ECE 174 – Lecture Supplement – SPRING 2009

Gradient-Descent GPS Algorithms

Kenneth Kreutz–Delgado
Electrical and Computer Engineering
Jacobs School of Engineering
University of California, San Diego

VERSION ECE174LS-GPS-S09v1.0

May 20, 2009

Linearization of the True Range Equation. Using the notation defined in the description of the GPS Computer Project we have that $S_\ell = (x_\ell, y_\ell, z_\ell)^T$ = position of ℓ^{th} Satellite, $\ell = 1, \cdots 4$; $S = (x, y, z)^T$ = position of receiver (the Station to be tracked); R_ℓ = *true range* to the receiver located at S from the satellite located at S_ℓ; and $\Delta S_\ell = S - S_\ell$ = satellite-to-receiver distance vector. For a fixed satellite location, S_ℓ, the true range, R_ℓ, is a nonlinear function of the receiver location, S, and is given by,

$$R_\ell(S) \;=\; \| S - S_\ell \| \;=\; \| \Delta S_\ell \| \;=\; \left(\Delta S_\ell^T \; \Delta S_\ell\right)^{\frac{1}{2}}, \quad \ell = 1, \cdots, 4\,. \tag{1}$$

Note that all points S for which $R_\ell(S) = r$ lie on a sphere of radius r centered at the satellite location S_ℓ.

The gradient of $R_\ell(S)$, $\nabla_S R_\ell(S)$, is the *direction of steepest increase of* $R_\ell(S)$. It corresponds to the direction in space for which a unit change in the position S results in the largest increase in the value of $R_\ell(S)$. Some thought provides the insight that it must be the case that

$$\nabla_S R_\ell(S) \;=\; r_\ell(S) \;\triangleq\; \frac{\Delta S_\ell}{R_\ell(S)}\,, \tag{2}$$

where $r_\ell(S)$ is the unit vector which points from the satellite location S_ℓ to the point S. This can be shown rigorously from the fact that

$$\nabla_S R_\ell(S) = \left(\frac{\partial R_\ell(S)}{\partial S}\right)^T$$

where (utilizing vector derivative identities)

$$\begin{aligned}
\frac{\partial R_\ell(S)}{\partial S} &= \frac{\partial \left(\Delta S_\ell^T\ \Delta S_\ell\right)^{\frac{1}{2}}}{\partial S} \\
&= \frac{\partial \left(\Delta S_\ell^T\ \Delta S_\ell\right)^{\frac{1}{2}}}{\partial \Delta S_\ell} \cdot \frac{\partial \Delta S_\ell}{\partial S} \\
&= \frac{1}{2}\left(\Delta S_\ell^T\ \Delta S_\ell\right)^{-\frac{1}{2}} \cdot \frac{\partial \Delta S_\ell^T\ \Delta S_\ell}{\partial \Delta S_\ell} \cdot I \\
&= \frac{1}{2\,R_\ell(S)} \cdot 2\,\Delta S_\ell^T \\
&= \frac{\Delta S_\ell^T}{R_\ell(S)} \ = \ r_\ell^T(S)\,.
\end{aligned}$$

The fact that the linearization of $R_\ell(S)$ is given by

$$\frac{\partial R_\ell(S)}{\partial S} \ = \ r_\ell^T(S) \tag{3}$$

is an important result which will be used below.

The Pseudorange Equation. The pseudorange measurement for satellite ℓ is denoted by y_ℓ, and modelled as

$$y_\ell = R_\ell(S) + b + \nu_\ell\,, \tag{4}$$

where the random noise term ν_ℓ is i.i.d. with p.d.f. $N(0,\sigma^2)$ for each $\ell = 1,2,3,4$. It is also assumed that ν_ℓ is independent of ν_μ for $\ell \neq \mu$. The (constant) systematic *clock bias* error b is caused by an inaccurate clock in the GPS receiver.

The same number of range measurements, $k = 1, \cdots, m$, are taken to each satellite ℓ,

$$y_\ell[k] = R_\ell(S) + b + \nu_\ell[k]\,,\ \ k = 1, \cdots, m\,;\ \ell = 1,2,3,4\,, \tag{5}$$

which we write as a collection of *vector–measurement equations,*

$$\mathbf{y}[k] \ = \ \mathbf{R}(S) + b\,\mathbf{e} + \boldsymbol{\nu}[k]\,,\ \ k = 1, \cdots, m\,, \tag{6}$$

where

$$\mathbf{y}[k] = \begin{pmatrix} y_1[k] \\ \vdots \\ y_4[k] \end{pmatrix}, \quad \mathbf{R}(S) = \begin{pmatrix} R_1(S) \\ \vdots \\ R_4(S) \end{pmatrix}, \quad \mathbf{e} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}, \quad \text{and} \quad \boldsymbol{\nu}[k] = \begin{pmatrix} \nu_1[k] \\ \vdots \\ \nu_4[k] \end{pmatrix}. \tag{7}$$

Now, let us rewrite (6) as

$$\mathbf{y}[k] \;=\; \mathbf{h}(X) + \boldsymbol{\nu}[k]\,, \;\; k = 1, \cdots, m\,, \tag{8}$$

where

$$X = \begin{pmatrix} S \\ b \end{pmatrix} \in \mathbb{R}^4 \quad \text{and} \quad \mathbf{h}(X) = \mathbf{R}(S) + \mathbf{e}\, b \in \mathbb{R}^4\,. \tag{9}$$

Note that for each measurement k, the expression shown in (8) contains four equations and four unknowns. The vector–measurement $\mathbf{y}[k] \in \mathbb{R}^4$ is comprised of range measurements made to all four satellites *at the same time* k.

The Least–Squares Parameter Estimation Problem. For the k–th measurement in (8) define the associated *single–measurement* least–squares loss function,

$$\boldsymbol{\ell}_k(X) = \frac{1}{2}\,\|\mathbf{y}[k] - \mathbf{h}(X)\|^2\,. \tag{10}$$

Because the noise vector $\boldsymbol{\nu}[k]$ has covariance matrix $\Sigma = \sigma^2 I_{4\times 4}$, minimizing the loss function (10) is equivalent to minimizing the weighted least–squares loss function with weighting matrix Σ^{-1},

$$\boldsymbol{\ell}_k(X) = \frac{\sigma^2}{2}\,\|\mathbf{y}[k] - \mathbf{h}(X)\|^2_{\Sigma^{-1}}\,.$$

With $\boldsymbol{\nu}[k] \sim \mathrm{N}(0, \Sigma)$, this means that the estimate for X obtained by minimizing $\boldsymbol{\ell}_k(X)$ yields the maximum likelihood extimate (MLE) of X given the single vector–measurement $\mathbf{y}[k]$.

Now define the *average* least–squares loss function,

$$\bar{\boldsymbol{\ell}}\,(X) = \langle \boldsymbol{\ell}_k(X) \rangle_m = \frac{1}{m}\sum_{k=1}^{m} \boldsymbol{\ell}_k(X)\,. \tag{11}$$

It is straightforward to show that the average loss function (11) is also equal to the *complete–data* least–squares loss function

$$\bar{\boldsymbol{\ell}}\,(X) = \frac{1}{2m}\left\| \begin{pmatrix} \mathbf{y}[1] \\ \vdots \\ \mathbf{y}[m] \end{pmatrix} - \begin{pmatrix} \mathbf{h}(X) \\ \vdots \\ \mathbf{h}(X) \end{pmatrix} \right\|^2 . \tag{12}$$

Minimizing the loss function (12) with respect to X is equivalent to minimizing a weighted least–squares loss function with weighting matrix Σ^{-1} when Σ has the trivial form

$$\Sigma = \sigma^2 I_{4m\times 4m}.$$

Under the gaussian assumption, the estimate obtained via this minimization corresponds to determining an MLE for X given the entire data set of vector–measurements $\{\,\mathbf{y}[1], \cdots, \mathbf{y}[m]\}$.

In order to utilize the entire data set of vector–measurements, we wish to find the MLE estimate for $X = (S^T, b)^T$ given the entire data set. We have shown that this estimate minimizes the complete–data least–squares loss function $\bar{\boldsymbol{\ell}}(X)$ given by Equation (12) which, in turn, is the same as minimizing the average loss function (11). Focussing, then, on the average loss function (11) *we now proceed to show that we can further simplify the loss function to be minimized.*

Define the *average* vector–measurement $\bar{\boldsymbol{y}} \in \mathbb{R}^4$ by

$$\bar{\boldsymbol{y}} = \langle \mathbf{y}[k] \rangle_m = \frac{1}{m} \sum_{k=1}^{m} \mathbf{y}[k] \,. \tag{13}$$

Noting that

$$2\,\boldsymbol{\ell}_k(X) = \|\mathbf{y}[k] - \mathbf{h}(X)\|^2 = \|\mathbf{y}[k]\|^2 - 2\,\langle \mathbf{y}[k], \mathbf{h}(X) \rangle + \|\mathbf{h}(X)\|^2$$

we can expand the average loss function (11) as

$$\begin{aligned} 2\,\bar{\boldsymbol{\ell}}\,(X) &= 2\,\langle \boldsymbol{\ell}_k(X) \rangle_m \\ &= \left\langle \|\mathbf{y}[k]\|^2 \right\rangle_m - 2\,\langle \bar{\boldsymbol{y}}, \mathbf{h}(X) \rangle + \|\mathbf{h}(X)\|^2 \\ &= \|\bar{\boldsymbol{y}}\|^2 - 2\,\langle \bar{\boldsymbol{y}}, \mathbf{h}(S) \rangle + \|\mathbf{h}(X)\|^2 + \left(\left\langle \|\mathbf{y}[k]\|^2 \right\rangle_m - \|\bar{\boldsymbol{y}}\|^2 \right) \\ &= \|\bar{\boldsymbol{y}} - \mathbf{h}(X)\|^2 + \left(\left\langle \|\mathbf{y}[k]\|^2 \right\rangle_m - \|\bar{\boldsymbol{y}}\|^2 \right) \\ &= 2\,\boldsymbol{\ell}(X) + \mathbf{g} \,, \end{aligned}$$

where

$$\boldsymbol{\ell}(X) = \frac{1}{2} \|\bar{\boldsymbol{y}} - \mathbf{h}(X)\|^2 \tag{14}$$

and

$$\mathbf{g} = \left(\left\langle \|\mathbf{y}[k]\|^2 \right\rangle_m - \|\bar{\boldsymbol{y}}\|^2 \right) \,.$$

Finally note that because $\mathbf{g}$ is independent of X, minimizing $\boldsymbol{\ell}(X)$ with respect to X is entirely equivalent to minimizing $\bar{\boldsymbol{\ell}}(X)$ (and hence to minimizing the complete–data loss function (12)). *Thus the problem of finding a least–squares solution to the complete–data loss function is equivalent to finding the minimum of the loss function $\boldsymbol{\ell}(X)$ given in (14).*

Let the *average noise vector* $\bar{\boldsymbol{\nu}} \in \mathbb{R}^4$ be defined by

$$\bar{\boldsymbol{\nu}} = \langle \boldsymbol{\nu}[k] \rangle_m = \frac{1}{m} \sum_{k=1}^{m} \boldsymbol{\nu}[k] \,. \tag{15}$$

Because of the iid assumption on the measurement noises $\boldsymbol{\nu}[\mathrm{k}] \sim \mathrm{N}(0, \Sigma) = \mathrm{N}(0, \sigma^2 I_{4\times 4})$, we have that the covariance matrix of $\bar{\boldsymbol{\nu}}$ is given by[1]

$$\boldsymbol{\Sigma} = \mathrm{E}\left\{ \bar{\boldsymbol{\nu}} \bar{\boldsymbol{\nu}}^T \right\} = \frac{1}{m^2} \sum_{k=1}^{m} \mathrm{E}\left\{ \boldsymbol{\nu}[k] \boldsymbol{\nu}^T[k] \right\} = \frac{1}{m} \Sigma = \frac{\sigma^2}{m} I_{4\times 4} \,. \tag{16}$$

[1] If you haven't had a probability course, just accept this fact. If you have had a probability course but you cannot easily prove this step, then there is a serious deficiency in your education.

Given the definitions (13) and (15) and the sequence of single vector–measurement equations (6) we have that

$$\bar{\boldsymbol{y}} = \mathbf{h}(X) + \bar{\boldsymbol{\nu}} \tag{17}$$

where $\bar{\boldsymbol{\nu}} \sim \mathrm{N}(0, \boldsymbol{\Sigma}) = \mathrm{N}(0, \frac{\sigma^2}{m} I_{4\times 4})$. The estimate of X obtained from minimizing the loss function (14) is equivalent to obtaining an MLE for (17) as it is readily shown that (14) is also given by

$$\boldsymbol{\ell}(X) = \frac{\sigma^2}{2m} \left\| \bar{\boldsymbol{y}} - \mathbf{h}(X) \right\|^2_{\boldsymbol{\Sigma}^{-1}} . \tag{18}$$

Generalized Gradient Descent Algorithms. As derived in Lecture Supplement 2, the Generalized Gradient Descent Algorithm (GDA) for minimizing the loss function (14) is given by

$$\widehat{X}_{j+1} = \widehat{X}_j + \alpha_k \, Q(\widehat{X}_j) \, \mathbf{H}^T(\widehat{X}_j) \left(\bar{\boldsymbol{y}} - \mathbf{h}(\widehat{X}_j) \right) . \tag{19}$$

where the estimate at iteration j of the algorithm is $\widehat{X}_j = (\widehat{S}_j^T, \widehat{b}_j)^T$ and

$$\mathbf{h}(\widehat{X}_j) = \mathbf{R}(\widehat{S}_j) + \mathbf{e}\, \hat{b}_j$$

where $\mathbf{R}(\widehat{S}_j)$ is computed using Equations (1) and (7).

The Jacobian matrix $\mathbf{H}(X) \in \mathbb{R}^{4\times 4}$ is determined from Equation (3) to be

$$\begin{aligned} \mathbf{H}(X) &= \frac{\partial \mathbf{h}(X)}{\partial X} = \begin{pmatrix} \frac{\partial \mathbf{R}(S)}{\partial S} & \frac{\partial \mathbf{e}\, b}{\partial b} \end{pmatrix} \\ &= \begin{pmatrix} r_1^T(S) & \\ \vdots & \mathbf{e} \\ r_4^T(S) & \end{pmatrix} = \begin{pmatrix} r_1^T(S) & 1 \\ r_2^T(S) & 1 \\ r_3^T(S) & 1 \\ r_4^T(S) & 1 \end{pmatrix} \end{aligned} \tag{20}$$

where $r_\ell\,(S)$ is given by Equation (2).

The (standard) *gradient descent algorithm* is based on the simple choice $Q(X) \equiv I$ in (19), yielding

$$\textbf{Gradient Descent:} \qquad \widehat{X}_{j+1} = \widehat{X}_j + \alpha_k \, \mathbf{H}^T(\widehat{X}_j) \left(\bar{\boldsymbol{y}} - \mathbf{h}(\widehat{X}_j) \right) . \tag{21}$$

The proper choice of a step–size parameter in the gradient descent algorithm is critical for good performance. Generally one expects this algorithm to have slow convergence.

The Gauss–Newton algorithm is based on the choice

$$Q(X) = \left(\mathbf{H}^T(X) \mathbf{H}(X) \right)^{-1} .$$

This results in

$$\widehat{X}_{j+1} = \widehat{X}_j + \alpha_k \, \mathbf{H}^+(\widehat{X}_j) \left(\bar{\boldsymbol{y}} - \mathbf{h}(\widehat{X}_j) \right) . \tag{22}$$

Note that for our problem $\mathbf{H}(X)$ is a square matrix, so that under the assumption that $\mathbf{h}(X)$ is one–to–one it is invertible.[2] Thus in our special case where $\mathbf{H}(X)$ just happens to be square, the Gauss–Newton algorithm corresponds to the choice

$$Q(X) = \mathbf{H}^{-1}(X)\mathbf{H}^{-T}(X)\,.$$

The use of this choice in (19) yields

$$\textbf{Gauss–Newton:} \qquad \widehat{X}_{j+1} = \widehat{X}_j + \alpha_k\, \mathbf{H}^{-1}(\widehat{X}_j)\left(\bar{\boldsymbol{y}} - \mathbf{h}(\widehat{X}_j)\right)\,, \tag{23}$$

as expected because $\mathbf{H}^{+}(\widehat{X}_j) = \mathbf{H}^{-1}(\widehat{X}_j)$ when $\mathbf{H}(\widehat{X}_j)$ is square and full rank.

The Gauss–Newton algorithm is less sensitive to the choice of the step–size parameter than the gradient descent method. Indeed, quite often good performance is obtained with the Gauss–Newton choice of $\alpha_k = 1$. Note, however, that the computational difference between the simple gradient descent method and the higher–performing Gauss–Newton method is significant.

Note that at any point $X = \widehat{X}$ for which $\mathbf{h}(X)$ is differentiable we have that

$$d\bar{\boldsymbol{y}} = \frac{\partial \mathbf{h}(\widehat{X})}{\partial X}\, dX = \mathbf{H}(\widehat{X})\, dX$$

which corresponds to the linear approximation

$$\Delta\bar{\boldsymbol{y}} = \mathbf{H}(\widehat{X})\, \Delta X\,.$$

Assuming the invertibility of $\mathbf{H}(S)$, this yields

$$\Delta X = \mathbf{H}^{-1}(\widehat{X})\Delta\bar{\boldsymbol{y}} \tag{24}$$

which corresponds to the Gauss–Newton update (23) if we make the identifications

$$\alpha_k = 1\,, \quad \widehat{X} = \widehat{X}_j\,, \quad \Delta\bar{\boldsymbol{y}} = \bar{\boldsymbol{y}} - \mathbf{h}(\widehat{X}_j)\,, \quad \Delta X = \widehat{X}_{j+1} - \widehat{X}_j\,.$$

This is as expected because the Gauss–Newton algorithm with step–size $\alpha_k = 1$ is equivalent to reiteratively linearizing $\mathbf{h}$ and solving the resulting least–squares problem.[3]

[2] Of course this will *not* be true if we have more than four satellites in the receiver line-of-sight. For example, if we had range measurements, $y_5[k]$, to a fifth satellite then $\mathbf{H}(X)$ would be 5×4 and hence noninvertible. Having additional satellite measurements improves the accuracy of the estimates, but at the expense of having to compute the more general pseudoinverse of $\mathbf{H}(X)$.

[3] The least–squares solutions involves the pseudoinverse of the Jacobian matrix $\mathbf{H}$. However in our case of a full–rank square matrix the Jacobian is invertible, $\mathbf{H}^{+} = \mathbf{H}^{-1}$.

Gauss–Newton Parameter Estimate Error Variance. We can determine accurate estimates of the parameter error covariance matrix for the Gauss–Newton algorithm by exploiting the fact that the Gauss–Newton algorithm is equivalent to reiteratively linearizing the nonlinear inverse problem and then solving the resulting approximate linear inverse problem.

Let X_0 denote the optimal MLE value of X which minimizes the loss function (14) and assume that the estimation algorithm (19) is convergent to X_0, $\widehat{X}_\infty \triangleq \lim_{j\to\infty} \widehat{X}_j = X_0$. Let $X_{\rm true} = (S_{\rm true}^T, b_{\rm true})^T$ denote the true, unknown, receiver location and clock bias and in the sequel let

$$\Delta X = X_{\rm true} - X_0 = X_{\rm true} - \widehat{X}_\infty$$

be the error between the true, unknown value $X_{\rm true}$ and the MLE $X_0 = \widehat{X}_\infty$. Also define $\widehat{X}_{\infty+1} \triangleq \lim_{j\to\infty} \widehat{X}_{j+1}$ and note that under the assumption of convergence of the Gauss–Newton algorithm to the MLE solution X_0 we must have that $\widehat{X}_{\infty+1} = \widehat{X}_\infty$.

Now consider the linearization of (13) about the learned value $\widehat{X}_\infty = X_0$,[4]

$$\Delta\bar{\boldsymbol{y}} = \mathbf{H}(X_0)\,\Delta X + \bar{\boldsymbol{\nu}}\,,$$

where $\Delta\bar{\boldsymbol{y}} = \bar{\boldsymbol{y}} - \mathbf{h}(X_0)$. *It is assumed that* $\Delta X = X_{true} - X_0 = X_{true} - \widehat{X}_\infty$ *is small so that the linearization is a very good approximation to (13).* As discussed in lecture, under the one–to–one (full column rank) assumption the MLE estimate of ΔX is determined as

$$\widehat{\Delta X} = \mathbf{H}^+(X_0)\Delta\bar{\boldsymbol{y}} = \Delta X + \mathbf{H}^+(X_0)\bar{\boldsymbol{\nu}}\,,$$

where $\widehat{\Delta X} = \widehat{X}_{\rm true} - X_0 = \widehat{X}_{\rm true} - \widehat{X}_\infty$ and

$$\mathbf{H}^+(X_0) = \left(\mathbf{H}^T(X_0)\mathbf{H}(X_0)\right)^{-1}\mathbf{H}^T(X_0)$$

is a left inverse of $\mathbf{H}(X_0)$. In the special case considered here $\mathbf{H}(X_0)$ is square, and hence invertible, so that $\mathbf{H}(X_0)^+ = \mathbf{H}(X_0)^{-1}$. Thus our least squares estimate of $\Delta X = X_{\rm true} - X_0$ obeys

$$\widehat{\Delta X} = \Delta X + \mathbf{H}^{-1}(X_0)\bar{\boldsymbol{\nu}}\,. \tag{25}$$

As noted above, because the MLE solution X_0 is found from running the Gauss–Newton algorithm until convergence it must be true that[5]

$$X_0 = \widehat{X}_\infty = \widehat{X}_{\infty+1}\,.$$

[4]The linearization of (13) at the point X_0 corresponds to performing a Taylor series expansion of the nonlinear term $\mathbf{h}(X)$ about the point X_0,

$$\mathbf{h}(X) = \mathbf{h}(X_0 + \Delta X) = \mathbf{h}(\widehat{X}_0) + \mathbf{H}(X_0)\,\Delta X + \text{h.o.t.}\,,$$

and then ignoring the terms higher than linear order.

[5]That is, at convergence (i.e., at $k = \infty$) running the algorithm for one more step does not change the answer.

Furthermore, from the interpretation of the Gauss–Newton algorithm as reiteratively solving the linearized least–squares problem it must be the case that running the Gauss–Newton algorithm for one more iteration corresponds to the update step

$$\widehat{X}_{\infty+1} = \widehat{X}_{\infty} + \alpha_{\infty}\widehat{\Delta X}$$

with $\alpha_{\infty} > 0$. *Therefore it must be the case under the convergence assumption that*

$$\widehat{\Delta X} = \widehat{X}_{\text{true}} - X_0 = 0$$

so that $\widehat{X}_{\text{true}} = X_0 = \widehat{X}_{\infty}$. Therefore from (25) and the fact that $\Delta X = X_{\text{true}} - X_0$ we obtain the important result that at convergence of the Gauss–Newton algorithm we have[6]

$$\widehat{X}_{\infty} = X_0 = X_{\text{true}} + \mathbf{H}^{-1}(X_0)\bar{\boldsymbol{\nu}}\,. \tag{26}$$

From Equation (26) we obtain the result that $\widehat{X}_{\infty}$ is unbiased, $\mathrm{E}\left\{\widehat{X}_{\infty}\right\} = X_{\text{true}}$ and that the parameter error covariance matrix is given by

$$\mathrm{E}\left\{\left(\widehat{X}_{\infty} - X_{\text{true}}\right)\left(\widehat{X}_{\infty} - X_{\text{true}}\right)^T\right\} = \frac{\sigma^2}{m}\mathbf{H}^{-1}(\widehat{X}_{\infty})\mathbf{H}^{-T}(\widehat{X}_{\infty}) = \frac{\sigma^2}{m}\left(\mathbf{H}^T(\widehat{X}_{\infty})\mathbf{H}(\widehat{X}_{\infty})\right)^{-1}.$$

Note that this precisely the result we expect from the MLE for the linear model which we have shown in lecture results in the parameter error covariance matrix,

$$\left(\mathbf{H}^T(\widehat{X}_{\infty})\boldsymbol{\Sigma}^{-1}\mathbf{H}(\widehat{X}_{\infty})\right)^{-1}$$

where $\boldsymbol{\Sigma} = \frac{\sigma^2}{m}I_{4\times 4}$. The square–roots of the diagonal elements of the parameter error covariance matrix give the error standard–deviations for the location and clock bias estimates obtained using the Gauss–Newton method.

Comments on Step-Size Determination. The Generalized Gradient Descent algorithm is of the form

$$\widehat{X}_{j+1} = \widehat{X}_j + \Delta\widehat{X}_j$$

where

$$\Delta\widehat{X}_j = -\alpha_j\, Q(\widehat{X}_j)\,\nabla_X \ell(\widehat{X}_j)$$

corresponds to a step in the opposite direction to the generalized gradient $Q(\widehat{X}_j)\,\nabla_X\ell(\widehat{X}_j)$[7] and the size of this step is controlled by the value of the step–size parameter $\alpha_j > 0$.

[6] Assuming that the error $\Delta X = X_{\text{true}} - X_0 = X_{\text{true}} - \widehat{X}_{\infty}$ is small so that the linear approximation is good.

[7] The fact that $Q(\widehat{X}_j)$ is positive definite ensures that the angle between the gradient $\nabla_X\ell(\widehat{X}_j)$ and the generalized gradient $Q(\widehat{X}_j)\,\nabla_X\ell(\widehat{X}_j)$ is less than 90^o so that a movement along the generalized gradient direction always has a component along the gradient direction.

The choice of step size can be very critical for ensuring convergence of a generalized gradient descent algorithm and is a major topic of concern in advanced textbooks on optimization theory. If the simple choice of a constant value of α_j is used, generally the smaller the value of α_j the more likely it is that the algorithm will converge but also that the rate of convergence will be slow. Fortunately, the Newton and Gauss–Newton methods tend to be quite stable and the choice of $\alpha_j = 1$ or α equal to a constant value slightly less than one often works well.

More generally, the step size choice can be chosen dynamically. Note that we need $\alpha_j \to 0$ more slowly than the generalized gradient goes to zero in order to avoid turning off the update step before we have learned the unknown parameter vector X. In principle we require that $\alpha_\infty > 0$ but practically we can have $\alpha_\infty = 0$ provided that the generalized gradient converges to zero before the step–size parameter has converged to zero. A simple dynamic choice for α_j is given by

$$\alpha_j = \alpha_0 \beta^j \,, \quad \beta = 0, 1, 2, \cdots$$

for $\alpha_0 > 0$ and $0 < \beta < 1$. More sophisticated ways to dynamically adjust the step–size are discussed in the textbooks on optimization theory.

A simple but computationally expensive way to enforce convergence[8] is to choose a step–size that guarantees that the loss function $\ell(\widehat{X}_j)$ is decreased as follows:

Begin

Choose values for $\alpha_0 > 0$ and $1 > \beta > 0$

Set $\ell_j = \ell(\widehat{X}_j)$ and $k = 0$

Loop Until $\ell_{j+1} < \ell_j$

$\alpha = \alpha_0 \, \beta^k$

$\widehat{X}_{j+1} = \widehat{X}_j - \alpha \, Q(\widehat{X}_j) \, \nabla_X \ell(\widehat{X}_j)$

$\ell_{j+1} = \ell(\widehat{X}_{j+1})$

$k \leftarrow k + 1$

End Loop

End

This will ensure that $\ell(\widehat{X}_{j+1}) < \ell(\widehat{X}_j)$ but at the potential cost of several expensive update and loss function evaluations at each iteration step j of the generalized gradient descent algorithm.

[8] But perhaps at the expense of the speed of convergence.