Levenberg-Marquardt minimisation

Philip F McLauchlan

The Levenberg-Marquardt algorithm [3, 2] is a general non-linear downhill minimisation algorithm for the case when derivatives of the objective function are known. It dynamically mixes Gauss-Newton and gradient-descent iterations. We shall develop the L-M algorithm for a simple case in our notation, which is derived from Kalman filtering theory [1]. The Ravl implementation of Levenberg-Marquardt will then be presented. Let the unknown parameters be represented by the vector $\mathbf{x}$, and let noisy measurements of $\mathbf{x}$ be made:

$$\mathbf{z}(j) = \mathbf{h}(j;\mathbf{x}) + \mathbf{w}(j), \ \ j = 1, \ldots, k \tag{1}$$

where $\mathbf{h}(j)$ is a measurement function and $\mathbf{w}(j)$ is zero-mean noise with covariance $N(j)$. Since we are describing an iterative minimization algorithm, we shall assume that we have already obtained an estimate $\hat{\mathbf{x}}^-$ of $\mathbf{x}$. Then the maximum likelihood solution for a new estimate $\hat{\mathbf{x}}$ minimizes

$$\chi^2(\hat{\mathbf{x}}) = \sum_{j=1}^{k} (\mathbf{z}(j) - \mathbf{h}(j;\hat{\mathbf{x}}))^\top N(j)^{-1} (\mathbf{z}(j) - \mathbf{h}(j;\hat{\mathbf{x}})). \tag{2}$$

We form a quadratic approximation to $\chi^2(.)$ around $\hat{\mathbf{x}}^-$, and minimize this approximation to $\chi^2(.)$ to obtain a new estimate $\hat{\mathbf{x}}^+$. In general we can write such a quadratic approximation as

$$\chi^2(\mathbf{x}) \approx a - 2\mathbf{a}^\top(\mathbf{x} - \hat{\mathbf{x}}^-) + (\mathbf{x} - \hat{\mathbf{x}}^-)^\top A(\mathbf{x} - \hat{\mathbf{x}}^-)$$

for scalar a, vectors $\mathbf{a}$, $\mathbf{b}$ and matrices A, B. Note that here and in equation (2) the signs and factors of two are chosen WLOG to simplify the resulting expressions for the solution. Differentiating, we obtain

$$\frac{\partial \chi^2}{\partial \mathbf{x}} = -2\mathbf{a} + 2A(\mathbf{x} - \hat{\mathbf{x}}^-),$$

$$\frac{\partial^2 \chi^2}{\partial \mathbf{x}^2} = 2A,$$

At the minimum point $\hat{\mathbf{x}}$ we have $\partial\chi^2/\partial\mathbf{x} = \mathbf{0}$ which means that

$$A(\hat{\mathbf{x}}^+ - \hat{\mathbf{x}}^-) = \mathbf{a}. \tag{3}$$

Thus we need to obtain $\mathbf{a}$ and A to compute the update. We now consider the form of $\chi^2(.)$ in (2). Writing the Jacobian of $\mathbf{h}(j,\mathbf{x})$ as

$$H(j) = \frac{\partial \mathbf{h}(j)}{\partial \mathbf{x}},$$

we have

$$\begin{aligned} \frac{\partial \chi^2}{\partial \mathbf{x}} &= -2\sum_{j=1}^{k} H(j)^\top N(j)^{-1}(\mathbf{z}(j) - \mathbf{h}(j;\mathbf{x})), && (4) \\ \frac{\partial^2 \chi^2}{\partial \mathbf{x}^2} &= 2\sum_{j=1}^{k} H(j)^\top N(j)^{-1} H(j) - 2\sum_{j=1}^{k} \left(\frac{\partial H(j)}{\partial \mathbf{x}}\right)^\top N(j)^{-1}(\mathbf{z}(j) - \mathbf{h}(j;\mathbf{x})) \\ &\approx 2\sum_{j=1}^{k} H(j)^\top N(j)^{-1} H(j), && (5) \end{aligned}$$

In the last formula for $\partial^2\chi^2/\partial\mathbf{x}^2$, the terms involving the second derivatives of $\mathbf{h}_{(j)}(.)$ have been omitted. This is done because these terms are generally much smaller and can in practice be omitted, as well as because the second derivatives are more difficult and complex to compute than the first derivatives.

Now we solve the above equations for $\mathbf{a}$ and A given the values of function $\mathbf{h}_{(j)}$ and the Jacobian $H_{(j)}$ evaluated at the previous estimate $\hat{\mathbf{x}}^-$. We have immediately

$$A = \sum_{j=1}^{k} H_{(j)}{}^\top N_{(j)}{}^{-1} H_{(j)}.$$

We now write the *innovation* vectors $\boldsymbol{\nu}_{(j)}$ as

$$\boldsymbol{\nu}_{(j)} = \mathbf{z}_{(j)} - \mathbf{h}(j; \hat{\mathbf{x}}^-)$$

Then we have

$$\mathbf{a} = \sum_{j=1}^{k} H_{(j)}{}^\top N_{(j)}{}^{-1} \boldsymbol{\nu}_{(j)} \tag{6}$$

Combining equations (3) and (6) we obtain the linear system

$$A(\hat{\mathbf{x}}^+ - \hat{\mathbf{x}}^-) = \mathbf{a} = \sum_{j=1}^{k} H_{(j)}{}^\top N_{(j)}{}^{-1} \boldsymbol{\nu}_{(j)} \tag{7}$$

to be solved for the adjustment $\hat{\mathbf{x}}^+ - \hat{\mathbf{x}}^-$. The covariance of the state is

$$P = A^{-1}.$$

The update (7) may be repeated, substituting the new $\hat{\mathbf{x}}^+$ as $\hat{\mathbf{x}}^-$, and improving the estimate until convergence is achieved according to some criterion. Levenberg-Marquardt modifies this updating procedure by adding a value λ to the diagonal elements of the linear system matrix before inverting it to obtain the update. λ is reduced if the last iteration gave an improved estimate, i.e. if χ^2 was reduced, and increased if J increased, in which case the estimate of $\mathbf{x}$ is reset to the estimate before the last iteration. It is this that allows the algorithm to smoothly switch between Gauss-Newton (small λ) and gradient descent (large λ).

This version is a generalization of Levenberg-Marquardt as it is normally presented (e.g. [4]) in that we incorporate vector measurements $\mathbf{z}_{(j)}$ with covariances $N_{(j)}$, rather than scalar measurements with variances. The full algorithm is as follows:

1. Start with a prior estimate $\hat{\mathbf{x}}^-$ of $\mathbf{x}$. Initialize λ to some starting value, e.g. 0.001.
2. Compute the updated estimate $\hat{\mathbf{x}}^+$ by solving the linear system (7) for the adjustment, having first added λ to each diagonal element of A.
3. Compute the least-squares residuals $\chi^2(\hat{\mathbf{x}}^-)$ and $\chi^2(\hat{\mathbf{x}}^+)$ from (2).
   (a) If $\chi^2(\hat{\mathbf{x}}^+) < \chi^2(\hat{\mathbf{x}}^-)$, reduce λ by a specified factor (say 10), set $\hat{\mathbf{x}}^-$ to $\hat{\mathbf{x}}^+$, and return to step 2.
   (b) Otherwise, the update failed to reduce the residual, so increase λ by a factor (say 10), forget the updated $\hat{\mathbf{x}}^+$ and return to step 2.

The algorithm continues until some pre-specified termination criterion has been met, such as a small change to the state vector, or a limit on the number of iterations.

If the measurements $\mathbf{z}_{(j)}$ are unbiased and normally distributed, the residual $\chi^2(\hat{\mathbf{x}}^+)$ is a χ^2 random variable, and testing the value of χ^2 against a χ^2 distribution is a good way of checking that the measurement noise model is reasonable. The number of degrees of freedom (DOF) of the χ^2 distribution can be determined as the total size of the measurement vectors, minus the size of the state. If the $SIZE(.)$ function returns the dimension of its vector argument, then the degrees of freedom may be computed as

$$DOF = \sum_{j=1}^{k} SIZE(\mathbf{z}_{(j)}) - SIZE(\mathbf{x}). \tag{8}$$

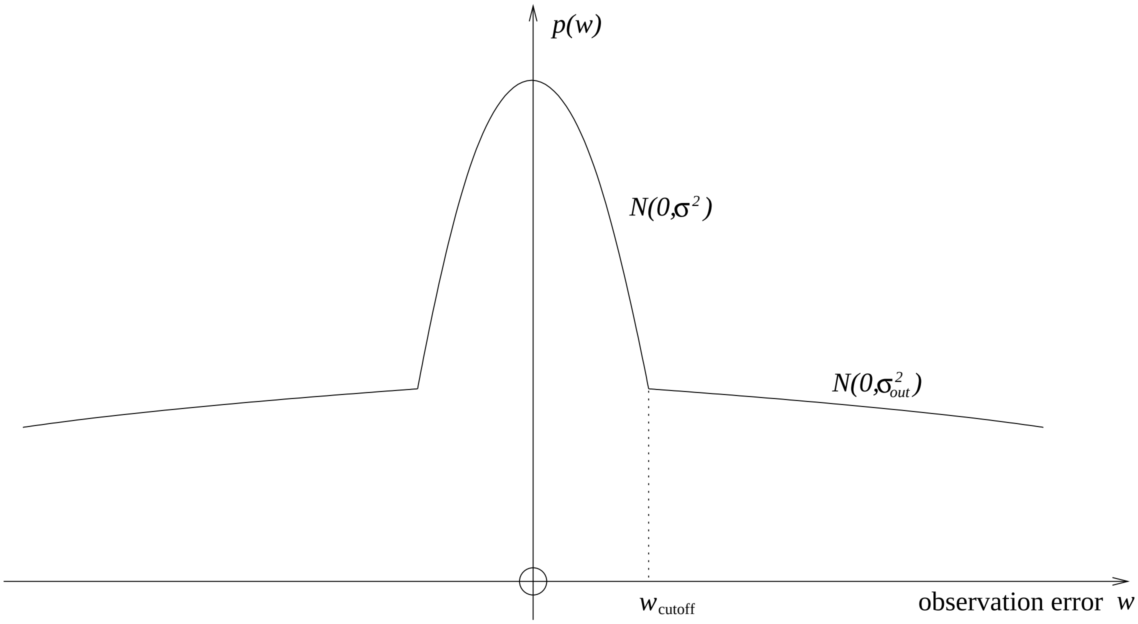

Figure 1: The error model used to model outliers in the observations incorporated in robust Levenberg-Marquardt, a combination of a narrow inlier Gaussian with variance σ^2, and wide Gaussian for outliers with variance σ^2_{out}. Both distributions on the observation error w have zero mean. The inlier and outlier Gaussians are scaled so that they meet continuously at a specific error value w_{cutoff}, which is effectively an upper error threshold for observations to be treated as inliers.

0.1 Robust observations

An important drawback of standard least-squares algorithm such as Levenberg-Marquardt is that they assume that all observations are correct. Various types of estimators have been successfully used to deal with the presence of outliers in the data. Examples are least median-of-squares, RANSAC and Hough transform estimators. These estimators involve a radical redesign of the measurement error model. We employ what is probably the simplest method of "robustifying" the standard Gaussian error model. The robust error model used here assumes that the errors follow a distribution combining a narrow "inlier" Gaussian with a wide "outlier" Gaussian, as shown for a one-dimensional distribution in Figure 1. The distribution is a function of the observation error[1] $\mathbf{w} = \mathbf{z} - \mathbf{h}(\mathbf{x})$. The relative vertical scaling of the two Gaussians is chosen so that the two distribution functions are equal at a chosen point w_{cutoff}.

For a general multi-dimensional observation, we have a inverse covariance matrix N^{-1} for the inlier distribution. We restrict the outlier distribution N^{-1}_{out} to be a rescaled version of the inlier distribution, so that

$$N^{-1}_{\text{out}} = \frac{1}{K} N^{-1}$$

for some value $K > 1$. We then choose a threshold χ^2_{cutoff} on the value of the normalised squared error χ^2 for switching between the two distributions. The probability distribution function is therefore

$$p(\mathbf{w}) = \begin{cases} e^{-\mathbf{w}^\top N^{-1}\mathbf{w}} & \text{if} \quad \mathbf{w}^\top N^{-1}\mathbf{w} < \chi^2_{\text{cutoff}} \ \text{(inlier)} \\ \\ e^{(K^{-1}-1)\chi^2_{\text{cutoff}}} e^{-\mathbf{w}^\top N^{-1}_{\text{out}}\mathbf{w}} & \text{otherwise (outlier)} \end{cases}$$

The scaling of the outlier distribution is chosen so that the two distributions are correctly aligned at the chosen cutoff point χ^2_{cutoff}. This leads directly to the correct "compensation" value for the likelihood function $(K^{-1}-1)\chi^2_{\text{cutoff}}$, to be added to the least-squares residual when the outlier distribution is selected during application of a minimisation

[1] The innovation $\boldsymbol{\nu}$ is the observation error relative to that computed from the *estimated* state: $\boldsymbol{\nu} = \mathbf{z} - \mathbf{h}(\hat{\mathbf{x}})$.

iteration. Note that each Levenberg-Marquardt observation can be chosen as robust or non-robust, and potentially each with a different choice for covariance scale factor K and squared error threshold χ^2_{cutoff}.

0.2 Implicit observations

Equation 1 does not encapsulate the most general form of observation, since it assumes that the observation vector $\mathbf{z}$ can be separated explicitly as a function $\mathbf{h}(\mathbf{x})$ of the state $\mathbf{x}$. It is sometimes therefore necessary to introduce an implicit observation equation of the form

$$\mathbf{F}(\mathbf{x}, \mathbf{z} - \mathbf{w}) = \mathbf{0}$$

where $\mathbf{w}$ again represents a random noise vector having covariance N. However with some manipulation and extra computation we can effectively convert the linearised version of the implicit $\mathbf{F}$-type function into an explicit $\mathbf{h}$-type function, allowing it to be incorporated in the same way. We linearise $\mathbf{F}(.)$ with respect to $\mathbf{x}$ and $\mathbf{z}$ around the estimated state $\hat{\mathbf{x}}$ and observation $\mathbf{z}$, assuming that the noise $\mathbf{w}$ is small:

$$\mathbf{F}(\mathbf{x}, \mathbf{z}_t) = \mathbf{F}(\hat{\mathbf{x}}, \mathbf{z}) + \frac{\partial \mathbf{F}}{\partial \mathbf{x}}(\mathbf{x} - \hat{\mathbf{x}}) - \frac{\partial \mathbf{F}}{\partial \mathbf{z}}\mathbf{w} = \mathbf{0}$$

where $\mathbf{x}$ here represents the true value of the state vector, and $\mathbf{z}_t$ is the true observation vector (as opposed to the actually measured vector $\mathbf{z}$), so that $\mathbf{w} = \mathbf{z} - \mathbf{z}_t$. We identify the following quantities with their equivalents for an $\mathbf{h}$-type observation:

The **innovation** vector is $\boldsymbol{\nu} = -\mathbf{F}(\hat{\mathbf{x}}, \mathbf{z})$.

The **Jacobian** matrix is $H = \frac{\partial \mathbf{F}}{\partial \mathbf{x}}$.

The **noise vector** is $\mathbf{w}' = \frac{\partial \mathbf{F}}{\partial \mathbf{z}}\mathbf{w}$.

The **noise covariance** matrix is $N' = \frac{\partial \mathbf{F}}{\partial \mathbf{z}} N \left(\frac{\partial \mathbf{F}}{\partial \mathbf{z}}\right)^\top$.

Extra computation is therefore needed to convert the observation covariance from N to N'. The innovation vector $\boldsymbol{\nu}$, Jacobian matrix H and observation covariance N' are substituted into the Levenberg-Marquardt algorithm in place of their equivalents for the $\mathbf{h}$-type observation.

References

[1] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association.* Academic Press, 1988.

[2] Åke Björck. *Numerical Methods for Least Squares Problems.* SIAM Press, Philadelphia, PA, 1996.

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

[4] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C.* Cambridge University Press, 1988.