Netwon-Like Methods

$x^{(t+1)} = x^{(t)} - [g''(x^{(t)})]^{-1} g'(x^{(t)})$: Newton's method

↳ Hessian matrix : Need to calculate second derivative

Some very effective methods rely on updating equations of the form

Multivatiate format

$l(\alpha, \beta)$ Hessian matrix

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - (\mathbf{M}^{(t)})^{-1} g'(\mathbf{x}^{(t)}),$$

$$\boxed{p\times 1} = p\times 1 - (p\times p)^{-1} (p\times 1)$$

where $\mathbf{M}^{(t)}$ is a $p \times p$ matrix approximating the Hessian, $g''(\mathbf{x}^{(t)})$.

$p \times p$ → # of parameter

In general optimization problems, there are several good reasons to consider replacing the Hessian by some simpler approximation.

- It may be computationally expensive to evaluate the Hessian.

  calculation of inverse matrix : computationally demanding

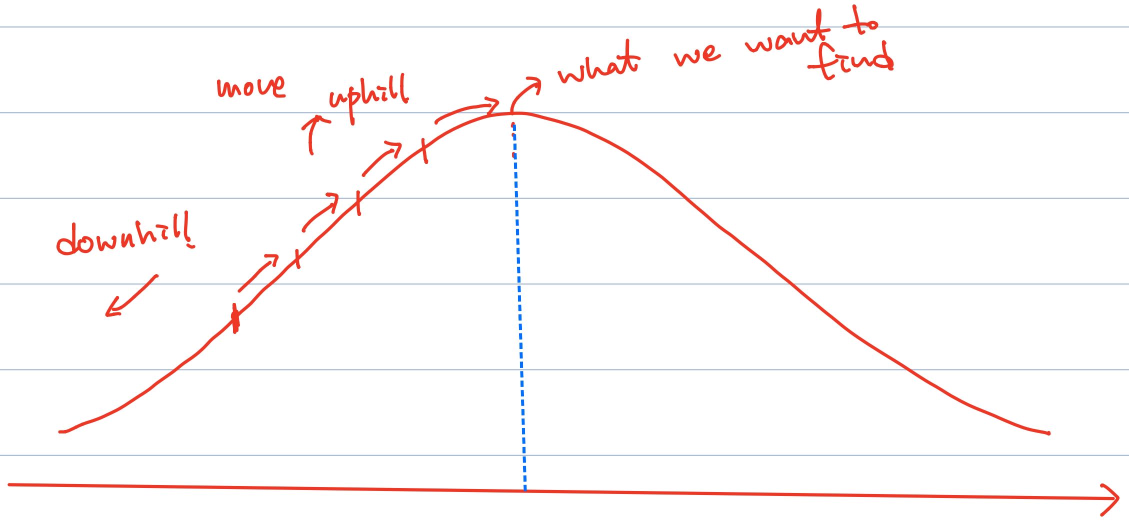
- The steps taken by Newton's method are not necessarily always uphill: At each iteration, there is no guarantee that $g(\mathbf{x}^{(t+1)}) > g(\mathbf{x}^{(t)})$. A suitable $\mathbf{M}^{(t)}$ can guarantee ascent.

If we chose a suitable $M^{(t)}$, it will guarantee the algorithm move uphill

ℓ Hassian matrix $= \begin{bmatrix} \frac{\partial^2 \ell(\alpha,\beta)}{\partial \alpha^2} & \frac{\partial^2 \ell(\alpha,\beta)}{\partial \alpha \partial \beta} \\ \frac{\partial^2 \ell(\alpha,\beta)}{\partial \alpha \partial \beta} & \frac{\partial^2 \ell(\alpha,\beta)}{\partial \beta^2} \end{bmatrix}$

As the number of parameter increase, the # of second derivative calculation increase exponentially.

⇒ In multivariate case, Hessian matrix calculation is computationally expansive.



Newton method does not guarantee the algorithm to move uphill

Ascent Algorithms

minimization problem : gradient descent algorithm
fixed-point iteration method => multivariate version

- To force uphill steps, one could resort to an ascent algorithm.
- The method of steepest ascent is obtained with the Hessian replacement $\mathbf{M}^{(t)} = -\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. inverse matrix.

  $M^{(t)} = -I$ => What is advantage ? We don't need to calculate

- Since the gradient of g indicates the steepest direction uphill on the surface of g at the point $\mathbf{x}^{(t)}$, setting $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + g'(\mathbf{x}^{(t)})$ amounts to taking a step in the direction of steepest ascent. $x^{(t+1)} =$

  When we use the identity matrix, => fixed point iteration method

- Scaled steps of the form $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \alpha^{(t)} g'(x(t))$ for some $\alpha^{(t)} > 0$ can be helpful for controlling convergence. $x^{(t+1)} = x^{(t)} + \alpha^{(t)} g'(x^{(t)})$

  we can change $\alpha^{(t)}$. by iteration.

- If $-\mathbf{M}^{(t)}$ is positive definite, ascent can be assured by choosing $\alpha^{(t)}$ sufficiently small, yielding $\underbrace{g(\mathbf{x}^{(t+1)}) - g(\mathbf{x}^{(t)})}_{\text{move uphill} \uparrow} > 0$.

Ascent Algorithms

↳ Deep learning use a lot $\alpha^{(t)}$: learning rate

- Therefore, an ascent algorithm involves a positive definite matrix $-\mathbf{M}^{(t)}$ to approximate the negative Hessian, and a contraction or step length parameter $\alpha^{(t)} > 0$ whose value can shrink to ensure ascent at each step.

$\alpha^{(t)} = 1 \Rightarrow$ update parameter values

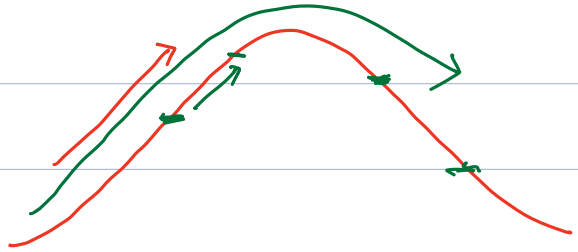
- For example,

$g(x^{(t+1)}) - g(x^{(t)}) > 0$

If $g(x^{(t+1)}) - g(x^{(t)}) < 0$, it mean

  - Start each step with $\alpha^{(t)} = 1$. If the original step turns out to be downhill, $\alpha^{(t)}$ can be halved. This is called backtracking.
  - If the step is still downhill, $\alpha^{(t)}$ is halved again until a sufficiently small step is found to be uphill.

- For Fisher scoring, $-\mathbf{M}^{(t)} = I(\boldsymbol{\theta}^{(t)})$, which is positive semidefinite. Therefore backtracking with Fisher scoring would avoid stepping downhill.



$$g(x^{(t+1)}) - g(x^{(t)}) \geq 0$$

If $\alpha^{(t)}$ is too large,

then we need to scale down.

$$\alpha^{(t)} = 1/2\, \alpha^{(t)}$$

Discrete Netwon and Fixed-Point Methods

↳ Secant method

- To avoid calculating the Hessian, one could resort to a secant-like method, yielding a discrete Newton method, or rely solely on an initial approximation, yielding a multivariate fixed-point method.
- Multivariate fixed-point methods use an initial approximation of g'' throughout the iterative updating.
- If this approximation is a matrix of constants, so $\mathbf{M}^{(t)} = \mathbf{M}$ for all t, then the updating equation is

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \mathbf{M}^{-1} g'(\mathbf{x}^{(t)}).$$

- A reasonable choice for $\mathbf{M}$ is $g''(\mathbf{x}^{(0)})$. Notice that if $\mathbf{M}$ is diagonal, then this amounts to applying the univariate scaled fixed-point algorithm separately to each component of g.

Gauss-Netwon Method

least square $y = x\beta + \varepsilon$

Nonlinear least squares problems with observed data $(y_i, \mathbf{z}_i)$ for $i = 1, \cdots, n$.

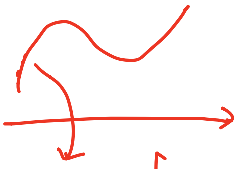
$y_i = f(z_i, \theta) + \varepsilon_i$

- Seeks to estimate θ by maximizing an objective function

min $\sum_{i=1}^{n} (y_i - f(z_i, \theta))^2$

↳ our objective

$$g(\boldsymbol{\theta}) = -\sum_{i=1}^{n} (y_i - f(\mathbf{z}_i, \boldsymbol{\theta}))^2 .$$



nonlinear format

$\boxed{f(z_i, \theta)}$

↳ alternatively   max $-\sum_{i=1}^{n} (y_i - f(z_i, \theta))^2$

- Such objective functions might be sensibly used, when estimating θ to fit the model

$$Y_i = f(\mathbf{z}_i, \boldsymbol{\theta}) + \epsilon_i$$

for some nonlinear function f and random error ϵ_i.

Rather than approximate g, the Gauss-Newton approach approximates f itself by its linear Taylor series expansion about $\boldsymbol{\theta}^{(t)}$. Replacing f by its linear approximation yields a linear least squares problem, which can be solved to derive an update $\boldsymbol{\theta}^{(t+1)}$.

Gauss-Netwon Method

$y_i = f(z_i, \theta) + \varepsilon_i \approx \underline{f(z_i, \theta^{(t)}) + (\theta - \theta^{(t)})^T f'(z_i, \theta^{(t)})} + \varepsilon$

$= \tilde{f}(z_i, \theta^{(t)}) + \varepsilon_i$

- The nonlinear model can be approximated by

$$Y_i \approx f\left(\mathbf{z}_i, \boldsymbol{\theta}^{(t)}\right) + \left(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t)}\right)' f'\left(\mathbf{z}_i, \boldsymbol{\theta}^{(t)}\right) + \epsilon_i = \tilde{f}\left(\mathbf{z}_i, \boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}\right) + \epsilon_i.$$

- A Gauss-Newton step is derived from the maximization of

$\max \tilde{g}(\theta) = -\sum_{i=1}^{n} (y_i - \tilde{f}(z_i, \theta^{(t)}))^2$

$$\tilde{g}(\boldsymbol{\theta}) = -\sum_{i=1}^{n} \left(y_i - \tilde{f}\left(\mathbf{z}_i, \boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}\right)\right)^2$$

with respect to $\boldsymbol{\theta}$, whereas a Newton step is derived from the maximization of a quadratic approximation to g itself,

$$g\left(\boldsymbol{\theta}^{(t)}\right) + \left(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t)}\right)' g'\left(\boldsymbol{\theta}^{(t)}\right) + \left(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t)}\right)' g''\left(\boldsymbol{\theta}^{(t)}\right)\left(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t)}\right).$$

Gauss-Netwon Method

$$x_i^{(t)} = y_i - f(z_i, \theta^{(t)}) = f(z_i, \cancel{\theta^{(t)}}) + (\theta - \theta^{(t)})^T f'(z_i, \theta^{(t)}) - f(\cancel{z_i, \theta}^{(t)}) + \varepsilon_i$$

$$= (\theta - \theta^{(t)})^T \underline{f'(z_i, \theta^{(t)})} + \varepsilon_i = (\theta - \theta^{(t)})^T \boxed{a_i^{(t)}} + \varepsilon_i$$

$a_i^{(t)}$

matrix A

Let $X_i^{(t)}$ denote a working response whose observed value is

$$x_i^{(t)} = y_i - f\left(\mathbf{z}_i, \boldsymbol{\theta}^{(t)}\right),$$

and define $\mathbf{a}_i^{(t)} = f'\left(\mathbf{z}_i, \boldsymbol{\theta}^{(t)}\right)$. Then the approximated problem can be reexpressed as minimizing the squared residuals of the linear regression model

$$\mathbf{X}^{(t)} = \mathbf{A}^{(t)}\left(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t)}\right) + \epsilon. \tag{3}$$

$Y_i = X\beta + \varepsilon_i$

$\hat{\beta} = (X^T X)^{-1} X^T y$

$$X^{(t)} = A^{(t)}(\theta - \theta^{(t)}) + \varepsilon$$

$$\theta - \theta^{(t)} = ((A^{(t)})^T A^{(t)})^{-1} (A^{(t)})^T X^{(t)}$$

Gauss-Netwon Method

The minimal squared error for fitting equation (3) is achieved when

$$\left(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t)}\right) = \left(\left(A^{(t)}\right)' A^{(t)}\right)^{-1} \left(A^{(t)}\right)' \mathbf{x}^{(t)}.$$

$a_i^{(t)} = f'(z_i, \theta^{(t)})$

$x_i^{(t)} = y_i - f(z_i, \theta^{(t)})$

Thus, the Gauss-Newton update for $\boldsymbol{\theta}^{(t)}$ is

$\theta^{(t+1)} = \theta^{(t)} + ((A^{(t)})^T A^{(t)})^{-1} (A^{(t)})^T x^{(t)}$

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \left(\left(A^{(t)}\right)' A^{(t)}\right)^{-1} \left(A^{(t)}\right)' \mathbf{x}^{(t)}.$$

- The potential advantage of the Gauss-Newton method is that it does not require computation of the Hessian.

converge. X.

- It is fast when f is nearly linear or when the model fits well.

will cause large residual.

- In other situations, particularly when the residuals at the true solution are large because the model fits poorly, the method may converge very slowly or not at all-even from good starting values.