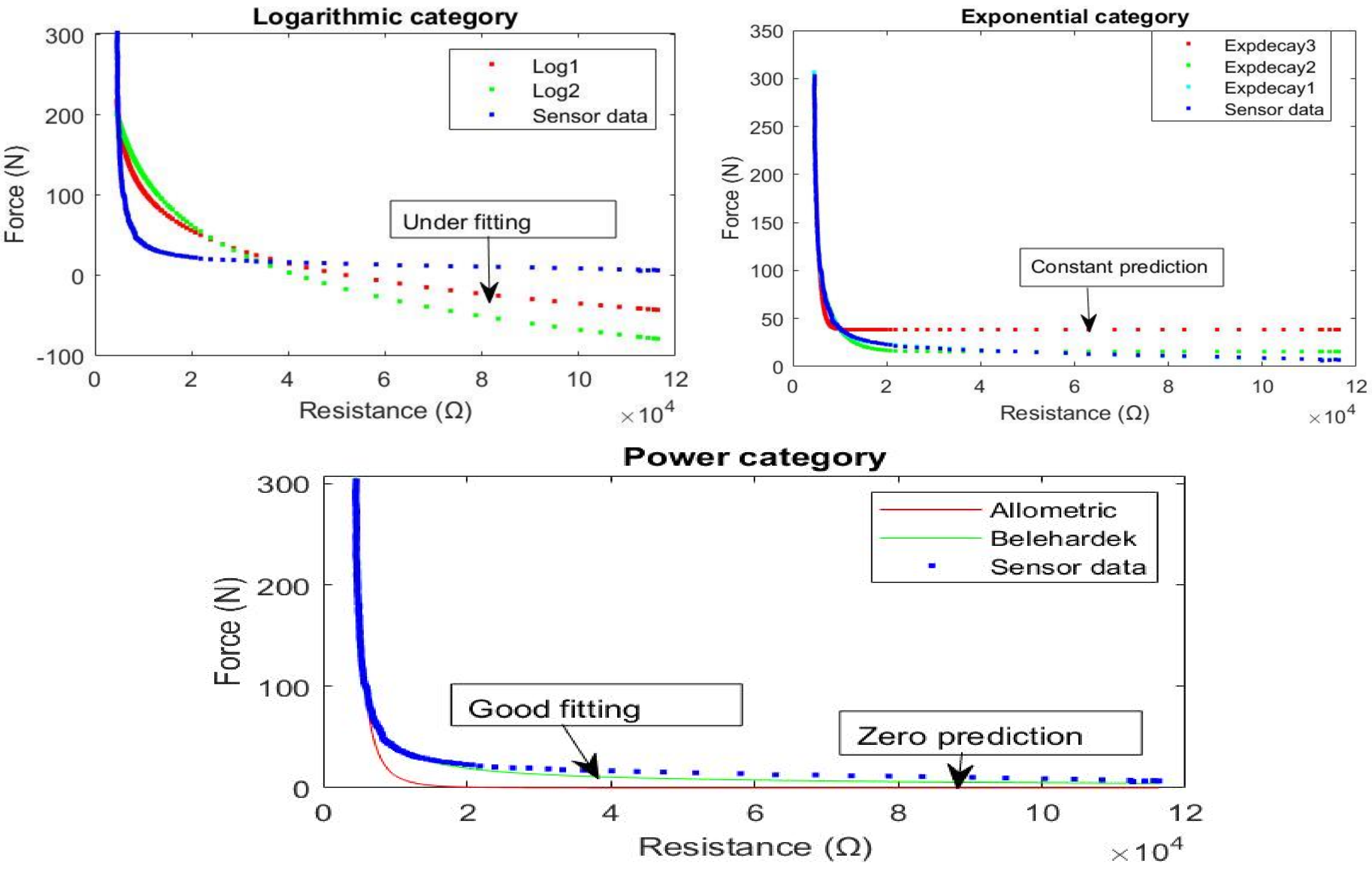


Fig 12: Non-linear fitting models

Selection of Model

Function	Fit status	Equation	R^2 value	MSE
Logarithmic category				
Log1	Under fitting	$y_i = a – b*\ln(x+c)$	0.6623	2522.18
Log2	Under fitting	$y_i = a*\ln(-b*\ln(x_i))$	0.5776	3153.18
Exponential Category				
Expdecay1	Constant prediction	$y_i = y_0+a_1*e^{(-(x_i - x_0)/t_1)}$	0.95991	301.73809
Expdecay2	Constant prediction	$y_i = y_0+a_1*e^{(-(x_i - x_0)/t_1)} + a_2*e^{(-(x_i - x_0)/t_2)}$	0.9772	172.4409
Expdecay3	Constant prediction	$y_i = y_0+a_1*e^{(-(x_i - x_0)/t_1)} + a_2*e^{(-(x_i - x_0)/t_2)} + a_3*e^{(-(x_i - x_0)/t_3)}$	0.97767	169.6951
Power Category				
Allometric	Overfitting	$y_i = a*x_i^b$	0.94594	405.013
Belehardek	Good fitting	$y_i = a*(x_i -b)^c$	0.9962	109.54

Optimization of Parameters

- Belehradek and Porodko independently published [1].
- Used in microbiology, to describe the rate of biological reactions as a function of temperature.
- X shifted power function, combination of power and hyperbolic.
- y = $2032.85915 * (x - 2137.74722)^{-0.7458}$

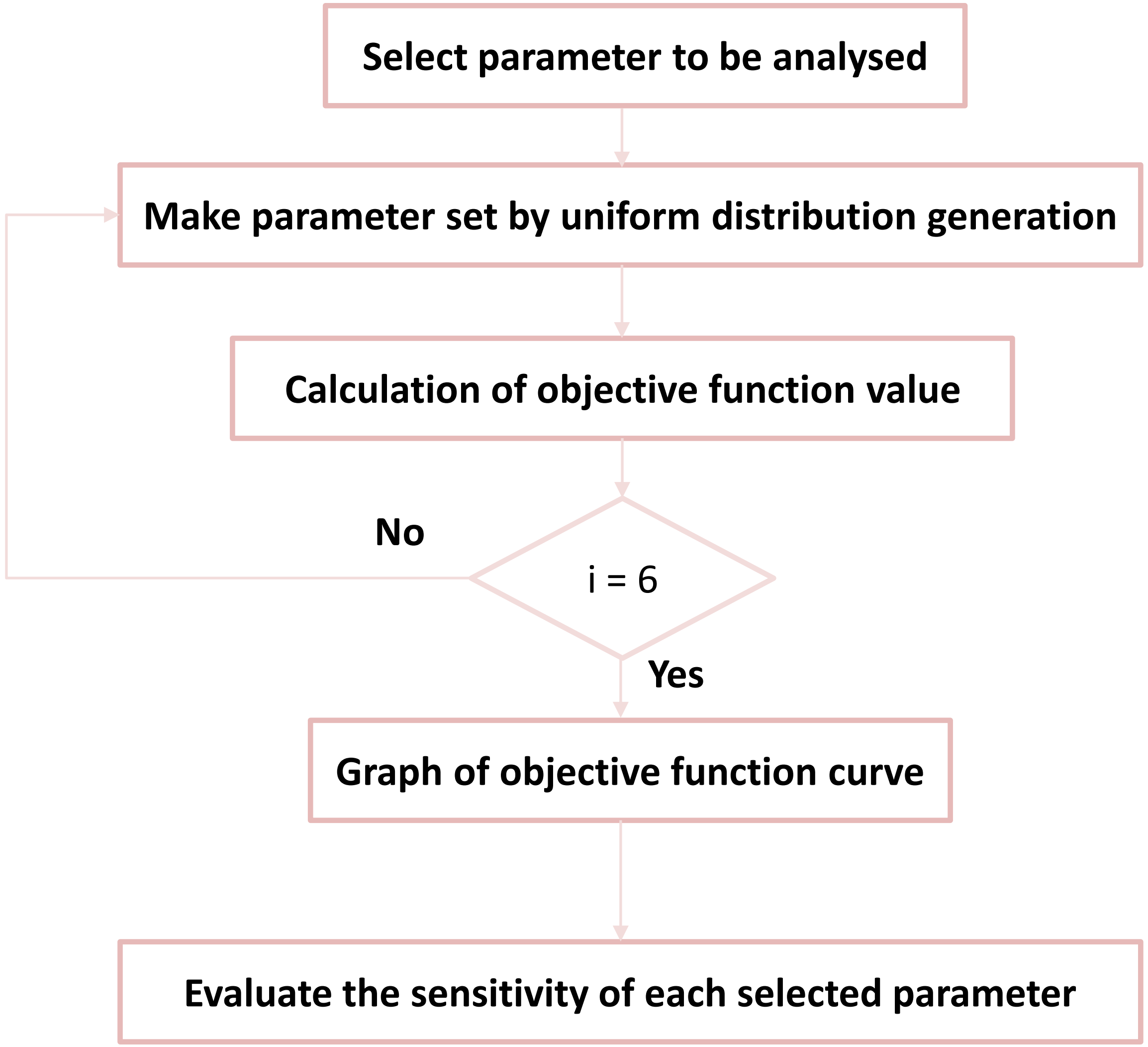
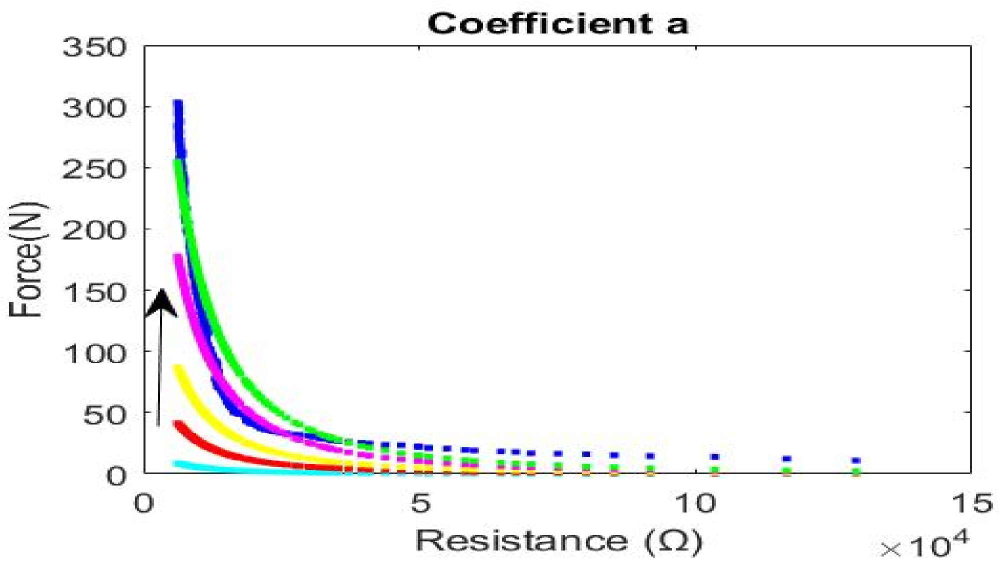
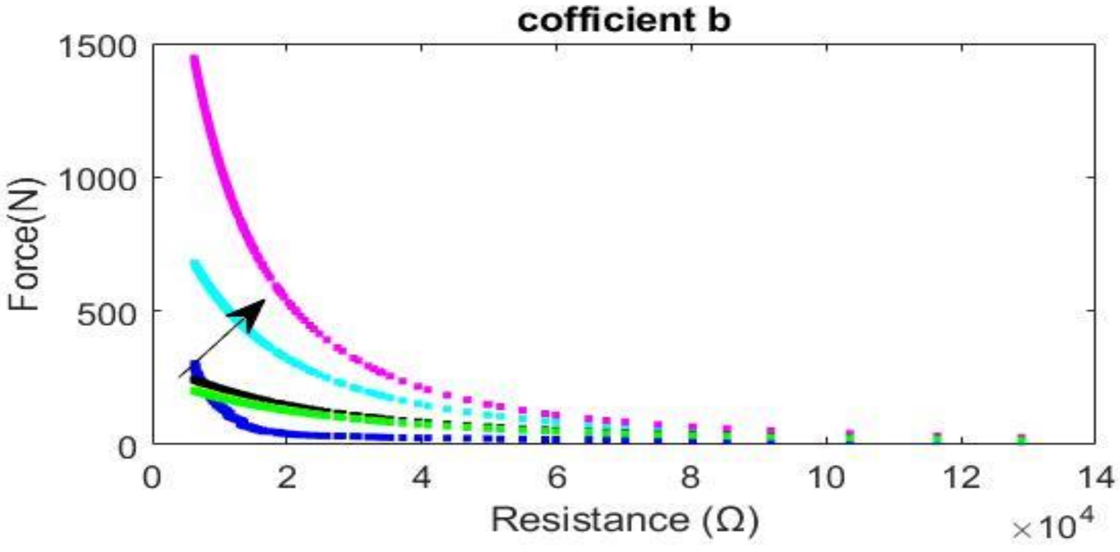


Fig 13: Flow chart for analysing the parameter sensitivity

Parameters sensitivity

Coefficients	Minimum value	Medium value	High value
a	*2032.85915*	48521.5487	6131.83939
b	*2137.74722*	6594.1248	8451.84213
c	*−0.7458*	-0.12548	0.02821





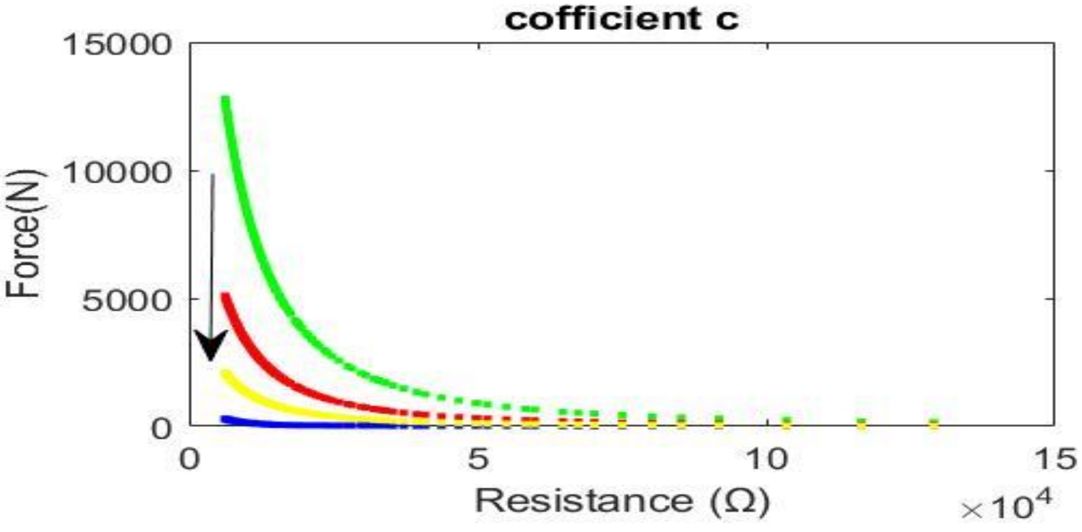


Fig 14: Parameter sensitivity of the model

Parameters sensitivity

- Analyse the effect of one parameter on the function at a time, keeping the other parameters fixed.
- The parameter a is offset, b is coefficient, and c is power of the function $y = a*(x - b)^c$

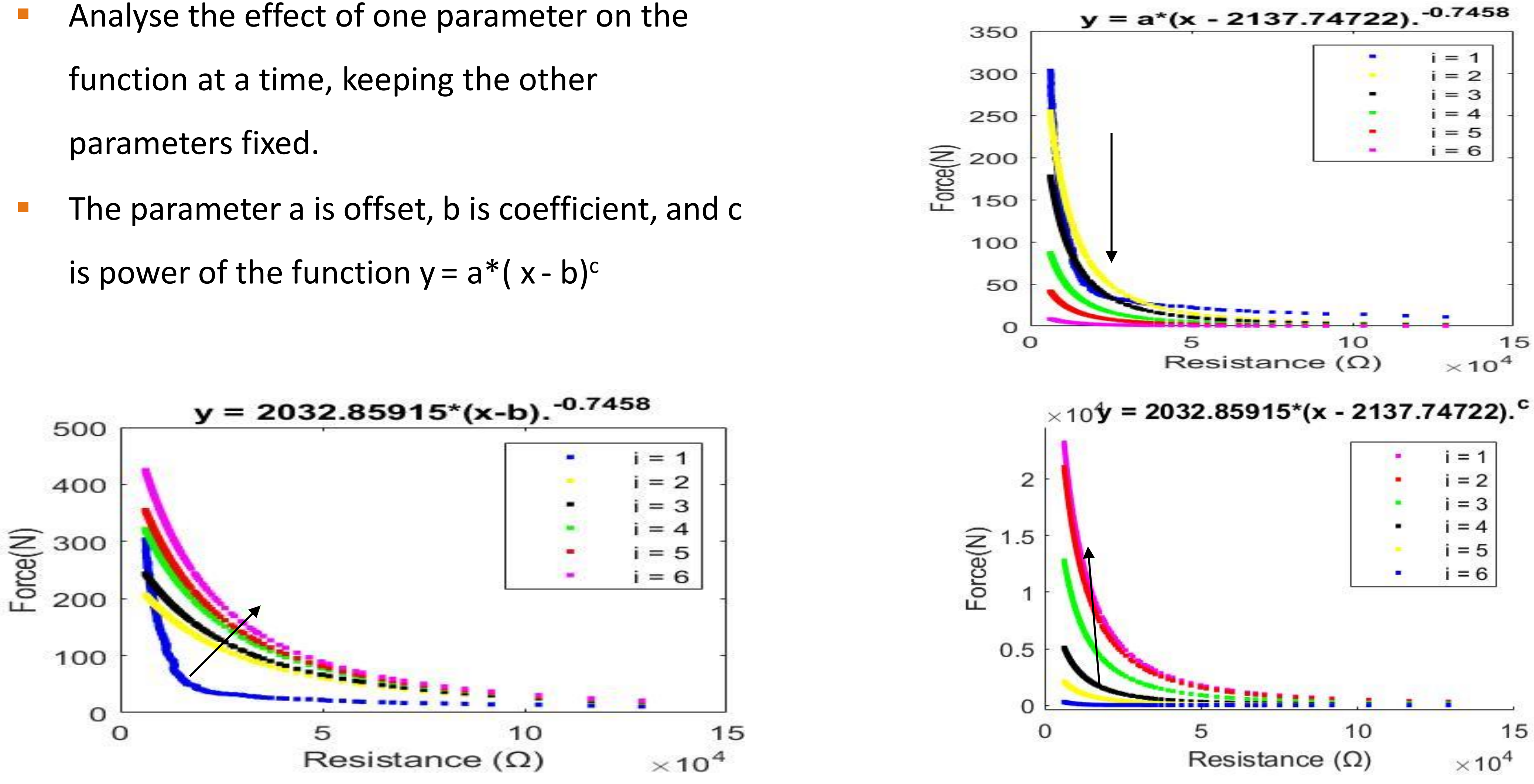


Fig 15: Parameters sensitivity of the model

Model Optimisation

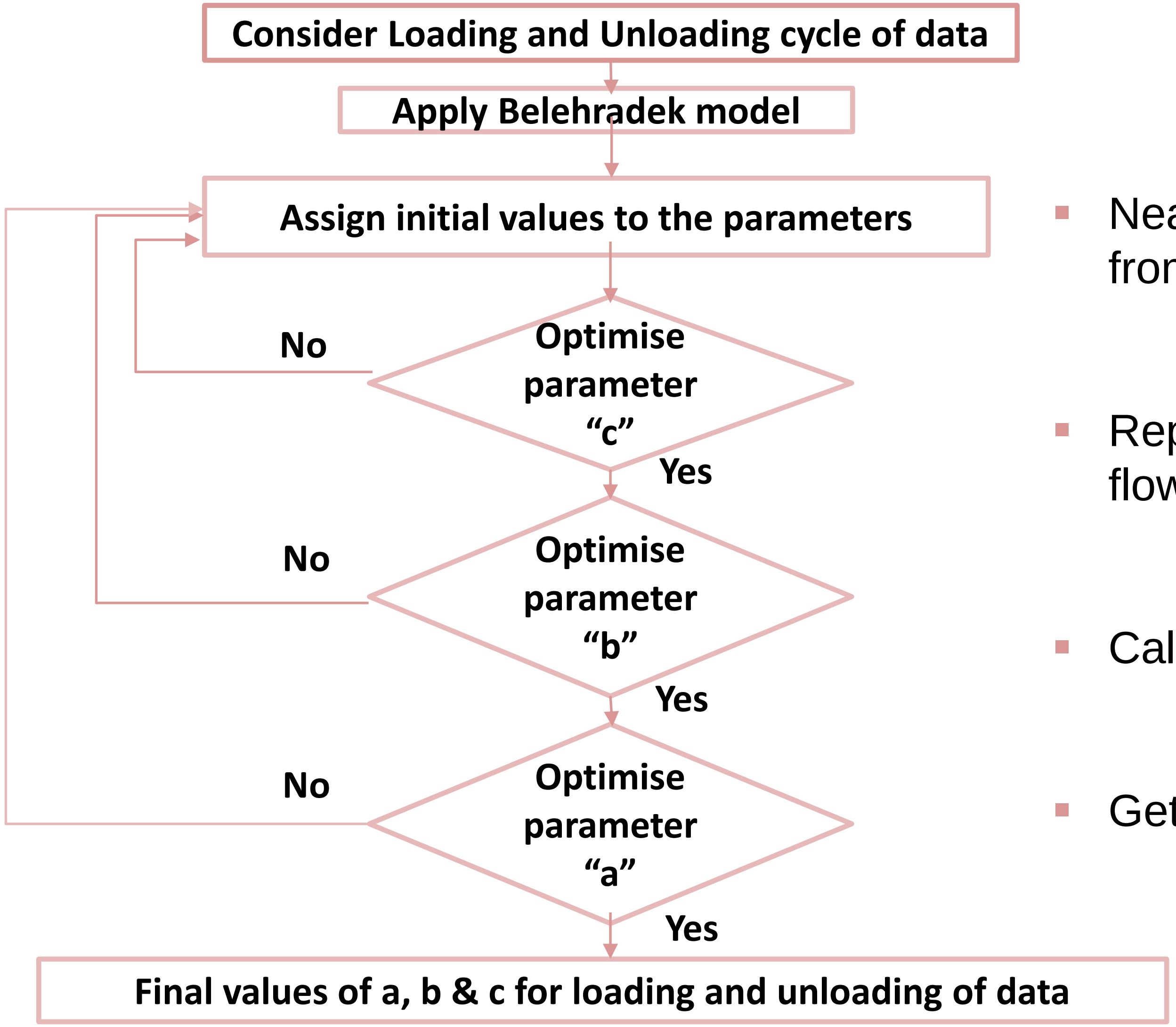
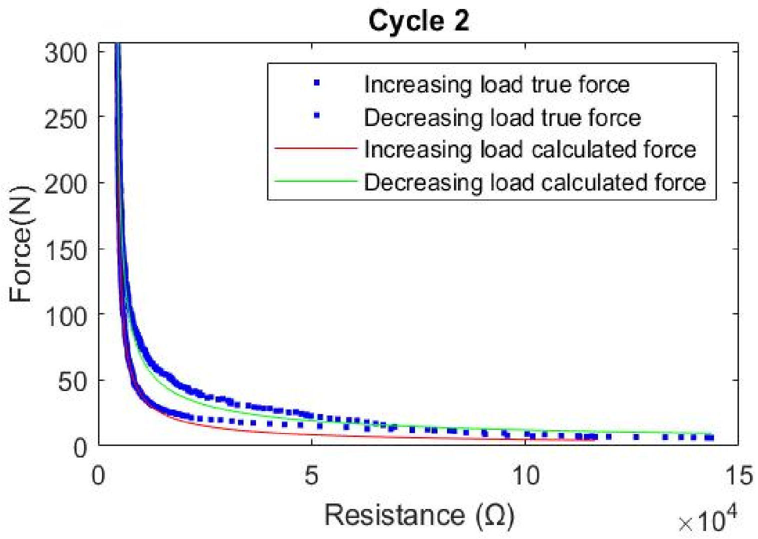
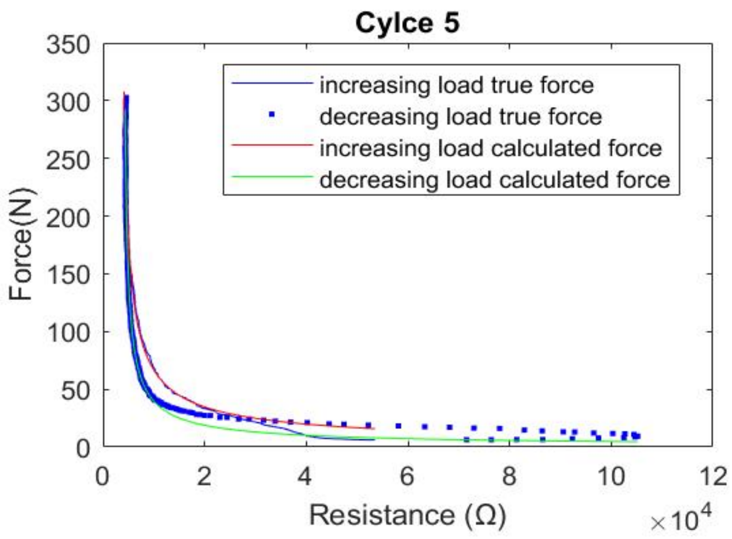


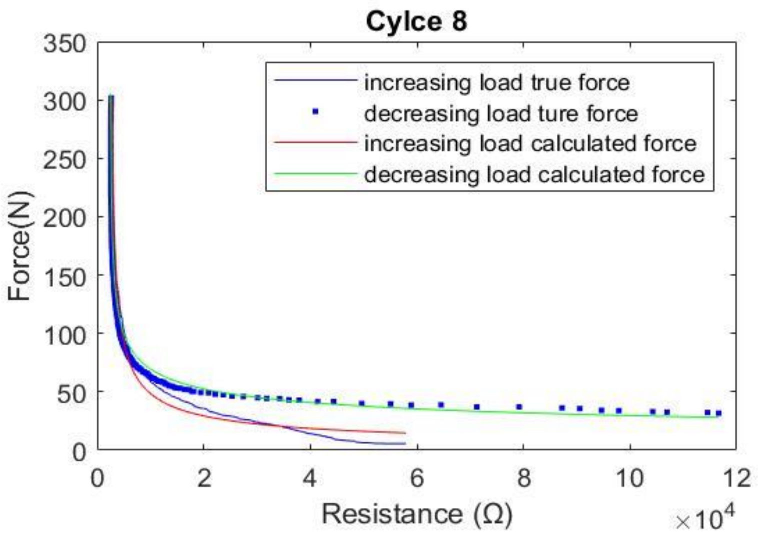
Fig 16: Flow chart for optimising the model

- Nearly 15cycles of loading and unloading data from 12000 sets of experimental data.
- Repeat the same procedure as shown in flowchart for all cycles of data.
- Calculate the average of each parameter.
- Get the optimized model.

Results and discussion







- Increasing Loading calculated formula

$$y1 = 24268.35857*(x – 2789.298881)^{-0.66939}$$

- Decreasing loading calculated formula

$$y2 = 2332.85915*(x – 3179.64618)^{-0.551925}$$

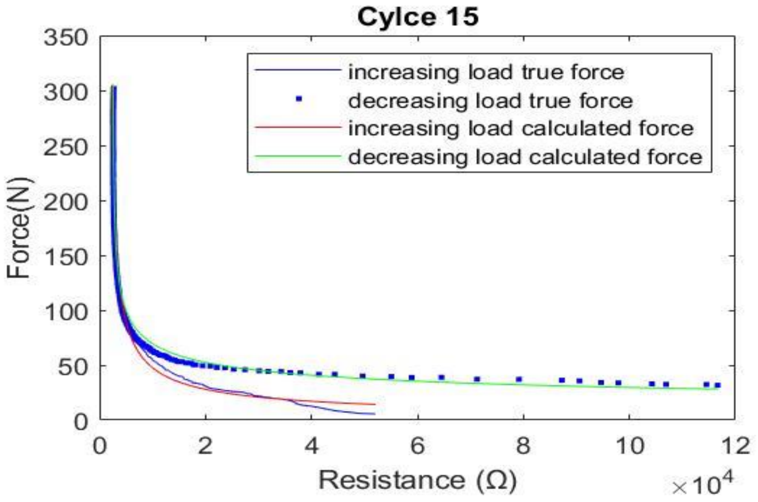


Fig 17: Model verification for different cycles of data

Conclusion

- Each and every sensor output is different from one other, while it measured on same conditions
- Even there is change in measured sensor value, whenever it is measured.
- The model can predict future behavior or result that has not yet been measured.
- When we substitute the measured resistance value into the model, we can calculate the force applied on the sensor.
- When we know the applied force, we can calculate the resistance of the sensor.

Reference

[1] T. RossDept. of Agricultural Science, University of Tasmania, G.P.O. Box 252C, Hobart 7001, Tasmania, Australia.

[2]Comparison_result_of_inversion_of_gravity_data_of_a_fault_by_particle_swarm_optimization_and_Levenberg-Marquardt_methods.

[3].https://eprints.cs.vt.edu/archive/00000707/01/odrTOMS04.pdf

[4]. N. Li, T. Zong and Z. Zhang, "Prediction of the Electronic Work Function by Regression Algorithm in Machine Learning," 2021 IEEE 6th International Conference on Big Data Analytics (ICBDA), 2021, pp. 87-91, doi: 10.1109/ICBDA51983.2021.9403202.

[5]. J. Ma, "Machine Learning in Predicting Diabetes in the Early Stage," 2020 2nd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), 2020, pp. 167-172, doi: 10.1109/MLBDBI51377.2020.00037.


Thank You