Algorithms for Unconstrained Optimization

LI-PING ZHANG

Department of Mathematical Sciences

Tsinghua University, Beijing 100084

Office: New Science Building #A302, Tel: 62798531

E-mail: lipingzhang@tsinghua.edu.cn

Outline

- Steepest Descent Method
- Newton Method
- Conjugate Direction Method
- Quasi-Newton Method

Introduction

Optimization algorithms tend to be iterative procedures.

Starting from a given point $\mathbf{x}^0$, they generate a sequence $\{\mathbf{x}^k\}$ of iterates (or trial solutions).

We study algorithms that produce iterates according to well determined rules–Deterministic Algorithm rather than some random selection process–Randomized Algorithm. 确定性的

The rules to be followed and the procedures that can be applied depend to a large extent on the characteristics of the problem to be solved.

Classes of problems

Some of the distinctions between optimization problems stem from

(a) differentiable versus nondifferentiable functions;

(b) unconstrained versus constrained variables;

(c) one-dimensional versus multi-dimensional variables;

(d) convex versus nonconvex minimization.

Iterative methods: Finite vs Convergence

For some classes of optimization problems (e.g., linear and quadratic optimization) there are algorithms that obtain a solution—or detect that the objective function is unbounded—in a finite number of iterations. For this reason, we call them finite algorithms.

Most algorithms encountered in Optimization are not finite, but instead are convergent—or at least they are designed to be so. Their object is to generate a sequence of trial or approximate solutions that converge to a “solution”.

The meaning of "solution"

What is meant by a solution may differ from one algorithm to another. In some cases, one seeks a local minimum; in some cases, one seeks a global minimum; in others, one seeks a KKT point of some sort as in the method of steepest descent discussed below. In fact, there are several possibilities for defining what a solution is. Once the definition is chosen, there must be a way of testing whether or not a point (trial solution) belongs to the set of solutions.

Search directions

Typically, a nonlinear optimization algorithm generates a sequence of points through an iterative scheme of the form

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{p}^k$$

where $\mathbf{p}^k$ is the search direction and α_k is the step size or step length.

The key is that once $\mathbf{x}^k$ is known, then $\mathbf{p}^k$ is chosen as some function of $\mathbf{x}^k$, and the scalar α_k may be chosen in accordance with some line (one-dimension) search rules.

The general idea

One selects a starting point and generates a possibly infinite sequence of trial solutions each of which is specified by the algorithm. An iterative algorithm is initiated by specifying a starting point.

The idea is to do this in such a way that the sequence of iterates generated by the algorithm converges to an element of the solution set of the problem.

Convergence to some other sort of point is undesirable—as is failure of the sequence to converge at all.

If for arbitrary starting points the algorithm is guaranteed to generate a sequence of points converging to a solution, then the algorithm is said to be globally convergent.

Convergent sequences of real numbers

Let $\{x^k\}$ be a sequence of real numbers. Then $\{x^k\}$ converges to 0 if and only if for all real numbers $\varepsilon > 0$ there exists a positive integer K such that

$$|x^k| < \varepsilon \quad \textbf{for all } k \geq K.$$

Let $\{x^k\}$ be a sequence of real numbers. Then $\{x^k\}$ converges to x^* if and only if $\{x^k - x^*\}$ converges to 0.

Criteria for convergence

- $\|x^{k+1} - x^k\| < \varepsilon$ or $\dfrac{\|x^{k+1} - x^k\|}{\|x^k\|} < \varepsilon$
- $f(x^k) - f(x^{k+1}) < \varepsilon$ or $\dfrac{f(x^k) - f(x^{k+1})}{f(x^k)} < \varepsilon$
- $\|\nabla f(x^k)\| < \varepsilon$

Speed of convergence

Let the sequence $\{x^k\}$ converge to x^*. Then the sequence has order p of Q-convergence if there exists $0 \leq r < \infty$ and $p \geq 1$ such that

$$\lim \sup_{k \to \infty} \frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|^p} = r.$$

- Q-Linear Convergence If $p = 1$ and $0 < r < 1$, the sequence $\{x^k\}$ is said to linearly converge to x^* with convergence ratio r.
- Q-Superlinear Convergence If $p = 1$ and $r = 0$, the sequence $\{x^k\}$ is said to superlinearly converge to x^*.
- Q-Quadratic Convergence If $p = 2$, the sequence $\{x^k\}$ is said to quadratically converge to x^*.

R-convergence

- R-Linear Convergence If there exists $\alpha \in (0, \infty)$ and $q \in (0, 1)$ such that

$$\|x^k - x^*\| \leq \alpha q^k.$$

- R-Superlinear Convergence If there exists $\alpha \in (0, \infty)$ and a positive sequence $\{q_k\}$ with $q_k \to 0$ such that

$$\|x^k - x^*\| \leq \alpha \prod_{i=1}^{k} q_i.$$

Unconstrained minimization of smooth functions

The meaning of the term smooth function differs from one author to another, but it is generally agreed that it means at least continuously differentiable.

A Generic Algorithm: Let f be a smooth function on R^n. We seek $\mathbf{x}^* \in R^n$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x} \in R^n$.

It may be necessary to adopt a more modest goal than finding a global minimum.

We might just look for a local minimum or a stationary point of the problem, f.

Assume we can decide whether any given point is a “solution.”

(A1) Test for convergence If the termination conditions are satisfied at $\mathbf{x}^k$, then it is taken (accepted) as a "solution." In practice, this may mean satisfying the desired conditions to within some tolerance. If so, stop. Otherwise, go to step (A2).

(A2) Compute a search direction, say $\mathbf{p}^k \neq 0$. This might be a direction in which the function value is known to decrease.

(A3) Compute a step length, say α_k such that

$$f(\mathbf{x}^k + \alpha_k \mathbf{p}^k) < f(x^k).$$

This may necessitate a one-dimensional (or line) search.

(A4) Define the new iterate by setting

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{p}^k$$

and return to step (A1).

Descent method

(A1) Choose a starting point $\mathbf{x}^0 \in \mathcal{R}^n$ and a scalar $\varepsilon > 0$. Set $k := 0$.

(A2) Compute a descent search direction $\mathbf{p}^k$ such that $(\mathbf{p}^k)^T \mathbf{g}^k < 0$ where $\mathbf{g}^k := \nabla f(\mathbf{x}^k)$.

(A3) Compute a step length α_k by the accurate line search rule or an inaccurate line search rule.

Accurate Line Search

$$\alpha_k = \textbf{arg}\min_{\alpha \geq 0}\{f(x^k + \alpha \mathbf{p}^k)\}.$$

This is called a one-dimensional accurate line search.

Inaccurate Line Search: Armijo step-rule, Wolfe step-rule

有时不必每一步都到最小值
（求最小值的计算量增加

(A4) Set $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{p}^k$. If $\|\mathbf{g}^{k+1}\| \leq \varepsilon$, stop. Otherwise, set $k := k + 1$ and return to step (A2).

Inaccurate Line Search

- Armijo step-rule Let $\beta > 0, \gamma \in (0,1)$. Compute $\alpha_k = \beta\gamma^{m_k}$, where m_k is the smallest nonnegative integer such that

$$f(x^k + \beta\gamma^m p^k) \leq f(x^k) + \sigma\beta\gamma^m (g^k)^T p^k,$$

  where $\sigma \in (0,1)$.

- Wolfe step-rule Compute α_k such that

$$\begin{aligned} f(x^k + \alpha p^k) &\leq f(x^k) + \sigma_1 \alpha (g^k)^T p^k, \\ \nabla f(x^k + \alpha p^k)^T p^k &\geq \sigma_2 (g^k)^T p^k, \end{aligned}$$

  where $0 < \sigma_1 < \sigma_2 < 1$.

The gradient method (steepest descent method)

Let f be a differentiable function and assume we can compute ∇f. We want to solve the unconstrained minimization problem

$$\min_{\mathbf{x}\in R^n} f(\mathbf{x}).$$

In the absence of further information, we seek a stationary point of f, that is, a point $\mathbf{x}^*$ at which $\nabla f(\mathbf{x}^*) = 0$.

Here we choose $\mathbf{p}^k = -\nabla f(\mathbf{x}^k)$ as the search direction at $\mathbf{x}^k$. The number $\alpha_k \geq 0$ is chosen "appropriately," namely to satisfy

$$\alpha_k \in \textbf{arg min} f(\mathbf{x}^k - \alpha \nabla f(\mathbf{x}^k)).$$

容易看出此时 $\langle \nabla f(x^{k+1}), p^k \rangle = 0$

Then the new iterate is defined as $\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha_k \nabla f(\mathbf{x}^k)$.

Now, if $\nabla f(\mathbf{x}^{k+1}) \neq 0$, then $-\nabla f(\mathbf{x}^{k+1})$ is a direction of descent at $\mathbf{x}^{k+1}$; in fact, it is the direction of steepest descent.

Example

Let $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T Q\mathbf{x}$ where $Q \in R^{n\times n}$ is symmetric and positive definite. This implies that the eigenvalues of Q are all positive. The unique minimum $\mathbf{x}^*$ of $f(\mathbf{x})$ exists and is given by the solution of the equation

$$\nabla f(\mathbf{x}) = \mathbf{c} + Q\mathbf{x} = \mathbf{0},$$

or equivalently

$$Q\mathbf{x} = -\mathbf{c}.$$

The iterative scheme

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{p}^k$$

becomes

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha_k(\mathbf{c} + Q\mathbf{x}^k)$$

by virtue of the definition, $\mathbf{p}^k = -(\mathbf{c} + Q\mathbf{x}^k)$.

To compute the step size, α_k, we consider

$$\begin{aligned} & f(\mathbf{x}^k + \alpha\mathbf{p}^k) \\ = \quad & \mathbf{c}^T(\mathbf{x}^k + \alpha\mathbf{p}^k) + \tfrac{1}{2}(\mathbf{x}^k + \alpha\mathbf{p}^k)^T Q(\mathbf{x}^k + \alpha\mathbf{p}^k) \\ = \quad & \mathbf{c}^T\mathbf{x}^k + \alpha\mathbf{c}^T\mathbf{p}^k + \tfrac{1}{2}(\mathbf{x}^k)^{\mathrm{T}} Q\mathbf{x}^k + \alpha(\mathbf{x}^k)^{\mathrm{T}} Q\mathbf{p}^k + \tfrac{1}{2}\alpha^2(\mathbf{p}^k)^{\mathrm{T}} Q\mathbf{p}^k \end{aligned}$$

Its minimizer α_k is the unique value of α where the derivative $f'(\mathbf{x}^k + \alpha \mathbf{p}^k)$ vanishes, i.e., where

$$\mathbf{c}^T \mathbf{p}^k + \mathbf{x}_k^{\mathrm{T}} Q \mathbf{p}^k + \alpha (\mathbf{p}^k)^{\mathrm{T}} Q \mathbf{p}^k = 0.$$

Thus

$$\alpha_k = -\frac{(\mathbf{c}^T + \mathbf{x}_k^{\mathrm{T}} Q)\mathbf{p}^k}{\mathbf{p}_k^{\mathrm{T}} Q \mathbf{p}^k} = \frac{(\mathbf{p}^k)^{\mathrm{T}} \mathbf{p}^k}{(\mathbf{p}^k)^{\mathrm{T}} Q \mathbf{p}^k}.$$

The recursion for the method of steepest descent now becomes

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \left(\frac{(\mathbf{p}^k)^{\mathrm{T}} \mathbf{p}^k}{(\mathbf{p}^k)^{\mathrm{T}} Q \mathbf{p}^k} \right) \mathbf{p}^k$$

where $\mathbf{p}^k = -(\mathbf{c} + Q\mathbf{x}^k)$.

Convergence of the steepest descent method

To show the convergence of the steepest descent method, we apply it to quadratic functions. Suppose Q is a symmetric positive definite matrix of order n and let its eigenvalues be $\lambda_1 \leq \cdots \leq \lambda_n$. Obviously, the global minimizer of the quadratic form $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q\mathbf{x}$ is at the origin, i.e., $x^* = 0$.

It can be shown that when the steepest descent method is started from any nonzero point $\mathbf{x}^0 \in R^n$,

$$\frac{f(\mathbf{x}^{k+1})}{f(\mathbf{x}^k)} \leq \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^2 < 1, \; k = 0, 1, \ldots ,$$

$$\frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|} \leq \sqrt{\lambda_n/\lambda_1}\,\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}.$$

This shows that the steepest descent method is convergent and the rate of convergence is R-linear.

Kantorovich Inequality

Let $Q \in \mathcal{R}^{n\times n}$ be a symmetric positive definite matrix and let its eigenvalues be $\lambda_1 \leq \cdots \leq \lambda_n$. Then for any vector $\mathbf{x} \in \mathcal{R}^n$

考虑Q的奇异值分解

$$\frac{(\mathbf{x}^T\mathbf{x})^2}{(\mathbf{x}^TQ\mathbf{x})(\mathbf{x}^TQ^{-1}\mathbf{x})} \geq \frac{4\lambda_1\lambda_n}{(\lambda_1+\lambda_n)^2}.$$

$\sum \lambda_i x_i^2$ $\sum \frac{1}{\lambda_i} x_i^2$

反向柯西不等式

左边上下可以同时除以x的平方和考虑作为λ的凸组合

Convergence analysis—quadratic form

二次型时的收敛性

Let $g^k = Q\mathbf{x}^k$. Then, by the accurate line search rule,

$$\alpha_k = \frac{(g^k)^T g^k}{(g^k)^T Q g^k}, \quad x^{k+1} = x^k - \alpha_k g^k.$$

Note that $x^* = 0$, we have

$$\begin{aligned}
\frac{f(\mathbf{x}^k) - f(\mathbf{x}^{k+1})}{f(\mathbf{x}^k)} &= \frac{(\mathbf{x}^k)^T Q\mathbf{x}^k - (\mathbf{x}^{k+1})^T Q\mathbf{x}^{k+1}}{(\mathbf{x}^k)^T Q\mathbf{x}^k} \\
&= \frac{2\alpha_k (g^k)^T g^k - \alpha_k^2 (g^k)^T Q g^k}{(\mathbf{x}^k)^T Q\mathbf{x}^k} \\
&= \frac{2((g^k)^T g^k)^2/(g^k)^T Q g^k - ((g^k)^T g^k)^2/(g^k)^T Q g^k}{(g^k)^T Q^{-1} g^k} \\
&= \frac{((g^k)^T g^k)^2}{((g^k)^T Q g^k)((g^k)^T Q^{-1} g^k)}.
\end{aligned}$$

Thus, by Kantorovich Inequality,

$$\frac{f(x^{k+1})}{f(x^k)} \leq 1 - \frac{4\lambda_1\lambda_n}{(\lambda_1+\lambda_n)^2} = \left(\frac{\lambda_n-\lambda_1}{\lambda_n+\lambda_1}\right)^2.$$

Note that

$$f(\mathbf{x}^k) \leq \left(\frac{\lambda_n-\lambda_1}{\lambda_n+\lambda_1}\right)^{2k} f(x^0), \quad \forall k \geq 0.$$

Since

$$\lambda_1(\mathbf{x}^k)^T\mathbf{x}^k \leq (\mathbf{x}^k)^T Q\mathbf{x}^k \leq \lambda_n(\mathbf{x}^k)^T\mathbf{x}^k, \quad (\mathbf{x}^k)^T Q\mathbf{x}^k = 2f(\mathbf{x}^k),$$

we have

$$\lambda_1\|\mathbf{x}^k-\mathbf{x}^*\|^2 \leq 2f(\mathbf{x}^k) \leq \lambda_n\|\mathbf{x}^k-\mathbf{x}^*\|^2.$$

Thus

$$\begin{aligned}
\|\mathbf{x}^k - \mathbf{x}^*\|^2 &\leq \frac{2}{\lambda_1} f(\mathbf{x}^k) \leq \frac{2}{\lambda_1} \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^{2k} f(x^0) \\
&\leq \frac{2}{\lambda_1} \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^{2k} \cdot \frac{\lambda_n}{2} \|\mathbf{x}^0 - \mathbf{x}^*\|^2 \\
&= \frac{\lambda_n}{\lambda_1} \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^{2k} \|\mathbf{x}^0 - \mathbf{x}^*\|^2.
\end{aligned}$$

This shows that the rate of convergence of the steepest descent method is R-linear.

Convergence of the steepest descent method—general form

The following theorem gives some conditions under which the steepest descent method will converge.

Theorem 1 *Let $f : R^n \to R$ be given. For some given point $\mathbf{x}^0 \in R^n$, let the level set*

$$X^0 = \{\mathbf{x} \in R^n : f(\mathbf{x}) \leq f(x^0)\}$$

be bounded. Assume further that f is continuously differentiable on the convex hull of X^0. Let $\{\mathbf{x}^k\}$ be the sequence of points generated by the steepest descent method initiated at $\mathbf{x}^0$. Then every accumulation point of $\{\mathbf{x}^k\}$ is a KKT point of f.

稳定点

Remark

According to this theorem, the steepest descent method initiated at any point of the level set X^0 will converge to a stationary point of f.

This is a nice feature; it means that (depending on the size of X^0), the starting point $\mathbf{x}^0$ could be far away from the point $\mathbf{x}^*$ to which the sequence converges.

In other words, it is not necessary to start the process in a neighborhood of the (unknown) solution.

This property is called global convergence.

The convergence rate of the steepest descent method applied to quadratic functions is known to be R-linear.

Application

One approach to the minimization of a function f is to consider solving the equations $\nabla f(x) = 0$ that represent the necessary conditions. It has been proposed that these equations could be solved by applying steepest descent to the function $h(x) = \|\nabla f(x)\|^2$. One advantage of the proposed method is that the minimum value is known. We ask whether this method is likely to be faster or slower than the application of steepest descent to the original function f itself.

Application continued

Consider the quadratic function $f = \frac{1}{2}x^T Q x$ where $Q \succ 0$. Then $h(x) = x^T Q^2 x$. Thus $h(x)$ is itself a quadratic function. The rate of convergence of steepest descent applied to h will be governed by the smallest and largest eigenvalues of the matrix Q^2. That is,

$$\frac{h(x^{k+1})}{h(x^k)} \leq \left(\frac{r^2-1}{r^2+1}\right)^2,$$

where $r = \lambda_n/\lambda_1$. It is clear that the convergence rate for the proposed method will be worse than for steepest descent applied to the original function f.

Newton method

All unconstrained local minimizers of a differentiable function f are stationary points: they make the gradient ∇f vanish.

Finding a solution of the stationarity condition

$$\nabla f(\mathbf{x}) = 0$$

is a matter of solving a system of (possibly nonlinear) equations.

For functions of a single real variable, the stationarity condition is

$$f'(x) = 0.$$

When f is twice continuously differentiable, Newton method can be a very effective way to solve such equations and hence to locate a stationary point of f.

Note: Newton method is a procedure for solving equations.

不是直接求极值

Newton method for solving the equation $g(x) = 0$

Although the iterative scheme associated with Newton method uses only first derivatives, the working hypothesis is that the function g—of which we want to find a zero—has continuous second derivatives.

The univariate case. Given a starting point x^0, Newton method for solving the equation $g(x) = 0$ is to generate the sequence of iterates

$$x^{k+1} = x^k - \frac{g(x^k)}{g'(x^k)}.$$

The iteration is well defined provided that $g'(x^k) \neq 0$ at each step. For obvious reasons, the iteration stops if $g(x^k) = 0$.

The interpretation of the iteration

$$x^{k+1} = x^k - \frac{g(x^k)}{g'(x^k)}$$

goes as follows. For a given iterate, x^k such that $g(x^k) \neq 0$, let $x = x^k + p$. By linearizing g at x^k we obtain

$$g(x) = g(x^k + p) \approx g(x^k) + pg'(x^k).$$

We seek a value of p at which the expression $g(x^k) + pg'(x^k) = 0$. The assumptions imply there will be such a value provided $g'(x^k) \neq 0$. It's a matter of where the tangent line crosses the horizontal axis. When $g'(x^k) = 0$, the tangent line is parallel to the horizontal axis, and there is no intersection. This anomaly is illustrated in Figure 1.

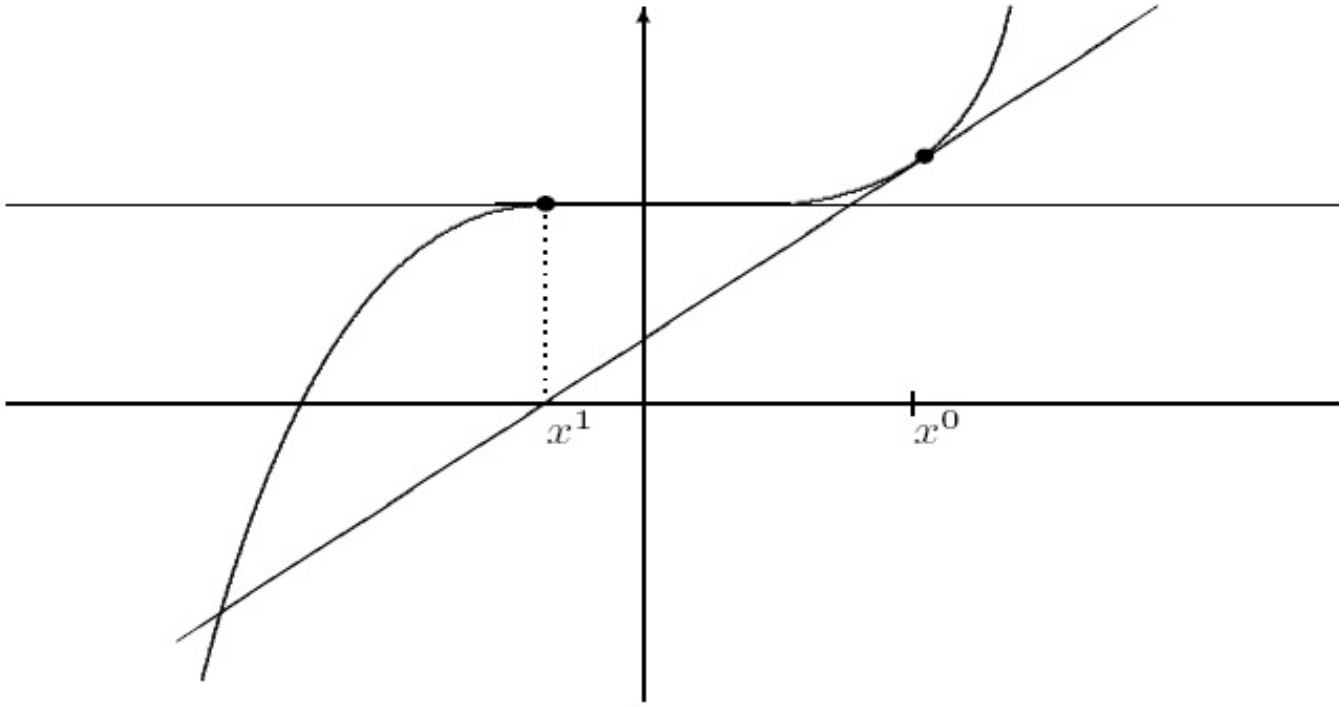


Figure 1

Without some further stipulations, there is no guarantee that the sequence of points defined by the iterative scheme above will converge to a zero of the function g. This can be illustrated by the case where

$$g(x) = x^{1/3}.$$

In this instance, the sequence diverges unless $x^0 = 0$, the zero of g. See Figure 2 below.

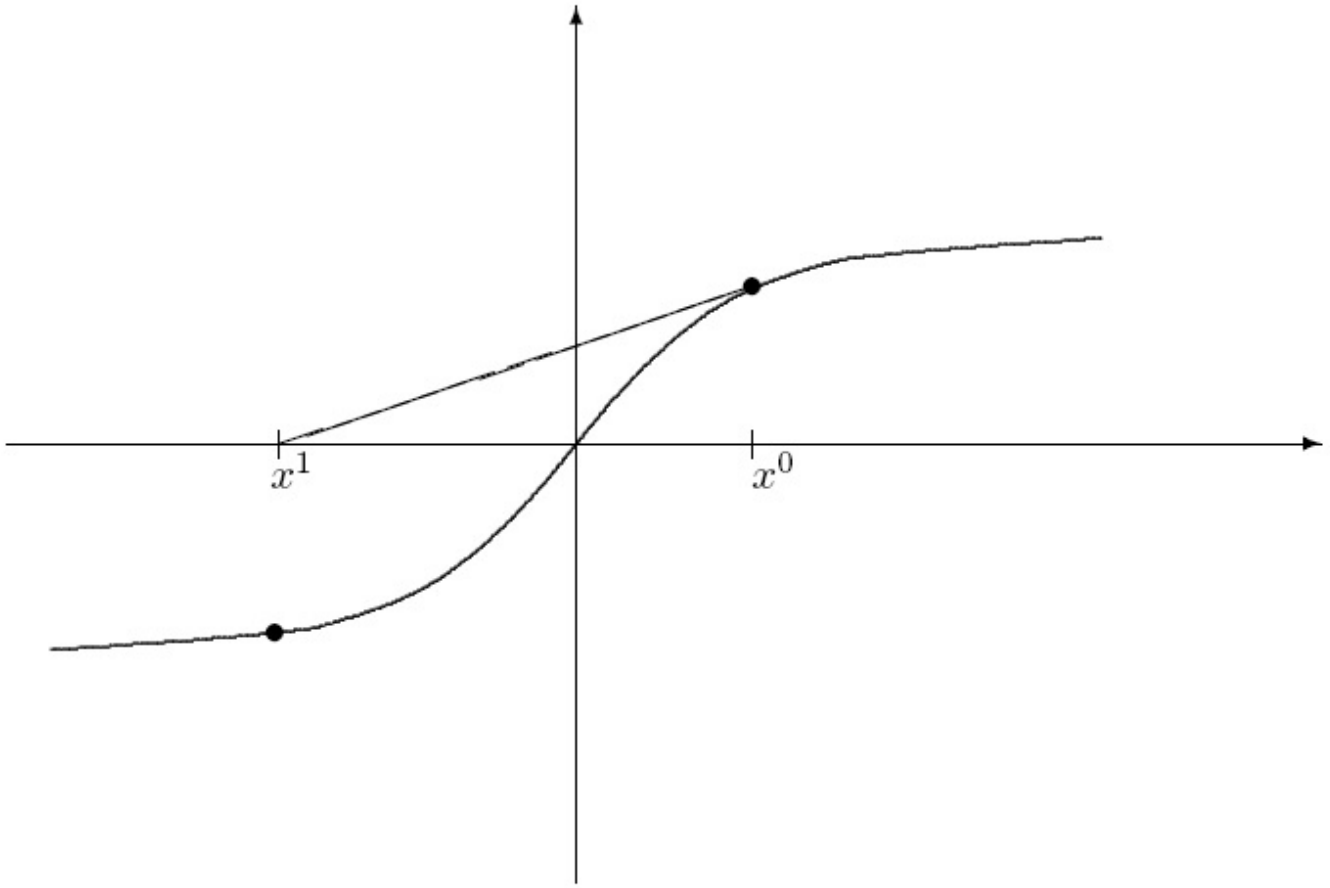


Figure 2

Locally Quadratic Convergence of Newton Method

Theorem 2 *If g is twice continuously differentiable and x^* is a zero of g at which $g'(x^*) \neq 0$, then provided that $|x^0 - x^*|$ is sufficiently small, the sequence generated by the iteration*

$$x^{k+1} = x^k - \frac{g(x^k)}{g'(x^k)}.$$

converges quadratically to x^* with rate constant $C = |g''(x^*)/2g'(x^*)|$.

局部二阶收敛

Proof

Since $g'(x^*) \neq 0$, when x^k is sufficiently close to x^*, $g'(x^k) \neq 0$. By the Taylor series expansion of g,

$$0 = g(x^*) = g(x^k) + g'(x^k)(x^* - x^k) + o(|x^k - x^*|).$$

Dividing by $g'(x^k)$, we have

$$x^k - x^* - \frac{g(x^k)}{g'(x^k)} = o(|x^k - x^*|),$$

i.e.,

$$|x^{k+1} - x^*| = o(|x^k - x^*|).$$

Hence, when $|x^0 - x^*|$ is sufficiently small, the sequence $\{x^k\}$ superlinearly converges to x^*.

When x^k is sufficiently close to x^*, we have

$$\begin{aligned}
|x^{k+1} - x^*| &= |x^k - x^* - \frac{g(x^k)}{g'(x^k)}| \\
&= \frac{|g'(x^k)(x^k - x^*) - g(x^k) + g(x^*)|}{|g'(x^k)|} \\
&\leq \frac{|\int_0^1 [g'(x^k) - g'(x^* + t(x^k - x^*))](x^k - x^*)dt|}{|g'(x^k)|} \\
&\leq |\frac{g''(x^* + \theta t(x^k - x^*))}{g'(x^k)}||x^k - x^*|^2 \int_0^1 (1-t)dt \\
&= |\frac{g''(x^* + \theta t(x^k - x^*))}{2g'(x^k)}||x^k - x^*|^2,
\end{aligned}$$

where $\theta \in (0, 1)$. This implies that

$$|x^{k+1} - x^*| \leq C|x^k - x^*|^2$$

as $x^k \to x^*$. That is, the sequence $\{x^k\}$ quadratically converges to x^*.

Remark

Newton method is finite for positive quadratic function.

Consider the quadratic function $f = \frac{1}{2}x^T Qx + c^T x$ where $Q \succ 0$. Newton method can find its minimum in one iteration from any starting point.

Clearly, the optimal solution is $x^* := -Q^{-1}c$. Let x^0 be any point in R^n. By virtue of Newton method,

$$x^1 = x^0 - Q^{-1}(Qx^0 + c) = -Q^{-1}c = x^*.$$

Newton method for solving a system of equations $g(x) = 0$

When we have a mapping

$$g(x) = \begin{bmatrix} g_1(x) \\ \vdots \\ g_\ell(x) \end{bmatrix},$$

we define the Jacobian of g as

$$\nabla g(x) = \left[\frac{\partial g_i(x)}{\partial x^j} \right].$$

The rows of $\nabla g(x)$ are the gradient vectors

$$\nabla g_1(x), \ldots, \nabla g_\ell(x).$$

For the system $g(x) = 0$, i.e.,

$$g_i(x) = 0, \quad i = 1, \ldots, n$$

the iteration is given by

$$x^{k+1} = x^k - (\nabla g(x^k))^{-\mathrm{T}} g(x^k).$$

This formula follows from the use of a Taylor series approximation to g at the point x^k, namely

$$g(x^k + p) \approx g(x^k) + \nabla g(x^k)^{\mathrm{T}} p.$$

When we set the right-hand side of this equation to zero, we can solve it for p, provided that the Jacobian matrix is nonsingular.

Some computational issues

As an optimization technique, Newton method, in its pure form, requires knowledge of (or the capacity to compute) the first and second derivatives of the minimand. In an environment where individual function evaluations are expensive, this could be a drawback.

In a large-scale problem, the computation of the search direction, p, could also turn out to be a time-consuming task. One solves

$$(\nabla g(x^k))^T p = -g(x^k)$$

using matrix factorization methods. In the optimization context, $g(x)$ would be the gradient vector of the minimand f so that the Jacobian matrix of g would be the Hessian matrix of f. In the neighborhood of a minimizer, the Hessian would be positive semidefinite, and if this matrix is to be nonsingular, it must then be positive definite. The LDL^T factorization would be used to compute p.

Storage

Storage can also be a factor in using Newton method, though with the storage capacity of today's computers, this issue does not loom as large as it once did.

Running time is another issue. There is a question of whether the relatively small number of iterations required by Newton method is worth the cost of the computing at each iteration. This consideration motivates the use of quasi-Newton methods which generally speaking, use a surrogate for the true Hessian matrix. Under some circumstances, it can be shown that such variants of Newton method possess superlinear convergence to a local minimizer.

Descent

For an arbitrary twice-continuously differentiable function f, there is no reason to expect Newton method produce a sequence of iterates that converge to a local minimizer of f.

In fact, as we have seen, convergence to anything is not guaranteed without additional hypotheses.

Inasmuch as Newton method strives to produce a stationary of f, it is only with a satisfactory second-order (or curvature) condition that we can be sure of having—or getting to—a local minimizer.

If our search direction at a point, say

$$p = -(\nabla^2 f(\bar{x}))^{-1} \nabla f(\bar{x}),$$

then it is a descent direction only if

$$p^T \nabla f(\bar{x}) = -(\nabla f(\bar{x}))^T (\nabla^2 f(\bar{x}))^{-1} \nabla f(\bar{x}) < 0$$

which is to say that

$$(\nabla f(\bar{x}))^T (\nabla^2 f(\bar{x}))^{-1} (\nabla f(\bar{x})) > 0.$$

This will hold if $\nabla^2 f(\bar{x})$ is a positive definite matrix. This is only a sufficient condition, however. It is often too strong a condition to impose when what we really want is $p^T \nabla f(\bar{x}) < 0$. In the case where $\nabla^2 f(\bar{x})$ is not positive definite, we can modify it by adding to it a suitable diagonal positive definite matrix E so that the sum $\nabla^2 f(\bar{x}) + E$ is positive definite.

Modified Newton Algorithm with Line Search

Initialize the algorithm with the point x^0 and the tolerance $\varepsilon > 0$. For $k = 0, 1, \ldots$, perform the following sequence of steps.

- (Termination criterion) If $\|\nabla f(x^k)\| < \varepsilon$, stop. Report x^k as an approximation to a stationary point, x^*.
- (Modify the Hessian) For some diagonal positive definite matrix E, so that $\nabla^2 f(x^k) + E$ is also positive definite, compute the matrix factorization

$$LDL^T = \nabla^2 f(x^k) + E.$$

- (Compute the search direction) Solve the equation

$$LDL^T p = -\nabla f(x^k).$$

Denote the solution by p^k.

- (Compute the step length) Do a line search to fine the step length α_k:

$$\alpha_k = \textbf{arg}\min\{f(x^k + \alpha p^k) : \ \alpha \geq 0\}$$

- (Compute the new iterate) Compute $x^{k+1} = x^k + \alpha_k p^k$. Repeat.

Note that Levenberg-Marquardt method: In the step of Modify the Hessian, consider the matrix $\nabla^2 f(x^k) + \lambda_k E$.

See M.S.Bazaraa, H.D.Sherali, C.M.Shetty, *Nonlinear Programming: Theory and Algorithms*, John Wiley & Sons, Inc., Publication, 2006.

Conjugate Direction Method

Conjugate direction methods can be regarded as being somewhat intermediate between the method of steepest descent and Newton method. They are motivated by the desire of accelerate the typically slow convergence associated with steepest descent while avoiding the information requirements associated with the evaluation, storage, and inversion of the Hessian as required by Newton method.

Conjugate direction methods invariably are invented and analyzed for the purely quadratic problem

$$f(\mathbf{x}) = \tfrac{1}{2}\mathbf{x}^T Q\mathbf{x} - \mathbf{b}^T\mathbf{x},$$

where Q is an $n \times n$ symmetric positive definite matrix.

Conjugate directions

Given a symmetric matrix Q, two vectors $\mathbf{d}_1$ and $\mathbf{d}_2$ are said to be Q-orthogonal, or conjugate with respect to Q, if $\mathbf{d}_1^T Q \mathbf{d}_2 = 0$.

A finite set of vectors $\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_k$ is said to be a Q-orthogonal if $\mathbf{d}_i^T Q \mathbf{d}_j = 0$ for all $i \neq j$.

Proposition 1 *If $Q \succ 0$ and the set of nonzero vectors $\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_k$ are Q-orthogonal, then these vectors are linearly independent.*

Conjugate Direction Theorem

Theorem 3 *Consider the purely quadratic problem* $\min f(\mathbf{x})$. *Let* $\{\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_{n-1}\}$ *be a set of nonzero* Q*-orthogonal vectors. For any* $\mathbf{x}^0 \in \mathcal{R}^n$ *the sequence* $\{\mathbf{x}^k\}$ *generated according to*

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}_k, \quad k \geq 0$$

with α_k *due to exact line search, converges to the unique solution* $\mathbf{x}^*$ *of the problem after* n *steps, that is,* $\mathbf{x}^n = \mathbf{x}^*$.

Proof

Since $\mathbf{d}_k's$ are linearly independent, we can write

$$\mathbf{x}^* - \mathbf{x}^0 = \beta_0\mathbf{d}_0 + \beta_1\mathbf{d}_1 + \ldots + \beta_{n-1}\mathbf{d}_{n-1}$$

for some set of $\beta_k's$. We multiply by $\mathbf{d}_k^T Q$ to find

$$\beta_k = \frac{\mathbf{d}_k^T Q(\mathbf{x}^* - \mathbf{x}^0)}{\mathbf{d}_k^T Q\mathbf{d}_k} = \frac{\mathbf{d}_k^T(\mathbf{b} - Q\mathbf{x}^0)}{\mathbf{d}_k^T Q\mathbf{d}_k}.$$

On the other hand, it follows from $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k\mathbf{d}_k$ that

$$\mathbf{x}^n - \mathbf{x}^0 = \alpha_0\mathbf{d}_0 + \alpha_1\mathbf{d}_1 + \ldots + \alpha_{n-1}\mathbf{d}_{n-1}.$$

The exact line search

$$\alpha_k = \textbf{arg}\min\{f(x^k + \alpha \mathbf{d}_k) : \ \alpha \geq 0\}$$

implies

$$\alpha_k = \frac{\mathbf{d}_k^T(\mathbf{b} - Q\mathbf{x}^k)}{\mathbf{d}_k^T Q \mathbf{d}_k}.$$

Since $\{\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_{n-1}\}$ is a set of nonzero Q-orthogonal vectors, $\mathbf{d}_k^T Q(\mathbf{x}^k - \mathbf{x}^0) = 0$. Hence,

$$\alpha_k = \frac{\mathbf{d}_k^T(\mathbf{b} - Q\mathbf{x}^0)}{\mathbf{d}_k^T Q \mathbf{d}_k} = \beta_k$$

for all $k = 0, 1, \ldots, n-1$. This yields $\mathbf{x}^n = \mathbf{x}^*$.

The Conjugate Gradient Method

The conjugate gradient method is the conjugate direction method that is obtained by selecting the successive direction vectors as a conjugate version of the successive gradients obtained as the method progresses. Thus, the directions are not specified beforehand, but rather are determined sequentially at each step of the iteration. At step k one evaluates the current negative gradient vector and adds to it a linear combination of the previous direction vectors to obtain a new conjugate direction vector along which to move.

The Conjugate Gradient Algorithm

Starting at any $\mathbf{x}^1 \in \mathcal{R}^n$ define $\mathbf{d}_1 = -\mathbf{g_1} = \mathbf{b} - \mathbf{Q}\mathbf{x}^\mathbf{1}$. We do the exact line search along the direction $\mathbf{d}_1$ to obtain the step-size

$$\alpha_1 = \frac{\mathbf{g_1^T g_1}}{\mathbf{d}_1^T Q \mathbf{d}_1}.$$

Let $\mathbf{x}^2 = \mathbf{x}^1 + \alpha_1 \mathbf{d}_1$ and $\mathbf{d}_2 = -\mathbf{g_2} + \beta_1 \mathbf{d_1}$. Let $\mathbf{d}_1$ and $\mathbf{d}_2$ be Q-orthogonal, we can take

$$\beta_1 = \frac{\mathbf{d}_1^T Q \mathbf{g_2}}{\mathbf{d}_1^T Q \mathbf{d}_1}.$$

To move along, we have

$$\begin{aligned} \mathbf{d}_k &= -\mathbf{g_k} + \beta_{\mathbf{k-1}} \mathbf{d_{k-1}}, \quad \beta_{\mathbf{k-1}} = \frac{\mathbf{d_{k-1}^T Q g_k}}{\mathbf{d_{k-1}^T Q d_{k-1}}}, \\ \mathbf{x}^{k+1} &= \mathbf{x}^k + \alpha_k \mathbf{d}_k, \quad \alpha_k = \frac{\mathbf{g_k^T g_k}}{\mathbf{d}_k^T Q \mathbf{d}_k}. \end{aligned} \tag{1}$$

Conjugate Gradient Theorem

Theorem 4 *Consider the purely quadratic problem* $\min f(\mathbf{x})$. *The conjugate gradient algorithm (1) terminates at most* n *steps. Moreover, for any* $1 \leq i \leq m$, *where* $m \leq n$, *we have*

(a) $\mathbf{d}_i^T Q \mathbf{d}_j = 0, \quad j = 1, 2, \ldots, i-1.$

(b) $\mathbf{g_i^T g_j = 0}, \quad \mathbf{j = 1, 2, \ldots, i-1}.$

(c) $\mathbf{g_i^T d_i = -g_i^T g_i}.$

Proof of Conjugate Gradient Theorem

By induction. When $i = 1$, (c) holds. When $i = 2$, $\mathbf{d}_2^T Q \mathbf{d}_1 = 0$ implies that (a) holds. By the exact line search rule, $\mathbf{g_2^T d_1 = 0}$, and then (b) holds. Since $\mathbf{g_2^T d_2 = g_2^T(-g_2 + \beta_1 d_1) = -g_2^T g_2}$, (c) holds.

Suppose that (a), (b) and (c) hold for $3 \leq i < m$. We prove they hold for $i + 1$.

To prove (b) we write

$$g_{i+1} = Q\mathbf{x}^{i+1} - \mathbf{b} = g_i + \alpha_i Q\mathbf{d}_i.$$

Hence,

$$\begin{aligned} g_{i+1}^T g_j &= (g_i + \alpha_i Q\mathbf{d}_i)^T g_j \\ &= g_i^T g_j + \alpha_i \mathbf{d}_i^T Q(-\mathbf{d}_j + \beta_{j-1}\mathbf{d}_{j-1}) \\ &= \begin{cases} 0, & i \neq j, \\ g_i^T g_j - \alpha_i \mathbf{d}_i^T Q\mathbf{d}_j = 0, & i = j. \end{cases} \end{aligned}$$

To prove (a) we write

$$\mathbf{d}_{i+1}^T Q\mathbf{d}_j = (-g_{i+1} + \beta_i \mathbf{d}_i)^T Q\mathbf{d}_j = -g_{i+1}^T Q\mathbf{d}_j + \beta_i \mathbf{d}_i^T Q\mathbf{d}_j.$$

Clearly, when $i = j$ we have $\mathbf{d}_{i+1}^T Q\mathbf{d}_j = 0$.

Consider the case $i \neq j$. It follows from $g_{i+1} - g_i = \alpha_i Q\mathbf{d}_i$ that

$$\begin{aligned} \mathbf{d}_{i+1}^T Q\mathbf{d}_j &= -g_{i+1}^T \left(\frac{g_{j+1} - g_j}{\alpha_i} \right) + \beta_i \mathbf{d}_i^T Q\mathbf{d}_j \\ &= -\frac{1}{\alpha_i} g_{i+1}^T g_{j+1} = 0. \end{aligned}$$

To prove (c) we write

$$g_{i+1}^T \mathbf{d}_{i+1} = g_{i+1}^T(-g_{i+1} + \beta_i \mathbf{d}_i) = -g_{i+1}^T g_{i+1} + \beta_i g_{i+1}^T \mathbf{d}_i.$$

Since

$$\begin{aligned} g_{i+1}^T \mathbf{d}_i &= g_{i+1}^T(-g_i + \beta_{i-1}\mathbf{d}_{i-1}) = \beta_{i-1} g_{i+1}^T \mathbf{d}_{i-1} \\ &= \beta_{i-1}\beta_{i-2}\cdots\beta_1 g_{i+1}^T(-g_1) = 0, \end{aligned}$$

(c) holds.

Remark. From Conjugate Gradient Theorem,

$$\beta_k = \frac{\mathbf{d}_k^T Q g_{k+1}}{\mathbf{d}_k^T Q \mathbf{d}_k} = \frac{\|g_{k+1}\|^2}{\|g_k\|^2}.$$

Fletcher-Reeves Conjugate Gradient Method

Step 1. Given $\mathbf{x}^0$ compute $g_0 = \nabla f(\mathbf{x}^0)$ and set $\mathbf{d}_0 = -g_0$.

Step 2. For $k = 0, 1, \ldots, n-1$:

(a) Set $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}_k$ where α_k minimizes $f(\mathbf{x}^k + \alpha \mathbf{d}_k)$.

(b) Compute $g_{k+1} = \nabla f(\mathbf{x}^{k+1})$.

(c) Unless $k = n-1$, set $d_{k+1} = -g_{k+1} + \beta_k \mathbf{d}_k$ where

FR算法在精确线性搜索时对凸函数收敛

$$\beta_k = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k} \quad \left[\textit{Polak-Ribiere method: } \beta_k = \frac{(g_{k+1} - g_k)^T g_{k+1}}{g_k^T g_k} \right]$$

Step 3. Replace $\mathbf{x}^0$ by $\mathbf{x}^n$ and go back to Step 1.

PRP算法对Wolf搜索的很多情况下收敛

Quasi-Newton Method

The main idea of Quasi-Newton Method is to construct a positive definite symmetric matrix that approximates the inverse of the Hessian matrix.

Let

$$g_k = \nabla f(x^k), \quad y_k = g_{k+1} - g_k, \quad s_k = x^{k+1} - x^k.$$

Then by the Taylor series expansion, 用割线近似切线

$$f(x) \approx f_{k+1} + g_{k+1}^T(x - x^{k+1}) + \frac{1}{2}(x - x^{k+1})^T G_{k+1}(x - x^{k+1}),$$

which implies that

$$\nabla f(x) \approx g_{k+1} + G_{k+1}(x - x^{k+1}).$$

If the Hessian G is constant, then we have

$$y_k = Gs_k, \tag{2}$$

and we see that the evaluation of the gradient at two points gives information about G. If n linearly independent directions $s_0, s_1, \ldots, s_{n-1}$ and the corresponding y_k's are known, then G is uniquely determined. Indeed, letting Y and S be the $n \times n$ matrices with columns y_k and s_k respectively, we have $G = YS^{-1}$.

It is natural to attempt to construct successive approximations H_k to G^{-1} based on data obtained from the first k steps of descent process in such a way that if G were constant the approximation would be consistent with (2) for these steps.

Thus we have the quasi-Newton condition:

$$H_{k+1}y_k = s_k, \tag{3}$$

or

$$B_{k+1}s_k = y_k, \tag{4}$$

where $H_{k+1}(B_{k+1})$ is an approximation of $G_{k+1}^{-1}(G_{k+1})$.

Specially, if G were constant H_{k+1} would satisfy

$$H_{k+1}y_i = s_i, \quad 0 \le i \le k. \tag{5}$$

After n linearly independent steps we would then have $H_n = G^{-1}$.

Rank one Correction

Define a recursion of the form

$$H_{k+1} = H_k + \alpha_k z_k z_k^T.$$

加一个半正定的秩一矩阵

By the quasi-Newton condition,

$$s_k = H_{k+1} y_k = H_k y_k + \alpha_k z_k z_k^T y_k,$$

and

$$y_k^T s_k - y_k^T H_k y_k = \alpha_k (z_k^T y_k)^2.$$

Then, we have

$$H_{k+1} = H_k + \frac{(s_k - H_k y_k)(s_k - H_k y_k)^T}{y_k^T (s_k - H_k y_k)}. \tag{6}$$

For the case where G is constant the rank one recursion converges to G^{-1} after at most n steps.

秩一校正

Theorem 5 *Let G be a fixed symmetric matrix and suppose that $s_0, s_1, \ldots, s_k$ are given vectors. Define the vectors $y_i = Gs_i,\ i = 0, 1, \ldots, k$. Starting with any initial symmetric matrix H_0 let*

$$H_{i+1} = H_i + \frac{(s_i - H_i y_i)(s_i - H_i y_i)^T}{y_i^T (s_i - H_i y_i)}.$$

Then

$$s_i = H_{k+1} y_i, \quad \textbf{for } i \leq k.$$

Proof: The proof is by induction. Suppose it is true for H_k and $i \leq k-1$. The relation was shown above to be true for H_{k+1} and $i = k$. For $i < k$

$$H_{k+1} y_i = H_k y_i + p_k (s_k^T y_i - y_k^T H_k y_i),$$

where

$$p_k = \frac{(s_k - H_k y_k)}{y_k^T (s_k - H_k y_k)}.$$

By the induction hypothesis, we have

$$H_{k+1}y_i = s_i + p_k(s_k^T y_i - y_k^T s_i).$$

From the calculation $y_i = Gs_i$, we have

$$y_k^T s_i = s_k^T G s_i = s_k^T y_i.$$

Thus it follows that the second term vanishes. Hence,

$$H_n G(s_0, s_1, \ldots, s_{n-1}) = (s_0, s_1, \ldots, s_{n-1}),$$

which together with the linear independence of $s_0, s_1, \ldots, s_{n-1}$, implies that $H_n G = I$.

Remark

To incorporate the approximate inverse Hessian in a descent procedure while simultaneously improving it, we calculate

$$d^k = -H_k g_k \quad \textbf{and} \quad \alpha_k = \textbf{arg}\min_{\alpha \geq 0} f(x^k + \alpha d^k).$$

This determines $s_k = \alpha_k d^k$, $x^{k+1} = x^k + s_k$, and g_{k+1}. Then H_{k+1} can be calculated according to (6).

There are some difficulties: First, the updating formula (6) preserves positive definiteness only if $y_k^T(s_k - H_k y_k) > 0$, which cannot be guaranteed. Also, even if $y_k^T(s_k - H_k y_k)$ is positive, it may be small, which can lead to numerical difficulties. Thus, although an excellent simple example of how information gathered during the descent process can in principle be used to update an approximation to the inverse Hessian, the rank one method possesses some limitations.

DFP Quasi-Newton Method

This method was proposed by Davidon [1959] and later developed by Fletcher and Powell [1963]. Performing a rank two correction procedure

$$H_{k+1} = H_k + as_k s_k^T + bH_k y_k y_k^T H_k$$

where a, b are parameter to be determined, for H_k such that the new matrix satisfies the quasi-Newton condition (3), we get the DFP formulation:

$$H_{k+1} = H_k + \frac{s_k s_k^T}{s_k^T y_k} - \frac{H_k y_k y_k^T H_k}{y_k^T H_k y_k}.$$

Theorem 6 *Let H_k be a positive definite symmetric matrix, then*

$$H_{k+1} \succ \mathbf{0} \Leftrightarrow s_k^T y_k > 0.$$

先考虑

$s_k^T y_k = (x_{k+1} - x_k)^T (g_{k+1} - g_k) > 0$

自己一般条件

如通过精确搜索或者wolf搜索得到的dk（此时也即Sk）或者当f为严格凸函数的时候这个就是梯度不等式

Proof: For any $x \in \mathcal{R}^n$ we have

$$x^T H_{k+1} x = x^T H_k x + \frac{(x^T s_k)^2}{s_k^T y_k} - \frac{(x^T H_k y_k)^2}{y_k^T H_k y_k}.$$

Defining $a = H_k^{1/2} x$ and $b = H_k^{1/2} y_k$, we may rewrite it as

$$x^T H_{k+1} x = \frac{(a^T a)(b^T b) - (a^T b)^2}{b^T b} + \frac{(x^T s_k)^2}{s_k^T y_k}. \tag{7}$$

Since $s_k^T y_k > 0$, both terms on the right side of (7) are nonnegative. We must show they do not vanish simultaneously. The first term vanishes only if a and b are proportional. This in turn implies that x and y_k are proportional, say $x = \beta y_k$. In that case, however,

$$x^T s_k = \beta y_k^T s_k \neq 0.$$

Hence, $H_{k+1} \succ \mathbf{0}$.

Conversely, suppose that $s_k^T y_k < 0$. Take $x = y_k$ and then $a = b$. Hence, it follows from (7) that

$$x^T H_{k+1} x = \frac{(y_k^T s_k)^2}{s_k^T y_k} = s_k^T y_k < 0,$$

which contradicts with $H_{k+1} \succ \mathbf{0}$.

Theorem 7 *If f is quadratic with positive definite Hessian F, then for the DFP method with exact line search*

$$s_i^T F s_j = 0, \quad 0 \le i < j \le k \tag{8}$$

$$H_{k+1} F s_i = s_i, \quad 0 \le i \le k. \tag{9}$$

Proof: We note that for the quadratic case

$$y_k = g_{k+1} - g_k = F x^{k+1} - F x^k = F s_k. \tag{10}$$

Also

$$H_{k+1} F s_k = H_{k+1} y_k = s_k \tag{11}$$

from the DFP formulation.

We now prove by induction. From (11) we see that (8) and (9) are true for $k = 0$. Assuming (8) and (9) are true for $k - 1$, we prove (8) and (9) are true for k. We have

$$g_k = g_{i+1} + F(s_{i+1} + \cdots + s_{k-1}).$$

Therefore, from the accurate line search rule and (8)

$$s_i^T g_k = s_i^T g_{i+1} = 0 \ \textbf{for} \ 0 \leq i < k.$$

Hence from (9)

$$s_i^T F H_k g_k = 0.$$

Thus since $s_k = -\alpha_k H_k g_k$ and $\alpha_k \neq 0$, we obtain

$$s_i^T F s_k = 0 \ \textbf{for} \ 0 \leq i < k,$$

which proves (8) for k.

Now since from (9) for $k-1$ and (10)

$$y_k^T H_k F s_i = y_k^T s_i = s_k^T F s_i = 0, \quad 0 \le i < k$$

we have

$$H_{k+1} F s_i = H_k F s_i = s_i, \quad 0 \le i < k.$$

This together with (11) proves (9) for k.

Remarks

- In the above theorem, $s_0, s_1, \ldots, s_{n-1}$ are F-orthogonal and hence they are linearly independent from (8). So, $H_n = F^{-1}$.
- DFP quasi-Newton method with exact line search terminates in a finite number of iterations if the function f is strictly convex.

We describe the DFP method for minimizing a differentiable function of several variables.

Initialization Step Let $\varepsilon > 0$ be a termination tolerance. Choose an initial point x_1 and an initial symmetric positive definite matrix H_1. Let $z_1 = x_1$, $k = j = 1$ and go to the Main Step.

Main Step

1. If $\|\nabla f(z_j)\| < \varepsilon$, stop; otherwise, let $p_j = -H_j \nabla f(z_j)$ and let λ_j be an optimal solution to the problem

$$\begin{aligned} \min \quad & f(z_j + \lambda p_j) \\ s.t. \quad & \lambda \geq 0. \end{aligned}$$

   Let $z_{j+1} = z_j + \lambda_j p_j$. If $j < n$, go to Step 2. If $j = n$, let $z_1 = x_{k+1} = z_{n+1}$, replace k by $k + 1$, let $j = 1$ and repeat Step 1.

2. Construct H_{j+1} as follows:

$$H_{j+1} = H_j + \frac{s_j s_j^T}{s_j^T y_j} - \frac{H_j y_j y_j^T H_j}{y_j^T H_j y_j},$$

where

$$s_j := \lambda_j p_j = z_{j+1} - z_j, \quad y_j = \nabla f(z_{j+1}) - f(z_j).$$

Replace j by $j+1$ and go to Step 1.

BFGS Quasi-Newton Method

This method was proposed independently by Broyden, Fletcher, Goldfarb, and Shanno. Performing a rank two correction procedure for B_k such that the new matrix satisfies the quasi-Newton condition (4), we get the BFGS formulation:

$$B_{k+1} = B_k + \frac{y_k y_k^T}{y_k^T s_k} - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k}.$$

It is easy to see that we can get BFGS or DFP formulation by simply exchanging y_k and s_k, H_k and B_k in (3) or (4).

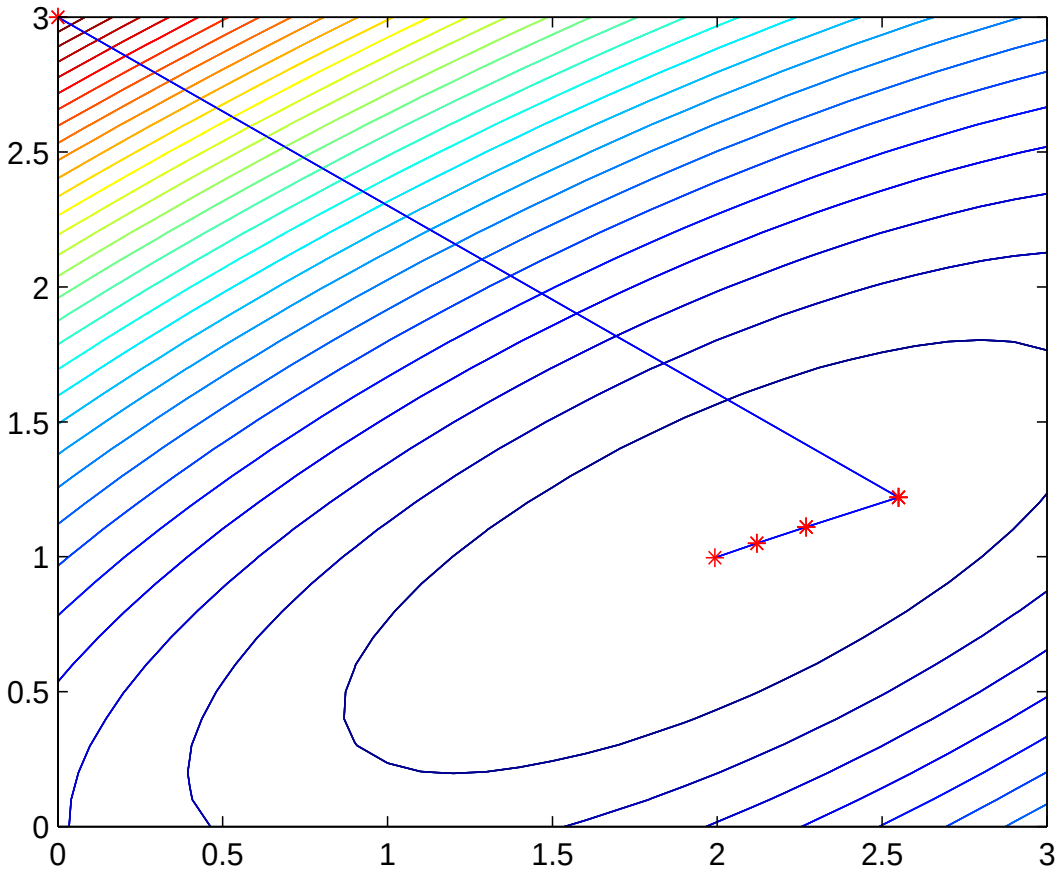
Example

Use the DFP method and the BFGS method to compute the following problem

$$\textbf{Minimize} \quad (x_1 - 2)^4 + (x_1 - 2x_2)^2.$$

The computing results are obtained by run *fminunc* in MatLab optimal toolbox and with the starting point $x^0 = (0; 3)$.

Algorithm	No.-iter	Opt.Sol	Opt.Val	Opt.Grad
DFP	114	(1.9921;0.9961)	3.9217e-009	2.3071e-005
BFGS	18	(1.9932;0.9966)	2.1839e-009	2.5295e-006


3
2.5
2
1.5
1
0.5
0
0
0.5
1
1.5
2
2.5
3