Analysis of gradient descent and extensions

Gradient descent is an optimization algorithm for finding a local minimum of a differentiable function. When we minimize a function $f(x_1,\ldots,x_n)$, the algorithm uses the derivatives $\partial f/\partial x_i$ for searching a direction that reduces *f(x)*. The steepest direction, in which *f(x)* decreases fastest, is given by the gradient $-\nabla f$.

Gradient descent $\boldsymbol{x_{k+1} = x_k - s_k \nabla f(x_k) \longrightarrow f(x_{k+1}) = f(x_k - s_k \nabla f(x_k))}$

In the equation above the variable s_k is a hyperparameter called learning rate or step size. The searching of the hyperparameter $\mathrm{s_k}$ would usually from [0.001,0.01,0.1,1,10,100] because when s_k is too small or too large, the loss function (our objective function in machine learning) would be converged before getting to the real minimum or fluctuated between high and low. We keep calculating the x_{k+1} before a local minimum is found. The equation below shows that the steepest direction is **perpendicular to the level direction**. Because when the algorithm stops it should be 0.

$$\frac{\partial f(\mathbf{x_{k+1}})}{\partial\, \mathbf{s_k}} = \nabla f(\mathbf{x_{k+1}}) \cdot \frac{\partial \mathbf{x_{k+1}}}{\partial\, \mathbf{s_k}} = \nabla f(\mathbf{x_{k+1}}) \cdot \frac{\partial}{\partial\, \mathbf{s_k}}(\mathbf{x_k} - \mathbf{s_k}\nabla f(\mathbf{x_k})) = \nabla f(\mathbf{x_{k+1}}) \cdot \nabla \mathbf{f}(\mathbf{x_k})$$

Below is an example of function $f(x,y) = \frac{1}{2}(x^2 + by^2)$ starting from $(x_0, y_0) = (b, 1)$. We can find the equation below:

$$f(x_k, y_k) = \left(\frac{1-b}{1+b}\right)^{2k} f(x_0, y_0)$$

We would find when b is small, the ratio is approaching 1 and virtually feezed. For handling this, we calculating the function's Hessian matrix H_{ij} and see the eigenvalues (set as $m \le \lambda \le \mathrm{M}$). In this case, the eigenvalues are 1 and b.

$$\boldsymbol{f(x_{k+1}) \le f(x_k) - s \cdot \|\nabla f\|^2 + \frac{Ms^2}{2}\|\nabla f\|^2}$$

We need to minimize the left side; the minimum of the right side is s=1/M. Substituting 1/M into the equation we would finally get the equation below, which says that every step decreases at least $\boldsymbol{1 - \frac{m}{M}}$. That's the linear convergence.

$$\boldsymbol{f(x_{k+1}) - f(x^*) \le (1 - \frac{m}{M})(f(x_k) - f(x^*))}$$

By the way, when b is small, the reduction factor is $\left(\frac{1-b}{1+b}\right)^{2k} \approx 1 - 4b$, which is four times better than the above inequalities.

What we can not do is to build a formula for minimizing a general function. So, another way is created. For example, **backtracking**, with the algorithm showed below:

Test If $f(\boldsymbol{X}) \le f(\boldsymbol{x}_k) - \frac{s}{3}||\nabla \boldsymbol{f}_k||^2$, with $s = 1$, stop and accept $\boldsymbol{X}$ as $\boldsymbol{x}_{k+1}$.

Otherwise backtrack: Reduce s to $\frac{1}{2}$ and try the test on $\boldsymbol{X} = \boldsymbol{x}_k - \frac{1}{2}\nabla \boldsymbol{f}_k$.