

Operations Research Techniques and Algorithms (620-361)

Dr Yao-ban Chan

y.chan@ms.unimelb.edu.au

Telephone: 8344 9073

Office: Room 198, Richard Berry Building

Christina Burt

c.burt@ms.unimelb.edu.au

Telephone: 8344 1797

Office: 139 Barry St

Wednesday 2nd April, 2008

Today's Lecture

Unimodal n -D unconstrained optimisation
Quasi-Newton

The main reason that Newton's Method converges faster than the steepest descent method, in some circumstances, is that it uses a better approximation of f to determine x^{k+1} from x^k .

The Steepest Descent Method approximates f using the affine (linear plus constant) tangent function

$$f(x) \approx f(x^k) + \langle \nabla f(x^k), x - x^k \rangle,$$

which takes into account the first two terms of the Taylor series for f at x^k . The descent direction is then taken to be the direction of steepest descent of the tangent plane, which is equivalent to the "instantaneous" direction of steepest descent of the original function f .

In contrast, Newton's Method approximates f using the quadratic function

$$f(x) \approx Q_k(x) = f(x^k) + \langle \nabla f(x^k), x - x^k \rangle + \frac{1}{2} \langle x - x^k, \nabla^2 f(x^k)(x - x^k) \rangle,$$

which consists of the first three terms of the Taylor series. The next iterate is taken to be the vector x^{k+1} which corresponds to the unique minimum of $Q_k(x)$ (provided the Hessian of f is invertible at x^k).

To find this, we solve $\nabla Q_k(x) = 0$:

$$0 = \nabla Q_k(x) = \nabla f(x^k) + \nabla^2 f(x^k)(x - x^k)$$

$$\nabla^2 f(x^k)(x - x^k) = -\nabla f(x^k)$$

$$x = x^k + \nabla^2 f(x^k)^{-1} \nabla f(x^k)$$

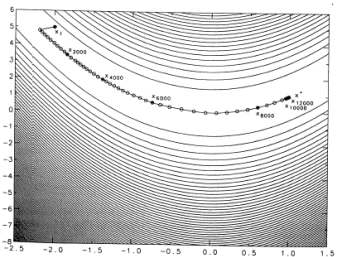
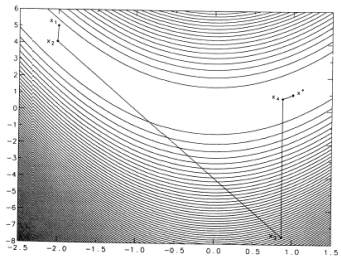
which is exactly the x^k plus the Newton direction! So taking step size $t_k = 1$ solves the approximation exactly.

To illustrate the faster convergence of Newton's Method over steepest descent, consider the problem

$$\min f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

The next figure ¹ shows the steps taken by Newton's Method, and those taken by the steepest descent method, to converge to an optimal solution. The former takes substantially fewer steps: it requires only 5 steps as opposed to the more than 12,000 required by steepest descent.

¹Courtesy of *Numerical Methods and Software*, D. Kahaner, C. Moler and S. Nash, Prentice-Hall, 1989.



Example.

Apply Newton's Method to the problem of minimizing

$$f(x) = x_1^2 x_2^2 + 2x_1^2 + 2x_2^2 - 4x_1 + 4x_2 \text{ using initial point } x^0 = (0, 0)^T.$$

Exercise.

Apply Newton's Method to the problem of minimizing

$$f(x) = 3x - 1 + \frac{3}{x}.$$

Consider what happens if you try different initial points, such as $x^0 = \frac{1}{2}$, or $x^0 = \frac{5}{4}$, or $x^0 = 2$, or $x^0 = 3$. Sketch the results on a graph. In which cases would you expect the sequence of points generated to have a cluster point?

Quasi-Newton methods

There are two particular drawbacks of Newton's method. The first is that we must be able to compute the Hessian matrix $\nabla^2 f(x^k)$ for each k . While this is straightforward for some problems, for many problems with large numbers of variables, just writing the mathematical expression of the Hessian matrix can be a major task, and the job of translating the mathematics into a computer code can be very error prone and time consuming.

The second drawback is that determining the Newton direction requires solution of a system of n linear equations in n unknowns, in particular finding d^k which solves

$$0 = \nabla f(x^k) + \nabla^2 f(x^k)d^k. \quad (1)$$

Using Gaussian elimination to calculate d^k requires around $n^3/3$ flops (floating point operations such as adding, subtracting, multiplying or dividing two numbers on a computer).

To give some perspective on the cost of Gaussian elimination, if we are given a problem with twice as many variables, $2n$ in all, then each time we determine d^k it takes $(2n)^3/2 = 8(n^3/2)$ flops, that is 8 times more work than before. In other words, the amount of work required per iteration of Newton's method increases dramatically as n increases.

For these reasons, approximate Newton or quasi-Newton methods have become a very important theoretical and practical part of optimization.

The basic idea is that, rather than defining the Newton direction $d^k = -\nabla^2 f(x^k)^{-1} \nabla f(x^k)$, we construct a matrix $H_k \approx \nabla^2 f(x^k)^{-1}$ and let

$$d^k = -H_k \nabla f(x^k);$$

this vector d^k is called the *quasi-Newton* direction.

Quasi-Newton methods can be relatively simple and cheap: $\nabla^2 f(x^k)$ is not needed, and the linear system $0 = \nabla f(x^k) + \nabla^2 f(x^k)d^k$ does not need to be solved. Note that the matrix-vector multiplication $H_k \nabla f(x^k)$ takes only around n^2 flops, and $n^2 \ll n^3/2$ for even moderate sized n .

Two of the most basic quasi-Newton methods are the DFP (Davidon-Fletcher-Powell) method and the BFGS (Broyden-Fletcher-Goldfarb-Shanno) method. The latter is generally accepted as the better method for optimization, hence we shall concentrate on it.

The secant method, familiar from solving a single equation with 1 variable, motivates the choice of H_k .

Suppose we want to find the minimum of C^2 function $f : \mathfrak{R} \rightarrow \mathfrak{R}$. This is the same as finding the root of the C^1 function $g = f' : \mathfrak{R} \rightarrow \mathfrak{R}$. Instead of using Newton's method,

$$\begin{aligned}x^{k+1} &= x^k - \frac{f'(x^k)}{f''(x^k)} \\ &= x^k - \frac{g(x^k)}{g'(x^k)},\end{aligned}$$

we approximate $f''(x^k) = g'(x^k)$ by the gradient of the secant on the graph of g which joins the $(x^{k-1}, g(x^{k-1}))$ to $(x^k, g(x^k))$.

The gradient of this secant is

$$\frac{g(x^k) - g(x^{k-1})}{x^k - x^{k-1}},$$

and the secant method calculates x^{k+1} as

$$\begin{aligned}x^{k+1} &= x^k - \frac{(x^k - x^{k-1})}{g(x^k) - g(x^{k-1})} g(x^k) \\ &= x^k - \frac{(x^k - x^{k-1})}{f'(x^k) - f'(x^{k-1})} f'(x^k)\end{aligned}$$

Putting it another way, the secant method sets

$$H_k = \left(\frac{f'(x^k) - f'(x^{k-1})}{x^k - x^{k-1}} \right)^{-1} = \frac{x^k - x^{k-1}}{f'(x^k) - f'(x^{k-1})}, \quad (2)$$

and $x^{k+1} := x^k - H_k f'(x^k)$.

In general, if Newton's Method for finding the minimum of f converges quadratically to x^* , the secant method converges superlinearly. So we hope, in higher dimensions ($n \geq 2$), that we can construct H_k in a relatively cheap way and still get superlinear convergence, without the cost of computing $\nabla^2 f(x^k)$ or solving an $n \times n$ linear system.

For $n \geq 2$, H_k is usually chosen

1. to be symmetric, because $\nabla^2 f(x^k)$, hence $\nabla^2 f(x^k)^{-1}$ is symmetric;
2. to satisfy

$$x^{k+1} - x^k = H_{k+1}(\nabla f(x^{k+1}) - \nabla f(x^k)) \quad (3)$$

3. to be positive definite.

The secant equation (3) is the n -dimensional analogue of the single-variable secant equation.

Symmetry and positive definiteness of H_k are useful because they ensure that the quasi-Newton direction $d^k = -H_k \nabla f(x^k)$ is a descent direction. This follows from Lemma 7 below which generalizes the fact that positive definiteness of $\nabla^2 f(x^k)$ implies the Newton direction is a descent direction.

Lemma 7:

Let B be a symmetric matrix in $\mathbb{R}^{n \times n}$. Then B is positive definite if and only if

- ▶ it is invertible and B^{-1} is positive definite.

If, in addition, $\nabla f(x) \neq 0$ then both $-B^{-1} \nabla f(x)$ and $-B \nabla f(x)$ are descent directions for f at x .

Proof: (only if) Let $d \neq 0$. Then, since B is positive definite, we know that $d^T B d > 0$, which implies that $Bd \neq 0$. Thus, for any $d \neq 0$ we know that $Bd \neq 0$. The fact that B is invertible follows from the well known result from linear algebra that a square matrix B is invertible if and only if $d \neq 0$ implies $Bd \neq 0$.

Alternatively we can set d to be an eigenvector of B with eigenvalue λ . Then $d^T B d = d^T \lambda d = \lambda \|d\|^2 > 0$, so $\lambda > 0$. Therefore the determinant is nonzero and B is invertible.

Again let $d \neq 0$. To show that B^{-1} is positive definite, we need to show that $d^T B^{-1} d > 0$. This follows since

$$\begin{aligned}d^T B^{-1} d &= d^T B^{-1} B B^{-1} d \\&= \left((B^{-1})^T d \right)^T B (B^{-1} d) \\&= (B^{-1} d)^T B (B^{-1} d) \\&> 0\end{aligned}$$

where the third equation follows because B is symmetric and the final inequality because B is positive definite.

(if) A similar argument with B^{-1} and B interchanged can be used to show that if B is invertible and B^{-1} is positive definite then B is positive definite.

To show that $-B^{-1} \nabla f(x)$ is a descent direction, we need to show that $\nabla f(x)^T (-B^{-1} \nabla f(x)) < 0$. This follows immediately because B^{-1} is positive definite. A similar argument shows that $-B \nabla f(x)$ is a descent direction.