Least-Squares Estimation of Nonlinear Parameters

Saumya Shah*, *20171193, IIIT Hyderabad,* **Anoushka Vyas***, *20171057, IIIT Hyderabad*

***Abstract*—The term paper is written as part of the course *Signal Detection and Estimation Theory* of *Team 5*. The survey describes various algorithms for least-squares estimation for nonlinear parameters as well as some experiments are done to validate the theoretical observations.**

I. Introduction

Many of the least-squares methods for nonlinear parameter estimation are based on expansion of Taylor series and corrections calculated by assuming a linear model and excluding second degree terms. These methods have problems of divergence of the successive iterates. Various modifications of the gradient descent method have also been introduced but they face the issue of slow convergence. As a result, a new method based on *maximum neighbourhood* method is developed which combined the Taylor series expansion and the gradient descent method known as the Levenberg-Marquardt method.

The term paper is organised in the following way, section II describes the problem statement mathematically, section III explains the two baselines- Gauss-Newton method and Gradient Descent method, section IV explains the Levenberg-Marquardt method, section V has the applications of least-squares methods, section VI explains in detail our experimental setup, figures and result tables, section VII has the observations and theoretical analysis, section VIII concludes our term paper, section IX contains the link to the GitHub repository which has the code for our experiments and the appendices A, B, C have the proves of the various theorems used in section IV.

II. Problem Definition

The model to be fitted to the data is defined as

$$E(y) = f(x_1, x_2, ..., x_m; b_1, b_2, ..., b_k) = f(\mathbf{x}, \mathbf{b}), \tag{1}$$

where $x_1, x_2, ..., x_m$ are independent variables, $b_1, b_2, ..., b_k$ are the k parameters to estimate, $E(y)$ is the expected value of the dependent variable y.

To define the objective function, let the data points be denoted by

$$(Y_i, X_{1i}, X_{2i}, ..., X_{mi}), \quad i = 1, 2, ..., n. \tag{2}$$

The objective function to minimize to get the k parameters is given as

$$\Phi = \sum_{i=1}^{n} [Y_i - \hat{Y}_i]^2 = ||\mathbf{Y} - \hat{\mathbf{Y}}||^2, \tag{3}$$

where $\hat{Y}_i$ is the value of predicted y at the ith data point.

III. Previous Algorithms

A. Gauss-Newton Method

The method is similar to expanding f in a Taylor series as follows

$$\langle Y(\mathbf{X_i}, \mathbf{b} + \boldsymbol{\delta_t})\rangle = f(\mathbf{X_i}, \mathbf{b}) + \sum_{j=1}^{k} (\frac{\partial f_i}{\partial b_j})(\delta_t)_j, \tag{4}$$

this can also be written in vectorized form as

$$\langle \mathbf{Y} \rangle = \mathbf{f_0} + \mathbf{P}\boldsymbol{\delta_t}. \tag{5}$$

The vector $\boldsymbol{\delta_t}$ is a small correction to $\mathbf{b}$, which can be found by least-squares method of setting $\frac{\partial \langle \Phi \rangle}{\partial \delta_j} = 0$, for all j. This can be written as

$$A\boldsymbol{\delta_t} = \mathbf{g}, \tag{6}$$

where

$$A^{[k \times k]} = P^T P, \tag{7}$$

$$P^{[n \times k]} = (\frac{\partial f_i}{\partial b_j}), \quad i = 1, 2, ..., n; \quad j = 1, 2, ..., k, \tag{8}$$

$$g^{[k \times 1]} = (\sum_{i=1}^{n} (Y_i - f_i) \frac{\partial f_i}{\partial b_j}) = P^T(\mathbf{Y} - \mathbf{f_0}), \quad j = 1, 2, ..., k. \tag{9}$$

The algorithm is implemented by correcting $\mathbf{b}$ by only a small value $\boldsymbol{\delta_t}$ given by equation 6. A step size $K\boldsymbol{\delta_t}$, $0 < K \leq 1$ is used to control the amount of change in $\mathbf{b}$ after $\boldsymbol{\delta_t}$ has specified the direction.

B. Gradient-Descent Method

In this method the correction in $\mathbf{b}$ is in the direction of the negative gradient of Φ. The step size is given as

$$\boldsymbol{\delta_g} = -(\frac{\partial \Phi}{\partial b_1}, \frac{\partial \Phi}{\partial b_2}, ..., \frac{\partial \Phi}{\partial b_k})^T. \tag{10}$$

IV. Proposed Algorithm

A. Theoretical Analysis

To provide theoretical basis, following theorems were established. Let $\lambda \geq 0$ be arbitrary and let $\boldsymbol{\delta_l}$ satisfy the equation

$$(A^{(r)} + \lambda^{(r)} I)\boldsymbol{\delta_l}^{(r)} = \mathbf{g}^{(r)}, \tag{11}$$

Then

1) *Theorem 1:* $\boldsymbol{\delta_l}$ minimizes Φ on the sphere whose radius $||\boldsymbol{\delta}||$ satisfies
$$||\boldsymbol{\delta}||^2 = ||\boldsymbol{\delta_l}||^2. \tag{12}$$
2) *Theorem 2:* $||\boldsymbol{\delta_l}(\lambda)||$ decreases to zero monotonically as $\lambda \to \infty$.
3) *Theorem 3:* $\boldsymbol{\delta_l}$ rotates from $\boldsymbol{\delta_t}$ to $\boldsymbol{\delta_g}$ monotonically as $\lambda \to \infty$.

*** indicates equal contribution**

B. Scale of Measurement

The properties of the gradient method are invariant under linear transformation of $\mathbf{b}$-space. However, they are not scale invariant. We should scale the $\mathbf{b}$-space in units of the standard deviations of the derivatives $\frac{\partial f_i}{\partial b_j}$, taken over the sample points $i = 1, 2, ..., n$. This choice of scale transforms A into the matrix of simple correlation coefficients among the $\frac{\partial f_i}{\partial b_j}$. This also improves the numerical aspects of computing procedures. We define a scaled matrix A^* and a scaled vector g^*:

$$A^* = a^*_{jj'} = \frac{a_{jj'}}{\sqrt{a_{jj}}\sqrt{a_{j'j'}}}, \tag{13}$$

and

$$g^* = (g^*_j) = (\frac{g_j}{\sqrt{a_{jj}}}), \tag{14}$$

and solve for the Taylor series correction using

$$A^* \boldsymbol{\delta_t}^* = \boldsymbol{g}^*. \tag{15}$$

Then

$$\delta_j = \frac{\delta^*_j}{\sqrt{a_{jj}}}. \tag{16}$$

C. Construction of the Algorithm

From observation, $\boldsymbol{\delta_t}$ is almost $\pi/2$ away from $\boldsymbol{\delta_g}$ in most problems because of the severe elongation of surface Φ, which makes the Gauss-Newton method slow. An idea to improve it is to interpolate between $\boldsymbol{\delta_t}$ and $\boldsymbol{\delta_g}$. In order to minimize Φ locally, equation for r^{th} iteration can be constructed as

$$(A^{*(r)} + \lambda^{(r)} I)\boldsymbol{\delta_l}^{*(r)} = \boldsymbol{g}^{*(r)}. \tag{17}$$

Once above equation is solved for $\boldsymbol{\delta_l}^{*(r)}$, $\boldsymbol{\delta_l}^{(r)}$ can be obtained using 16. The new trial vector is given by

$$\mathbf{b}^{(r+1)} = \mathbf{b}^{(r)} + \boldsymbol{\delta_l}^{(r)}. \tag{18}$$

It is necessary to select $\lambda^{(r)}$ such that

$$\Phi^{(r+1)} < \Phi^{(r)}. \tag{19}$$

At each iteration, we need to minimize Φ in the maximum neighbourhood over which the linearized function will adequate representation of the nonlinear function. We need to choose small value of $\lambda^{(r)}$ when better convergence is required. This is especially useful in the later stages of the convergence, when guesses are near minimum and we want linear expansion of the model to be a good approximation over only a very small region. The value of λ should be sufficiently large to satisfy 19. However, if the value of λ is too large, it would make the method work like steepest descent i.e. rapid initial progress followed by increasingly slower progress. Based on these considerations, the proposed iterative algorithm is as follows:

1) Let $v > 1$, and initialize $\lambda^{(0)} = 10^{-2}$.
2) Compute $\Phi(\lambda^{(r-1)}/v)$ and $\Phi(\lambda^{(r-1)})$.
    a) If $\Phi(\lambda^{(r-1)}/v) \leq \Phi^{(r)}$, let $\lambda^{(r)} = \lambda^{(r-1)}/v$.
    b) If $\Phi(\lambda^{(r-1)}/v) > \Phi^{(r)}$ and $\Phi(\lambda^{(r-1)}) \leq \Phi^{(r)}$, let $\lambda^{(r)} = \lambda^{(r-1)}$.
    c) If $\Phi(\lambda^{(r-1)}/v) > \Phi^{(r)}$ and $\Phi(\lambda^{(r-1)}) > \Phi^{(r)}$, increase λ by successive multiplication by v until for some smallest w, $\Phi(\lambda^{(r-1)}v^w) \leq \Phi^{(r)}$. Let $\lambda^{(r)} = \lambda^{(r-1)}v^w$.
3) Set $\mathbf{b}^{(r)} \leftarrow \mathbf{b}^{(r-1)} + \boldsymbol{\delta_l}$, where $\boldsymbol{\delta_l}$ is obtained using 17 with $\lambda = \lambda^{(r)}$.
4) Set $r \leftarrow r + 1$.
5) Repeat steps 2 through 4.

V. Applications

A. Curve and Surface Fitting

In computer vision, robotics, and geometric modelling it is common to derive an implicit representation of an object from raw shape data obtained with an imaging or touch sensing system. Such an implicit representation often takes the form of the zero set of a polynomial of even degree. In surface fitting, a family of quartic polynomials with stably bounded zero sets is generally considered. The effectiveness of representation and computational efficiency of fitting are considered while choosing the family. The parameters to be determined are polynomial coefficients using a least-squares method. For example, minimizing the sum of squares of the distances from individual data points to the polynomial surface. Let's say, we need to fit a surface over n data points $p_1, ..., p_n$ in R^2 or R^3. We consider the family of polynomial curves/surfaces of form

$$f(x, a_0, ..., a_k) = 0, \tag{20}$$

where $x = (x, y)$ or (x, y, z) and $a_0, ..., a_k$ are the coefficients. Let the distance from p_i to the surface 20 be d_i, where $i = 1, ..., n$. The fitting problem can then be modeled in a least-squares fashion as

$$\min_{a_0, ... a_k} \sum_{i=0}^{k} d_i^2 \tag{21}$$

$d_i, 1 \leq i \leq n$ cannot be determined until $a_0, ..., a_k$ are known, which according to 21 depends on the knowledge of d_i. To get out of this chicken-and-egg situation, we approximate d_i as follows. Let q_i be the closest point to p_i on the surface to be determined. Obviously, $f(q_i; a_0, ..., a_k) = 0$. The best fitting f will place q_i close enough to p_i. So value of f can be approximated at q_i using Taylor's series at p_i, discarding all terms of the second order and above

$$f(q_i; a_0, ..., a_k) \approx f(p_i, a_0, ..., a_k) + \nabla f(p_i; a_0, ..., a_k).(q_i - p_i). \tag{22}$$

As the left hand side of the above equation is zero, we have

$$\nabla f(p_i; a_0, ..., a_k).(q_i - p_i) \approx -f(p_i; a_0, ...a_k). \tag{23}$$

As q_i and p_i are very close to each other, the vector $q_i - p_i$ and the gradient $\nabla f(p_i; a_0, ...a_k)$ are nearly parallel. The above equation yields an approximation

$$d_i = ||q_i - p_i|| \approx \frac{|f(p_i; a_0, ...a_k)|}{||\nabla f(p_i; a_0, ..., a_k)||}. \tag{24}$$

Substituting above approximation in 21, the fitting problem can be reformulated as

$$\min_{a_0, ..., a_k} \frac{f^2(p_i; a_0, ..., a_k)}{||\nabla f(p_i; a_0, ..., a_k)||^2}. \tag{25}$$

B. Surface Patch Reconstruction

A robotic hand can reconstruct an unknown surface patch by touch. The goal is to track along three concurrent curves on the surface while collecting tactile data points (x_k, y_k, z_k), $1 \leq k \leq n$, in the meantime. Each curve represents the intersection of patch with a separate plane (referred to as the sampling plane) within which the tracking motion is presently constrained. Let p be the intersection point of the three curves. By fitting a parabola to the data points along each curve, its curvature at p is estimated. The surface's normal curvature at p in the tangent direction of this curve is the product of the (estimated) curvature with the cosine of the angle between the sampling plane and the normal plane containing the tangent direction. Thus three normal curvatures are obtained. From these curvature and the relative orientations of the corresponding tangent vectors at p, we solve for the principal curvatures k_1 and k_2 and the Darboux frame at the point p. Under this frame, the surface patch locally takes the form

$$z(x, y) = \frac{1}{2}(k_1 x^2 + k_2 y^2) + \sum_{3 \leq i+j \leq d} a_{ij} x^i y^j, \tag{26}$$

Here the terms of degree greater than two are added to describe a larger area of the surface. The coefficients of the polynomial gathered into a vector a, are determined in a least-squares sense as

$$\min_a f(a) \tag{27}$$

where

$$f(a) = \frac{1}{n} \sum_{k=1}^{n} (z(x_k, y_k) - z_k)^2. \tag{28}$$

VI. Experimental Setup

For the purpose of analysis and comparison, we implemented the Gradient Descent method, Gauss-Newton method and Levenberg-Marquardt method. To compare the algorithms different types of experimental setups are created which are described below.

A. Synthetic Dataset

The experiments are performed by considering f as a gaussian function

$$f(x) = a * \frac{1}{\sqrt{2\pi\sigma^2}} e^{(-\frac{1}{2}(\frac{x-\mu}{\sigma})^2)}, \tag{29}$$

where $a = 10$ is the scaling factor, $\mu = 0$ is the mean and $\sigma = 20$ is the standard deviation. These three are the nonlinear parameters to be estimated.

B. Experiments

- **Baseline:** The experiment is performed for initilization of all one, learning rate of 0.01 and 100 observation points.
- **Experiment 1:** In this experiment the initialization of the parameters is changed to $30, 35, 40$.
- **Experiment 2:** The number of observation points are changed to 20.
- **Experiment 3:** A gaussian noise of $\mu = 0.1$ and $\sigma = 0.1$ is added to the observations.

C. Figures

The following figures are plotted as a part of analysis:

- Groundtruth Gaussian
- Predicted Gaussian
- Objective function ($||\Phi||^2$) vs number of iterations

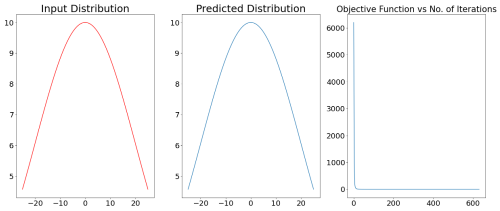

Fig. 1: Baseline plots for the Gradient Descent Method

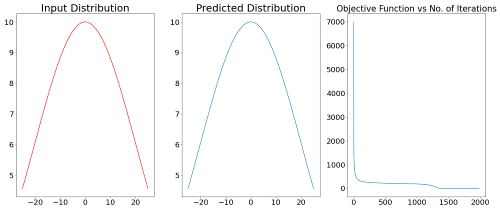

Fig. 2: Experiment-1 plots for the Gradient Descent Method

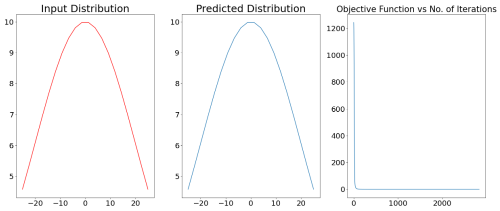

Fig. 3: Experiment-2 plots for the Gradient Descent Method

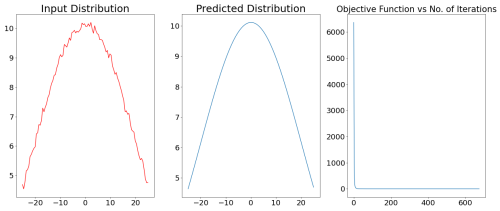

Fig. 4: Experiment-3 plots for the Gradient Descent Method

VII. Analysis

- The predictions made by Gauss-Newton and Levinberg-Marquardt methods are better as compared to Gradient Descent method (evident in **TABLE** II) as the gradient approximation involves linearly approximated second order terms.
- Unlike Gradient Descent, Gauss-Newton and Levenberg-Marquardt methods have a normalisation proportional

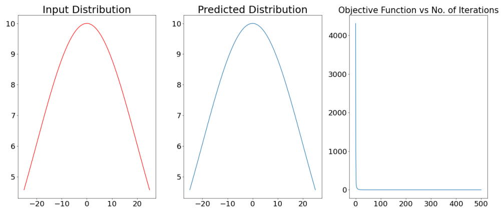

Fig. 5: Baseline plots for the Gauss-Newton Method

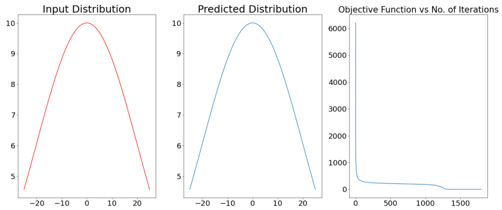

Fig. 6: Experiment-1 plots for the Gauss-Newton Method

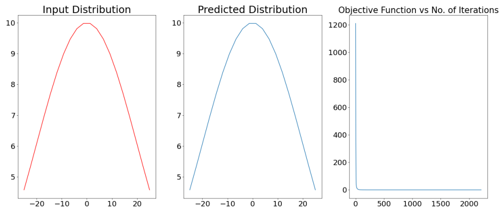

Fig. 7: Experiment-2 plots for the Gauss-Newton Method

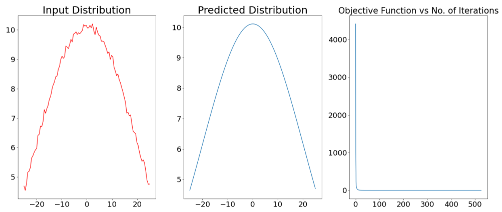

Fig. 8: Experiment-3 plots for the Gauss-Newton Method

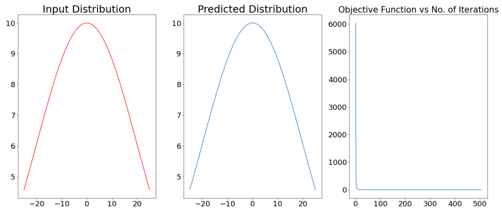

Fig. 9: Baseline plots for the Levenberg-Marquardt Method

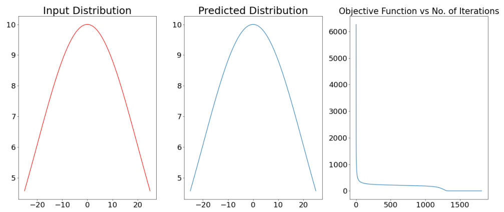

Fig. 10: Experiment-1 plots for the Levenberg-Marquardt Method

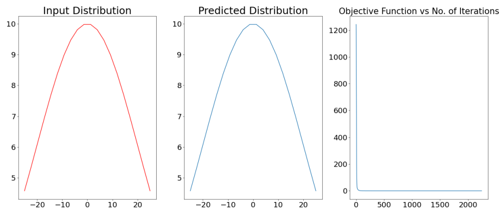

Fig. 11: Experiment-2 plots for the Levenberg-Marquardt Method

to the gradient magnitude. So very large gradients are normalised such that they do not cause oscillations and very small gradients are boosted for updates.

- Gauss-Newton has a normalisation of magnitude of gradient, whereas Levenberg-Marquardt has a normalisation of (λ + magnitude of gradient). For some cases (depending on whether λ is greater than or less than magnitude of gradient) it can wrongly increase or decrease gradients, which leads to more number of iterations to converge (shown in **TABLE I**).
- The number of iterations required to converge is in the order:
  Gauss-Newton $>$ Levenberg-Marquardt $>$ Gradient Descent.
- When we make a different initialisation, we need more iterations to converge and this also involves some sudden drops in the cost function versus iteration maps. That is because here we are approximating an exponential function and our initialization is far away. So initially, predictions values will come closer to ground truth rapidly.
- Noisy samples have almost no effect on rate of convergence but it does have on the cost value and subsequently the predictions obtained, which are little off from the ground truth values.

VIII. Conclusion

The Levenberg-Marquardt method combines the property of Gradient Descent methods ability to converge from an initial guess and the property of Gauss-Newton method to converge to the solution faster after the neighbourhood values are reached but is able to eliminate the shortcomings of both algorithms which is slow convergence and divergence from the solution.

Algorithm	Baseline	Experiment 1	Experiment 2	Experiment 3
Gradient Descent	631	1985	2835	682
Gauss-Newton	500	1797	2212	548
Levenberg-Marquardt	506	1814	2244	524

TABLE I: Number of iteration required by the algorithms to converge to minimum

	Baseline			Experiment 1			Experiment 2			Experiment 3		
Algorithms	a	μ	σ	a	μ	σ	a	μ	σ	a	μ	σ
Gradient Descent	10.00	9.48e-14	20	10.00	-9.41e-14	20.00	10.00	3.17e-14	20.00	10.08	0.03	20.14
Gauss-Newton	10.00	7.76e-14	20	10.00	-7.75e-14	20.00	10.00	2.60e-14	20.00	10.08	0.03	20.14
Levenberg-Marquardt	10.00	7.79e-14	20	10.00	-7.81e-14	20.00	10.00	2.61e-14	20.00	10.08	0.03	20.14

TABLE II: Estimated parameters of gaussian where a is the scaling factor, μ is the mean, and σ is the standard deviation

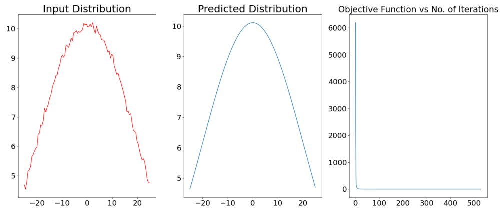

Fig. 12: Experiment-3 plots for the Levenberg-Marquardt Method

IX. Code

The source code of our experiments and theoretical analysis can be found in the following GitHub repository [Link].

Appendix A
Proof of Theorem 1

In order to find $\boldsymbol{\delta}$ that minimizes

$$\Phi = ||\mathbf{Y} - \mathbf{f_0} - P\boldsymbol{\delta}||^2, \tag{30}$$

under the constraint

$$||\boldsymbol{\delta}||^2 = ||\boldsymbol{\delta_l}||^2, \tag{31}$$

Lagrange method is used where the final equation is given as

$$u(\boldsymbol{\delta}, \lambda) = ||\mathbf{Y} - \mathbf{f_0} - P\boldsymbol{\delta}||^2 + \lambda(||\boldsymbol{\delta}||^2 - ||\boldsymbol{\delta_l}||^2), \tag{32}$$

and λ is a Lagrange multiplier.

After taking the derivatives, the equation that solves $\boldsymbol{\delta}$ is given by

$$(P^TP + \lambda I)\boldsymbol{\delta} = P^T(\mathbf{Y} - \mathbf{f_0}), \tag{33}$$

and after premultiplying 33 by $(P^TP)^{-1}$

$$\boldsymbol{\delta} = (P^TP)^{-1}P^T(\mathbf{Y} - \mathbf{f_0}) - (P^TP)^{-1}\lambda\boldsymbol{\delta} \tag{34}$$

and substituting this in the Lagrange derivative will give the answer. Thus, it can be seen that equations 11 and 33 are identical. Thus, this stationary point is actually a minimum.

Appendix B
Proof of Theorem 2

As matrix A is symmetric and positive definite, it can be diagonalized as $S^TAS = D$, S is orthogonal and D has positive diagonal elements. Putting this in 11

$$\boldsymbol{\delta_0} = S(D + \lambda I)^{-1}S^T\mathbf{g} \tag{35}$$

Let $\mathbf{v} = S^T\mathbf{g}$ then,

$$\begin{aligned} ||\boldsymbol{\delta_l}(\lambda)||^2 &= \mathbf{g}^TS(D + \lambda I)^{-1}S^TS(D + \lambda I)^{-1}S^T\mathbf{g} \\ &= \mathbf{v}^T[(D + \lambda I)^2]^{-1}\mathbf{v} \\ &= \sum_{j=1}^{k} \frac{v_j^2}{(D_j + \lambda)^2} \end{aligned} \tag{36}$$

which clearly is a decreasing function in λ.

Appendix C
Proof of Theorem 3

From the above theory $\boldsymbol{\delta_g} = \mathbf{g}$. Let γ be the angle between $\boldsymbol{\delta_l}$ and $\boldsymbol{\delta_g}$, then

$$\begin{aligned} cos\gamma &= \frac{\boldsymbol{\delta}^T\mathbf{g}}{(||\boldsymbol{\delta}||)(||\mathbf{g}||)} \\ &= \frac{\mathbf{v}^T(D + \lambda I)^{-1}\mathbf{v}}{(\mathbf{v}^T[(D + \lambda I)^2]^{-1}\mathbf{v})^{1/2}(\mathbf{g}^T\mathbf{g})^{1/2}} \\ &= \frac{\sum_{j=1}^{k} \frac{v_j^2}{(D_j+\lambda)}}{[\sum_{j=1}^{k} \frac{v_j^2}{(D_j+\lambda)^2}]^{1/2}(\mathbf{g}^T\mathbf{g})^{1/2}}. \end{aligned} \tag{37}$$

After differentiating and simplifying

$$\frac{dcos\gamma}{d\lambda} = \frac{[\sum_{j=1}^{k} v_j^2\Pi_{1j}][\sum_{j=1}^{k} v_j^2\Pi_{3j}] - [\sum_{j=1}^{k} v_j^2\Pi_{2j}]^2}{[\sum_{j=1}^{k} \frac{v_j^2}{(D_j+\lambda)^2}]^{3/2}[\Pi_{j=1}^{k}(D_j + \lambda)^2]^2(\mathbf{g}^T\mathbf{g})^{1/2}} \tag{38}$$

where $\Pi_{1j} = \Pi^k_{j'=1,j'\neq j}\ (D_{j'}\ + \lambda)$, $\Pi_{2j} = \Pi^k_{j'=1,j'\neq j}\ (D_{j'}\ + \lambda)^2$, $\Pi_{3j} = \Pi^k_{j'=1,j'\neq j}\ (D_{j'}\ + \lambda)^3$. The sign of the derivative depends on the numerator. The numerator can be changed to

$$[\sum_{j=1}^{k} v_j^2\Pi_{1j}][\sum_{j=1}^{k} v_j^2\Pi_{3j}] - [\sum_{j=1}^{k}(v_j\Pi_{1j}^{1/2})(v_j\Pi_{3j}^{1/2})]^2. \tag{39}$$

By Cauchy Schwarz Inequality, 39 is positive. Thus, the derivative is always positive. As a result, γ is monotonically decreasing function of λ.

For very large value of $\lambda(\infty)$, the equatio 11 gives $\mathbf{g}/\lambda$, the angle between $\boldsymbol{\delta}_\mathbf{0}$ and $\mathbf{g}$ approaches zero. When $\lambda = 0$, the vectors $\boldsymbol{\delta}_\mathbf{0}$ and $\mathbf{g}$ meet at some angle between 0 and $\pi/2$. This analysis also shows γ is a continuous monotone decreasing function of λ.

References

[1] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. Journal of the Society for Industrial and Applied Mathematics, 11(2):431–441, 1963.

[2] M. M. Blane, Z. Lei, H. Civi, D. B. Cooper. The 3L algorithm for fitting implicit polynomial curves and surfaces to data. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(3):298–313, 2000

[3] Y.-B. Jia and J. Tian. Surface patch reconstruction from “one-dimensional” tactile data. IEEE Transactions on Automation Science and Engineering, 7(2):400–407, 2010.