Second Order Optimization

Introduction

The second order algorithm is any algorithm that uses any second derivative, in the scalar case.
Compared with first order methods such as gradient descent, second order methods basically have the following advantages:
1. A full step jumps directly to the minimum of the local squared approx.
2. Often this is already a good heuristic.
3. More efficient on solving nonlinear problems such as solving root, minimizer and least square

There are many **second order optimization** methods, including Newton Method, Gauss-Newton Method and The Levenberg – Marquardt Method.

Newton Method

For finding roots (zero points) of f(x):

Newton's method is a root-finding algorithm which produces successively better approximations to the roots (or zeroes) of a real-valued function. The most basic version starts with a single-variable function *f* defined for a real variable *x*, the function's derivative f', and an initial guess x_0 for a root of *f*. If the function satisfies sufficient assumptions and the initial guess is close, then

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

For finding optima of f(x) in 1D:

Similarly, based on the second order Taylor expansion, $f(x) = f(x^{(0)}) + f'(x^{(0)})(x - x^{(0)}) + \frac{1}{2}f''(x - x^{(0)})^2$

Deriving from the above formula and setting $f'(x) = 0$

This method can be used to find extremum value(not always global)

$$x = x^{(0)} - \frac{f'(x^{(0)})}{f''(x^{(0)})}$$

For High dimensional function, sililarly, the Iterative formula is $\theta := \theta - H^{-1}J(\theta)$
when

$$\nabla^2 f(x) = \begin{pmatrix} \frac{\partial^2}{\partial_{x_1}\partial_{x_1}} f(x) & \frac{\partial^2}{\partial_{x_1}\partial_{x_2}} f(x) & \cdots & \frac{\partial^2}{\partial_{x_1}\partial_{x_n}} f(x) \\ \frac{\partial^2}{\partial_{x_1}\partial_{x_2}} f(x) & & & \vdots \\ \vdots & & & \vdots \\ \frac{\partial^2}{\partial_{x_n}\partial_{x_2}} f(x) & \cdots & \cdots & \frac{\partial^2}{\partial_{x_n}\partial_{x_n}} f(x) \end{pmatrix} \in \mathbb{R}^{n\times n} \qquad J(\theta) = \begin{bmatrix} J_1 \\ \vdots \\ J_n \end{bmatrix},$$

Here, the first one is the Hessian Matrix, which is a square matrix of second-order partial derivatives of a scalar-valued function, or scalar field (It describes the local curvature of a function of many variables).
J denotes the Jacobian, which is a vector of partial derivatives of all parameters.

Gauss-Newton Method

To solve an extrem complex non-linear problem, it will take a huge computing resources for Newton method to calculate the Hessian matrix. Gauss-Newton method is using $2\nabla r^T \nabla r$ to approxiamate Hessian matrix, which can improve the efficient.

Levenberg – Marquardt Method

The Levenberg-Marquardt method acts more like a gradient-descent method when the parameters are far from their optimal value, and acts more like the Gauss-Newton method when the parameters are close to their optimal value. This document describes these methods and illustrates the use of software to solve nonlinear least squares curve-fitting problems.