Non-linear Optimization: Newton — Gauss Method

The Newton–Gauss method is a second-order optimization technique for quadratic functions, utilizing a linear approximation of the optimized function at each step. It is applied to solve nonlinear least squares problems, effectively reducing them to a sequence of linear least squares problems.

Gradient and Hessian of the Loss Function.

Given the quadratic loss function

$$Q(\boldsymbol{x}) = \sum_{\boldsymbol{x} \in X^\ell} (a(\boldsymbol{x}, \boldsymbol{\theta}) - y(\boldsymbol{x}))^2 \tag{1}$$

we can express the gradient and Hessian of the function in terms of the model's parameters:

gradient is the column vector:

$$\nabla f(\boldsymbol{x}) := \begin{pmatrix} \frac{\partial f(\boldsymbol{x})}{\partial x_1} \\ \vdots \\ \frac{\partial f(\boldsymbol{x})}{\partial x_k} \end{pmatrix},$$

and f_j' denotes jth component of the column

1. The gradient components are

$$\begin{aligned} Q_j' &= \frac{\partial Q}{\partial \theta_j} \\ &= 2 \sum_{\boldsymbol{x} \in X^\ell} (a(\boldsymbol{x}, \boldsymbol{\theta}) - y(\boldsymbol{x})) \cdot \frac{\partial a(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \theta_j} \end{aligned} \tag{2}$$

2. The Hessian components are

$$\begin{aligned} Q_{i,j}'' &= \frac{\partial^2 Q}{\partial \theta_i \partial \theta_j} \\ &= 2 \sum_{\boldsymbol{x} \in X^\ell} \frac{\partial a(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \theta_i} \frac{\partial a(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \theta_j} - 2 \sum_{\boldsymbol{x} \in X^\ell} (a(\boldsymbol{x}, \boldsymbol{\theta}) - y(\boldsymbol{x})) \cdot \frac{\partial^2 a(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \theta_i \partial \theta_j}. \end{aligned} \tag{3}$$

Linear Approximation of the Algorithm.

Apply a Taylor series expansion of the algorithm up to the linear term near the current approximation of the parameter vector $\hat{\boldsymbol{\theta}}$:

$$a(\boldsymbol{x}, \boldsymbol{\theta}) = \underbrace{a\left(\boldsymbol{x}, \hat{\boldsymbol{\theta}}\right)}_{\text{const}} + \sum_j \underbrace{\frac{\partial a\left(\boldsymbol{x}, \hat{\boldsymbol{\theta}}\right)}{\partial \theta_j}}_{\text{const}_j} \underbrace{\left(\theta_j - \hat{\theta}_j\right)}_{\delta\theta_j} + O\left(\left\|\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}\right\|^2\right), \tag{4}$$

$a\left(\boldsymbol{x}, \hat{\boldsymbol{\theta}}\right)$ is constant, and the linear term is the sum of the partial derivatives of $a\left(\boldsymbol{x}, \hat{\boldsymbol{\theta}}\right)$ with respect to the parameters θ_j. The higher-order terms are negligible and will be omitted below.

Differentiate the linear approximation of the algorithm:

$$\begin{aligned} \frac{\partial}{\partial \theta_j} a(\boldsymbol{x}, \boldsymbol{\theta}) &\approx 0 + \underbrace{\frac{\partial a\left(\boldsymbol{x}, \hat{\boldsymbol{\theta}}\right)}{\partial \theta_j}}_{\text{const}_k} \cdot 1 + \cancel{O\left(\left\|\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}\right\|^2\right)} \\ &= \text{const}_j \end{aligned} \tag{5}$$

The components of the sum depending on $\theta_{j \neq k}$ was zeroed out in the differentiation over θ_k.

Substitute the obtained derivative into the expression for the Hessian:

$$Q_{i,j}'' \approx 2 \sum_{\boldsymbol{x} \in X^\ell} \underbrace{\frac{\partial a\left(\boldsymbol{x}, \hat{\boldsymbol{\theta}}\right)}{\partial \theta_i}}_{\text{const}_i} \underbrace{\frac{\partial a\left(\boldsymbol{x}, \hat{\boldsymbol{\theta}}\right)}{\partial \theta_j}}_{\text{const}_j} - \cancel{2 \sum_{\boldsymbol{x} \in X^\ell} (a(\boldsymbol{x}, \boldsymbol{\theta}) - y(\boldsymbol{x})) \cdot 0} \tag{6}$$

The linear term will be zeroed out in the second differentiation and will not enter the Hessian.

Matrix Formulation of the Optimization Step.

Introduce the matrix of first partial derivatives and the algorithm's response vector at the current approximation of the parameters $\hat{\boldsymbol{\theta}}$:

$$D := \left\{ \frac{\partial a\left(\boldsymbol{x}_i, \hat{\boldsymbol{\theta}}\right)}{\partial \theta_j} \right\}_{i,j}, \quad \boldsymbol{a} := \begin{pmatrix} a\left(\boldsymbol{x}_1, \hat{\boldsymbol{\theta}}\right) \\ \vdots \\ a\left(\boldsymbol{x}_\ell, \hat{\boldsymbol{\theta}}\right) \end{pmatrix} \tag{7}$$

matrix D and vector $\boldsymbol{a}$ depend on the point of expansion $\hat{\boldsymbol{\theta}}$ and are recalculated at each optimization step.

The gradient and Hessian (at each step) are calculated using the matrix D:

$$Q' = D^\mathsf{T} \, (\boldsymbol{a} - \boldsymbol{y}), \quad Q'' = D^\mathsf{T} \, D(\boldsymbol{a} - \boldsymbol{y}) \tag{8}$$

The optimization step of the Newton — Rafson method is also expressed in terms of the matrix D:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma \cdot \overbrace{\underbrace{\left(D^\mathsf{T} \, D\right)^{-1} D^\mathsf{T}}_{D^+} \, \underbrace{(\boldsymbol{a} - \boldsymbol{y})}_{\boldsymbol{\varepsilon}}}^{\delta\boldsymbol{\theta}} \tag{9}$$

Newton — Rafson method is a second-order optimization technique that provides fast convergence. Newton–Gauss method is an approximate second-order method that uses a linear approximation of the optimized function at each step.

The optimization step vector at each iteration can be determined from the linear system in any of these formulations:

$$\underbrace{\boldsymbol{\varepsilon}}_{\boldsymbol{y}} = D \cdot \underbrace{\delta\boldsymbol{\theta}}_{\boldsymbol{\beta}} \quad \Leftrightarrow \quad \delta\boldsymbol{\theta} = D^+ \boldsymbol{\varepsilon} \quad \Leftrightarrow \quad \|D \cdot \delta\boldsymbol{\theta} - \boldsymbol{\varepsilon}\|^2 \to \min_{\boldsymbol{\beta}} \tag{10}$$

The nonlinear optimization problem is reduced to a sequence of linear problems: at each iteration, a linear expansion of the function is made, matrices are calculated, and a (new) system of linear equations is solved.

The method is a second-order approximation method, providing fast convergence and slightly inferior accuracy compared to the Newton–Raphson method.