Precision of the ATLAS muon spectrometer
Woudstra, M.J.

Citation for published version (APA):
Woudstra, M. J. (2002). Precision of the ATLAS muon spectrometer

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
Appendix C  Fitting a model to data using $\chi^2$ minimisation

We want to fit a model with $M$ parameters $\alpha_i$ to a set of $N$ uncorrelated measurements $y_i$ with error $\sigma_i$. The model gives a set of $N$ predictions $f_i(\alpha)$ for $y_i$. To determine the values of $\alpha_i$, we minimise the $\chi^2$, which is defined as

$$\chi^2 = \sum_{i=1}^{N} \left( \frac{y_i - f_i(\hat{\alpha})}{\sigma_i} \right)^2 \quad \text{C-1}$$

This minimisation problem can be converted into a set of $M$ non-linear equations by setting the derivatives to zero:

$$\frac{\partial \chi^2}{\partial \alpha_k} = -2 \sum_{i=1}^{N} \frac{[y_i - f_i(\hat{\alpha})]}{\sigma_i^2} \cdot \frac{\partial f_i(\hat{\alpha})}{\partial \alpha_k} = 0 \quad k = 1 \ldots M. \quad \text{C-2}$$

The Newton-Raphson method [57] of solving these equations linearises the $\chi^2$ around initial values $\hat{\alpha}_0$, which gives the matrix equation

$$\left. \frac{\partial \chi^2}{\partial \alpha} \right|_{\hat{\alpha} = \hat{\alpha}_0} + (\hat{\alpha}_e - \hat{\alpha}_0) \left. \frac{\partial^2 \chi^2}{\partial \alpha^2} \right|_{\hat{\alpha} = \hat{\alpha}_0} = \hat{\delta}, \quad \text{C-3}$$

where $\hat{\alpha}_e$ is the next estimation of the parameters. The second derivative of the $\chi^2$, which appears in this equation, is given by

$$\frac{\partial^2 \chi^2}{\partial \alpha_k \partial \alpha_l} = 2 \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \left[ \frac{\partial f_i(\hat{\alpha})}{\partial \alpha_k} \cdot \frac{\partial f_i(\hat{\alpha})}{\partial \alpha_l} - \left( y_i - f_i(\hat{\alpha}) \right) \cdot \frac{\partial^2 f_i(\hat{\alpha})}{\partial \alpha_k \partial \alpha_l} \right]. \quad \text{C-4}$$

The terms containing the second derivative of $f_i$ tend to cancel out if the model is correct, and can in fact be destabilising if the model fits badly or if the data is contaminated by one-sided outlier points. We will therefore leave out these terms. It is conventional (and faster in calculation) to remove the factors of 2 by defining

$$a_{kl} = \frac{1}{2} \cdot \frac{\partial^2 \chi^2}{\partial \alpha_k \partial \alpha_l}, \quad b_k = -\frac{1}{2} \cdot \frac{\partial \chi^2}{\partial \alpha_k}, \quad \delta \alpha_l = \alpha_{e,l} - \alpha_{0,l} \quad \text{C-5}$$

so that equation C-3 can be written as

$$\sum_{l=1}^{M} a_{kl} \delta \alpha_l = b_k. \quad \text{C-6}$$
The matrix elements $a_{kl}$ and vector elements $b_k$ can be expressed in $y_i, f_i$ and $df/da$:

$$a_{kl} = \sum_{i=1}^{N} \frac{1}{\sigma_i^2} \frac{\partial f_i(\bar{a})}{\partial \alpha_k} \frac{\partial f_i(\bar{a})}{\partial \alpha_l}$$  \hspace{1cm} \text{C-7}$$

$$b_k = \sum_{i=1}^{N} \frac{1}{\sigma_i} (y_i - f_i(\bar{a})) \frac{\partial f_i(\bar{a})}{\partial \alpha_k} .$$  \hspace{1cm} \text{C-8}$$

If the matrix $A$ is nearly singular far away from the minimum, the solution to the Newton-Raphson method oozes off into infinity. In this case we can use the steepest descent formula:

$$\delta \alpha_i = \text{constant} \cdot b_i,$$  \hspace{1cm} \text{C-9}$$

which just moves down hill with a step size depending on the constant.

The Levenberg-Marquardt method [57] chooses the constant as $1/(\lambda a_{ll})$ with fudge factor $\lambda$ to regulate the step size. It allows for a continuous transition between the Newton-Raphson method and the steepest descent method. This method works very well in practice and has become the standard of non-linear least-squares routines. The transition between the two methods is done by replacing the matrix elements $a_{kl}$ by $a'_{kl}$:

$$a'_{kl} = a_{kl} (1 + \delta_{kl} \lambda).$$  \hspace{1cm} \text{C-10}$$

with $\delta_{kl}$ the Kronecker delta. When $\lambda$ is very large, we are in the steepest descent regime, and as $\lambda$ approaches zero, we go into Newton-Raphson regime.

Given an initial guess $\bar{a}_0$, the recommended Marquardt recipe is as follows:

1. Compute $\chi^2(\bar{a}_0)$.
2. Pick a modest value for $\lambda$, say $\lambda = 0.001$.
3. Solve the linear equations with $a'_{kl}$ and evaluate $\chi^2(\bar{a}_0 + \delta \bar{a})$.
4. If $\chi^2(\bar{a}_0 + \delta \bar{a}) > \chi^2(\bar{a}_0)$, increase $\lambda$ by a substantial factor (e.g. 10) and go back to 3.
5. If $\chi^2(\bar{a}_0 + \delta \bar{a}) < \chi^2(\bar{a}_0)$, decrease $\lambda$ by a substantial factor, update the trial solution ($\bar{a}_0 = \bar{a}_0 + \delta \bar{a}$), and go back to 3.
6. Once the iteration is converged (according to some criterion), set $\lambda = 0$, solve the linear equations and compute the covariance matrix of the parameters $V_{\bar{a}}$:

$$V_{\bar{a}} = [a_{kl}]^{-1} .$$  \hspace{1cm} \text{C-11}$$

For a linear model (i.e. $\partial^2 f(\bar{a})/\partial \bar{a}^2 = 0$), the solution is independent of $\bar{a}_0$ and only one iteration is needed (with $\lambda = 0$).

For a non-linear model one needs to iterate. It gives the correct answer (i.e. converges to the global minimum), if $\bar{a}_0$ is sufficiently close to $\bar{a}_*$, where 'sufficiently' depends on the non-linearity of the problem.
If the measurements are correlated, we need both the errors and the correlations, which we represent in the $N \times N$ covariance matrix $V$. The inverse matrix $W = V^{-1}$ appears in the $\chi^2$, so that equation C-1 changes to:

$$\chi^2 = [y - \hat{f}(\alpha)]^T \cdot W \cdot [y - \hat{f}(\alpha)] = \sum_{i=1}^{N} \sum_{j=1}^{N} (y_i - f_i(\alpha))W_{ij}(y_j - f_j(\alpha)).$$

C-12

The derivative of $\chi^2$ to parameter $\alpha_k$ is then very similar to the uncorrelated one, except that now there is a double summation:

$$\frac{\partial \chi^2}{\partial \alpha_k} = -2 \sum_{i=1}^{N} \sum_{j=1}^{N} W_{ij}(y_i - f_i(\alpha)) \cdot \frac{\partial f_i(\alpha)}{\partial \alpha_k} = -2[y - \hat{f}(\alpha)]^T \cdot W \cdot \frac{\partial \hat{f}(\alpha)}{\partial \alpha_k}.$$

C-13

Similarly for the second derivative:

$$\frac{\partial^2 \chi^2}{\partial \alpha_k \partial \alpha_l} = 2 \sum_{i=1}^{N} \sum_{j=1}^{N} W_{ij} \left( \frac{\partial f_i(\alpha)}{\partial \alpha_k} \cdot \frac{\partial f_j(\alpha)}{\partial \alpha_l} - (y_i - f_i(\alpha)) \cdot \frac{\partial^2 f_j(\alpha)}{\partial \alpha_k \partial \alpha_l} \right),$$

C-14

$$= 2 \left[ \frac{\partial \hat{f}(\alpha)}{\partial \alpha_k} \right]^T \cdot W \cdot \frac{\partial \hat{f}(\alpha)}{\partial \alpha_l} - [y - \hat{f}(\alpha)]^T \cdot W \cdot \frac{\partial^2 \hat{f}(\alpha)}{\partial \alpha_k \partial \alpha_l}$$

so that the elements $a_{kl}$ and $b_k$ for correlated data are:

$$a_{kl} = \left[ \frac{\partial \hat{f}(\alpha)}{\partial \alpha_k} \right]^T \cdot W \cdot \frac{\partial \hat{f}(\alpha)}{\partial \alpha_l},$$

C-15

$$b_k = [y - \hat{f}(\alpha)]^T \cdot W \cdot \frac{\partial \hat{f}(\alpha)}{\partial \alpha_k},$$

C-16

which reduces to the uncorrelated case if $W$ is a diagonal matrix with $1/\sigma_i^2$ as elements. The notation has become equal for both cases, which is exploited in the implementation of this algorithm in the C++ programming language by writing generic code using templates and operator overloading. A nice feature of this method is that the user only needs to supply the data points and the parametric function (including the derivatives w.r.t. the parameters), and does not need to first build the $\chi^2$. 

Appendix C  Fitting a model to data using $\chi^2$ minimisation 151