# CISC 371 Class 10

## Nonlinear Least Squares and the Levenberg-Marquardt Algorithm

Texts: [1] pp. 254–262; [2] pp. 67–72

*Main Concepts:*

- *Nonlinear least squares*
- *Linear approximation*
- *Gauss-Newton method is altered descent*
- *Levenberg-Marquardt algorithm is altered descent*

**Sample Problem, Geographic Location:** How can we solve the GPS localization equations for an ideal receiving station?

Some problems that require optimization involve multiple independent data, or "readings", where there is a nonlinear model function for each reading. An example, which we will examine more closely in the class, is GPS localization. Each "reading" is the time delay between a receiver and each satellite in the constellation for which a broadcast message has been acquired. The nonlinear function is the distance between the receiver and the satellite that broadcast the acquired message.

In general, such problems can be managed by using a nonlinear least squares (NLS) formulation. We will write the $i^{\text{th}}$ reading as $y_i$ and the corresponding function as $g_i(\vec{w})$. For $m$ readings, the presumption is that an optimal point $\vec{w}^*$ will be evaluated to a scalar that is approximately the corresponding reading. This is equivalent to the presumption that the residual error between the that is, for the $i^{\text{th}}$ reading is zero, so

$$\begin{array}{rrcl} & g_i(\vec{w}^*) & \approx & y_i \\ \equiv & g_i(\vec{w}^*) - y_i & \approx & 0 \\ \equiv & r_i(\vec{w}^*) & \approx & 0 \end{array} \tag{10.1}$$

Gathering the $m$ readings and the $m$ functions of Equation 10.1 into vectors, the basic presumption is that the residual vector is approximately the zero vector, so

$$\vec{r}(\vec{w}^*) \approx \vec{0} \tag{10.2}$$

Equation 10.2 can be posed as a NLS problem by forming an objective function that is the sum of the squares of the residual errors. We can define such a function as

$$F(\vec{w}) \stackrel{\text{def}}{=} \frac{1}{2}\|\vec{r}(\vec{w})\|^2 \tag{10.3}$$

The optimization problem for Equation 10.3 is to find

$$\vec{w}^* = \underset{\vec{w}\in\mathbb{R}^n}{\operatorname{argmin}} F(\vec{w}) \tag{10.4}$$

A solution to Equation 10.4 is a least-squares solution to the original nonlinear problem. One way that we can compute a minimizer is to use the method of steepest descent. The gradient $\underline{\nabla} F$ and the descent direction $\vec{d}$ are

$$\begin{aligned} \underline{\nabla} F(\vec{w}) &= [\vec{r}(\vec{w}))]^T J_{\vec{g}}(\vec{w}) \\ \vec{d}(\vec{w}) &= -[\underline{\nabla} F(\vec{w})]^T \\ &= -[J_{\vec{g}}(\vec{w})]^T \vec{r}(\vec{w}) \end{aligned} \tag{10.5}$$

A derivation of Equation 10.5 is provided in the extra notes for this class.

Instead of using Equation 10.5, a more common way to solve Equation 10.3 is to locally approximate each nonlinear function $g_i(\vec{w})$ with a linear model, and then to find the direction of steepest descent for the locally linear model.

## 10.1 Gauss-Newton Iteration

The insight into the most commonly used solution of a NLS problem is that any function can be approximated using a Taylor series. Expanding around some point $\vec{w}_0$, the first-order approximation to each function $g_i(\vec{w})$ in Equation 10.1 is

$$g_i(\vec{w}) \approx g_i(\vec{w}_0) + \underline{\nabla} g_i(\vec{w}_0)[\vec{w} - \vec{w}_0] \tag{10.6}$$

The approximated residual error for each reading $y_i$, using the linear approximation of $g_i(\vec{w})$ in Equation 10.6, is

$$q_i(\vec{w}) \stackrel{\text{def}}{=} g_i(\vec{w}_0) + \underline{\nabla} g_i(\vec{w}_0)[\vec{w} - \vec{w}_0] - y_i \tag{10.7}$$

Gathering the residual errors $r_i(\vec{w})$ in Equation 10.7 into a vector, the objective function that minimizes the sum of the squares of the residual errors is

$$f(\vec{w}) \stackrel{\text{def}}{=} \|\vec{q}(\vec{w})\|^2 \tag{10.8}$$

The *Gauss-Newton* iteration, which solves Equation 10.8 by steepest descent with a unit step-size $s = 1$, is

$$\vec{w}_{k+1} = \vec{w}_k - \left[[J_{\vec{g}}(\vec{w}_k)]^T [J_{\vec{g}}(\vec{w}_k)]\right]^{-1} [J_{\vec{g}}(\vec{w}_k)]^T \vec{r}\,(\vec{w}_k) \tag{10.9}$$

The extra notes for this class provide a derivation of Equation 10.9. We can temporarily abbreviate

$$\begin{aligned} J_k &\stackrel{\text{def}}{=} J_{\vec{g}}(\vec{w}_k) \\ \vec{r}_k &\stackrel{\text{def}}{=} \vec{r}(\vec{w}_k) \end{aligned}$$

and dropping the argument $\vec{w}_k$, an equivalent way of writing Equation 10.9 is

$$\begin{aligned} \vec{w}_{k+1} &= \vec{w}_k - [J_k^T J_k]^{-1} J_k^T \vec{r}(\vec{w}_k) \\ &= \vec{w}_k - [J_k^T J_k]^{-1} J_k^T \vec{r}_k \end{aligned} \tag{10.10}$$

Equation 10.10 clarifies the Gauss-Newton iteration as a scaled version of steepest descent, where the symmetric positive [semi-]definite matrix $[J_k^T J_k]^{-1}$ is used in iteration $k$ to scale the direction of steepest descent for the objective function $F(\vec{w})$ in Equation 10.3.

## 10.2 Levenberg-Marquardt Algorithm

The Gauss-Newton iteration of Equation 10.9 may fail to converge if the initial estimate $\vec{w}_1$ is "far" from a local minimizer $\vec{w}^*$, or if the Jacobian matrix $J_{\vec{g}}(\vec{w}_k)$ is rank-deficient. These problems were observed in 1944 by Kenneth Levenberg [3] and were solved, using a matrix-vector formulation, in 1963 by Donald Marquardt [4]. Their proposed modification was to "damp" the Gauss-Newton solution by using a non-negative scalar value, usually written as $\lambda \geq 0$. This iteration is

$$\begin{aligned} \vec{w}_{k+1} &= \vec{w}_k - \left[ [J_{\vec{g}}(\vec{w}_k)]^T [J_{\vec{g}}(\vec{w}_k)] + \lambda I \right]^{-1} J^T \vec{r}(\vec{w}_k) \\ &= \vec{w}_k + [J^T J + \lambda I]^{-1} J^T \vec{r}_k \end{aligned} \tag{10.11}$$

## 10.3 Example – GPS Localization

The Global Positioning System (GPS) works by having satellites that transmit highly accurate timing information [5]. A receiver can sense these transmissions and estimate its 3D location as an optimization problem. To avoid a detail that is related to clock bias, we will assume that the receiver also has a highly accurate timer such as an atomic clock; examples of such a receiver are a ground station or a military craft.

At any instant, the receiver can find the current receiver time from the onboard clock; the receiver can also find the time that a satellite broadcasted a message. The difference between these

times, multiplied by the speed of light $\mathrm{c}$, is a calculation of the *pseudorange* from the receiver to the satellite. The satellite's message includes the 3D location of the satellite at the instant of the broadcast. As the receiver processes the information for the $i^{\text{th}}$ satellite, the relevant values are:

$s_i$**:** time that the $i^{\text{th}}$ satellite sent this message

$t_i$**:** time that the receiver acquired the $i^{\text{th}}$ satellite's message

$y_i$**:** pseudorange for the $i^{\text{th}}$ satellite, the time difference multiplied by speed of light $\mathrm{c}$, so $y_i = (t_i - s_i)\mathrm{c}$

$\vec{x}_i$**:** 3D location of the $i^{\text{th}}$ satellite at time $s_i$

$g_i(\vec{w})$**:** actual distance from the $i^{\text{th}}$ satellite to the receiver, so $g_i(\vec{w}) = \|\vec{w} - \vec{x}_i\|$

Illustrations in Figure 10.1 show a 2D version of the 3D GPS geometry and the receiver's point of view. The receiver knows only that satellites are at an approximate altitude above the surface and at approximately estimated distances.

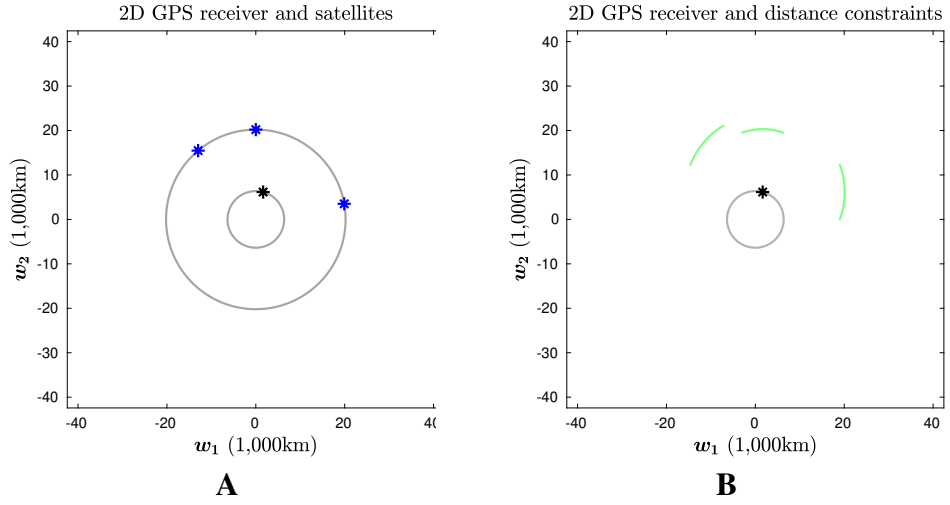


**Figure 10.1:** A 2D version of the GPS problem, with the receiver shown as a black asterisk and three visible satellites shown as blue asterisks. Iterative descent of the function $f_3$ using a damped Newton's Method with backtracking. (A) The receiver is on the Earth's surface, approximated as a circle with a radius of 6,361 km; three satellites have orbits that are approximated as a circle with a radius of 20,200 km. (B) To the receiver, the satellites are known to be at distances that are shown as arcs in green; the receiver knows only the approximate altitude of the satellites.

The functions $g_i(\vec{w})$ and the values $y_i$ can be used to formulate a NLS problem. We can write each distance from the receiver to a satellite as

$$\begin{aligned} g_i(\vec{w}) &= \|\vec{w} - \vec{x}_i\| \\ &= \left(\|\vec{w} - \vec{x}_i\|^2\right)^{1/2} \\ &= \left([\vec{w} - \vec{x}_i]^T[\vec{w} - \vec{x}_i]\right)^{1/2} \\ &= \left([\vec{w}^T - \vec{x}_i^T][\vec{w} - \vec{x}_i]\right)^{1/2} \\ &= \left(\vec{w}^T\vec{w} - 2\vec{x}_i^T\vec{w} + \vec{x}_i^T\vec{x}_i\right)^{1/2} \end{aligned} \tag{10.12}$$

From Equation 10.12, the gradient of each function $g_i(\vec{w})$ can be found using the Chain Rule as

$$\begin{aligned} \underline{\nabla} g_i(\vec{w}) &= 1/2\left(\|\vec{w} - \vec{x}_i\|^2\right)^{-1/2}\underline{\nabla}\left(\vec{w}^T\vec{w} - 2\vec{x}_i^T\vec{w} + \vec{x}_i^T\vec{x}_i\right) \\ &= 1/2\underline{\nabla}\left(\vec{w}^T\vec{w} - 2\vec{x}_i^T\vec{w} + \vec{x}_i^T\vec{x}_i\right)\left(\|\vec{w} - \vec{x}_i\|^2\right)^{-1/2} \\ &= 1/2\left[2\vec{w}^T - 2\vec{x}_i^T\right]\left(\|\vec{w} - \vec{x}_i\|^2\right)^{-1/2} \\ &= [\vec{w} - \vec{x}_i]^T/\|\vec{w} - \vec{x}_i\| \end{aligned} \tag{10.13}$$

For $m$ satellites, the Jacobian matrix for our GPS problem is

$$J_{\vec{g}}(\vec{w}) = \begin{bmatrix} [\vec{w} - \vec{x}_1]^T/\|\vec{w} - \vec{x}_1\| \\ [\vec{w} - \vec{x}_2]^T/\|\vec{w} - \vec{x}_2\| \\ [\vec{w} - \vec{x}_3]^T/\|\vec{w} - \vec{x}_3\| \\ \vdots \\ [\vec{w} - \vec{x}_m]^T/\|\vec{w} - \vec{x}_m\| \end{bmatrix} \tag{10.14}$$

The residual vector for our GPS problem is

$$\vec{r}(\vec{w}) = \vec{g}(\vec{w}) - \vec{y} = \begin{bmatrix} g_1(\vec{w}) - y_1 \\ g_2(\vec{w}) - y_2 \\ g_3(\vec{w}) - y_3 \\ \vdots \\ g_m(\vec{w}) - y_m \end{bmatrix} \tag{10.15}$$

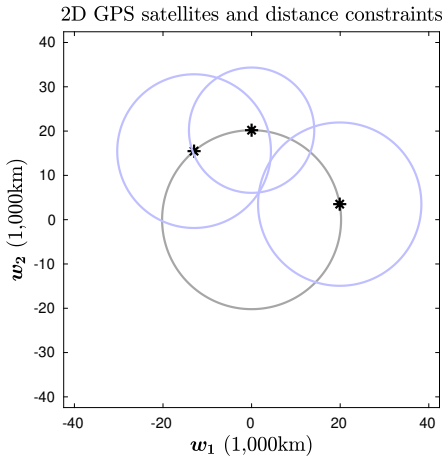


**Figure 10.2:** A 2D version of the GPS problem, with three visible satellites shown as blue asterisks. The receiver is at a known approximate distance from each satellite, so the receiver must be close to some point on each blue circle. A point that is close to the intersection of the three circles will minimize the residual error of the NLS problem.

Equation 10.15 has a simple geometrical interpretation. As illustrated in Figure 10.2, the receiver is at a point that is close to the intersection of circles, each centered at $\vec{x}_i$ with an approximate radius of $y_i$.

The Jacobian matrix of Equation 10.14, plus the residual vector of Equation 10.15, can be used to solve our GPS problem by using the Gauss-Newton iteration or by using the Levenberg-Marquardt Algorithm. The choice of the algorithm, and the selection of a value of $\lambda$, may depend on the accuracy and speed requirements of a particular implementation, among many other factors.

## 10.4 Example: Fermat-Weber Problem

The introductory class for this course used one of Fermat's problems as a motivating example. The problem, stated as a NLS problem, is similar to the GPS problem of Section 10.3. The "anchor" points of Fermat's problem are the points $\vec{x}_i$. The $i^{\text{th}}$ distance function, to any point $\vec{w}$, is Equation 10.12. The only difference between Fermat's problem and the GPS problem is that, in Fermat's problem, there is no "reading" so $y_i = 0$ for all $1 \leq i \leq m$. The objective function simplifies to

$$\begin{aligned} & g_i(\vec{w}) &=& \|\vec{w} - \vec{x}_i\| \\ \text{so} \quad & F_F(\vec{w}) &=& \|\vec{g}(\vec{w})\|^2 \end{aligned} \tag{10.16}$$

The Fermat-Weber problem is closely related to Fermat's problem. From our point of view, the

difference is that in Weber's problem there is a positive "weight" $b_i > 0$ for each reading. We can create a diagonal weighting matrix $B$, which is

$$B = \begin{bmatrix} b_1 & 0 & 0 & \cdots & 0 \\ 0 & b_2 & 0 & \cdots & 0 \\ 0 & 0 & b_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & b_m \end{bmatrix} \tag{10.17}$$

The objective for the Fermat-Weber problem is a substitution of $B$ in Equation 10.17 into the Fermat objective function of Equation 10.16, which is

$$F_W(\vec{w}) \quad = \quad \|B\vec{g}(\vec{w})\|^2 \tag{10.18}$$

The Fermat-Weber problem can also be solved using either the Gauss-Newton iteration or the Levenberg-Marquardt Algorithm.

## Gauss-Newton and Levenberg-Marquardt: Comments

We can observe that, for a full-rank Jacobian matrix, both the matrix in Equation 10.9 and the matrix in Equation 10.11 are symmetric and positive definite. For $\lambda > 0$, the matrix in Equation 10.11 is *always* symmetric and positive definite, regardless of the rank of the Jacobian matrix.

We also observe that both methods are scaled versions of steepest descent. The scaling matrix in the Gauss-Newton iteration is sometimes – *incorrectly* – called a "Hessian" at the point $\vec{w}_k$. Although it may be an approximation to the Hessian matrix, in general $J^T J \neq \underline{\nabla}^2 F$.

The argument $\lambda$ has a critical role in the Levenberg-Marquardt Algorithm. Let us consider the limiting cases. If $\lambda = 0$ then the Levenberg-Marquardt scaling matrix is equal to the Gauss-Netwon scaling matrix. As $\lambda \to \infty$, the Levenberg-Marquardt scaling matrix approaches $\lambda I$, so the inverse approaches $1/\lambda I$; for a large value of $\lambda$, the Levenberg-Marquardt Algorithm is equivalent to the method of steepest descent with a stepsize of $s = 1/\lambda$.

## 10.5 Extra Notes on Nonlinear Least Squares

These extra notes include derivations of Equation 10.5 and Equation 10.9. As a preliminary matter, we can derive the gradient of the squared norm of the difference between a vector $\vec{u}$ and any vector $\vec{v}$ as

$$\begin{aligned} \|\vec{u}-\vec{v}\|^2 &= [\vec{u}-\vec{v}]^T[\vec{u}-\vec{v}] \\ &= [\vec{u}^T-\vec{v}^T][\vec{u}-\vec{v}] \\ &= \vec{u}^T\vec{u}-2\vec{v}^T\vec{u}+\vec{v}^T\vec{v} \\ \text{so} \quad \frac{\partial}{\partial\vec{u}}\|\vec{u}-\vec{v}\|^2 &= \frac{\partial}{\partial\vec{u}}\left(\vec{u}^T\vec{u}-2\vec{v}^T\vec{u}+\vec{v}^T\vec{v}\right) \\ &= 2\vec{u}^T-2\vec{u}^T \\ &= 2[\vec{u}-\vec{v}^T \end{aligned} \tag{10.19}$$

### 10.5.1 Gradient of the NLS Objective Function

The gradient of $F(\vec{w})$ in Equation 10.3 can be written, using the Chain Rule and a substitution into Equation 10.19, as

$$\begin{aligned} \underline{\nabla}F(\vec{w}) &= \frac{1}{2}\left[\frac{\partial F}{\partial\vec{r}}\right]\left[\frac{\partial\vec{r}}{\partial\vec{g}}\right]\left[\frac{\partial\vec{g}}{\partial\vec{w}}\right] \\ &= \frac{1}{2}2[\vec{r}]^T I J_{\vec{g}}(\vec{w}) \\ &= [\vec{r}]^T J_{\vec{g}}(\vec{w}) \end{aligned} \tag{10.20}$$

Equation 10.20 is a derivation of Equation 10.5 above.

### 10.5.2 The Gauss-Newton Iteration

Using a local linear approximation of the given function $g_i(\vec{w})$, the approximated residual error of each term is given in Equation 10.7, which can also be written as

$$\begin{aligned} q_i(\vec{w}) &= g_i(\vec{w}_0) + \underline{\nabla} g_i(\vec{w}_0)[\vec{w} - \vec{w}_0] - y_i \\ &= g_i(\vec{w}_0) + \underline{\nabla} g_i(\vec{w}_0)\vec{w} - \underline{\nabla} g_i(\vec{w}_0)\vec{w}_0 - y_i \\ &= \underline{\nabla} g_i(\vec{w}_0)\vec{w} - \underline{\nabla} g_i(\vec{w}_0)\vec{w}_0 + g_i(\vec{w}_0) - y_i \\ &= \underline{\nabla} g_i(\vec{w}_0)\vec{w} - (\underline{\nabla} g_i(\vec{w}_0)\vec{w}_0 - (g_i(\vec{w}_0) - y_i)) \\ &= \underline{\nabla} g_i(\vec{w}_0)\vec{w} - (\underline{\nabla} g_i(\vec{w}_0)\vec{w}_0 - (r_i(\vec{w}_0))) \\ \text{so} \quad \vec{q}(\vec{w}) &= J_{\vec{g}}(\vec{w}_0)\vec{w} - [J_{\vec{g}}(\vec{w}_0)\vec{w}_0 - \vec{r}(\vec{w}_0)] \end{aligned} \tag{10.21}$$

The objective function $f(\vec{w})$ of Equation 10.8 can be solved explicitly. We can begin by substituting Equation 10.21 for the term $\vec{q}(\vec{w})$, which is

$$\begin{aligned} \vec{w}^* &= \underset{\vec{w} \in \mathbb{R}^n}{\operatorname{argmin}} f(\vec{w}) \\ &= \underset{\vec{w} \in \mathbb{R}^n}{\operatorname{argmin}} \|\vec{q}(\vec{w})\|^2 \\ &= \underset{\vec{w} \in \mathbb{R}^n}{\operatorname{argmin}} \|J_{\vec{g}}(\vec{w}_0)\vec{w} - [J_{\vec{g}}(\vec{w}_0)\vec{w}_0 - \vec{r}(\vec{w}_0)]\|^2 \end{aligned} \tag{10.22}$$

The problem stated in Equation 10.22 is the overdetermined linear equation

$$J_{\vec{g}}(\vec{w}_0)\vec{w}^* \approx [J_{\vec{g}}(\vec{w}_0)\vec{w}_0 - \vec{r}(\vec{w}_0)] \tag{10.23}$$

Equation 10.23 has the normal equation as its solution, which can be written as

$$\begin{aligned} \left[[J_{\vec{g}}]^T[J_{\vec{g}}(\vec{w}_0)]\right]\vec{w}^* &= [J_{\vec{g}}]^T[J_{\vec{g}}(\vec{w}_0)\vec{w}_0 - \vec{r}(\vec{w}_0)] \\ &= [J_{\vec{g}}]^T[J_{\vec{g}}(\vec{w}_0)]\vec{w}_0 - [J_{\vec{g}}]^T\vec{r}(\vec{w}_0) \\ \text{so} \quad \vec{w}^* &= \left[[J_{\vec{g}}]^T[J_{\vec{g}}(\vec{w}_0)]\right]^{-1}[J_{\vec{g}}]^T[J_{\vec{g}}(\vec{w}_0)]\vec{w}_0 \\ &\quad - \left[[J_{\vec{g}}]^T[J_{\vec{g}}(\vec{w}_0)]\right]^{-1}[J_{\vec{g}}]^T\vec{r}(\vec{w}_0) \\ &= \vec{w}_0 - \left[[J_{\vec{g}}]^T[J_{\vec{g}}(\vec{w}_0)]\right]^{-1}[J_{\vec{g}}]^T\vec{r}(\vec{w}_0) \end{aligned} \tag{10.24}$$

Converting Equation 10.24 into an iteration, by substituting $\vec{w}_0$ with $\vec{w}_k$ and $\vec{w}^*$ with $\vec{w}_{k+1}$, produces the Gauss-Newton iteration of Equation 10.9.

End of Extra Notes

## References


[1] Antoniou A, Lu WS: Practical Optimization: Algorithms and Engineering Applications. Springer Science & Business Media, 2007

[2] Beck A: Introduction to Nonlinear Optimization: Theory, Algorithms, and Applications with MATLAB. Siam Press, 2014

[3] Levenberg K: A method for the solution of certain non-linear problems in least squares. *Q Appl Math* 2(2):164–168, 1944

[4] Marquardt DW: An algorithm for least-squares estimation of nonlinear parameters. *SIAM J Appl Math* 11(2):431–441, 1963

[5] Tsui JBY: Fundamentals of Global Positioning System Receivers: A Software Approach, volume 173. John Wiley & Sons, 2005