



Figure 8.6: The method of steepest descent applied to a quadratic cost surface. The method of steepest descent involves jumping to the point of lowest cost along the line defined by the gradient at the initial point on each step. This resolves some of the problems seen with using a fixed learning rate in figure 4.6, but even with the optimal step size the algorithm still makes back-and-forth progress toward the optimum. By definition, at the minimum of the objective along a given direction, the gradient at the final point is orthogonal to that direction.

the form:

$$\mathbf{d}_t = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) + \beta_t \mathbf{d}_{t-1} \quad (8.29)$$

where β_t is a coefficient whose magnitude controls how much of the direction, \mathbf{d}_{t-1} , we should add back to the current search direction.

Two directions, \mathbf{d}_t and \mathbf{d}_{t-1} , are defined as conjugate if $\mathbf{d}_t^\top \mathbf{H} \mathbf{d}_{t-1} = 0$, where \mathbf{H} is the Hessian matrix.

The straightforward way to impose conjugacy would involve calculation of the eigenvectors of \mathbf{H} to choose β_t , which would not satisfy our goal of developing a method that is more computationally viable than Newton’s method for large problems. Can we calculate the conjugate directions without resorting to these calculations? Fortunately the answer to that is yes.

Two popular methods for computing the β_t are:

1. Fletcher-Reeves:

$$\beta_t = \frac{\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_t)^\top \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_t)}{\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_{t-1})^\top \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_{t-1})} \quad (8.30)$$