Gradient descent: (# of iterations for $f(x^k) - f(x^*) \le \epsilon$)

f is Lipschitz: $k = O(1/\epsilon^2)$ [$t_k \approx 1/\sqrt{k}$]

$\nabla f(x)$ is Lipschitz: $k = O(1/\epsilon)$ [t_k from BTLS]

f is strongly convex & Lipschitz: $k = O(\log(1/\epsilon))$ [t_k from BTLS]

Newton:

$\nabla^2 f$ is Lipschitz & f is strongly convex $\to$ Damped sublinear

$\to$ Quad conv. $O(\log\log(1/\epsilon))$

Stochastic Gradient: $f = E(f_i(x))$

At i^{th} iteration: $x^{k+1} = x^k - \nabla f_i(x^k)$

f is Lipschitz: $O(1/\epsilon^2)$

f is strongly convex & ∇f is Lipschitz: $O\left(\frac{\log(1/\epsilon)}{\epsilon}\right)$

Low cost per iteration!

- In machine learning: $f = E\left(\left(\omega^T \phi(x_i) - y_i\right)^2\right)$ (x_i, y_i) are i/p & o/p for i^{th} example in a dataset . . .
- Stochastic gradient descent: More interesting from generalization perspective (learning theory)

Proof: If f is Lipschitz & convex

This proof is exactly the same proof as that for subgradient descent algorithm with Lipschitz continuity and convexity assumptions on the function:
http://www.seas.ucla.edu/~vandenbe/236C/lectures/sgmethod.pdf

Implementation

main effort in each iteration: evaluate derivatives and solve Newton system

$$H\Delta x = g$$

where $H = \nabla^2 f(x)$, $g = -\nabla f(x)$

via Cholesky factorization

$$H = LL^T, \qquad \Delta x_{\rm nt} = L^{-T}L^{-1}g, \qquad \lambda(x) = \|L^{-1}g\|_2$$

- cost $(1/3)n^3$ flops for unstructured system
- cost $\ll (1/3)n^3$ if H sparse, banded

example of dense Newton system with structure

$$f(x) = \sum_{i=1}^{n} \psi_i(x_i) + \psi_0(Ax+b), \qquad H = D + A^T H_0 A$$

- assume $A \in \mathbf{R}^{p\times n}$, dense, with $p \ll n$
- D diagonal with diagonal elements $\psi_i''(x_i)$; $H_0 = \nabla^2 \psi_0(Ax+b)$

method 1: form H, solve via dense Cholesky factorization: (cost $(1/3)n^3$)

method 2 (page 9–15): factor $H_0 = L_0 L_0^T$; write Newton system as

$$D\Delta x + A^T L_0 w = -g, \qquad L_0^T A \Delta x - w = 0$$

eliminate Δx from first equation; compute w and Δx from

$$(I + L_0^T A D^{-1} A^T L_0) w = -L_0^T A D^{-1} g, \qquad D\Delta x = -g - A^T L_0 w$$

cost: $2p^2 n$ (dominated by computation of $L_0^T A D^{-1} A^T L_0$)

Implementation

main effort in each iteration: evaluate derivatives and solve Newton system

$$H\Delta x = g$$

where $H = \nabla^2 f(x)$, $g = -\nabla f(x)$

via Cholesky factorization

$$H = LL^T, \qquad \Delta x_{\rm nt} = L^{-T}L^{-1}g, \qquad \lambda(x) = \|L^{-1}g\|_2$$

- cost $(1/3)n^3$ flops for unstructured system
- cost $\ll (1/3)n^3$ if H sparse, banded

example of dense Newton system with structure

$$f(x) = \sum_{i=1}^{n} \psi_i(x_i) + \psi_0(Ax+b), \qquad H = D + A^T H_0 A$$

- assume $A \in \mathbf{R}^{p\times n}$, dense, with $p \ll n$
- D diagonal with diagonal elements $\psi_i''(x_i)$; $H_0 = \nabla^2\psi_0(Ax+b)$

method 1: form H, solve via dense Cholesky factorization: (cost $(1/3)n^3$)

method 2 (page 9–15): factor $H_0 = L_0 L_0^T$; write Newton system as

$$D\Delta x + A^T L_0 w = -g, \qquad L_0^T A\Delta x - w = 0$$

eliminate Δx from first equation; compute w and Δx from

$$(I + L_0^T A D^{-1} A^T L_0) w = -L_0^T A D^{-1} g, \qquad D\Delta x = -g - A^T L_0 w$$

cost: $2p^2 n$ (dominated by computation of $L_0^T A D^{-1} A^T L_0$)