# Uncertainty analysis

# References

- Booth, C. H. "Statistical measure of confidence" Int. Tables Crystallogr. I (2022)
- Bevington, P. R. & Robinson, D. K. (2023). Data Reduction and Error Analysis for the Physical Sciences, 3nd ed., ch. 11. Boston: WBC/McGraw-Hi
- User guide fro MINPACK-1, Argonne National Laboratory, Report ANL-80-74, 1980.

# Levenberg-Marquardt algorithm

- The Levenberg-Marquardt algorithm is a method for solving non-linear least squares problems, which arise in many applications in science and engineering. The algorithm is an iterative technique that starts from an initial estimate of the solution and improves the solution iteratively.

# Nonlinear equations

If $m$ functions of $f_1, f_2, \cdots, f_m$ of the $n$ variables $x_1, x_2, \cdots, x_n$ are given, with $m \geq n$, the MINPACK-1 subroutines can be used to find $x_1, x_2, \cdots, x_n$ such that solve the nonlinear system of equations:

$$\min \left\{ \sum_{i=1}^{m} f_i(x)^2 : x \in \mathbb{R}^n \right\}$$

For $\mathbf{x} \in \mathbb{R}^n$, define the vector function $\mathbf{F} : \mathbb{R}^n \to \mathbb{R}^m$ by

$$\mathbf{F}(\mathbf{x}) = \begin{pmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_m(\mathbf{x}) \end{pmatrix}$$

where the $f_i$ is the i-th residual of the system of equations. The residual means that $f_i(\mathbf{x}) = 0$ is the solution of model $g_i(\mathbf{x})$ with given data $y_i$.

$$f(\mathbf{x}) = y_i - g_i(\mathbf{x})$$

The problem of finding a $\mathbf{x} \in \mathbb{R}^n$ such that

$$\mathrm{argmin}_{\mathbf{x} \in \mathbb{R}^n} \left\{ \sum_{i=1}^{m} f_i(\mathbf{x})^2 \right\}$$

can be refrazed as finding a $\mathbf{x}^* \in \mathbb{R}^n$ such that

$$||\mathbf{F}(\mathbf{x}^*)|| \leq ||\mathbf{F}(\mathbf{x})|| \quad N(\mathbf{x}) \in \mathbb{R}^n$$

where $N(\mathbf{x})$ is a neighborhood of $\mathbf{x}^*$.

If $\mathbf{x}^*$ is a solution of the nonlinear least square, then the gradient of the sum of squares of the residuals is zero at $\mathbf{x}^*$.

$$\nabla \left( \sum_{i=1}^{m} f_i(\mathbf{x})^2 \right) = 0$$

$$\sum_{i=1}^{m} f_i(\mathbf{x}) \nabla f_i(\mathbf{x}) = 0$$

In terms of Jacobian matrix, the above equation can be written as

$$\mathbf{J}(\mathbf{x})^T \mathbf{F}(\mathbf{x}) = 0$$

Note that maximum also satisfies this condition, but but the algorithm avoids the maximum.

# Basic concept of Levenberg-Marquardt algorithm

The basic concept of Levenberg-Marquard algorithm determines the correction $\mathbf{p}$ to the current estimate $\mathbf{x}$ that produces sufficient decrese in the residuals of $\mathbf{F}$ at the new point $\mathbf{x} + \mathbf{p}$.

$$\mathbf{x}_+ = \mathbf{x} + \mathbf{p}$$

In other words, the algorithm finds a $\mathbf{p}$ such that satisfies the following condition:

$$||\mathbf{F}(\mathbf{x}_+)|| \leq ||\mathbf{F}(\mathbf{x})||$$

The correction of p depends on the diagonal scaling matrix $\mathbf{D}$, a step bound $\boldsymbol{\Delta}$, and an approximation $\mathbf{J}$ to the Jacobian matrix $\mathbf{J}(\mathbf{x})$.

$\mathbf{J}$ is calculated by the foward-difference approximation to $\mathbf{F}'(\mathbf{x})$, unless the user supplies a function to calculate $\mathbf{J}$.

To compute the correction $\mathbf{p}$, the algorithm solves the linear least squares problem

$$\min\{||\mathbf{Jp}+\mathbf{f}|| : ||\mathbf{Dp}|| \leq \boldsymbol{\Delta}\} \quad (1)$$

where $\mathbf{f}$ is the vector $\mathbf{F}(\mathbf{x})$. If the solution does not provide suitable correction, then $\boldsymbol{\Delta}$ is decreased and $\mathbf{J}$ is updated. Before the start of next iteration, $\boldsymbol{\Delta}$, $\mathbf{J}$, and $\mathbf{D}$ are updated.

(1) is the approximate solution of the following problem:

$$\min \{||\mathbf{F}(\mathbf{x}+\mathbf{p})|| : ||\mathbf{D}\mathbf{p}|| \leq \Delta\}$$

if there is a solution,

$$||D(\mathbf{x}-\mathbf{x}^*)|| \leq \Delta$$

then $\mathbf{x}+\mathbf{p}$ is close to $\mathbf{x}^*$, If this is not the case then $\mathbf{x}+\mathbf{p}$ is close to $\mathbf{x}$.

# The sum of $||\mathbf{J}\mathbf{p} + \mathbf{f}||$

The sum of $||\mathbf{J}\mathbf{p} + \mathbf{f}||$ is minimized by solving the least squares problem

$$S(\mathbf{x} + \mathbf{p}) \approx ||\mathbf{y} - \mathbf{f}(\mathbf{x}) - \mathbf{J}\mathbf{p}||$$

$$= [\mathbf{y} - \mathbf{f}(\mathbf{x}) - \mathbf{J}\mathbf{p}]^T[\mathbf{y} - \mathbf{f}(\mathbf{x}) - \mathbf{J}\mathbf{p}]$$

$$[\mathbf{y} - \mathbf{f}(\mathbf{x})]^T[\mathbf{y} - \mathbf{f}(\mathbf{x})] + \mathbf{p}^T\mathbf{J}^T\mathbf{J}\mathbf{p} - 2\mathbf{p}^T\mathbf{J}^T[\mathbf{y} - \mathbf{f}(\mathbf{x})]$$

Taking the derivative with respect to $\mathbf{p}$ and setting it to zero, we obtain

$$\mathbf{J}^T\mathbf{J}\mathbf{p} = \mathbf{J}^T[\mathbf{y} - \mathbf{f}(\mathbf{x})]$$

TODO: updated this part.

# Approximation of the Jacobian matrix

The Jacobian matrix $\mathbf{J}$ is approximated by the forward-difference approximation to $\mathbf{F}'(\mathbf{x})$.

$$\mathbf{J}(\mathbf{x}) \approx \frac{\mathbf{F}(\mathbf{x} + h\mathbf{e}_i) - \mathbf{F}(\mathbf{x})}{h}$$

where $\mathbf{e}_i$ is the $i$-th column of the identity matrix and $h$ is the difference parameter.

# Error bound

Let $\mathbf{x}^*$ be a solution of the nonlinear least squares problem. For $\epsilon > 0$, the sensitivity (upper) bonds $\sigma_1, \cdots, \sigma_n$ such that, for each $i$, the condition

$$|x_i - x_i^*| \leq \sigma_i$$

with

$$||F(\mathbf{x})|| \leq (1+\epsilon)||F(\mathbf{x}^*)||$$

# First order approximation (Is this correct?)

The first order approximation of $\mathbf{F}(\mathbf{x})$ is given by

$$\sigma_i = \epsilon^{1/2}\left(\frac{||\mathbf{F}(\mathbf{x}^*)||}{||\mathbf{F}'(\mathbf{x}^*)\cdot\mathbf{e}_i||}\right)$$

If $\mathbf{x}$ is the approximation of the sollution and $\mathbf{J}$ is the approximation of $\mathbf{F}'(\mathbf{x})$, then the first order approximation of $\mathbf{F}(\mathbf{x})$ is given by

$$\sigma_i = \epsilon^{1/2}\left(\frac{||\mathbf{F}(\mathbf{x})||}{||\mathbf{J}\cdot\mathbf{e}_i||}\right)$$

The covariance matrix in this case is given by

$$\mathbf{\Sigma} = \left(\mathbf{J}^T\mathbf{J}\right)^{-1}$$

# Approximate Hessian

If we take an first order Taylor expansion of $\mathbf{F}(\mathbf{x})$ around $\mathbf{x}^*$,

$$\mathbf{F}(\mathbf{x})^T\mathbf{F}(\mathbf{x}) = \sum_{i=1}^{m} \left(f_i(\mathbf{x}^*) + \nabla f_i(\mathbf{x}^*)(\mathbf{x} - \mathbf{x}^*)\right)^2$$

The hessian matrix $\mathbf{H}$ of $\mathbf{F}(\mathbf{x})^T\mathbf{F}(\mathbf{x})$ is given by

$$\begin{aligned}\mathbf{H} = \nabla^2(\mathbf{F}(\mathbf{x})^T\mathbf{F}(\mathbf{x})) &= \sum_{i=1}^{m} \nabla f_i(\mathbf{x}^*)\nabla f_i(\mathbf{x}^*) \\ &= \mathbf{J}(\mathbf{x}^*)^T\mathbf{J}(\mathbf{x}^*)\end{aligned}$$

# How is the covariance matrix calculated in MINPACK and Scipy.optimize.leastsq()?

The covariance matrix is calculated by

$$\Sigma = \left(\mathbf{J}^T\mathbf{J}\right)^{-1} = \mathbf{P}\left(\mathbf{R}^T\mathbf{R}\right)^{-1}\mathbf{P}$$

where $\mathbf{P}$ is the permutation matrix and $\mathbf{R}$ is the upper triangular matrix obtained from the QR decomposition of $\mathbf{J}$.

$$\mathbf{PJ} = \mathbf{QR}$$

# Covariance matrix and hessian matrix

# Covariance matrix and hessian matrix

Consider a Gaussian random vector $\theta$ with mean $\theta^*$ and covariance matrix $\mathbf{\Sigma}_\theta$. It's joint probability density function is given by

$$p(\theta) = \frac{1}{(2\pi)^{n/2}|\mathbf{\Sigma}_\theta|^{1/2}}\exp\left(-\frac{1}{2}(\theta-\theta^*)^T{\mathbf{\Sigma}_\theta}^{-1}(\theta-\theta^*)\right)$$

the negative log likelihood function $J(\theta)$ is given by

$$J(\theta) = -\log p(\theta) = \frac{1}{2}(\theta-\theta^*)^T{\mathbf{\Sigma}_\theta}^{-1}(\theta-\theta^*) + \frac{1}{2}\log|\mathbf{\Sigma}_\theta| + \frac{n}{2}\log 2\pi$$

The Hessian matrix $\mathbf{H}$ of $J(\theta)$ is given by

$$\mathbf{H} = \frac{\partial^2 J(\theta)}{\partial\theta_i\partial\theta_j} = {\mathbf{\Sigma}_\theta}^{-1}$$

# Implementation used in Larch

Larch uses lmfit package for the Levenberg-Marquardt algorithm.
lmfit calls Scipy.optimize.leastsq() to call MINPACK's lmdif() and lmder() functions.

The hessian matrix return by leastsq() is calculated using an approximate hessian matrix calculated by

$$\mathbf{H} = \mathbf{J}(\mathbf{x})^T \mathbf{J}(\mathbf{x})$$

# $\chi^2$ distribution

If $Z_1, \cdots, Z_n$ are independent standard normal random variables, then the random variable

$$Q = \sum_{i=1}^{n} Z_i^2$$