Nonlinear Regression Models with Iterative Methods

Aleksandr Chernega, Frank Lynch, PhD

Mathematics Department | Eastern Washington University

Contents

1. Introduction

After diving into multiple regression, a student of statistics will likely wonder what equation can they fail to fit to a set of data. As long as the regression parameters are coefficients of transformations of the dependent variables, as such:

$$\hat{y} = \beta_1 + \beta_2 f_1(\boldsymbol{x}) + \cdots + \beta_2 f_m(\boldsymbol{x})$$

then practically any function can be fit with the typical least-squares multiple regression procedure. However, what if a student wishes to fit a nonlinear function where the regression parameters are within the transformations themselves? Equations such as,

$$\hat{y} = \beta_1 \, sin(\beta_2 \boldsymbol{x} + \beta_3) + \beta_4$$

and

$$\hat{y} = \beta_1 (\beta_2 \boldsymbol{x} + \beta_3)^{\beta_4 \boldsymbol{x} + \beta_5} + \beta_6$$

require an altogether different technique. Luckily, drawing heavily from numerical analysis, we are able to develop numerical solutions to satisfy this exact curiosity.

Numerical methods are taught to math students for good reason: they offer accurate approximations to mathematical models that often do not have analytic solutions and, critically, are many students' first forays into programming a computer to do math for them. Emphasis is often placed on solving linear systems due to their clean properties of convergence. This thesis will step beyond that and develop the Newton-Raphson method for approximating solutions to *nonlinear* systems as the application we hope to apply it to will almost always have no analytic solution.

Once developed, we will observe that the ability to solve a system of nonlinear systems of equations allows us to attack the problem of minimizing the sum of squared errors of a nonlinear regression model. We will quickly recognize that regression models are all but guaranteed to not have exact solutions, however, differentiating the system with respect to our regression parameters will then yield a system that will directly lead us to the Gauss-Newton method of solving nonlinear least squares problems iteratively.

Proving convergence is difficult with such methods but we will do our best to intuitively lay out the groundwork without relying on the substantial amount of prerequisite knowledge required to completely prove convergence.

Given the applied nature of this paper, a worked example with sample code will provide the reader with a tangible application of this method.

2. Newton-Raphson Method for Nonlinear Systems

We will assume that the reader is familiar with Newton's method of root finding in $\mathbb{R}$, in which an initial guess $\boldsymbol{x_0}$ is improved upon iteratively according to the algorithm,

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

We wish to extend Newton's algorithm to approximate roots of any n-dimensional systems of nonlinear equations of the form,

$$\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0} = \begin{cases} f_1(\boldsymbol{x}) \\ \vdots \\ f_n(\boldsymbol{x}) \end{cases}$$

where argument $\boldsymbol{x} = [x_1, x_2, \dots, x_m]^T$ with $\boldsymbol{x} \in \mathbb{R}^m$.

We begin derivation with an approximation of $\boldsymbol{F}$ about a small change in $\boldsymbol{x}$, which we achieve with a multivariable Taylor expansion of $\boldsymbol{F(x + \Delta x)}$ for every $f_i(\boldsymbol{x} + \Delta\boldsymbol{x})$,

$$\begin{aligned} f_1(\boldsymbol{x} + \Delta\boldsymbol{x}) &= f_1(\boldsymbol{x}) + \sum_{i=1}^{m} \frac{\partial f_1(\boldsymbol{x})}{\partial x_i} \Delta x_1 + \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \frac{\partial^2 f_1(\boldsymbol{\xi})}{\partial x_i \partial x_j} \Delta x_i \Delta x_j \\ f_2(\boldsymbol{x} + \Delta\boldsymbol{x}) &= f_2(\boldsymbol{x}) + \sum_{i=1}^{m} \frac{\partial f_2(\boldsymbol{x})}{\partial x_i} \Delta x_2 + \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \frac{\partial^2 f_2(\boldsymbol{\xi})}{\partial x_i \partial x_j} \Delta x_i \Delta x_j \\ &\vdots \\ f_n(\boldsymbol{x} + \Delta\boldsymbol{x}) &= f_n(\boldsymbol{x}) + \sum_{i=1}^{m} \frac{\partial f_n(\boldsymbol{x})}{\partial x_i} \Delta x_n + \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \frac{\partial^2 f_n(\boldsymbol{\xi})}{\partial x_i \partial x_j} \Delta x_i \Delta x_j \end{aligned}$$

for some $\boldsymbol{\xi} \in \mathbb{R}$. We can succinctly write this system as,

$$\boldsymbol{F}(\boldsymbol{x} + \Delta\boldsymbol{x}) = \boldsymbol{F}(\boldsymbol{x}) + \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_m} \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \vdots \\ \Delta x_m \end{bmatrix} + O(\Delta\boldsymbol{x}^2)$$

We immediately recognize the matrix of partial derivatives as the Jacobian matrix of $\boldsymbol{F}$. Thus,

$$\boldsymbol{F}(\boldsymbol{x} + \Delta\boldsymbol{x}) = \boldsymbol{F}(\boldsymbol{x}) + J(\boldsymbol{x})\Delta\boldsymbol{x} + O(\Delta\boldsymbol{x}^2)$$

Dropping the error term for now, we solve for $\Delta\boldsymbol{x}$:

$$\boldsymbol{F}(\boldsymbol{x} + \Delta\boldsymbol{x}) = \boldsymbol{F}(\boldsymbol{x}) + J(\boldsymbol{x})\Delta\boldsymbol{x}$$

$$\Leftrightarrow \qquad \boldsymbol{F}(\boldsymbol{x}+\varDelta\boldsymbol{x})-\boldsymbol{F}(\boldsymbol{x})=J(\boldsymbol{x})\varDelta\boldsymbol{x}$$

$$\Leftrightarrow \qquad J^{-1}(\boldsymbol{x})[\boldsymbol{F}(\boldsymbol{x}+\varDelta\boldsymbol{x})-\boldsymbol{F}(\boldsymbol{x})]=\varDelta\boldsymbol{x}$$

Recall that we aim to find where $\boldsymbol{F}(\boldsymbol{x})=\mathbf{0}$. Let us assume that our approximation $\boldsymbol{x}+\varDelta\boldsymbol{x}$ is a root. It follows that,

$$\begin{aligned}\varDelta\boldsymbol{x}&=J^{-1}(\boldsymbol{x})[\boldsymbol{F}(\boldsymbol{x}+\varDelta\boldsymbol{x})-\boldsymbol{F}(\boldsymbol{x})]\\&=J^{-1}(\boldsymbol{x})[-\boldsymbol{F}(\boldsymbol{x})]\\&=-J^{-1}(\boldsymbol{x})\boldsymbol{F}(\boldsymbol{x})\end{aligned}$$

Adding $\boldsymbol{x}$ to both sides results in an expression for the subsequent iteration of $\boldsymbol{x}$ that is a more accurate approximation to the true root:

$$\boldsymbol{x}+\varDelta\boldsymbol{x}=\boldsymbol{x}-J^{-1}(\boldsymbol{x})\boldsymbol{F}(\boldsymbol{x})$$

The result is that if we begin with an initial guess $\boldsymbol{x_0}$ for the true root $\boldsymbol{x_r}$, we can update $\boldsymbol{x}_0$ with $\boldsymbol{\varDelta x}$ to obtain a more accurate approximation for $\boldsymbol{x_r}$. If we continue in this manner, iterating every subsequent approximation with a new updating term, we will eventually converge on an approximation for $\boldsymbol{x_r}$ within any chosen error tolerance. In summary,

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\varDelta\boldsymbol{x}_k=\boldsymbol{x}_k-J^{-1}(\boldsymbol{x}_k)\boldsymbol{F}(\boldsymbol{x}_k)$$

which we recognize as the higher-dimensional analog to the simple iterative Newton's method we started with. Given a reasonable guess for the initial root approximation, the Newton-Raphson method will converge, which we will discuss shortly. However, the method critically assumes that the existence of a solution exists; in other words, that the inverse of the Jacobian exists. Because we aim to apply this method to nonlinear fitting of regression curves, we are all but guaranteed that the inverse of the Jacobian will not exist as the probability our regression curve interpolates exactly through every data point in any real-world dataset is astronomically low. The solution to this problem will directly lead us to the Gauss-Newton method of approximation.

3. Gauss-Newton Method

Our ultimate aim is to be able to fit any curve $f(x,\boldsymbol{\beta})$ to a set D of n data points where $D=\{(x_i,y_i)|i\epsilon\mathbb{N}_n\}$, aiming beyond tradition regression which requires that the fitting curve and data points be able to be transformed into a linear system of equations.

Let $\boldsymbol{\beta}$ be the vector consisting of model parameters. We begin by seeking to minimize the sum of squared errors (or squared residuals) between the set of functions $f_i(\boldsymbol{\beta}) = f(x_i, \boldsymbol{\beta}) \in \boldsymbol{F}(\boldsymbol{\beta})$ and the data we are fitting it to. Let

$$\boldsymbol{r} = \begin{bmatrix} y_1 - f_1(\boldsymbol{\beta}) \\ \vdots \\ y_n - f_n(\boldsymbol{\beta}) \end{bmatrix} = \boldsymbol{y} - \boldsymbol{F}(\boldsymbol{\beta})$$

be the vector of residuals, a nonlinear function in $\mathbb{R}^n$. Then, the sum of squared residuals is equal to

$$\sum_{i=1}^{n} r_i^2 = \boldsymbol{r}^T\boldsymbol{r} = ||\boldsymbol{y} - \boldsymbol{F}(\boldsymbol{\beta})||^2$$

We minimize $\boldsymbol{r}^T\boldsymbol{r}$ by deriving where the derivative of $\boldsymbol{r}^T\boldsymbol{r}$ with respect to $\boldsymbol{\beta}$ is equal to zero:

$$\frac{d[\boldsymbol{r}^T\boldsymbol{r}]}{d\boldsymbol{\beta}} = \mathbf{0}$$

$$\Leftrightarrow \quad \frac{d}{d\boldsymbol{\beta}}||\boldsymbol{y} - \boldsymbol{F}(\boldsymbol{\beta})||^2 = \mathbf{0}$$

$$\Leftrightarrow \quad \frac{d}{d\boldsymbol{\beta}}\sum_{i=1}^{n}\big(y_i - f_i(\boldsymbol{\beta})\big)^2 = \mathbf{0}$$

$$\Leftrightarrow \quad \sum_{i=1}^{n}\left(-\frac{d}{d\boldsymbol{\beta}}f_i(\boldsymbol{\beta})\right) * 2(y_i - f_i(\boldsymbol{\beta})) = \mathbf{0}$$

$$\Leftrightarrow \quad -2J^T(\boldsymbol{y} - \boldsymbol{F}(\boldsymbol{\beta})) = \mathbf{0}$$

$$\Leftrightarrow \quad J^T(\boldsymbol{y} - \boldsymbol{F}(\boldsymbol{\beta})) = \mathbf{0}$$

where J is the Jacobian matrix of the residuals $\boldsymbol{r}$ where $J(x) \in \mathbb{R}^{m\times n}$, $J(\boldsymbol{\beta})_{ij} = \frac{\partial \boldsymbol{r}_i(\beta_j)}{\partial \beta_j}$. We are **not** assuming closed form solutions for this equation; thus, we develop an iterative approach to finding the roots of the above equation. We apply a similar technique as with the Newton-Raphson method we developed above for solving roots of nonlinear systems of equations. We aim to find,

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k + \Delta\boldsymbol{\beta}$$

As with the Newton-Raphson method, we found that the Taylor expansion of $\boldsymbol{F}(\boldsymbol{\beta}_{k+1})$ about $\boldsymbol{\beta}_k$ is equal to

$$\boldsymbol{F}(\boldsymbol{\beta}_{k+1}) = \boldsymbol{F}(\boldsymbol{\beta}_k) + \boldsymbol{J}\Delta\boldsymbol{\beta}$$

Substituting into $J^T(\boldsymbol{y} - \boldsymbol{F}(\boldsymbol{\beta}_{k+1})) = \mathbf{0}$,

$$J^T(\boldsymbol{y} - (\boldsymbol{F}(\boldsymbol{\beta}_k) + J\Delta\boldsymbol{\beta})) = \mathbf{0}$$

$$\Leftrightarrow \qquad J^T\boldsymbol{y} - J^T\boldsymbol{F}(\boldsymbol{\beta}_k) - J^T J\Delta\boldsymbol{\beta} = \mathbf{0}$$

$$\Leftrightarrow \qquad J^T J\Delta\boldsymbol{\beta} = J^T\boldsymbol{y} - J^T\boldsymbol{F}(\boldsymbol{\beta}_k)$$

$$\Leftrightarrow \qquad J^T J\Delta\boldsymbol{\beta} = J^T\big(\boldsymbol{y} - \boldsymbol{F}(\boldsymbol{\beta}_k)\big)$$

$$\Leftrightarrow \qquad \Delta\boldsymbol{\beta} = [J^T J]^{-1}\ J^T\big(\boldsymbol{y} - \boldsymbol{F}(\boldsymbol{\beta}_k)\big)$$

Thus,

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k + \Delta\boldsymbol{\beta} = \boldsymbol{\beta}_k + [J^T J]^{-1}\ J^T\big(\boldsymbol{y} - \boldsymbol{F}(\boldsymbol{\beta}_k)\big)$$

Aside: notice that if the sum of squares had a unique solution (by extension, J was an invertible matrix) then

$$\begin{aligned}[J^T J]^{-1}\ J^T &= J^{-1}(J^T)^{-1}J^T \\ &= J^{-1}\end{aligned}$$

Then our expression for $\boldsymbol{\beta}_{k+1}$ would reduce to exactly the result guaranteed by the Newton-Raphson method:

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k + J^{-1}\big(\boldsymbol{y} - \boldsymbol{F}(\boldsymbol{\beta}_k)\big)$$

Had we proceeded with less rigor, we could have begun with the Newton-Raphson method and simply replaced J^{-1} with the Moore-Penrose matrix inverse: $[J^T J]^{-1}\ J^T$, the most widely used generalization of the matrix inverse[8], also known as the matrix pseudoinverse.

Note that random oscillations are to be expected with iterative techniques so we will typically employ a damping coefficient $\alpha < 1$ to maximize likelihood of convergence as such:

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k + \alpha\Delta\boldsymbol{\beta}$$

4. Convergence of Nonlinear Iterative Methods

Due to vast scope of requisite background required to rigorously prove convergence, we will be using a geometric approach drawing heavily from Ake Bjorck[1]. To guarantee convergence we must assume existence of the Jacobian of the residual vector and that the elements of the residual vector are all twice-differentiable. Whilst deriving the Gauss-Newton method we found that the first derivative of $f(\boldsymbol{\beta}) = \boldsymbol{r}^T\boldsymbol{r}$ is

$$\nabla f(\boldsymbol{\beta}) = J(\boldsymbol{\beta})^T\boldsymbol{r}(\boldsymbol{\beta})$$

We assumed that for a critical point $\boldsymbol{\beta}^*$ to be a local minimum of $f(\boldsymbol{\beta})$, it was necessary that

$$\nabla f(\boldsymbol{\beta}^*) = J(\boldsymbol{\beta}^*)^T \boldsymbol{r}(\boldsymbol{\beta}^*) = 0$$

We will now establish that a critical point $\boldsymbol{\beta}^*$ is a local minimum of $f(\boldsymbol{\beta})$ using the geometric approach used by Wedin[7]. Using a similar approach in finding the first derivative of $f(\boldsymbol{\beta})$ we find that the second derivative is then equal to,

$$\nabla^2 f(\boldsymbol{\beta}) = J(\boldsymbol{\beta})^T J(\boldsymbol{\beta}) + \sum_{i=1}^{n} r_i(\boldsymbol{\beta}) \nabla^2 \boldsymbol{r}_i(\boldsymbol{\beta})$$

If we assume that the Jacobian has full-column rank (all columns are linearly independent) then we can rewrite the above expression (removing parameter designation for clarity) as,

$$\nabla^2 f(\boldsymbol{\beta}) = J^T \left([J^T J]^{-1}\ J^T - ||\boldsymbol{r}|| [(J^T J)^{-1}\ J^T]^T \sum_{i=1}^{n} (-\frac{\boldsymbol{r}}{||\boldsymbol{r}||} \nabla^2 \boldsymbol{r}_i(\boldsymbol{\beta})) [(J^T J)^{-1}\ J^T] \right) J$$

which we can vastly simplify by defining key components as,

$$\alpha = ||\boldsymbol{r}||$$

$$w_i = -\frac{\boldsymbol{r}}{\alpha}$$

$$D_i^2 = \nabla^2 \boldsymbol{r}_i(\boldsymbol{\beta})$$

$$D_w^2 = \sum_{i=1}^{n} w_i D_i^2$$

$$J^* = [J^T J]^{-1}\ J^T$$

We can then succinctly write the second derivative of f as,

$$\nabla^2 f(\boldsymbol{\beta}) = J^T (I - \alpha {J^*}^T D_w^2 J^*) J$$

The following[1] uses results and definitions beyond the scope of this paper from differential geometry to establish a *necessary* condition for $\boldsymbol{\beta}^*$ to be a critical point.

We interpret minimizing $f(\boldsymbol{\beta})$ as finding the closest point to the origin of the surface $\boldsymbol{r}(\boldsymbol{\beta})$. From what we derived $\nabla^2 f(\boldsymbol{\beta})$ to be, observe that $K = {J^*}^T D_w^2 J^*$ is a symmetric matrix defined to be the *normal curvature matrix* of our residual surface $\boldsymbol{r}(\boldsymbol{\beta})$ with *principal radii of curvature* equal to $\frac{1}{\lambda_i}$, the inverses of the eigenvalues of K with respect to the normal $w_i = -\frac{\boldsymbol{r}}{\alpha}$. If the Jacobian matrix at our critical point $J^*(\boldsymbol{\beta}^*)$ is of full column rank, then $J^T(I - \alpha K)J$ must be positive definite at the critical point $\boldsymbol{\beta}^*$ i.e. its eigenvalues are all positive, and that $\boldsymbol{\beta}^*$ is a local minimum if and only if $1 - \alpha\lambda_1 > 0$, where λ_1 is the first eigenvalue of K.

It can be shown[1] that if $1-\alpha\lambda_1 \leq 0$ then $f(\boldsymbol{\beta}^*)$ has a *saddle point* at $\boldsymbol{\beta}^*$, if $1-\alpha\lambda_n < 0$ then $f(\boldsymbol{\beta}^*)$ has a local *maximum* at $\boldsymbol{\beta}^*$, and that (critically) if $1-\alpha\lambda_1 > 0$, then $f(\boldsymbol{\beta}^*)$ has a local *minimum* at $\boldsymbol{\beta}^*$.

Again, with methods beyond the scope of this paper, it can be shown[3] that at the critical point $\boldsymbol{\beta}^*$, $||\boldsymbol{r}|| < \frac{1}{\lambda_1}$ which implies that $1-\alpha\lambda_1 = 1 - \frac{||\boldsymbol{r}||}{\frac{1}{\lambda_1}} > 0$, which is consistent with the conditions that $\boldsymbol{\beta}^*$ is indeed a critical point.

5. A Worked Application in Curve Fitting

We demonstrate the utility of the method with a worked example. We generate 100 data points by adding random error to a known sine curve,

$$y = (1.5)\sin(2x+4) + 0.8$$

We will then attempt to fit a regression curve of the form:

$$\hat{y} = \beta_1 \sin(\beta_2 x + \beta_3) + \beta_4$$

The Jacobian matrix and vector of residuals then follow,

$$J = \begin{bmatrix} \sin(\beta_2 x_1 + \beta_3) & \beta_1 x \sin(\beta_2 x_1 + \beta_3) & \beta_1 \sin(\beta_2 x_1 + \beta_3) & 1 \\ \vdots & & \vdots & \vdots \\ \sin(\beta_2 x_n + \beta_3) & \beta_1 x \sin(\beta_2 x_n + \beta_3) & \beta_1 \sin(\beta_2 x_n + \beta_3) & 1 \end{bmatrix}$$

$$\boldsymbol{r} = \begin{bmatrix} y_1 - \hat{y}(\hat{\boldsymbol{\beta}}, x_1) \\ \vdots \\ y_n - \hat{y}(\hat{\boldsymbol{\beta}}, x_n) \end{bmatrix}$$

With a damping coefficient of $\boldsymbol{\alpha = 0.01}$ and an initial guess of $\hat{\boldsymbol{\beta}}_0 = [2\ 2\ 2\ 2]^T$, we arrive at the following vector of estimating parameters (for this particular set of random data; running the program multiple times will yield similar but unique results),

$$\hat{\boldsymbol{\beta}} = \begin{bmatrix} 1.4978 \\ 1.9952 \\ 2.9895 \\ 0.81886 \end{bmatrix}$$

and consequently, the regression curve,

$$\hat{y} = (1.4978)\sin\big((1.9952)x + 2.9895\big) + 0.81886$$

Compare this to the true equation with no random error,

$$y = (1.5)\sin(2x+4) + 0.8$$

and observe the clear convergence to a best-fit curve.

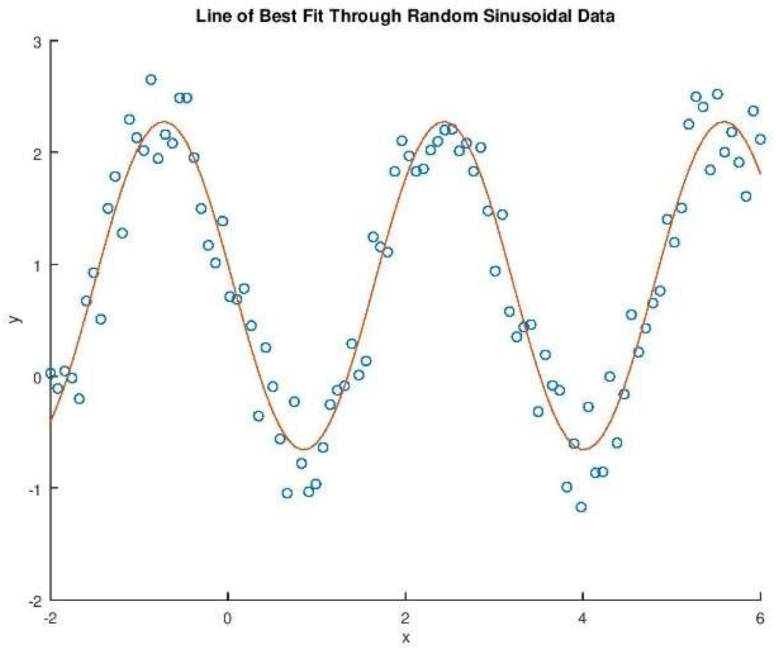

Figure 1. Regression through sinusoidal data with curve of form $\hat{y} = \beta_1 \sin(\beta_2 x + \beta_3) + \beta_4$

6. MATLAB/GNU Octave Code

```
% Implentation is Octave code but is compatible w/ MATLAB

clc;clear all;close all;format shortG; % Formatting

n = 100; % Number of data points

% Generates data points with random errors based on known sine curve
x = linspace(-2,6,n);
y = 1.5*sin(2*x+3)+0.8;
randErrors = (rand(n,1)'-0.5);
y = y + randErrors;
% ----------------------------------------------------------------

b = [2, 2, 2, 2]'; % Starting guess for beta parameters

% Iterates through
```

```
for i=1 : 400
  J = [sin(b(2)*x + b(3)); % Jacobian matrix construction
      (b(1)*x.*cos(b(2)*x + b(3)));
      (b(1)*cos(b(2)*x + b(3)));
      ones(n,1)']';

  r = (y - (b(1)*sin(b(2)*x + b(3)) + b(4)))'; % Residual calculation

  delta_b = inv(J'*J)*J'*r; % Calculating delta_beta

  b = b + 0.01*delta_b; % Iterating step; damping coefficient = 0.01
end
% ------------------------------------------------------------------

vector_of_betas = b % Displays final beta vector
y_reg = b(1)*sin(b(2)*x + b(3)) + b(4);

hold
scatter(x, y); % Plots data points
plot(x, y_reg); % Plots regression curve
title("Line of Best Fit Through Random Sinusoidal Data");
xlabel("x");
ylabel("y");
```

7. References

[1] Bjorck, A. (1996). *Numerical Methods for Least Squares Problems*. Society for Industrial and Applied Mathematics.

[2] Bradie, B. (2006). *A Friendly Introduction to Numerical Analysis*. Pearson Education.

[3] Draper, N. R., & Smith, H. (1998). *Applied Regression Analysis*. John Wiley & Sons.

[4] *Gauss-Newton algorithm for nonlinear models*. (2020). *Fourier.eng.hmc.edu*. Retrieved 4 June 2020, from http://fourier.eng.hmc.edu/e176/lectures/NM/node36.html

[5] Given a data set, a., & Jagy, W. (2013). *Given a data set, how do you do a sinusoidal regression on paper? What are the equations, algorithms?*. *Mathematics Stack Exchange*. Retrieved 4 June 2020, from

https://math.stackexchange.com/questions/301194/given-a-data-set-how-do-you-do-a-sinusoidal-regression-on-paper-what-are-the-e

[6] *Newton-Raphson method (multivariate)*. (2020). *Fourier.eng.hmc.edu*. Retrieved 4 June 2020, from http://fourier.eng.hmc.edu/e176/lectures/NM/node21.html

[7] Wedin, P.A. (1974). On the Gauss-Newton method for the nonlinear least squares problems. *Institute for Applied Mathematics*. Working Paper 24.

[8] Stallings, W. T., Boullion, T. L. (1972). The Pseudoinverse of an r-Circulant Matrix. *Proceedings of the American Mathematical Society.* 34 (2): 385–88.

[9] *Taylor series expansion*. (2020). *Fourier.eng.hmc.edu*. Retrieved 4 June 2020, from http://fourier.eng.hmc.edu/e176/lectures/NM/node45.html