最佳化方法與應用
MA5037-*

Chapter 3. Line Search Methods

Introduction

Each iteration of a line search method computes a search direction p_k and then decides how far to move along that direction. The iteration is given by

$$x_{k+1} = x_k + \alpha_k p_k\,,$$

where the **positive** scalar α_k is called the **step length**. The success of a line search method depends on effective choices of both the direction p_k and the step length α_k.

Introduction

Most line search algorithms require p_k to be a descent direction satisfying

$$p_k^{\mathrm{T}} \nabla f_k < 0$$

because this property guarantees that the function f can be reduced along this direction, as discussed in the previous chapter. Moreover, the search direction often has the form

$$p_k = -B_k^{-1} \nabla f_k \,, \qquad (1)$$

where B_k is a symmetric and non-singular matrix.

1. In the steepest descent method, B_k is the identity matrix I.
2. In Newton's method, B_k is the exact Hessian $(\nabla^2 f)(x_k)$.
3. In quasi-Newton methods, B_k is an approximation to the Hessian that is updated at every iteration by means of a low-rank formula.

Introduction

Most line search algorithms require p_k to be a descent direction satisfying

$$p_k^{\mathrm{T}} \nabla f_k < 0$$

because this property guarantees that the function f can be reduced along this direction, as discussed in the previous chapter. Moreover, the search direction often has the form

$$p_k = -B_k^{-1} \nabla f_k \,, \tag{1}$$

where B_k is a symmetric and non-singular matrix.

1. In the steepest descent method, B_k is the identity matrix I.
2. In Newton's method, B_k is the exact Hessian $(\nabla^2 f)(x_k)$.
3. In quasi-Newton methods, B_k is an approximation to the Hessian that is updated at every iteration by means of a low-rank formula.

Introduction

Most line search algorithms require p_k to be a descent direction satisfying

$$p_k^{\mathrm{T}} \nabla f_k < 0$$

because this property guarantees that the function f can be reduced along this direction, as discussed in the previous chapter. Moreover, the search direction often has the form

$$p_k = -B_k^{-1} \nabla f_k \,, \tag{1}$$

where B_k is a symmetric and non-singular matrix.

1. In the steepest descent method, B_k is the identity matrix I.
2. In Newton's method, B_k is the exact Hessian $(\nabla^2 f)(x_k)$.
3. In quasi-Newton methods, B_k is an approximation to the Hessian that is updated at every iteration by means of a low-rank formula.

Introduction

When p_k is defined by (1) and B_k is positive definite, we have

$$p_k^{\mathrm{T}} \nabla f_k = -\nabla f_k^{\mathrm{T}} B_k^{-1} \nabla f_k < 0$$

and therefore p_k is a descent direction.

In this chapter, we discuss how to choose α_k and p_k to promote convergence from remote starting points. We also study the rate of convergence of steepest descent, quasi-Newton, and Newton methods. Since the pure Newton iteration is not guaranteed to produce descent directions when the current iterate is not close to a solution, we discuss modifications in Section 3.4 that allow it to start from any initial point.

Introduction

When p_k is defined by (1) and B_k is positive definite, we have

$$p_k^{\mathrm{T}} \nabla f_k = -\nabla f_k^{\mathrm{T}} B_k^{-1} \nabla f_k < 0$$

and therefore p_k is a descent direction.

In this chapter, we discuss how to choose α_k and p_k to promote convergence from remote starting points. We also study the rate of convergence of steepest descent, quasi-Newton, and Newton methods. Since the pure Newton iteration is not guaranteed to produce descent directions when the current iterate is not close to a solution, we discuss modifications in Section 3.4 that allow it to start from any initial point.

Introduction

When p_k is defined by (1) and B_k is positive definite, we have

$$p_k^{\mathrm{T}} \nabla f_k = -\nabla f_k^{\mathrm{T}} B_k^{-1} \nabla f_k < 0$$

and therefore p_k is a descent direction.

In this chapter, we discuss how to choose α_k and p_k to promote convergence from remote starting points. We also study the rate of convergence of steepest descent, quasi-Newton, and Newton methods. Since the pure Newton iteration is not guaranteed to produce descent directions when the current iterate is not close to a solution, we discuss modifications in Section 3.4 that allow it to start from any initial point.

§3.1 Step Length

In computing the step length α_k, we face a **tradeoff**. We would like to choose α_k to give a substantial reduction of f, but at the same time we do not want to spend too much time making the choice. The ideal choice would be the global minimizer of the univariate function $\varphi(\cdot)$ defined by

$$\varphi(\alpha) = f(x_k + \alpha p_k), \quad \alpha > 0, \tag{2}$$

but in general, it is too expensive to identify this value. To find even a local minimizer of φ to moderate precision generally requires too many evaluations of the objective function f and possibly the gradient ∇f. More practical strategies perform an inexact line search to identify a step length that achieves adequate reductions in f at minimal cost.

§3.1 Step Length

In computing the step length α_k, we face a **tradeoff**. We would like to choose α_k to give a substantial reduction of f, but at the same time we do not want to spend too much time making the choice. The ideal choice would be the global minimizer of the univariate function $\varphi(\cdot)$ defined by

$$\varphi(\alpha) = f(x_k + \alpha p_k), \quad \alpha > 0, \tag{2}$$

but in general, it is too expensive to identify this value. To find even a local minimizer of φ to moderate precision generally requires too many evaluations of the objective function f and possibly the gradient ∇f. More practical strategies perform an inexact line search to identify a step length that achieves adequate reductions in f at minimal cost.

§3.1 Step Length

In computing the step length α_k, we face a **tradeoff**. We would like to choose α_k to give a substantial reduction of f, but at the same time we do not want to spend too much time making the choice. The ideal choice would be the global minimizer of the univariate function $\varphi(\cdot)$ defined by

$$\varphi(\alpha) = f(x_k + \alpha p_k), \quad \alpha > 0, \tag{2}$$

but in general, it is too expensive to identify this value. To find even a local minimizer of φ to moderate precision generally requires too many evaluations of the objective function f and possibly the gradient ∇f. More practical strategies perform an inexact line search to identify a step length that achieves adequate reductions in f at minimal cost.

§3.1 Step Length

In computing the step length α_k, we face a **tradeoff**. We would like to choose α_k to give a substantial reduction of f, but at the same time we do not want to spend too much time making the choice. The ideal choice would be the global minimizer of the univariate function $\varphi(\cdot)$ defined by

$$\varphi(\alpha) = f(x_k + \alpha p_k), \quad \alpha > 0, \tag{2}$$

but in general, it is too expensive to identify this value. To find even a local minimizer of φ to moderate precision generally requires too many evaluations of the objective function f and possibly the gradient ∇f. More practical strategies perform an inexact line search to identify a step length that achieves adequate reductions in f at minimal cost.

§3.1 Step Length

Typical line search algorithms try out a sequence of candidate values for α, stopping to accept one of these values when certain conditions are satisfied. The line search is done in two stages:

1. A bracketing phase finds an interval containing desirable step lengths, and
2. a bisection or interpolation phase computes a good step length within this interval.

Sophisticated line search algorithms can be quite complicated, so we defer a full description until Section 3.5.

§3.1 Step Length

Typical line search algorithms try out a sequence of candidate values for α, stopping to accept one of these values when certain conditions are satisfied. The line search is done in two stages:

1. A bracketing phase finds an interval containing desirable step lengths, and
2. a bisection or interpolation phase computes a good step length within this interval.

Sophisticated line search algorithms can be quite complicated, so we defer a full description until Section 3.5.

§3.1 Step Length

Typical line search algorithms try out a sequence of candidate values for α, stopping to accept one of these values when certain conditions are satisfied. The line search is done in two stages:

1. A bracketing phase finds an interval containing desirable step lengths, and
2. a bisection or interpolation phase computes a good step length within this interval.

Sophisticated line search algorithms can be quite complicated, so we defer a full description until Section 3.5.

§3.1 Step Length

We now discuss various termination conditions for line search algorithms and show that effective step lengths need not lie near minimizers of the univariate function $\varphi(\alpha)$ defined in (2). A simple condition we could impose on α_k is to require a reduction in f; that is, $f(x_k + \alpha_k p_k) < f(x_k)$. One example of that this requirement is **not** enough to produce convergence to x_* is illustrated in Figure 1.

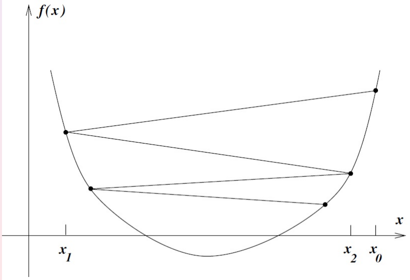

Figure 1: Insufficient reduction in f

§3.1 Step Length

We now discuss various termination conditions for line search algorithms and show that effective step lengths need not lie near minimizers of the univariate function $\varphi(\alpha)$ defined in (2). A simple condition we could impose on α_k is to require a reduction in f; that is, $f(x_k + \alpha_k p_k) < f(x_k)$. One example of that this requirement is **not** enough to produce convergence to x_* is illustrated in Figure 1.

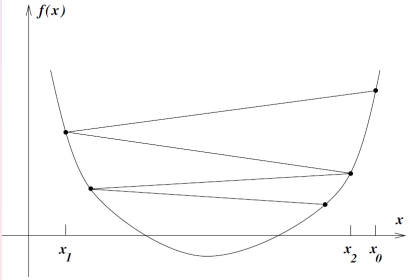

Figure 1: Insufficient reduction in f

§3.1 Step Length

In the example given in the previous page, the minimum function value is $f_* = -1$, but a sequence of iterates $\{x_k\}$ for which $f(x_k) = 5/k$, $k = 0, 1, \cdots$ yields a decrease at each iteration but has a limiting function value of zero. The insufficient reduction in f at each step causes it to fail to converge to the minimizer of this convex function. To avoid this behavior we need to enforce a sufficient decrease condition, a concept we discuss next.

§3.1 Step Length

In the example given in the previous page, the minimum function value is $f_* = -1$, but a sequence of iterates $\{x_k\}$ for which $f(x_k) = 5/k$, $k = 0, 1, \cdots$ yields a decrease at each iteration but has a limiting function value of zero. The insufficient reduction in f at each step causes it to fail to converge to the minimizer of this convex function. To avoid this behavior we need to enforce a sufficient decrease condition, a concept we discuss next.

§3.1 Step Length

- **The Wolfe Conditions**:

A popular inexact line search condition stipulates that α_k should first of all give sufficient decrease in the objective function f, as measured by the following inequality:

$$f(x_k + \alpha p_k) \leqslant f(x_k) + c_1 \alpha \nabla f_k^{\mathrm{T}} p_k \tag{3}$$

for some constant $c_1 \in (0, 1)$. In other words, the reduction in f should be proportional to both the step length α_k and the directional derivative $\nabla f_k^{\mathrm{T}} p_k$. Inequality (3) is sometimes called the **Armijo condition**.

Let $\ell(\alpha)$ denote the right-hand-side of (3); that is,

$$\ell(\alpha) = f(x_k) + c_1 \alpha \nabla f_k^{\mathrm{T}} p_k.$$

This function a linear function with **negative** slope $c_1 \nabla f_k^{\mathrm{T}} p_k$.

§3.1 Step Length

- **The Wolfe Conditions**:

A popular inexact line search condition stipulates that α_k should first of all give sufficient decrease in the objective function f, as measured by the following inequality:

$$f(x_k + \alpha p_k) \leqslant f(x_k) + c_1 \alpha \nabla f_k^{\mathrm{T}} p_k \tag{3}$$

for some constant $c_1 \in (0, 1)$. In other words, the reduction in f should be proportional to both the step length α_k and the directional derivative $\nabla f_k^{\mathrm{T}} p_k$. Inequality (3) is sometimes called the **Armijo condition**.

Let $\ell(\alpha)$ denote the right-hand-side of (3); that is,

$$\ell(\alpha) = f(x_k) + c_1 \alpha \nabla f_k^{\mathrm{T}} p_k .$$

This function a linear function with **negative** slope $c_1 \nabla f_k^{\mathrm{T}} p_k$.

§3.1 Step Length

- **The Wolfe Conditions**:

A popular inexact line search condition stipulates that α_k should first of all give sufficient decrease in the objective function f, as measured by the following inequality:

$$f(x_k + \alpha p_k) \leqslant f(x_k) + c_1 \alpha \nabla f_k^{\mathrm{T}} p_k \tag{3}$$

for some constant $c_1 \in (0,1)$. In other words, the reduction in f should be proportional to both the step length α_k and the directional derivative $\nabla f_k^{\mathrm{T}} p_k$. Inequality (3) is sometimes called the **Armijo condition**.

Let $\ell(\alpha)$ denote the right-hand-side of (3); that is,

$$\ell(\alpha) = f(x_k) + c_1 \alpha \nabla f_k^{\mathrm{T}} p_k.$$

This function a linear function with **negative** slope $c_1 \nabla f_k^{\mathrm{T}} p_k$.

§3.1 Step Length

- **The Wolfe Conditions**:

A popular inexact line search condition stipulates that α_k should first of all give sufficient decrease in the objective function f, as measured by the following inequality:

$$f(x_k + \alpha p_k) \leqslant f(x_k) + c_1 \alpha \nabla f_k^{\mathrm{T}} p_k \tag{3}$$

for some constant $c_1 \in (0, 1)$. In other words, the reduction in f should be proportional to both the step length α_k and the directional derivative $\nabla f_k^{\mathrm{T}} p_k$. Inequality (3) is sometimes called the **Armijo condition**.

Let $\ell(\alpha)$ denote the right-hand-side of (3); that is,

$$\ell(\alpha) = f(x_k) + c_1 \alpha \nabla f_k^{\mathrm{T}} p_k.$$

This function a linear function with **negative** slope $c_1 \nabla f_k^{\mathrm{T}} p_k$.

§3.1 Step Length

The sufficient decrease condition is illustrated in Figure 2.

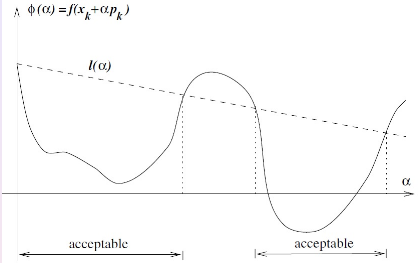

Figure 2: Sufficient decrease condition

Because $c_1 \in (0,1)$, it lies above the graph of φ for small positive values of α. The sufficient decrease condition states that α is acceptable only if $\varphi(\alpha) \leqslant \ell(\alpha)$. The intervals on which this condition is satisfied are shown in Figure 2. In practice, c_1 is chosen to be quite small, say $c_1 = 10^{-4}$.

§3.1 Step Length

The sufficient decrease condition is illustrated in Figure 2.

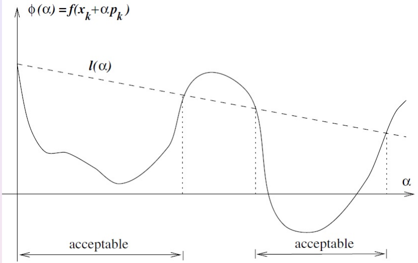

Figure 2: Sufficient decrease condition

Because $c_1 \in (0,1)$, it lies above the graph of φ for small positive values of α. The sufficient decrease condition states that α is acceptable only if $\varphi(\alpha) \leqslant \ell(\alpha)$. The intervals on which this condition is satisfied are shown in Figure 2. In practice, c_1 is chosen to be quite small, say $c_1 = 10^{-4}$.

§3.1 Step Length

The sufficient decrease condition is illustrated in Figure 2.

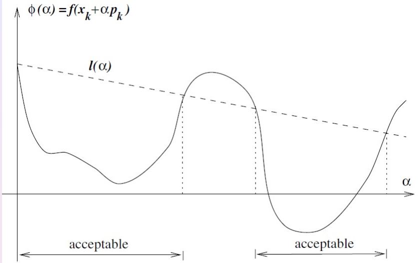

Figure 2: Sufficient decrease condition

Because $c_1 \in (0,1)$, it lies above the graph of φ for small positive values of α. The sufficient decrease condition states that α is acceptable only if $\varphi(\alpha) \leqslant \ell(\alpha)$. The intervals on which this condition is satisfied are shown in Figure 2. In practice, c_1 is chosen to be quite small, say $c_1 = 10^{-4}$.

§3.1 Step Length

The sufficient decrease condition is not enough by itself to ensure that the algorithm makes reasonable progress because, as we see from Figure 2, it is satisfied for all sufficiently small values of α. To **rule out unacceptably short steps** we introduce a second requirement, called the **curvature condition**, which requires α_k to satisfy

$$\nabla f(x_k + \alpha_k p_k)^{\mathrm{T}} p_k \geqslant c_2 \nabla f_k^{\mathrm{T}} p_k \tag{4}$$

for some constant $c_2 \in (c_1, 1)$, where c_1 is the constant from (3). Note that the left-hand side is simply the derivative $\varphi'(\alpha_k)$, so the curvature condition ensures that the slope of φ at α_k is greater than c_2 times the initial slope $\varphi'(0)$.

The curvature condition is illustrated in Figure 3 in the next page.

§3.1 Step Length

The sufficient decrease condition is not enough by itself to ensure that the algorithm makes reasonable progress because, as we see from Figure 2, it is satisfied for all sufficiently small values of α. To **rule out unacceptably short steps** we introduce a second requirement, called the **curvature condition**, which requires α_k to satisfy

$$\nabla f(x_k + \alpha_k p_k)^{\mathrm{T}} p_k \geqslant c_2 \nabla f_k^{\mathrm{T}} p_k \tag{4}$$

for some constant $c_2 \in (c_1, 1)$, where c_1 is the constant from (3). Note that the left-hand side is simply the derivative $\varphi'(\alpha_k)$, so the curvature condition ensures that the slope of φ at α_k is greater than c_2 times the initial slope $\varphi'(0)$.

The curvature condition is illustrated in Figure 3 in the next page.

§3.1 Step Length

The sufficient decrease condition is not enough by itself to ensure that the algorithm makes reasonable progress because, as we see from Figure 2, it is satisfied for all sufficiently small values of α. To **rule out unacceptably short steps** we introduce a second requirement, called the **curvature condition**, which requires α_k to satisfy

$$\nabla f(x_k + \alpha_k p_k)^{\mathrm{T}} p_k \geqslant c_2 \nabla f_k^{\mathrm{T}} p_k \tag{4}$$

for some constant $c_2 \in (c_1, 1)$, where c_1 is the constant from (3). Note that the left-hand side is simply the derivative $\varphi'(\alpha_k)$, so the curvature condition ensures that the slope of φ at α_k is greater than c_2 times the initial slope $\varphi'(0)$.

The curvature condition is illustrated in Figure 3 in the next page.

§3.1 Step Length

The sufficient decrease condition is not enough by itself to ensure that the algorithm makes reasonable progress because, as we see from Figure 2, it is satisfied for all sufficiently small values of α. To **rule out unacceptably short steps** we introduce a second requirement, called the **curvature condition**, which requires α_k to satisfy

$$\nabla f(x_k + \alpha_k p_k)^{\mathrm{T}} p_k \geqslant c_2 \nabla f_k^{\mathrm{T}} p_k \tag{4}$$

for some constant $c_2 \in (c_1, 1)$, where c_1 is the constant from (3). Note that the left-hand side is simply the derivative $\varphi'(\alpha_k)$, so the curvature condition ensures that the slope of φ at α_k is greater than c_2 times the initial slope $\varphi'(0)$.

The curvature condition is illustrated in Figure 3 in the next page.

§3.1 Step Length

On the other hand, if $\varphi'(\alpha_k)$ is only slightly negative or even positive, it is a sign that we cannot expect much more decrease in f in this direction, so it makes sense to terminate the line search. Typical values of c_2 are 0.9 when the search direction p_k is chosen by a Newton or quasi-Newton method, and 0.1 when p_k is obtained from a nonlinear conjugate gradient method.

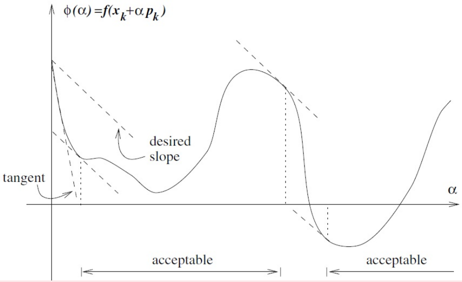

Figure 3: The curvature condition

§3.1 Step Length

The sufficient decrease and curvature conditions are known collectively as the **Wolfe conditions**:

$$f(x_k + \alpha_k p_k) \leqslant f(x_k) + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \,, \tag{5a}$$

$$\nabla f(x_k + \alpha_k p_k)^{\mathrm{T}} p_k \geqslant c_2 \nabla f_k^{\mathrm{T}} p_k \,, \tag{5b}$$

with $0 < c_1 < c_2 < 1$. We illustrate them in Figure 4.

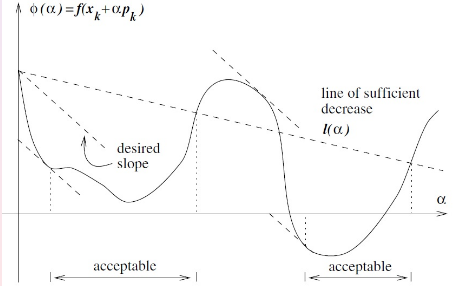

Figure 4: Step lengths satisfying the Wolfe conditions

§3.1 Step Length

A step length may satisfy the Wolfe conditions without being particularly close to a minimizer of φ, as we show in Figure 4. We can, however, modify the curvature condition to force α_k to lie in at least a broad neighborhood of a local minimizer or stationary point of φ. The strong Wolfe conditions require α_k to satisfy

$$f(x_k + \alpha_k p_k) \leqslant f(x_k) + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \,, \tag{6a}$$

$$\left|\nabla f(x_k + \alpha_k p_k)^{\mathrm{T}} p_k\right| \leqslant c_2 \left|\nabla f_k^{\mathrm{T}} p_k\right| \,, \tag{6b}$$

with $0 < c_1 < c_2 < 1$. The only difference with the Wolfe conditions is that we no longer allow the derivative $\varphi'(\alpha_k)$ to be too positive. Hence, we exclude points that are far from stationary points of φ.

§3.1 Step Length

A step length may satisfy the Wolfe conditions without being particularly close to a minimizer of φ, as we show in Figure 4. We can, however, modify the curvature condition to force α_k to lie in at least a broad neighborhood of a local minimizer or stationary point of φ. The strong Wolfe conditions require α_k to satisfy

$$f(x_k + \alpha_k p_k) \leqslant f(x_k) + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \,, \tag{6a}$$

$$\left|\nabla f(x_k + \alpha_k p_k)^{\mathrm{T}} p_k\right| \leqslant c_2 \left|\nabla f_k^{\mathrm{T}} p_k\right| \,, \tag{6b}$$

with $0 < c_1 < c_2 < 1$. The only difference with the Wolfe conditions is that we no longer allow the derivative $\varphi'(\alpha_k)$ to be too positive. Hence, we exclude points that are far from stationary points of φ.

§3.1 Step Length

Lemma

Suppose that $f:\mathbb{R}^n\to\mathbb{R}$ is continuously differentiable. Let p_k be a descent direction at x_k, and assume that f is bounded from below along the ray $\{x_k+\alpha p_k \,|\, \alpha>0\}$. Then if $0<c_1<c_2<1$, there exist intervals of step lengths satisfying the Wolfe conditions (5) and the strong Wolfe conditions (6).

Proof.

Define $\varphi(\alpha)\equiv f(x_k+\alpha p_k)$ and $\ell(\alpha)\equiv f(x_k)+\alpha c_1\nabla f_k^{\rm T} p_k$. By the differentiability of f,

$$f(x_k+\alpha p_k)-f(x_k)-\alpha\nabla f_k^{\rm T} p_k = o(\|\alpha p_k\|) = o(|\alpha|)\,.$$

Since p_k is a descent direction, $\nabla f_k^{\rm T} p_k<0$. By the fact that $c_1\in(0,1)$, there exists $\delta>0$ such that

$$\varphi(\alpha)-\ell(\alpha) = (1-c_1)\alpha\nabla f_k^{\rm T} p_k + o(|\alpha|) < 0 \quad \text{if } 0<\alpha<\delta\,. \qquad \square$$

§3.1 Step Length

Lemma

Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable. Let p_k be a descent direction at x_k, and assume that f is bounded from below along the ray $\{x_k + \alpha p_k \,|\, \alpha > 0\}$. Then if $0 < c_1 < c_2 < 1$, there exist intervals of step lengths satisfying the Wolfe conditions (5) and the strong Wolfe conditions (6).

Proof.

Define $\varphi(\alpha) \equiv f(x_k + \alpha p_k)$ and $\ell(\alpha) \equiv f(x_k) + \alpha c_1 \nabla f_k^{\mathrm{T}} p_k$. By the differentiability of f,

$$f(x_k + \alpha p_k) - f(x_k) - \alpha \nabla f_k^{\mathrm{T}} p_k = o(\|\alpha p_k\|) = o(|\alpha|)\,.$$

Since p_k is a descent direction, $\nabla f_k^{\mathrm{T}} p_k < 0$. By the fact that $c_1 \in (0,1)$, there exists $\delta > 0$ such that

$$\varphi(\alpha) - \ell(\alpha) = (1 - c_1)\alpha \nabla f_k^{\mathrm{T}} p_k + o(|\alpha|) < 0 \quad \text{if } 0 < \alpha < \delta\,. \qquad \square$$

§3.1 Step Length

Lemma

Suppose that $f:\mathbb{R}^n \to \mathbb{R}$ is continuously differentiable. Let p_k be a descent direction at x_k, and assume that f is bounded from below along the ray $\{x_k + \alpha p_k \,|\, \alpha > 0\}$. Then if $0 < c_1 < c_2 < 1$, there exist intervals of step lengths satisfying the Wolfe conditions (5) and the strong Wolfe conditions (6).

Proof.

Define $\varphi(\alpha) \equiv f(x_k + \alpha p_k)$ and $\ell(\alpha) \equiv f(x_k) + \alpha c_1 \nabla f_k^{\mathrm{T}} p_k$. By the differentiability of f,

$$f(x_k + \alpha p_k) - f(x_k) - \alpha \nabla f_k^{\mathrm{T}} p_k = o(\|\alpha p_k\|) = o(|\alpha|)\,.$$

Since p_k is a descent direction, $\nabla f_k^{\mathrm{T}} p_k < 0$. By the fact that $c_1 \in (0,1)$, there exists $\delta > 0$ such that

$$\varphi(\alpha) - \ell(\alpha) = (1 - c_1)\alpha \nabla f_k^{\mathrm{T}} p_k + o(|\alpha|) < 0 \quad \text{if } 0 < \alpha < \delta\,. \quad \square$$

§3.1 Step Length

Proof (cont'd).

Therefore, $\varphi(\alpha) < \ell(\alpha)$ whenever $0 < \alpha < \delta$.

By assumption, there exists $m \in \mathbb{R}$ such that $\varphi(\alpha) \geqslant m$ for all $\alpha > 0$, while the fact that $\nabla f_k^{\mathrm{T}} p_k < 0$ implies that

$$\lim_{\alpha \to \infty} \ell(\alpha) = -\infty .$$

Therefore, the continuity of φ and ℓ implies that the set $\{\alpha > 0 \,|\, \varphi(\alpha) = \ell(\alpha)\}$ is non-empty. Let

$$\bar{\alpha} = \inf\left\{\alpha > 0 \,\middle|\, f(x_k + \alpha p_k) = f(x_k) + \alpha c_1 \nabla f_k^{\mathrm{T}} p_k\right\}.$$

Then $\bar{\alpha} \geqslant \delta$, and the sufficient decrease condition (5a)/(6a) clearly holds for all step lengths less than $\bar{\alpha}$. □

§3.1 Step Length

Proof (cont'd).

Therefore, $\varphi(\alpha) < \ell(\alpha)$ whenever $0 < \alpha < \delta$.

By assumption, there exists $m \in \mathbb{R}$ such that $\varphi(\alpha) \geqslant m$ for all $\alpha > 0$, while the fact that $\nabla f_k^{\mathrm{T}} p_k < 0$ implies that

$$\lim_{\alpha \to \infty} \ell(\alpha) = -\infty \,.$$

Therefore, the continuity of φ and ℓ implies that the set $\{\alpha > 0 \,|\, \varphi(\alpha) = \ell(\alpha)\}$ is non-empty. Let

$$\bar{\alpha} = \inf \left\{ \alpha > 0 \,\middle|\, f(x_k + \alpha p_k) = f(x_k) + \alpha c_1 \nabla f_k^{\mathrm{T}} p_k \right\}.$$

Then $\bar{\alpha} \geqslant \delta$, and the sufficient decrease condition (5a)/(6a) clearly holds for all step lengths less than $\bar{\alpha}$. □

§3.1 Step Length

Proof (cont'd).

Therefore, $\varphi(\alpha) < \ell(\alpha)$ whenever $0 < \alpha < \delta$.

By assumption, there exists $m \in \mathbb{R}$ such that $\varphi(\alpha) \geqslant m$ for all $\alpha > 0$, while the fact that $\nabla f_k^{\mathrm{T}} p_k < 0$ implies that

$$\lim_{\alpha \to \infty} \ell(\alpha) = -\infty\,.$$

Therefore, the continuity of φ and ℓ implies that the set $\{\alpha > 0 \,|\, \varphi(\alpha) = \ell(\alpha)\}$ is non-empty. Let

$$\bar{\alpha} = \inf\left\{\alpha > 0 \,\middle|\, f(x_k + \alpha p_k) = f(x_k) + \alpha c_1 \nabla f_k^{\mathrm{T}} p_k\right\}.$$

Then $\bar{\alpha} \geqslant \delta$, and the sufficient decrease condition (5a)/(6a) clearly holds for all step lengths less than $\bar{\alpha}$. □

§3.1 Step Length

Proof (cont'd).

By the mean value theorem, there exists $\tilde{\alpha} \in (0, \bar{\alpha})$ such that

$$f(x_k + \bar{\alpha} p_k) - f(x_k) = \bar{\alpha} (\nabla f)(x_k + \tilde{\alpha} p_k)^{\mathrm{T}} p_k \,.$$

By the definition of $\bar{\alpha}$ and the continuity of φ and ℓ,

$$f(x_k + \bar{\alpha} p_k) = \varphi(\bar{\alpha}) = \ell(\bar{\alpha}) = f(x_k) + \bar{\alpha} c_1 \nabla f_k^{\mathrm{T}} p_k \,;$$

thus the fact that $0 < c_1 < c_2 < 1$ implies that

$$(\nabla f)(x_k + \tilde{\alpha} p_k)^{\mathrm{T}} p_k = c_1 \nabla f_k^{\mathrm{T}} p_k > c_2 \nabla f_k^{\mathrm{T}} p_k \,. \tag{7}$$

Therefore, $\tilde{\alpha}$ satisfies the Wolfe conditions (5), and the inequalities hold **strictly** in both (5a) and (5b). Hence, by our smoothness assumption on f, there is an interval around $\tilde{\alpha}$ for which the Wolfe conditions hold. The negativity of the left-hand side of (7) shows that the strong Wolfe conditions (6) hold in the same interval. □

§3.1 Step Length

Proof (cont'd).

By the mean value theorem, there exists $\tilde{\alpha} \in (0, \bar{\alpha})$ such that

$$f(x_k + \bar{\alpha} p_k) - f(x_k) = \bar{\alpha} (\nabla f)(x_k + \tilde{\alpha} p_k)^{\mathrm{T}} p_k \,.$$

By the definition of $\bar{\alpha}$ and the continuity of φ and ℓ,

$$f(x_k + \bar{\alpha} p_k) = \varphi(\bar{\alpha}) = \ell(\bar{\alpha}) = f(x_k) + \bar{\alpha} c_1 \nabla f_k^{\mathrm{T}} p_k \,;$$

thus the fact that $0 < c_1 < c_2 < 1$ implies that

$$(\nabla f)(x_k + \tilde{\alpha} p_k)^{\mathrm{T}} p_k = c_1 \nabla f_k^{\mathrm{T}} p_k > c_2 \nabla f_k^{\mathrm{T}} p_k \,. \tag{7}$$

Therefore, $\tilde{\alpha}$ satisfies the Wolfe conditions (5), and the inequalities hold **strictly** in both (5a) and (5b). Hence, by our smoothness assumption on f, there is an interval around $\tilde{\alpha}$ for which the Wolfe conditions hold. The negativity of the left-hand side of (7) shows that the strong Wolfe conditions (6) hold in the same interval. □

§3.1 Step Length

Proof (cont'd).

By the mean value theorem, there exists $\tilde{\alpha} \in (0, \bar{\alpha})$ such that

$$f(x_k + \bar{\alpha} p_k) - f(x_k) = \bar{\alpha}(\nabla f)(x_k + \tilde{\alpha} p_k)^{\mathrm{T}} p_k \,.$$

By the definition of $\bar{\alpha}$ and the continuity of φ and ℓ,

$$f(x_k + \bar{\alpha} p_k) = \varphi(\bar{\alpha}) = \ell(\bar{\alpha}) = f(x_k) + \bar{\alpha} c_1 \nabla f_k^{\mathrm{T}} p_k \,;$$

thus the fact that $0 < c_1 < c_2 < 1$ implies that

$$(\nabla f)(x_k + \tilde{\alpha} p_k)^{\mathrm{T}} p_k = c_1 \nabla f_k^{\mathrm{T}} p_k > c_2 \nabla f_k^{\mathrm{T}} p_k \,. \tag{7}$$

Therefore, $\tilde{\alpha}$ satisfies the Wolfe conditions (5), and the inequalities hold **strictly** in both (5a) and (5b). Hence, by our smoothness assumption on f, there is an interval around $\tilde{\alpha}$ for which the Wolfe conditions hold. The negativity of the left-hand side of (7) shows that the strong Wolfe conditions (6) hold in the same interval. □

§3.1 Step Length

The Wolfe conditions are scale-invariant in a broad sense: Multiplying the objective function by a constant or making an affine change of variables does not alter them. They can be used in most line search methods, and are particularly important in the implementation of quasi-Newton methods.

Remark: For the purpose of the analysis it sometimes requires that the step length obtained by the exact line search is used. Suppose that $f(x) = \frac{1}{2}x^{\mathrm{T}}Qx$ for some positive definite matrix Q. For a descent direction p_k, the exact line search step length α_k is given by

$$\alpha_k = -\frac{x_k^{\mathrm{T}}Qp_k}{p_k^{\mathrm{T}}Qp_k}$$

since if $\varphi(\alpha) = f(x_k + \alpha p_k)$, then $\varphi'(\alpha) = x_k^{\mathrm{T}}Qp_k + \alpha p_k^{\mathrm{T}}Qp_k$.

§3.1 Step Length

The Wolfe conditions are scale-invariant in a broad sense: Multiplying the objective function by a constant or making an affine change of variables does not alter them. They can be used in most line search methods, and are particularly important in the implementation of quasi-Newton methods.

Remark: For the purpose of the analysis it sometimes requires that the step length obtained by the exact line search is used. Suppose that $f(x) = \frac{1}{2}x^{\mathrm{T}}Qx$ for some positive definite matrix Q. For a descent direction p_k, the exact line search step length α_k is given by

$$\alpha_k = -\frac{x_k^{\mathrm{T}}Qp_k}{p_k^{\mathrm{T}}Qp_k}$$

since if $\varphi(\alpha) = f(x_k + \alpha p_k)$, then $\varphi'(\alpha) = x_k^{\mathrm{T}}Qp_k + \alpha p_k^{\mathrm{T}}Qp_k$.

§3.1 Step Length

Therefore, for the Armijo condition (5a) to hold with this α_k, we must have $c_1 \leqslant \frac{1}{2}$ since

$$\begin{aligned}
&\frac{1}{2}(x_k+\alpha_k p_k)^{\mathrm{T}} Q (x_k+\alpha_k p_k) \leqslant \frac{1}{2} x_k^{\mathrm{T}} Q x_k - c_1 \alpha_k x_k^{\mathrm{T}} Q p_k \\
&\Leftrightarrow \ \alpha_k x_k^{\mathrm{T}} Q p_k + \frac{1}{2}\alpha_k^2 p_k^{\mathrm{T}} Q p_k \leqslant -c_1 \alpha_k x_k^{\mathrm{T}} Q p_k \\
&\Leftrightarrow \ x_k^{\mathrm{T}} Q p_k + \frac{1}{2}\alpha_k p_k^{\mathrm{T}} Q p_k \leqslant -c_1 x_k^{\mathrm{T}} Q p_k \\
&\Leftrightarrow \ -\alpha_k + \frac{1}{2}\alpha_k \leqslant -c_1 \alpha_k \\
&\Leftrightarrow \ c_1 \leqslant \frac{1}{2}\,.
\end{aligned}$$

This implies that if $c_1 > 1/2$, then the line search would exclude the minimizer of a quadratic, so later on we usually assume that $c_1 \leqslant 1/2$ in the Armijo condition.

§3.1 Step Length

Moreover, for this particular quadratic function f, at the k-th iterate x_k, the Newton direction $p_k^{\rm N}$ is given by

$$p_k^{\rm N} = -\big[(\nabla f)^2(x_k)\big]^{-1}\nabla f_k = -Q^{-1}(Qx_k) = -x_k\,;$$

thus for the Armijo condition (5a) to hold with $p_k = p_k^{\rm N}$ and $\alpha_k = 1$, we must have $c_1 \leqslant \frac{1}{2}$ since

$$\begin{aligned} \frac{1}{2}(x_k - x_k)^{\rm T}Q(x_k - x_k) &\leqslant \frac{1}{2}x_k^{\rm T}Qx_k - c_1 x_k^{\rm T}Qx_k \\ \Leftrightarrow\ \ c_1 x_k^{\rm T}Qx_k &\leqslant \frac{1}{2}x_k^{\rm T}Qx_k \\ \Leftrightarrow\ \ c_1 &\leqslant \frac{1}{2}\,. \end{aligned}$$

Therefore, if $c_1 > 1/2$, then the unit step lengths may not be admissible. This is another way of seeing that one needs $c_1 \leqslant 1/2$ in the Armijo condition.

§3.1 Step Length

Moreover, for this particular quadratic function f, at the k-th iterate x_k, the Newton direction $p_k^{\rm N}$ is given by

$$p_k^{\rm N} = -\big[(\nabla f)^2(x_k)\big]^{-1}\nabla f_k = -Q^{-1}(Qx_k) = -x_k\,;$$

thus for the Armijo condition (5a) to hold with $p_k = p_k^{\rm N}$ and $\alpha_k = 1$, we must have $c_1 \leqslant \dfrac{1}{2}$ since

$$\begin{aligned}
\frac{1}{2}(x_k - x_k)^{\rm T}Q(x_k - x_k) &\leqslant \frac{1}{2}x_k^{\rm T}Qx_k - c_1 x_k^{\rm T}Qx_k \\
\Leftrightarrow\quad c_1 x_k^{\rm T}Qx_k &\leqslant \frac{1}{2}x_k^{\rm T}Qx_k \\
\Leftrightarrow\quad c_1 &\leqslant \frac{1}{2}\,.
\end{aligned}$$

Therefore, if $c_1 > 1/2$, then the unit step lengths may not be admissible. This is another way of seeing that one needs $c_1 \leqslant 1/2$ in the Armijo condition.

§3.1 Step Length

- **The Goldstein Conditions**:

Like the Wolfe conditions, the *Goldstein conditions* ensure that the step length α achieves sufficient decrease but is not too short. The Goldstein conditions can also be stated as a pair of inequalities:

$$f(x_k) + (1-c)\alpha_k \nabla f_k^{\mathrm{T}} p_k \leqslant f(x_k + \alpha_k p_k) \leqslant f(x_k) + c\alpha_k \nabla f_k^{\mathrm{T}} p_k \quad (8)$$

with $0 < c < 1/2$. The second inequality is the sufficient decrease (Armijo) condition (3), whereas the first inequality is introduced to control the step length from below. See Figure 5 on the next page.

§3.1 Step Length

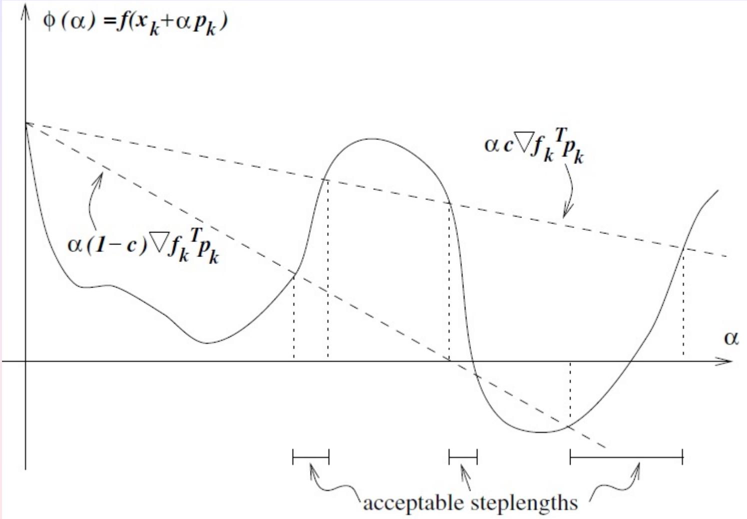

Figure 5: The Goldstein conditions

§3.1 Step Length

Compared with the Wolfe conditions, a **disadvantage** of the Goldstein conditions is that the first inequality in (8) may exclude all minimizers of φ. However, the Goldstein and Wolfe conditions have much in common, and their convergence theories are quite similar. The Goldstein conditions are often used in Newton-type methods but are not well suited for quasi-Newton methods that maintain a positive definite Hessian approximation.

§3.1 Step Length

- **Sufficient Decrease and Backtracking**:

The sufficient decrease (Armijo) condition (3) alone is not sufficient to ensure that the algorithm makes reasonable progress along the given search direction. However, if the line search algorithm chooses its candidate step lengths using a so-called **backtracking** approach, we can dispense with the extra condition (5b) and use just the sufficient decrease condition to terminate the line search procedure. In its most basic form, backtracking proceeds as follows.

Algorithm 3.1 (Backtracking Line Search):

Choose $\bar{\alpha} > 0$, $\rho \in (0, 1)$, $c \in (0, 1)$; Set $\alpha \leftarrow \bar{\alpha}$;

while $f(x_k + \alpha p_k) > f(x_k) + c\alpha \nabla f_k^T p_k$

$\alpha \leftarrow \rho\alpha$;

end

Terminate with $\alpha_k = \alpha$.

§3.1 Step Length

- **Sufficient Decrease and Backtracking**:

The sufficient decrease (Armijo) condition (3) alone is not sufficient to ensure that the algorithm makes reasonable progress along the given search direction. However, if the line search algorithm chooses its candidate step lengths using a so-called **backtracking** approach, we can dispense with the extra condition (5b) and use just the sufficient decrease condition to terminate the line search procedure. In its most basic form, backtracking proceeds as follows.

Algorithm 3.1 (Backtracking Line Search):

Choose $\bar{\alpha} > 0$, $\rho \in (0,1)$, $c \in (0,1)$; Set $\alpha \leftarrow \bar{\alpha}$;

while $f(x_k + \alpha p_k) > f(x_k) + c\alpha \nabla f_k^T p_k$

$\alpha \leftarrow \rho\alpha$;

end

Terminate with $\alpha_k = \alpha$.

§3.1 Step Length

- **Sufficient Decrease and Backtracking**:

The sufficient decrease (Armijo) condition (3) alone is not sufficient to ensure that the algorithm makes reasonable progress along the given search direction. However, if the line search algorithm chooses its candidate step lengths using a so-called **backtracking** approach, we can dispense with the extra condition (5b) and use just the sufficient decrease condition to terminate the line search procedure. In its most basic form, backtracking proceeds as follows.

Algorithm 3.1 (Backtracking Line Search):

Choose $\bar{\alpha} > 0$, $\rho \in (0,1)$, $c \in (0,1)$; Set $\alpha \leftarrow \bar{\alpha}$;

while $f(x_k + \alpha p_k) > f(x_k) + c\alpha \nabla f_k^T p_k$

$\alpha \leftarrow \rho\alpha$;

end

Terminate with $\alpha_k = \alpha$.

§3.1 Step Length

In this procedure, the initial step length $\bar{\alpha}$ is chosen to be 1 in Newton and quasi-Newton methods, but can have different values in other algorithms such as steepest descent or conjugate gradient. An acceptable step length will be found after a finite number of trials, because α_k will eventually become small enough that the sufficient decrease condition holds. In practice, the contraction factor ρ is often allowed to vary at each iteration of the line search. For example, it can be chosen by safeguarded interpolation, as we describe later. We need ensure only that at each iteration we have $\rho \in [\rho_{\text{lo}}, \rho_{\text{hi}}]$, for some fixed constants $0 < \rho_{\text{lo}} < \rho_{\text{hi}} < 1$.

§3.1 Step Length

In this procedure, the initial step length $\bar{\alpha}$ is chosen to be 1 in Newton and quasi-Newton methods, but can have different values in other algorithms such as steepest descent or conjugate gradient. An acceptable step length will be found after a finite number of trials, because α_k will eventually become small enough that the sufficient decrease condition holds. In practice, the contraction factor ρ is often allowed to vary at each iteration of the line search. For example, it can be chosen by safeguarded interpolation, as we describe later. We need ensure only that at each iteration we have $\rho \in [\rho_{\text{lo}}, \rho_{\text{hi}}]$, for some fixed constants $0 < \rho_{\text{lo}} < \rho_{\text{hi}} < 1$.

§3.1 Step Length

In this procedure, the initial step length $\bar{\alpha}$ is chosen to be 1 in Newton and quasi-Newton methods, but can have different values in other algorithms such as steepest descent or conjugate gradient. An acceptable step length will be found after a finite number of trials, because α_k will eventually become small enough that the sufficient decrease condition holds. In practice, the contraction factor ρ is often allowed to vary at each iteration of the line search. For example, it can be chosen by safeguarded interpolation, as we describe later. We need ensure only that at each iteration we have $\rho \in [\rho_{\text{lo}}, \rho_{\text{hi}}]$, for some fixed constants $0 < \rho_{\text{lo}} < \rho_{\text{hi}} < 1$.

§3.1 Step Length

The backtracking approach ensures either that the selected step length α_k is some fixed value (the initial choice $\bar{\alpha}$), or else that it is short enough to satisfy the sufficient decrease condition but not too short. The latter claim holds because the accepted value α_k is within a factor ρ of the previous trial value, α_k/ρ, which was rejected for violating the sufficient decrease condition; that is, for being too long. This simple and popular strategy for terminating a line search is well suited for Newton methods but is less appropriate for quasi-Newton and conjugate gradient methods.

§3.1 Step Length

The backtracking approach ensures either that the selected step length α_k is some fixed value (the initial choice $\bar{\alpha}$), or else that it is short enough to satisfy the sufficient decrease condition but not too short. The latter claim holds because the accepted value α_k is within a factor ρ of the previous trial value, α_k/ρ, which was rejected for violating the sufficient decrease condition; that is, for being too long. This simple and popular strategy for terminating a line search is well suited for Newton methods but is less appropriate for quasi-Newton and conjugate gradient methods.

§3.1 Step Length

The backtracking approach ensures either that the selected step length α_k is some fixed value (the initial choice $\bar{\alpha}$), or else that it is short enough to satisfy the sufficient decrease condition but not too short. The latter claim holds because the accepted value α_k is within a factor ρ of the previous trial value, α_k/ρ, which was rejected for violating the sufficient decrease condition; that is, for being too long. This simple and popular strategy for terminating a line search is well suited for Newton methods but is less appropriate for quasi-Newton and conjugate gradient methods.

§3.2 Convergence of Line Search Methods

To obtain global convergence, we must not only have well chosen step lengths but also well chosen search directions p_k. We discuss requirements on the search direction in this section, focusing on one key property: the angle θ_k between p_k and the steepest descent direction $-\nabla f_k$, defined by

$$\cos\theta_k = \frac{-\nabla f_k^{\mathrm{T}} p_k}{\|\nabla f_k\| \|p_k\|}.$$

§3.2 Convergence of Line Search Methods

The following theorem, due to Zoutendijk, has far-reaching consequences. It quantifies the effect of properly chosen step lengths α_k, and shows, for example, that the steepest descent method is globally convergent. For other algorithms, it describes how far p_k can deviate from the steepest descent direction and still produce a globally convergent iteration. Various line search termination conditions can be used to establish this result, but for concreteness we will consider only the Wolfe conditions (5). Though Zoutendijk's result appears at first to be technical and obscure, its power will soon become evident.

§3.2 Convergence of Line Search Methods

Theorem (Zoutendijk)

Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable, and $\{x_k\}$ be a sequence of iterates taking the form $x_{k+1} = x_k + \alpha_k p_k$, where x_0 is the starting point of the iteration, p_k is a descent direction, and α_k satisfies the Wolfe conditions (5). *Suppose in addition that f is bounded from below in the level set $S = \{x \,|\, f(x) \leqslant f(x_0))\}$, and the gradient ∇f is Lipschitz continuous on an open set $\mathcal{N}$ containing S; that is, there exists a constant $L > 0$ such that*

$$\|(\nabla f)(x) - (\nabla f)(\tilde{x})\| \leqslant L\|x - \tilde{x}\| \quad \forall\, x, \tilde{x} \in \mathcal{N}.$$

Then it holds the inequality

$$\sum_{k=0}^{\infty} \cos^2\theta_k \|\nabla f_k\|^2 < \infty. \tag{9}$$

§3.2 Convergence of Line Search Methods

Theorem (Zoutendijk)

Let $f:\mathbb{R}^n\to\mathbb{R}$ be continuously differentiable, and $\{x_k\}$ be a sequence of iterates taking the form $x_{k+1}=x_k+\alpha_k p_k$, where x_0 is the starting point of the iteration, p_k is a descent direction, and α_k satisfies the Wolfe conditions (5). Suppose in addition that f is bounded from below in the level set $S=\{x \,\big|\, f(x)\leqslant f(x_0)\}$, and the gradient ∇f is Lipschitz continuous on an open set $\mathcal{N}$ containing S; that is, there exists a constant $L>0$ such that

$$\|(\nabla f)(x)-(\nabla f)(\tilde{x})\|\leqslant L\|x-\tilde{x}\| \quad \forall\, x,\tilde{x}\in\mathcal{N}.$$

Then it holds the inequality

$$\sum_{k=0}^{\infty}\cos^2\theta_k\|\nabla f_k\|^2<\infty. \tag{9}$$

§3.2 Convergence of Line Search Methods

Theorem (Zoutendijk)

Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable, and $\{x_k\}$ be a sequence of iterates taking the form $x_{k+1} = x_k + \alpha_k p_k$, where x_0 is the starting point of the iteration, p_k is a descent direction, and α_k satisfies the Wolfe conditions (5)*. Suppose in addition that f is bounded from below in the level set $S = \{x \,|\, f(x) \leqslant f(x_0)\}$, and the gradient ∇f is Lipschitz continuous on an open set $\mathcal{N}$ containing S; that is, there exists a constant $L > 0$ such that*

$$\|(\nabla f)(x) - (\nabla f)(\tilde{x})\| \leqslant L\|x - \tilde{x}\| \quad \forall\, x, \tilde{x} \in \mathcal{N}.$$

Then it holds the inequality

$$\sum_{k=0}^{\infty} \cos^2\theta_k \|\nabla f_k\|^2 < \infty. \tag{9}$$

§3.2 Convergence of Line Search Methods

Proof.

From the second Wolfe condition (5b),

$$(\nabla f_{k+1} - \nabla f_k)^{\mathrm{T}} p_k \geqslant (c_2 - 1)\nabla f_k^{\mathrm{T}} p_k \,,$$

and the Lipschitz condition and the Cauchy-Schwartz inequality further imply that

$$(\nabla f_{k+1} - \nabla f_k)^{\mathrm{T}} p_k \leqslant L\alpha_k \|p_k\|^2 \,.$$

The two inequalities above show that

$$(c_2 - 1)\nabla f_k^{\mathrm{T}} p_k \leqslant L\alpha_k \|p_k\|^2 \quad \text{or equivalently} \quad \alpha_k \geqslant \frac{c_2 - 1}{L} \frac{\nabla f_k^{\mathrm{T}} p_k}{\|p_k\|^2} \,.$$

By substituting this inequality into the first Wolfe condition (5a), we obtain that

$$f_{k+1} \leqslant f_k + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \leqslant f_k - c_1 \frac{1 - c_2}{L} \cos^2\theta_k \|\nabla f_k\|^2 \,.$$

□

§3.2 Convergence of Line Search Methods

Proof.

From the second Wolfe condition (5b),

$$(\nabla f_{k+1} - \nabla f_k)^{\mathrm{T}} p_k \geqslant (c_2 - 1)\nabla f_k^{\mathrm{T}} p_k ,$$

and the Lipschitz condition and the Cauchy-Schwartz inequality further imply that

$$(\nabla f_{k+1} - \nabla f_k)^{\mathrm{T}} p_k \leqslant L\alpha_k \|p_k\|^2 .$$

The two inequalities above show that

$$(c_2 - 1)\nabla f_k^{\mathrm{T}} p_k \leqslant L\alpha_k \|p_k\|^2 \quad \text{or equivalently} \quad \alpha_k \geqslant \frac{c_2 - 1}{L} \frac{\nabla f_k^{\mathrm{T}} p_k}{\|p_k\|^2} .$$

By substituting this inequality into the first Wolfe condition (5a), we obtain that

$$f_{k+1} \leqslant f_k + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \leqslant f_k - c_1 \frac{1 - c_2}{L} \cos^2\theta_k \|\nabla f_k\|^2 .$$

□

§3.2 Convergence of Line Search Methods

Proof.

From the second Wolfe condition (5b),

$$(\nabla f_{k+1} - \nabla f_k)^{\mathrm{T}} p_k \geqslant (c_2 - 1)\nabla f_k^{\mathrm{T}} p_k \,,$$

and the Lipschitz condition and the Cauchy-Schwartz inequality further imply that

$$(\nabla f_{k+1} - \nabla f_k)^{\mathrm{T}} p_k \leqslant L\alpha_k \|p_k\|^2 \,.$$

The two inequalities above show that

$$(c_2 - 1)\nabla f_k^{\mathrm{T}} p_k \leqslant L\alpha_k \|p_k\|^2 \quad \text{or equivalently} \quad \alpha_k \geqslant \frac{c_2 - 1}{L} \frac{\nabla f_k^{\mathrm{T}} p_k}{\|p_k\|^2} \,.$$

By substituting this inequality into the first Wolfe condition (5a), we obtain that

$$f_{k+1} \leqslant f_k + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \leqslant f_k - c_1 \frac{1 - c_2}{L} \cos^2\theta_k \|\nabla f_k\|^2 \,. \qquad \square$$

§3.2 Convergence of Line Search Methods

Proof (cont'd).

From previous page:

$$f_{k+1} \leqslant f_k + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \leqslant f_k - c_1 \frac{1-c_2}{L} \cos^2\theta_k \|\nabla f_k\|^2 \,.$$

Summing over all indices k less that ℓ, we find that

$$f_{\ell+1} \leqslant f_0 - c_1 \frac{1-c_2}{L} \sum_{k=0}^{\ell} \cos^2\theta_k \|\nabla f_k\|^2 \,.$$

Since f is bounded from below in S, from the inequality above it follows that for all $\ell \in \mathbb{N}$,

$$c_1 \frac{1-c_2}{L} \sum_{k=0}^{\ell} \cos^2\theta_k \|\nabla f_k\|^2 \leqslant f_0 - \inf_{x \in S} f(x) < \infty \,.$$

This concludes the theorem. □

§3.2 Convergence of Line Search Methods

Proof (cont'd).

From previous page:

$$f_{k+1} \leqslant f_k + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \leqslant f_k - c_1 \frac{1-c_2}{L} \cos^2\theta_k \|\nabla f_k\|^2 .$$

Summing over all indices k less that ℓ, we find that

$$f_{\ell+1} \leqslant f_0 - c_1 \frac{1-c_2}{L} \sum_{k=0}^{\ell} \cos^2\theta_k \|\nabla f_k\|^2 .$$

Since f is bounded from below in S, from the inequality above it follows that for all $\ell \in \mathbb{N}$,

$$c_1 \frac{1-c_2}{L} \sum_{k=0}^{\ell} \cos^2\theta_k \|\nabla f_k\|^2 \leqslant f_0 - \inf_{x \in S} f(x) < \infty .$$

This concludes the theorem. □

§3.2 Convergence of Line Search Methods

The Zoutendijk condition (9) implies that

$$\lim_{k\to\infty} \cos^2\theta_k \|\nabla f_k\|^2 = 0\,.$$

This limit can be used to derive **global convergence** results for line search algorithms. If our method for choosing the search direction p_k in the iteration scheme ensures that the angle θ_k defined by

$$\cos\theta_k = \frac{-\nabla f_k^{\mathrm{T}} p_k}{\|\nabla f_k\|\|p_k\|}$$

is bounded away from 90 degree so that $\cos\theta_k \geqslant \delta > 0$ for some positive constant δ, then it follows immediately that

$$\lim_{k\to\infty} \|\nabla f_k\| = 0\,. \tag{10}$$

In other words, we can be sure that the gradient norms $\|\nabla f_k\|$ converge to zero, provided that the search directions are never too close to orthogonality with the gradient.

§3.2 Convergence of Line Search Methods

The Zoutendijk condition (9) implies that

$$\lim_{k\to\infty} \cos^2\theta_k \|\nabla f_k\|^2 = 0\,.$$

This limit can be used to derive **global convergence** results for line search algorithms. If our method for choosing the search direction p_k in the iteration scheme ensures that the angle θ_k defined by

$$\cos\theta_k = \frac{-\nabla f_k^{\mathrm{T}} p_k}{\|\nabla f_k\|\|p_k\|}$$

is bounded away from 90 degree so that $\cos\theta_k \geqslant \delta > 0$ for some positive constant δ, then it follows immediately that

$$\lim_{k\to\infty} \|\nabla f_k\| = 0\,. \tag{10}$$

In other words, we can be sure that the gradient norms $\|\nabla f_k\|$ converge to zero, provided that the search directions are never too close to orthogonality with the gradient.

§3.2 Convergence of Line Search Methods

The Zoutendijk condition (9) implies that

$$\lim_{k\to\infty} \cos^2\theta_k \|\nabla f_k\|^2 = 0 .$$

This limit can be used to derive **global convergence** results for line search algorithms. If our method for choosing the search direction p_k in the iteration scheme ensures that the angle θ_k defined by

$$\cos\theta_k = \frac{-\nabla f_k^{\mathrm{T}} p_k}{\|\nabla f_k\|\|p_k\|}$$

is bounded away from 90 degree so that $\cos\theta_k \geqslant \delta > 0$ for some positive constant δ, then it follows immediately that

$$\lim_{k\to\infty} \|\nabla f_k\| = 0 . \tag{10}$$

In other words, we can be sure that the gradient norms $\|\nabla f_k\|$ converge to zero, provided that the search directions are never too close to orthogonality with the gradient.

§3.2 Convergence of Line Search Methods

We use the term **globally convergent** to refer to algorithms for which the property

$$\lim_{k\to\infty} \|\nabla f_k\| = 0 \tag{10}$$

is satisfied, but note that this term is sometimes used in other contexts to mean different things. For line search methods of the general form $x_{k+1} = x_k + \alpha_k p_k$, the limit (10) is the strongest global convergence result that can be obtained: We cannot guarantee that the method converges to a minimizer, but only that it is attracted by stationary points. Only by making additional requirements on the search direction p_k – by introducing negative curvature information from the Hessian $(\nabla^2 f)(x_k)$, for example – can we strengthen these results to include convergence to a local minimum.

§3.2 Convergence of Line Search Methods

We use the term **globally convergent** to refer to algorithms for which the property

$$\lim_{k\to\infty} \|\nabla f_k\| = 0 \tag{10}$$

is satisfied, but note that this term is sometimes used in other contexts to mean different things. For line search methods of the general form $x_{k+1} = x_k + \alpha_k p_k$, the limit (10) is the strongest global convergence result that can be obtained: We cannot guarantee that the method converges to a minimizer, but only that it is attracted by stationary points. Only by making additional requirements on the search direction p_k – by introducing negative curvature information from the Hessian $(\nabla^2 f)(x_k)$, for example – can we strengthen these results to include convergence to a local minimum.

§3.2 Convergence of Line Search Methods

We use the term **globally convergent** to refer to algorithms for which the property

$$\lim_{k\to\infty} \|\nabla f_k\| = 0 \tag{10}$$

is satisfied, but note that this term is sometimes used in other contexts to mean different things. For line search methods of the general form $x_{k+1} = x_k + \alpha_k p_k$, the limit (10) is the strongest global convergence result that can be obtained: We cannot guarantee that the method converges to a minimizer, but only that it is attracted by stationary points. Only by making additional requirements on the search direction p_k – by introducing negative curvature information from the Hessian $(\nabla^2 f)(x_k)$, for example – can we strengthen these results to include convergence to a local minimum.

§3.2 Convergence of Line Search Methods

Consider now the Newton-like method $x_{k+1} = x_k - \alpha_k B_k^{-1} \nabla f_k$ and assume that the matrices B_k are positive definite with a uniformly bounded condition number; that is, there is a constant M such that

$$\|B_k\| \|B_k^{-1}\| \leqslant M \qquad \forall\, k \in \mathbb{N}\,.$$

It is easy to show from the definition of θ_k that $\cos\theta_k \geqslant 1/M$; thus we find that $\lim\limits_{k \to \infty} \|\nabla f_k\| = 0$. Therefore, we have shown that Newton and quasi-Newton methods are globally convergent if the matrices B_k have a bounded condition number and are positive definite (which is needed to ensure that p_k is a descent direction), and if the step lengths satisfy the Wolfe conditions.

§3.2 Convergence of Line Search Methods

Consider now the Newton-like method $x_{k+1} = x_k - \alpha_k B_k^{-1} \nabla f_k$ and assume that the matrices B_k are positive definite with a uniformly bounded condition number; that is, there is a constant M such that

$$\|B_k\| \|B_k^{-1}\| \leqslant M \qquad \forall\, k \in \mathbb{N} .$$

It is easy to show from the definition of θ_k that $\cos\theta_k \geqslant 1/M$; thus we find that $\lim_{k\to\infty} \|\nabla f_k\| = 0$. Therefore, we have shown that Newton and quasi-Newton methods are globally convergent if the matrices B_k have a bounded condition number and are positive definite (which is needed to ensure that p_k is a descent direction), and if the step lengths satisfy the Wolfe conditions.

§3.2 Convergence of Line Search Methods

Consider now the Newton-like method $x_{k+1} = x_k - \alpha_k B_k^{-1} \nabla f_k$ and assume that the matrices B_k are positive definite with a uniformly bounded condition number; that is, there is a constant M such that

$$\|B_k\| \|B_k^{-1}\| \leqslant M \qquad \forall\, k \in \mathbb{N}\,.$$

It is easy to show from the definition of θ_k that $\cos\theta_k \geqslant 1/M$; thus we find that $\lim\limits_{k\to\infty} \|\nabla f_k\| = 0$. Therefore, we have shown that Newton and quasi-Newton methods are globally convergent if the matrices B_k have a bounded condition number and are positive definite (which is needed to ensure that p_k is a descent direction), and if the step lengths satisfy the Wolfe conditions.

§3.2 Convergence of Line Search Methods

For some algorithms, such as conjugate gradient methods, we will be able to prove only the **weaker** result

$$\liminf_{k\to\infty} \|\nabla f_k\| = 0\,; \tag{11}$$

that is, only a subsequence of the gradient norms $\|\nabla f_{k_j}\|$ converges to zero. This result usually can be proved **by contradiction** using Zoutendijk's condition $\sum_{k=0}^{\infty} \cos^2\theta_k \|\nabla f_k\|^2 < \infty$. Suppose that (11) does not hold. Then there exists $\gamma > 0$ such that

$$\|\nabla f_k\| \geqslant \gamma \qquad \forall\, k \gg 1\,.$$

This shows that $\lim_{k\to\infty} \cos\theta_k = 0$. To establish (11), it is then enough to show that a subsequence $\{\cos\theta_{k_j}\}_{k=1}^{\infty}$ is bounded away from zero. We will use this strategy in Chapter 5 to study the convergence of nonlinear conjugate gradient methods.

§3.2 Convergence of Line Search Methods

For some algorithms, such as conjugate gradient methods, we will be able to prove only the **weaker** result

$$\liminf_{k\to\infty} \|\nabla f_k\| = 0\,; \tag{11}$$

that is, only a subsequence of the gradient norms $\|\nabla f_{k_j}\|$ converges to zero. This result usually can be proved **by contradiction** using Zoutendijk's condition $\sum_{k=0}^{\infty} \cos^2\theta_k \|\nabla f_k\|^2 < \infty$. Suppose that (11) does not hold. Then there exists $\gamma > 0$ such that

$$\|\nabla f_k\| \geqslant \gamma \qquad \forall\, k \gg 1\,.$$

This shows that $\lim_{k\to\infty} \cos\theta_k = 0$. To establish (11), it is then enough to show that a subsequence $\{\cos\theta_{k_j}\}_{k=1}^{\infty}$ is bounded away from zero. We will use this strategy in Chapter 5 to study the convergence of nonlinear conjugate gradient methods.

§3.2 Convergence of Line Search Methods

For some algorithms, such as conjugate gradient methods, we will be able to prove only the **weaker** result

$$\liminf_{k\to\infty} \|\nabla f_k\| = 0\,; \tag{11}$$

that is, only a subsequence of the gradient norms $\|\nabla f_{k_j}\|$ converges to zero. This result usually can be proved **by contradiction** using Zoutendijk's condition $\sum_{k=0}^{\infty} \cos^2\theta_k \|\nabla f_k\|^2 < \infty$. Suppose that (11) does not hold. Then there exists $\gamma > 0$ such that

$$\|\nabla f_k\| \geqslant \gamma \qquad \forall\, k \gg 1\,.$$

This shows that $\lim_{k\to\infty} \cos\theta_k = 0$. To establish (11), it is then enough to show that a subsequence $\{\cos\theta_{k_j}\}_{k=1}^{\infty}$ is bounded away from zero. We will use this strategy in Chapter 5 to study the convergence of nonlinear conjugate gradient methods.

§3.2 Convergence of Line Search Methods

By applying this proof technique, we can prove global convergence in the sense of

$$\lim_{k\to\infty} \|\nabla f_k\| = 0 \qquad (10)$$

or

$$\liminf_{k\to\infty} \|\nabla f_k\| = 0 \qquad (11)$$

for a general class of algorithms. Consider any algorithm for which

1. every iteration produces a decrease in the objective function;
2. every m-th iteration is a steepest descent step, with step length chosen to satisfy the Wolfe or Goldstein conditions.

Then, since $\cos\theta_k = 1$ for the steepest descent steps, the result (11) holds. Of course, we would design the algorithm so that it does something "better" than steepest descent at the other $m-1$ iterates. The occasional steepest descent steps may not make much progress, but they at least guarantee overall global convergence.

§3.2 Convergence of Line Search Methods

By applying this proof technique, we can prove global convergence in the sense of

$$\lim_{k\to\infty} \|\nabla f_k\| = 0 \tag{10}$$

or

$$\liminf_{k\to\infty} \|\nabla f_k\| = 0 \tag{11}$$

for a general class of algorithms. Consider any algorithm for which

1. every iteration produces a decrease in the objective function;
2. every m-th iteration is a steepest descent step, with step length chosen to satisfy the Wolfe or Goldstein conditions.

Then, since $\cos\theta_k = 1$ for the steepest descent steps, the result (11) holds. Of course, we would design the algorithm so that it does something "better" than steepest descent at the other $m-1$ iterates. The occasional steepest descent steps may not make much progress, but they at least guarantee overall global convergence.

§3.2 Convergence of Line Search Methods

By applying this proof technique, we can prove global convergence in the sense of

$$\lim_{k\to\infty} \|\nabla f_k\| = 0 \tag{10}$$

or

$$\liminf_{k\to\infty} \|\nabla f_k\| = 0 \tag{11}$$

for a general class of algorithms. Consider any algorithm for which

1. every iteration produces a decrease in the objective function;
2. every m-th iteration is a steepest descent step, with step length chosen to satisfy the Wolfe or Goldstein conditions.

Then, since $\cos\theta_k = 1$ for the steepest descent steps, the result (11) holds. Of course, we would design the algorithm so that it does something "better" than steepest descent at the other $m-1$ iterates. The occasional steepest descent steps may not make much progress, but they at least guarantee overall global convergence.

§3.2 Convergence of Line Search Methods

By applying this proof technique, we can prove global convergence in the sense of

$$\lim_{k\to\infty} \|\nabla f_k\| = 0 \tag{10}$$

or

$$\liminf_{k\to\infty} \|\nabla f_k\| = 0 \tag{11}$$

for a general class of algorithms. Consider any algorithm for which

1. every iteration produces a decrease in the objective function;
2. every m-th iteration is a steepest descent step, with step length chosen to satisfy the Wolfe or Goldstein conditions.

Then, since $\cos\theta_k = 1$ for the steepest descent steps, the result (11) holds. Of course, we would design the algorithm so that it does something "better" than steepest descent at the other $m-1$ iterates. The occasional steepest descent steps may not make much progress, but they at least guarantee overall global convergence.

§3.2 Convergence of Line Search Methods

Note that throughout this section we have used only the fact that Zoutendijk's condition

$$\sum_{k=0}^{\infty} \cos^2\theta_k \|\nabla f_k\|^2 < \infty \tag{9}$$

implies the limit

$$\lim_{k\to\infty} \cos^2\theta_k \|\nabla f_k\|^2 = 0\,.$$

In later chapters we will make use of the bounded sum condition (9), which forces the sequence $\left\{\cos^2\theta_k \|\nabla f_k\|^2\right\}_{k=1}^{\infty}$ to converge to zero at a **sufficiently rapid rate**.

§3.3 Rate of Convergence

It would seem that designing optimization algorithms with good convergence properties is easy, since all we need to ensure is that the search direction p_k does not tend to become orthogonal to the gradient ∇f_k, or that steepest descent steps are taken regularly. We could simply compute $\cos\theta_k$ at every iteration and turn p_k toward the steepest descent direction if $\cos\theta_k$ is smaller than some preselected constant $\delta > 0$. Angle tests of this type ensure global convergence, but they are undesirable for two reasons. First, they may impede a fast rate of convergence, because for problems with an ill-conditioned Hessian, it may be necessary to produce search directions that are almost orthogonal to the gradient, and an inappropriate choice of the parameter δ may cause such steps to be rejected. Second, angle tests destroy the invariance properties of quasi-Newton methods.

§3.3 Rate of Convergence

It would seem that designing optimization algorithms with good convergence properties is easy, since all we need to ensure is that the search direction p_k does not tend to become orthogonal to the gradient ∇f_k, or that steepest descent steps are taken regularly. We could simply compute $\cos\theta_k$ at every iteration and turn p_k toward the steepest descent direction if $\cos\theta_k$ is smaller than some preselected constant $\delta > 0$. Angle tests of this type ensure global convergence, but they are undesirable for two reasons. First, they may impede a fast rate of convergence, because for problems with an ill-conditioned Hessian, it may be necessary to produce search directions that are almost orthogonal to the gradient, and an inappropriate choice of the parameter δ may cause such steps to be rejected. Second, angle tests destroy the invariance properties of quasi-Newton methods.

§3.3 Rate of Convergence

Algorithmic strategies that achieve rapid convergence can sometimes conflict with the requirements of global convergence, and vice versa. For example, the steepest descent method is the quintessential globally convergent algorithm, but it is quite slow in practice, as we shall see below. On the other hand, the pure Newton iteration converges rapidly when started close enough to a solution, but its steps may not even be descent directions away from the solution. The challenge is to design algorithms that incorporate both properties: good global convergence guarantees and a rapid rate of convergence.

§3.3 Rate of Convergence

Definition

Let $\{x_k\}_{k=1}^{\infty}$ be a sequence in $\mathbb{R}^n$ and x_* be the limit of the sequence.

1. $\{x_k\}_{k=1}^{\infty}$ is said to converge to x_* **superlinearly** if
$$\lim_{k\to\infty} \frac{\|x_{k+1} - x_*\|}{\|x_k - x_*\|} = 0\,.$$
2. $\{x_k\}_{k=1}^{\infty}$ is said to converge to x_* **quadratically** if there exists a constant $M > 0$ such that
$$\frac{\|x_{k+1} - x_*\|}{\|x_k - x_*\|^2} \leqslant M \qquad \forall\, k \gg 1\,.$$

Example

1. The sequence $x_k = 1 + k^{-k}$ converges superlinearly to 1.
2. The sequence $x_k = 1 + k^{-2^k}$ converges quadratically to 1.

§3.3 Rate of Convergence

Definition

Let $\{x_k\}_{k=1}^{\infty}$ be a sequence in $\mathbb{R}^n$ and x_* be the limit of the sequence.

1. $\{x_k\}_{k=1}^{\infty}$ is said to converge to x_* **superlinearly** if
$$\lim_{k\to\infty} \frac{\|x_{k+1} - x_*\|}{\|x_k - x_*\|} = 0\,.$$
2. $\{x_k\}_{k=1}^{\infty}$ is said to converge to x_* **quadratically** if there exists a constant $M > 0$ such that
$$\frac{\|x_{k+1} - x_*\|}{\|x_k - x_*\|^2} \leqslant M \qquad \forall\, k \gg 1\,.$$

Example

1. The sequence $x_k = 1 + k^{-k}$ converges superlinearly to 1.
2. The sequence $x_k = 1 + k^{-2^k}$ converges quadratically to 1.

§3.3 Rate of Convergence

- **Convergence Rate of Steepest Descent**:

We begin our study of convergence rates of line search methods by considering the most basic approach of all: the steepest descent method.

We can learn much about the steepest descent method by considering the ideal case, in which the objective function is quadratic and the line searches are exact. Let us suppose that

$$f(x) = \frac{1}{2}x^{\mathrm{T}}Qx - b^{\mathrm{T}}x,$$

where Q is symmetric and positive definite. The gradient is given by $(\nabla f)(x) = Qx - b$ and the minimizer x_* is the unique solution of the linear system $Qx = b$.

§3.3 Rate of Convergence

It is easy to compute the step length α_k that minimizes $f(x_k - \alpha\nabla f_k)$. By differentiating the function

$$f(x_k - \alpha\nabla f_k) = \frac{1}{2}(x_k - \alpha\nabla f_k)^{\mathrm{T}} Q(x_k - \alpha\nabla f_k) - b^{\mathrm{T}}(x_k - \alpha\nabla f_k)$$

with respect to α, and setting the derivative to zero, we obtain that

$$\alpha_k = \frac{\nabla f_k^{\mathrm{T}}\nabla f_k}{\nabla f_k^{\mathrm{T}} Q\nabla f_k}\,.$$

If we use this exact minimizer α_k, the steepest descent iteration for f given above is given by

$$x_{k+1} = x_k - \left(\frac{\nabla f_k^{\mathrm{T}}\nabla f_k}{\nabla f_k^{\mathrm{T}} Q\nabla f_k}\right)\nabla f_k\,. \tag{12}$$

Since $\nabla f_k = Qx_k - b$, this equation yields a closed-form expression for x_{k+1} in terms of x_k.

§3.3 Rate of Convergence

It is easy to compute the step length α_k that minimizes $f(x_k - \alpha\nabla f_k)$. By differentiating the function

$$f(x_k - \alpha\nabla f_k) = \frac{1}{2}(x_k - \alpha\nabla f_k)^{\mathrm{T}} Q (x_k - \alpha\nabla f_k) - b^{\mathrm{T}}(x_k - \alpha\nabla f_k)$$

with respect to α, and setting the derivative to zero, we obtain that

$$\alpha_k = \frac{\nabla f_k^{\mathrm{T}} \nabla f_k}{\nabla f_k^{\mathrm{T}} Q \nabla f_k} .$$

If we use this exact minimizer α_k , the steepest descent iteration for f given above is given by

$$x_{k+1} = x_k - \left(\frac{\nabla f_k^{\mathrm{T}} \nabla f_k}{\nabla f_k^{\mathrm{T}} Q \nabla f_k}\right) \nabla f_k . \tag{12}$$

Since $\nabla f_k = Q x_k - b$, this equation yields a closed-form expression for x_{k+1} in terms of x_k.

§3.3 Rate of Convergence

In Figure 6 we plot a typical sequence of iterates generated by the steepest descent method on a two-dimensional quadratic objective function. The contours of f are ellipsoids whose axes lie along the orthogonal eigenvectors of Q. Note that the iterates zigzag toward the solution.

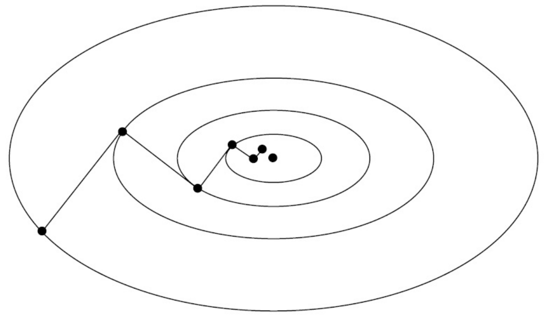

Figure 6: Steepest descent steps

§3.3 Rate of Convergence

To quantify the rate of convergence we introduce the weighted norm $\|x\|_Q^2 \equiv x^{\mathrm{T}}Qx$. Using the relation $Qx_* = b$,

$$\begin{aligned}
\frac{1}{2}\|x - x_*\|_Q^2 &= \frac{1}{2}(x - x_*)^{\mathrm{T}}Q(x - x_*) \\
&= \frac{1}{2}x^{\mathrm{T}}Qx - \frac{1}{2}x_*^{\mathrm{T}}Qx - \frac{1}{2}x^{\mathrm{T}}Qx_* + \frac{1}{2}x_*^{\mathrm{T}}Qx_* \\
&= \frac{1}{2}x^{\mathrm{T}}Qx - \frac{1}{2}b^{\mathrm{T}}x - \frac{1}{2}x^{\mathrm{T}}b - \left(\frac{1}{2}x_*^{\mathrm{T}}Qx_* - x_*^{\mathrm{T}}Qx_*\right) \\
&= f(x) - f(x_*)
\end{aligned}$$

so this norm measures the difference between the current objective value and the optimal value. Using the iteration scheme (12) and noting that $\nabla f_k = Q(x_k - x_*)$, we now derive the equality

$$\|x_{k+1} - x_*\|_Q^2 = \left[1 - \frac{(\nabla f_k^{\mathrm{T}}\nabla f_k)^2}{(\nabla f_k^{\mathrm{T}}Q\nabla f_k)(\nabla f_k^{\mathrm{T}}Q^{-1}\nabla f_k)}\right]\|x_k - x_*\|_Q^2 .$$

§3.3 Rate of Convergence

To quantify the rate of convergence we introduce the weighted norm $\|x\|_Q^2 \equiv x^{\mathrm{T}}Qx$. Using the relation $Qx_* = b$,

$$\begin{aligned}\frac{1}{2}\|x - x_*\|_Q^2 &= \frac{1}{2}(x - x_*)^{\mathrm{T}}Q(x - x_*) \\ &= \frac{1}{2}x^{\mathrm{T}}Qx - \frac{1}{2}x_*^{\mathrm{T}}Qx - \frac{1}{2}x^{\mathrm{T}}Qx_* + \frac{1}{2}x_*^{\mathrm{T}}Qx_* \\ &= \frac{1}{2}x^{\mathrm{T}}Qx - \frac{1}{2}b^{\mathrm{T}}x - \frac{1}{2}x^{\mathrm{T}}b - \Big(\frac{1}{2}x_*^{\mathrm{T}}Qx_* - x_*^{\mathrm{T}}Qx_*\Big) \\ &= f(x) - f(x_*)\end{aligned}$$

so this norm measures the difference between the current objective value and the optimal value. Using the iteration scheme (12) and noting that $\nabla f_k = Q(x_k - x_*)$, we now derive the equality

$$\|x_{k+1} - x_*\|_Q^2 = \left[1 - \frac{(\nabla f_k^{\mathrm{T}}\nabla f_k)^2}{(\nabla f_k^{\mathrm{T}}Q\nabla f_k)(\nabla f_k^{\mathrm{T}}Q^{-1}\nabla f_k)}\right]\|x_k - x_*\|_Q^2 .$$

§3.3 Rate of Convergence

By the substitution of variable $y = x - Q^{-1}b$, we find that

$$f(x) = \frac{1}{2}x^{\mathrm{T}}Qx - b^{\mathrm{T}}x = \frac{1}{2}(x - Q^{-1}b)^{\mathrm{T}}Q(x - Q^{-1}b) - \frac{1}{2}b^{\mathrm{T}}Q^{-1}b$$
$$= \frac{1}{2}y^{\mathrm{T}}Qy - \frac{1}{2}b^{\mathrm{T}}Q^{-1}b \equiv g(y)\,.$$

Setting $y_k = x_k - Q^{-1}b$ for all $k \in \mathbb{N}$ and $\nabla g_k = (\nabla g)(y_k)$. Since

$$(\nabla f)(x) = Qx - b = Q(x - Q^{-1}b) = Qy = (\nabla g)(y)\,,$$

we have $p_k = -\nabla g_k$ and the step length α_k for the steepest descent method satisfies

$$\alpha_k = \frac{\nabla f_k^{\mathrm{T}}\nabla f_k}{\nabla f_k^{\mathrm{T}}Q\nabla f_k} = \frac{\nabla g_k^{\mathrm{T}}\nabla g_k}{\nabla g_k^{\mathrm{T}}Q\nabla g_k}\,.$$

Therefore, $x_{k+1} = x_k - \alpha_k\nabla f_k$ if and only if $y_{k+1} = y_k - \alpha_k\nabla g_k$ which shows that the steepest descent method with the exact line search for both f and g are identical.

§3.3 Rate of Convergence

Since $x_* = Q^{-1}b$, $y = x - x_*$. Moreover, since $p_k = -Qy_k$, we also have

$$p_k^{\mathrm{T}} Q y_k = -p_k^{\mathrm{T}} p_k = -\alpha_k p_k^{\mathrm{T}} Q p_k \quad \text{and} \quad p_k^{\mathrm{T}} Q^{-1} p_k = y_k^{\mathrm{T}} Q y_k = \|y_k\|_Q^2 .$$

Therefore,

$$\begin{aligned}
\|x_{k+1} - x_*\|_Q^2 = y_{k+1}^{\mathrm{T}} Q y_{k+1} &= (y_k + \alpha_k p_k)^{\mathrm{T}} Q (y_k + \alpha_k p_k) \\
&= y_k^{\mathrm{T}} Q y_k + 2\alpha_k p_k^{\mathrm{T}} Q y_k + \alpha_k^2 p_k^{\mathrm{T}} Q p_k \\
&= \|y_k\|_Q^2 + \alpha_k p_k^{\mathrm{T}} Q y_k \\
&= \|y_k\|_Q^2 + \alpha_k \frac{p_k^{\mathrm{T}} Q y_k}{p_k^{\mathrm{T}} Q^{-1} p_k} \|y_k\|_Q^2 \\
&= \left[1 + \alpha_k \frac{p_k^{\mathrm{T}} Q y_k}{p_k^{\mathrm{T}} Q^{-1} p_k}\right] \|y_k\|_Q^2 \\
&= \left[1 - \frac{(p_k^{\mathrm{T}} p_k)^2}{(p_k^{\mathrm{T}} Q p_k)(p_k^{\mathrm{T}} Q^{-1} p_k)}\right] \|x_k - x_*\|_Q^2 .
\end{aligned}$$

§3.3 Rate of Convergence

The expression

$$\|x_{k+1} - x_*\|_Q^2 = \left[1 - \frac{(\nabla f_k^{\mathrm{T}} \nabla f_k)^2}{(\nabla f_k^{\mathrm{T}} Q \nabla f_k)(\nabla f_k^{\mathrm{T}} Q^{-1} \nabla f_k)}\right] \|x_k - x_*\|_Q^2 .$$

describes the exact decrease in f at each iteration, but since the term inside the brackets is difficult to interpret, it is more useful to bound it in terms of the condition number of the problem.

Theorem

When the steepest descent method with exact line searches is applied to the strongly convex quadratic function $f(x) = \frac{1}{2}x^{\mathrm{T}}Qx - b^{\mathrm{T}}x$, the error norm $\|x_k - x_*\|_Q^2$ satisfies

$$\|x_{k+1} - x_*\|_Q^2 \leqslant \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^2 \|x_k - x_*\|_Q^2 \qquad \forall\, k \in \mathbb{N}, \tag{13}$$

where $0 < \lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$ are the eigenvalues of Q.

§3.3 Rate of Convergence

The expression

$$\|x_{k+1} - x_*\|_Q^2 = \left[1 - \frac{(\nabla f_k^{\mathrm{T}} \nabla f_k)^2}{(\nabla f_k^{\mathrm{T}} Q \nabla f_k)(\nabla f_k^{\mathrm{T}} Q^{-1} \nabla f_k)}\right] \|x_k - x_*\|_Q^2 .$$

describes the exact decrease in f at each iteration, but since the term inside the brackets is difficult to interpret, it is more useful to bound it in terms of the condition number of the problem.

Theorem

When the steepest descent method with exact line searches is applied to the strongly convex quadratic function $f(x) = \frac{1}{2}x^{\mathrm{T}}Qx - b^{\mathrm{T}}x$, *the error norm* $\|x_k - x_*\|_Q^2$ *satisfies*

$$\|x_{k+1} - x_*\|_Q^2 \leqslant \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^2 \|x_k - x_*\|_Q^2 \qquad \forall\, k \in \mathbb{N}, \tag{13}$$

where $0 < \lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$ *are the eigenvalues of* Q.

§3.3 Rate of Convergence

Sketch of the proof.

Since Q is symmetric, $Q = P\Lambda P^{\mathrm{T}}$ for some diagonal matrix $\Lambda = \mathrm{diag}(\lambda_1, \cdots, \lambda_n)$ and orthogonal matrix P. Let $u_k = P^{\mathrm{T}}\nabla f_k$. Write $u_k = (z_1, z_2, \cdots, z_n)$. By the fact that $u_k^{\mathrm{T}} u_k = \nabla f_k^{\mathrm{T}} \nabla f_k$,

$$\frac{(\nabla f_k^{\mathrm{T}} \nabla f_k)^2}{(\nabla f_k^{\mathrm{T}} Q \nabla f_k)(\nabla f_k^{\mathrm{T}} Q^{-1} \nabla f_k)} = \frac{(\sum_{j=1}^n z_j^2)^2}{(u_k^{\mathrm{T}} \Lambda u_k)(u_k^{\mathrm{T}} \Lambda^{-1} u_k)}$$

$$= \frac{(\sum_{j=1}^n z_j^2)^2}{(\sum_{j=1}^n \lambda_j z_j^2)(\sum_{j=1}^n \lambda_j^{-1} z_j^2)} = \frac{1/\sum_{j=1}^n \lambda_j \xi_j}{\sum_{j=1}^n \lambda_j^{-1} \xi_j} \equiv \frac{\phi(\xi)}{\psi(\xi)},$$

where $\xi_j = z_j^2 / \sum_{j=1}^n z_j^2$ (satisfies $\sum_{j=1}^n \xi_j = 1$ and $\xi_j \geqslant 0$ for all j).

A lower bound for the ratio is $\dfrac{4\lambda_1\lambda_n}{(\lambda_1 + \lambda_n)^2}$ (see Figure 7 on the next page). □

§3.3 Rate of Convergence

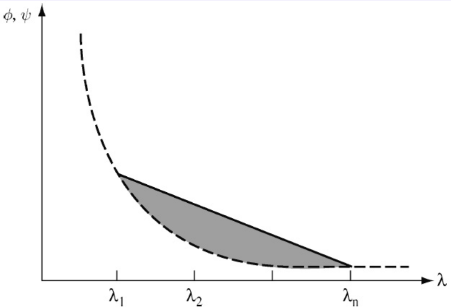

Figure 7: Kantorovich inequality: The dashed curve represents the function $1/\lambda$, and the value of $\phi(\xi)$ is a point on this curve. On the other hand, the value of $\psi(\xi)$ is a convex combination of points on the curve and its value corresponds to a point in the shaded region. For the same vector ξ both functions are represented by points on the same vertical line. The minimum value of this ratio is achieved for some $\lambda = \xi_1\lambda_1 + \xi_n\lambda_n$ with $\xi_1 + \xi_n = 1$.

§3.3 Rate of Convergence

Sketch of the proof.

Since Q is symmetric, $Q = P\Lambda P^{\mathrm{T}}$ for some diagonal matrix $\Lambda = \mathrm{diag}(\lambda_1, \cdots, \lambda_n)$ and orthogonal matrix P. Let $u_k = P^{\mathrm{T}}\nabla f_k$. Write $u_k = (z_1, z_2, \cdots, z_n)$. By the fact that $u_k^{\mathrm{T}} u_k = \nabla f_k^{\mathrm{T}} \nabla f_k$,

$$\frac{(\nabla f_k^{\mathrm{T}}\nabla f_k)^2}{(\nabla f_k^{\mathrm{T}} Q \nabla f_k)(\nabla f_k^{\mathrm{T}} Q^{-1}\nabla f_k)} = \frac{(\sum_{j=1}^n z_j^2)^2}{(u_k^{\mathrm{T}}\Lambda u_k)(u_k^{\mathrm{T}}\Lambda^{-1} u_k)}$$

$$= \frac{(\sum_{j=1}^n z_j^2)^2}{(\sum_{j=1}^n \lambda_j z_j^2)(\sum_{j=1}^n \lambda_j^{-1} z_j^2)} = \frac{1/\sum_{j=1}^n \lambda_j \xi_j}{\sum_{j=1}^n \lambda_j^{-1}\xi_j} \equiv \frac{\phi(\xi)}{\psi(\xi)},$$

where $\xi_j = z_j^2 / \sum_{j=1}^n z_j^2$ (satisfies $\sum_{j=1}^n \xi_j = 1$ and $\xi_j \geqslant 0$ for all j).

A lower bound for the ratio is $\dfrac{4\lambda_1\lambda_n}{(\lambda_1+\lambda_n)^2}$ (see Figure 7 on the next page). □

§3.3 Rate of Convergence

Sketch of the proof.

Since Q is symmetric, $Q = P\Lambda P^{\mathrm{T}}$ for some diagonal matrix $\Lambda = \mathrm{diag}(\lambda_1, \cdots, \lambda_n)$ and orthogonal matrix P. Let $u_k = P^{\mathrm{T}}\nabla f_k$. Write $u_k = (z_1, z_2, \cdots, z_n)$. By the fact that $u_k^{\mathrm{T}} u_k = \nabla f_k^{\mathrm{T}} \nabla f_k$,

$$\frac{(\nabla f_k^{\mathrm{T}} \nabla f_k)^2}{(\nabla f_k^{\mathrm{T}} Q \nabla f_k)(\nabla f_k^{\mathrm{T}} Q^{-1} \nabla f_k)} = \frac{(\sum_{j=1}^n z_j^2)^2}{(u_k^{\mathrm{T}} \Lambda u_k)(u_k^{\mathrm{T}} \Lambda^{-1} u_k)}$$

$$= \frac{(\sum_{j=1}^n z_j^2)^2}{(\sum_{j=1}^n \lambda_j z_j^2)(\sum_{j=1}^n \lambda_j^{-1} z_j^2)} = \frac{1/\lambda}{(\lambda_1 + \lambda_n - \lambda)/(\lambda_1 \lambda_n)} \equiv \frac{\phi(\xi)}{\psi(\xi)},$$

where $\xi_j = z_j^2 / \sum_{j=1}^n z_j^2$ (satisfies $\sum_{j=1}^n \xi_j = 1$ and $\xi_j \geqslant 0$ for all j).

A lower bound for the ratio is $\dfrac{4\lambda_1\lambda_n}{(\lambda_1 + \lambda_n)^2}$ (see Figure 7 on the next page). □

§3.3 Rate of Convergence

Sketch of the proof.

Since Q is symmetric, $Q = P\Lambda P^{\mathrm{T}}$ for some diagonal matrix $\Lambda = \operatorname{diag}(\lambda_1, \cdots, \lambda_n)$ and orthogonal matrix P. Let $u_k = P^{\mathrm{T}}\nabla f_k$. Write $u_k = (z_1, z_2, \cdots, z_n)$. By the fact that $u_k^{\mathrm{T}} u_k = \nabla f_k^{\mathrm{T}}\nabla f_k$,

$$\frac{(\nabla f_k^{\mathrm{T}}\nabla f_k)^2}{(\nabla f_k^{\mathrm{T}}Q\nabla f_k)(\nabla f_k^{\mathrm{T}}Q^{-1}\nabla f_k)} = \frac{(\sum_{j=1}^n z_j^2)^2}{(u_k^{\mathrm{T}}\Lambda u_k)(u_k^{\mathrm{T}}\Lambda^{-1}u_k)}$$

$$= \frac{(\sum_{j=1}^n z_j^2)^2}{(\sum_{j=1}^n \lambda_j z_j^2)(\sum_{j=1}^n \lambda_j^{-1} z_j^2)} = \frac{1/\lambda}{(\lambda_1 + \lambda_n - \lambda)/(\lambda_1\lambda_n)} \equiv \frac{\phi(\xi)}{\psi(\xi)},$$

where $\xi_j = z_j^2 / \sum_{j=1}^n z_j^2$ (satisfies $\sum_{j=1}^n \xi_j = 1$ and $\xi_j \geqslant 0$ for all j).

A lower bound for the ratio is $\dfrac{4\lambda_1\lambda_n}{(\lambda_1 + \lambda_n)^2}$ (see Figure 7 on the next page). □

§3.3 Rate of Convergence

The inequalities

$$\|x_{k+1} - x_*\|_Q^2 \leqslant \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^2 \|x_k - x_*\|_Q^2 \qquad \forall\, k \in \mathbb{N} \qquad (13)$$

and

$$\frac{1}{2}\|x - x_*\|_Q^2 = f(x) - f(x_*)$$

show that the function values f_k converge to the minimum f_* at a **linear rate**. As a special case of this result, we see that convergence is achieved in one iteration if all the eigenvalues are equal. In this case, the contours in Figure 6 are circles and the steepest descent direction always points at the solution. In general, as the condition number $\kappa(Q) = \lambda_n/\lambda_1$ increases, the contours of the quadratic become more elongated, the zigzagging in Figure 6 becomes more pronounced, and (13) implies that the convergence degrades. Even though (13) is a worst-case bound, it gives an accurate indication of the behavior of the algorithm when $n > 2$.

§3.3 Rate of Convergence

The inequalities

$$\|x_{k+1} - x_*\|_Q^2 \leqslant \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^2 \|x_k - x_*\|_Q^2 \qquad \forall\, k \in \mathbb{N} \qquad (13)$$

and

$$\frac{1}{2}\|x - x_*\|_Q^2 = f(x) - f(x_*)$$

show that the function values f_k converge to the minimum f_* at a **linear rate**. As a special case of this result, we see that convergence is achieved in one iteration if all the eigenvalues are equal. In this case, the contours in Figure 6 are circles and the steepest descent direction always points at the solution. In general, as the condition number $\kappa(Q) = \lambda_n/\lambda_1$ increases, the contours of the quadratic become more elongated, the zigzagging in Figure 6 becomes more pronounced, and (13) implies that the convergence degrades. Even though (13) is a worst-case bound, it gives an accurate indication of the behavior of the algorithm when $n > 2$.

§3.3 Rate of Convergence

The inequalities

$$\|x_{k+1} - x_*\|_Q^2 \leqslant \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^2 \|x_k - x_*\|_Q^2 \qquad \forall\, k \in \mathbb{N} \tag{13}$$

and

$$\frac{1}{2}\|x - x_*\|_Q^2 = f(x) - f(x_*)$$

show that the function values f_k converge to the minimum f_* at a **linear rate**. As a special case of this result, we see that convergence is achieved in one iteration if all the eigenvalues are equal. In this case, the contours in Figure 6 are circles and the steepest descent direction always points at the solution. In general, as the condition number $\kappa(Q) = \lambda_n/\lambda_1$ increases, the contours of the quadratic become more elongated, the zigzagging in Figure 6 becomes more pronounced, and (13) implies that the convergence degrades. Even though (13) is a worst-case bound, it gives an accurate indication of the behavior of the algorithm when $n > 2$.

§3.3 Rate of Convergence

The inequalities

$$\|x_{k+1} - x_*\|_Q^2 \leqslant \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^2 \|x_k - x_*\|_Q^2 \qquad \forall\, k \in \mathbb{N} \tag{13}$$

and

$$\frac{1}{2}\|x - x_*\|_Q^2 = f(x) - f(x_*)$$

show that the function values f_k converge to the minimum f_* at a **linear rate**. As a special case of this result, we see that convergence is achieved in one iteration if all the eigenvalues are equal. In this case, the contours in Figure 6 are circles and the steepest descent direction always points at the solution. In general, as the condition number $\kappa(Q) = \lambda_n/\lambda_1$ increases, the contours of the quadratic become more elongated, the zigzagging in Figure 6 becomes more pronounced, and (13) implies that the convergence degrades. Even though (13) is a worst-case bound, it gives an accurate indication of the behavior of the algorithm when $n > 2$.

§3.3 Rate of Convergence

The rate-of-convergence behavior of the steepest descent method is essentially the same on general nonlinear objective functions. In the following result we assume that the step length is the global minimizer along the search direction.

Theorem

Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable, and that the iterates generated by the steepest-descent method with exact line searches converge to a point x_* at which the Hessian matrix $(\nabla^2 f)(x_*)$ is positive definite. Let r be any scalar satisfying

$$r \in \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}, 1 \right)$$

where $\lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$ are the eigenvalues of $(\nabla^2 f)(x_*)$. Then

$$f(x_{k+1}) - f(x_*) \leqslant r^2 \big[f(x_k) - f(x_*) \big] \qquad \forall\, k \gg 1\,.$$

§3.3 Rate of Convergence

The rate-of-convergence behavior of the steepest descent method is essentially the same on general nonlinear objective functions. In the following result we assume that the step length is the global minimizer along the search direction.

Theorem

Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable, and that the iterates generated by the steepest-descent method with exact line searches converge to a point x_* at which the Hessian matrix $(\nabla^2 f)(x_*)$ is positive definite. Let r be any scalar satisfying

$$r \in \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}, 1\right)$$

where $\lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$ are the eigenvalues of $(\nabla^2 f)(x_*)$. Then

$$f(x_{k+1}) - f(x_*) \leqslant r^2 \big[f(x_k) - f(x_*)\big] \qquad \forall\, k \gg 1\,.$$

§3.3 Rate of Convergence

In general, we cannot expect the rate of convergence to improve if an inexact line search is used. Therefore, the theorem in the previous page shows that the steepest descent method can have an unacceptably slow rate of convergence, even when the Hessian is reasonably well conditioned. For example, if $\kappa(Q) = 800$, $f(x_1) = 1$, and $f(x_*) = 0$, the theorem in the previous page suggests that the function value will still be about 0.08 after one thousand (?500?) iterations of the steepest descent method with exact line search.

§3.3 Rate of Convergence

In general, we cannot expect the rate of convergence to improve if an inexact line search is used. Therefore, the theorem in the previous page shows that the steepest descent method can have an unacceptably slow rate of convergence, even when the Hessian is reasonably well conditioned. For example, if $\kappa(Q) = 800$, $f(x_1) = 1$, and $f(x_*) = 0$, the theorem in the previous page suggests that the function value will still be about 0.08 after one thousand (?500?) iterations of the steepest descent method with exact line search.

§3.3 Rate of Convergence

- **Convergence Rate of Newton's Method**:

We now consider Newton's method, for which the search is given by

$$p_k^{\mathrm{N}} = -(\nabla^2 f_k)^{-1} \nabla f_k \,.$$

Since the Hessian matrix $\nabla^2 f_k$ may not always be positive definite, p_k^{N} may not always be a descent direction, and many of the ideas discussed so far in this chapter no longer apply. In Section 3.4 and Chapter 4 we will describe two approaches for obtaining a globally convergent iteration based on the Newton step: a **line search approach**, in which the Hessian $\nabla^2 f_k$ is modified, if necessary, to make it positive definite and thereby yield descent, and a **trust region approach**, in which $\nabla^2 f_k$ is used to form a quadratic model that is minimized in a ball around the current iterate x_k.

§3.3 Rate of Convergence

- **Convergence Rate of Newton's Method**:

We now consider Newton's method, for which the search is given by

$$p_k^{\mathrm{N}} = -(\nabla^2 f_k)^{-1} \nabla f_k .$$

Since the Hessian matrix $\nabla^2 f_k$ may not always be positive definite, p_k^{N} may not always be a descent direction, and many of the ideas discussed so far in this chapter no longer apply. In Section 3.4 and Chapter 4 we will describe two approaches for obtaining a globally convergent iteration based on the Newton step: a **line search approach**, in which the Hessian $\nabla^2 f_k$ is modified, if necessary, to make it positive definite and thereby yield descent, and a **trust region approach**, in which $\nabla^2 f_k$ is used to form a quadratic model that is minimized in a ball around the current iterate x_k.

§3.3 Rate of Convergence

- **Convergence Rate of Newton's Method**:

We now consider Newton's method, for which the search is given by

$$p_k^{\mathrm{N}} = -(\nabla^2 f_k)^{-1} \nabla f_k \,.$$

Since the Hessian matrix $\nabla^2 f_k$ may not always be positive definite, p_k^{N} may not always be a descent direction, and many of the ideas discussed so far in this chapter no longer apply. In Section 3.4 and Chapter 4 we will describe two approaches for obtaining a globally convergent iteration based on the Newton step: a **line search approach**, in which the Hessian $\nabla^2 f_k$ is modified, if necessary, to make it positive definite and thereby yield descent, and a **trust region approach**, in which $\nabla^2 f_k$ is used to form a quadratic model that is minimized in a ball around the current iterate x_k.

§3.3 Rate of Convergence

In the following we discuss just the local rate-of-convergence properties of Newton's method.

Theorem

*Suppose that f is twice differentiable and that the Hessian $\nabla^2 f$ is **Lipschitz continuous** in a neighborhood of a solution x_* at which $(\nabla f)(x_*) = 0$ and $(\nabla^2 f)(x_*)$ is positive definitive. Consider the iteration $x_{k+1} = x_k + p_k^N = x_k - (\nabla^2 f_k)^{-1} \nabla f_k$. Then*

1. *if the starting point x_0 is sufficiently close to x_*, the sequence of iterates converges to x_*;*
2. *the rate of convergence of $\{x_k\}_{k=1}^{\infty}$ is quadratic; and*
3. *the sequence of gradient norms $\{\|\nabla f_k\|\}_{k=1}^{\infty}$ converges quadratically to zero.*

§3.3 Rate of Convergence

Proof.

First, since $(\nabla^2 f)(x_*)$ is non-singular and $\nabla^2 f$ is Lipschitz in a neighborhood of x_*, there exist $L, \delta > 0$ such that

$$\|(\nabla^2 f)^{-1}(x)\| \leqslant 2\|(\nabla^2 f)^{-1}(x_*)\| \qquad \forall\, x \in B(x_*, \delta)$$

and

$$\|(\nabla^2 f)(x) - (\nabla^2 f)(y)\| \leqslant L\|x - y\| \qquad \forall\, x, y \in B(x_*, \delta)\,.$$

From the definition of the Newton step and the condition $\nabla f_* = 0$,

$$\begin{aligned} x_{k+1} - x_* = x_k + p_k^N - x_* &= x_k - x_* - (\nabla^2 f_k)^{-1}\nabla f_k \\ &= (\nabla^2 f_k)^{-1}\big[(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*)\big]\,, \end{aligned} \tag{14}$$

where, by the chain rule, the last term can be written as

$$\nabla f_k - \nabla f_* = \int_0^1 \frac{d}{dt}(\nabla f)((1-t)x_* + tx_k)\,dt$$

□

§3.3 Rate of Convergence

Proof.

First, since $(\nabla^2 f)(x_*)$ is non-singular and $\nabla^2 f$ is Lipschitz in a neighborhood of x_*, there exist $L, \delta > 0$ such that

$$\left\|(\nabla^2 f)^{-1}(x)\right\| \leqslant 2\left\|(\nabla^2 f)^{-1}(x_*)\right\| \qquad \forall\, x \in B(x_*, \delta)$$

and

$$\left\|(\nabla^2 f)(x) - (\nabla^2 f)(y)\right\| \leqslant L\|x - y\| \qquad \forall\, x, y \in B(x_*, \delta)\,.$$

From the definition of the Newton step and the condition $\nabla f_* = 0$,

$$\begin{aligned} x_{k+1} - x_* = x_k + p_k^{\mathrm{N}} - x_* &= x_k - x_* - (\nabla^2 f_k)^{-1}\nabla f_k \\ &= (\nabla^2 f_k)^{-1}\left[(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*)\right], \end{aligned} \tag{14}$$

where, by the chain rule, the last term can be written as

$$\nabla f_k - \nabla f_* = \int_0^1 \frac{d}{dt}(\nabla f)((1-t)x_* + t x_k)\, dt$$

□

§3.3 Rate of Convergence

Proof.

First, since $(\nabla^2 f)(x_*)$ is non-singular and $\nabla^2 f$ is Lipschitz in a neighborhood of x_*, there exist $L, \delta > 0$ such that

$$\big\|(\nabla^2 f)^{-1}(x)\big\| \leqslant 2\big\|(\nabla^2 f)^{-1}(x_*)\big\| \qquad \forall\, x \in B(x_*, \delta)$$

and

$$\big\|(\nabla^2 f)(x) - (\nabla^2 f)(y)\big\| \leqslant L\|x - y\| \qquad \forall\, x, y \in B(x_*, \delta)\,.$$

From the definition of the Newton step and the condition $\nabla f_* = 0$,

$$\begin{aligned} x_{k+1} - x_* = x_k + p_k^{\mathrm{N}} - x_* &= x_k - x_* - (\nabla^2 f_k)^{-1}\nabla f_k \\ &= (\nabla^2 f_k)^{-1}\big[(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*)\big]\,, \end{aligned} \tag{14}$$

where, by the chain rule, the last term can be written as

$$\nabla f_k - \nabla f_* = \int_0^1 \frac{d}{dt}(\nabla f)((1-t)x_* + t x_k)\, dt$$

□

§3.3 Rate of Convergence

Proof.

First, since $(\nabla^2 f)(x_*)$ is non-singular and $\nabla^2 f$ is Lipschitz in a neighborhood of x_*, there exist $L, \delta > 0$ such that

$$\big\|(\nabla^2 f)^{-1}(x)\big\| \leqslant 2\big\|(\nabla^2 f)^{-1}(x_*)\big\| \qquad \forall\, x \in B(x_*, \delta)$$

and

$$\big\|(\nabla^2 f)(x) - (\nabla^2 f)(y)\big\| \leqslant L\|x - y\| \qquad \forall\, x, y \in B(x_*, \delta)\,.$$

From the definition of the Newton step and the condition $\nabla f_* = 0$,

$$\begin{aligned} x_{k+1} - x_* = x_k + p_k^{\rm N} - x_* &= x_k - x_* - (\nabla^2 f_k)^{-1}\nabla f_k \\ &= (\nabla^2 f_k)^{-1}\big[(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*)\big]\,, \end{aligned} \tag{14}$$

where, by the chain rule, the last term can be written as

$$\nabla f_k - \nabla f_* = \int_0^1 \frac{d}{dt}(\nabla f)(x_* + t(x_k - x_*))\, dt$$

□

§3.3 Rate of Convergence

Proof.

First, since $(\nabla^2 f)(x_*)$ is non-singular and $\nabla^2 f$ is Lipschitz in a neighborhood of x_*, there exist $L, \delta > 0$ such that

$$\|(\nabla^2 f)^{-1}(x)\| \leqslant 2\|(\nabla^2 f)^{-1}(x_*)\| \qquad \forall\, x \in B(x_*, \delta)$$

and

$$\|(\nabla^2 f)(x) - (\nabla^2 f)(y)\| \leqslant L\|x - y\| \qquad \forall\, x, y \in B(x_*, \delta)\,.$$

From the definition of the Newton step and the condition $\nabla f_* = 0$,

$$\begin{aligned} x_{k+1} - x_* = x_k + p_k^{\mathrm{N}} - x_* &= x_k - x_* - (\nabla^2 f_k)^{-1}\nabla f_k \\ &= (\nabla^2 f_k)^{-1}\big[(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*)\big]\,, \end{aligned} \tag{14}$$

where, by the chain rule, the last term can be written as

$$\nabla f_k - \nabla f_* = \int_0^1 (\nabla^2 f)(x_* + t(x_k - x_*))(x_k - x_*)\, dt\,.$$

□

§3.3 Rate of Convergence

Proof (cont'd).

Therefore, if $x_k \in B(x_*, \delta)$,

$$\begin{aligned}
&\big\|(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*)\big\| \\
&\quad = \Big\| \int_0^1 \big[(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\big](x_k - x_*)dt \Big\| \\
&\quad \leqslant \int_0^1 \big\|\big[(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\big](x_k - x_*)\big\| \, dt \\
&\quad \leqslant \int_0^1 L \, \big\|x_k - [x_* + t(x_k - x_*)]\big\| \|x_k - x_*\| \, dt \\
&\quad \leqslant \int_0^1 L(1-t)\|x_k - x_*\|^2 \, dt = \frac{L}{2}\|x_k - x_*\|^2
\end{aligned}$$

and the identity (14) shows that

$$\|x_{k+1} - x_*\| \leqslant \frac{L}{2}\big\|(\nabla^2 f)^{-1}(x_k)\big\|\|x_k - x_*\|^2$$

□

§3.3 Rate of Convergence

Proof (cont'd).

Therefore, if $x_k \in B(x_*, \delta)$,

$$\begin{aligned}
&\big\|(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*)\big\| \\
&\quad = \Big\| \int_0^1 \big[(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\big](x_k - x_*) dt \Big\| \\
&\quad \leqslant \int_0^1 \big\| \big[(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\big](x_k - x_*)\big\| \, dt \\
&\quad \leqslant \int_0^1 L \, \big\| x_k - [x_* + t(x_k - x_*)]\big\| \|x_k - x_*\| \, dt \\
&\quad \leqslant \int_0^1 L(1-t)\|x_k - x_*\|^2 \, dt = \frac{L}{2}\|x_k - x_*\|^2
\end{aligned}$$

and the identity (14) shows that

$$\|x_{k+1} - x_*\| \leqslant \frac{L}{2}\big\|(\nabla^2 f)^{-1}(x_k)\big\| \|x_k - x_*\|^2$$

□

§3.3 Rate of Convergence

Proof (cont'd).

Therefore, if $x_k \in B(x_*, \delta)$,

$$\begin{aligned}
&\left\|(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*)\right\| \\
&= \left\| \int_0^1 \left[(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\right](x_k - x_*) dt \right\| \\
&\leqslant \int_0^1 \left\| \left[(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\right](x_k - x_*)\right\| dt \\
&\leqslant \int_0^1 L \left\| x_k - [x_* + t(x_k - x_*)]\right\| \|x_k - x_*\| \, dt \\
&\leqslant \int_0^1 L(1-t)\|x_k - x_*\|^2 \, dt = \frac{L}{2}\|x_k - x_*\|^2
\end{aligned}$$

and the identity (14) shows that

$$\|x_{k+1} - x_*\| \leqslant L \left\|(\nabla^2 f)^{-1}(x_*)\right\| \|x_k - x_*\|^2 .$$

□

§3.3 Rate of Convergence

Proof (cont'd).

Let $\widetilde{L} = L \left\|(\nabla^2 f)^{-1}(x_*)\right\|$. Then

$$\|x_{k+1} - x_*\| \leqslant \widetilde{L}\|x_k - x_*\|^2 \qquad \text{if } x_k \in B(x_*, \delta).$$

Choose x_0 satisfying $\|x_0 - x_*\| < r \equiv \min\left\{\delta, \frac{1}{2\widetilde{L}}\right\}$. Then

$$x_k \in B(x_*, r) \subseteq B(x_*, \delta) \qquad \forall\, k \in \mathbb{N}\,;$$

thus the sequence $\{x_k\}_{k=1}^{\infty}$ converges to x_*, and the rate of convergence is quadratic.

To see that the sequence $\{\|\nabla f_k\|\}_{k=1}^{\infty}$ converges to 0 quadratically, we note that

$$\nabla f_k + \nabla^2 f_k p_k^{\mathrm{N}} = 0\,;$$

thus by the chain rule again, □

§3.3 Rate of Convergence

Proof (cont'd).

Let $\widetilde{L} = L\left\|(\nabla^2 f)^{-1}(x_*)\right\|$. Then

$$\|x_{k+1} - x_*\| \leqslant \widetilde{L}\|x_k - x_*\|^2 \qquad \text{if } x_k \in B(x_*, \delta).$$

Choose x_0 satisfying $\|x_0 - x_*\| < r \equiv \min\left\{\delta, \dfrac{1}{2\widetilde{L}}\right\}$. Then

$$x_k \in B(x_*, r) \subseteq B(x_*, \delta) \qquad \forall\, k \in \mathbb{N};$$

thus the sequence $\{x_k\}_{k=1}^{\infty}$ converges to x_*, and the rate of convergence is quadratic.

To see that the sequence $\{\|\nabla f_k\|\}_{k=1}^{\infty}$ converges to 0 quadratically, we note that

$$\nabla f_k + \nabla^2 f_k\, p_k^{\mathrm{N}} = 0\,;$$

thus by the chain rule again, □

§3.3 Rate of Convergence

Proof (cont'd).

thus by the chain rule again,

$$
\begin{aligned}
\left\|\nabla f_{k+1}\right\| &= \left\|(\nabla f)(x_{k+1}) - (\nabla f)(x_k) - (\nabla^2 f)(x_k)p_k^{\mathrm{N}}\right\| \\
&= \left\|\int_0^1 \frac{d}{dt}(\nabla f)((1-t)x_k + tx_{k+1})\,dt - (\nabla^2 f)(x_k)p_k^{\mathrm{N}}\right\| \\
&= \left\|\int_0^1 (\nabla^2 f)(x_k + tp_k^{\mathrm{N}})p_k^{\mathrm{N}}\,dt - \int_0^1 (\nabla^2 f)(x_k)p_k^{\mathrm{N}}\,dt\right\| \\
&= \left\|\int_0^1 \left[(\nabla^2 f)(x_k + tp_k^{\mathrm{N}}) - (\nabla^2 f)(x_k)\right]p_k^{\mathrm{N}}\,dt\right\| \\
&\leqslant \int_0^1 Lt\left\|p_k^{\mathrm{N}}\right\|^2 dt = \frac{L}{2}\left\|p_k^{\mathrm{N}}\right\|^2 \leqslant \frac{L}{2}\left\|(\nabla^2 f)(x_k)^{-1}\right\|^2\|\nabla f_k\|^2 \\
&\leqslant 2L\left\|(\nabla^2 f)(x_*)^{-1}\right\|^2\|\nabla f_k\|^2 .
\end{aligned}
$$

Therefore, $\{\|\nabla f_k\|\}_{k=1}^{\infty}$ converges quadratically to zero. □

§3.3 Rate of Convergence

Remark: If f is assumed to be twice continuously differentiable only but **not** necessarily Lipschitz in a neighborhood of x_*, the sequence of iterates generated by Newton's method may **not** achieve quadratic convergence. Nevertheless, the convergence is still superlinear since for $x_k \in B(x_*, \delta)$ in the proof,

$$\begin{aligned}
&\left\|(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*)\right\| \\
&\quad = \left\| \int_0^1 \left[(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\right](x_k - x_*)dt \right\| \\
&\quad \leqslant \int_0^1 \left\| \left[(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\right](x_k - x_*)\right\| dt \\
&\quad \leqslant \int_0^1 \left\|(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\right\| \|x_k - x_*\| \, dt \\
&\quad = o(\|x_k - x_*\|) \, ,
\end{aligned}$$

where the last equality follows from the continuity of $\nabla^2 f$.

§3.3 Rate of Convergence

Remark: If f is assumed to be twice continuously differentiable only but **not** necessarily Lipschitz in a neighborhood of x_*, the sequence of iterates generated by Newton's method may **not** achieve quadratic convergence. Nevertheless, the convergence is still superlinear since for $x_k \in B(x_*, \delta)$ in the proof,

$$\begin{aligned}
&\left\|(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*)\right\| \\
&\qquad = \left\| \int_0^1 \left[(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\right](x_k - x_*) dt \right\| \\
&\qquad \leqslant \int_0^1 \left\| \left[(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\right](x_k - x_*)\right\| dt \\
&\qquad \leqslant \int_0^1 \left\|(\nabla^2 f)(x_k) - (\nabla^2 f)(x_* + t(x_k - x_*))\right\| \|x_k - x_*\| \, dt \\
&\qquad = o(\|x_k - x_*\|) \,,
\end{aligned}$$

where the last equality follows from the continuity of $\nabla^2 f$.

§3.3 Rate of Convergence

Therefore, using

$$\begin{aligned} x_{k+1} - x_* &= x_k + p_k^N - x_* = x_k - x_* - (\nabla^2 f_k)^{-1} \nabla f_k \\ &= (\nabla^2 f_k)^{-1} \big[(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*) \big] , \end{aligned} \tag{14}$$

and

$$\big\| (\nabla^2 f)^{-1}(x) \big\| \leqslant 2 \big\| (\nabla^2 f)^{-1}(x_*) \big\| \qquad \forall\, x \in B(x_*, \delta)$$

we obtain

$$\|x_{k+1} - x_*\| = o(\|x_k - x_*\|) \, .$$

Even though we always "assume" that the sequence of iterates generated by Newton's method converges quadratically, in most of the situations (when we only assume the continuity of $\nabla^2 f$) superlinear convergence is the best rate of convergence result we can have.

§3.3 Rate of Convergence

Therefore, using

$$\begin{aligned} x_{k+1} - x_* &= x_k + p_k^{\mathrm{N}} - x_* = x_k - x_* - (\nabla^2 f_k)^{-1} \nabla f_k \\ &= (\nabla^2 f_k)^{-1} \big[(\nabla^2 f_k)(x_k - x_*) - (\nabla f_k - \nabla f_*) \big] , \end{aligned} \tag{14}$$

and

$$\big\| (\nabla^2 f)^{-1}(x) \big\| \leqslant 2 \big\| (\nabla^2 f)^{-1}(x_*) \big\| \qquad \forall\, x \in B(x_*, \delta)$$

we obtain

$$\|x_{k+1} - x_*\| = o(\|x_k - x_*\|) .$$

Even though we always "assume" that the sequence of iterates generated by Newton's method converges quadratically, in most of the situations (when we only assume the continuity of $\nabla^2 f$) superlinear convergence is the best rate of convergence result we can have.

§3.3 Rate of Convergence

- **Convergence Rate of Quasi-Newton Method**:

Suppose now that the search direction has the form $p_k = -B_k^{-1}\nabla f_k$, where the symmetric and positive definite matrix B_k is updated at every iteration by a quasi-Newton updating formula. In this part of the section we aim for showing the superlinear convergence of quasi-Newton method under the assumption that B_k satisfies

$$\lim_{k\to\infty} \frac{\|(B_k - \nabla^2 f(x_*))p_k\|}{\|p_k\|} = 0\,. \tag{15}$$

We note that in the case of Newton's method, $B_k = (\nabla^2 f)(x_k)$ so (15) holds if f is twice continuously differentiable:

$$\lim_{k\to\infty} \frac{\|\big((\nabla^2 f)(x_k) - (\nabla^2 f)(x_*)\big)p_k\|}{\|p_k\|} = 0\,.$$

§3.3 Rate of Convergence

An amazing consequence of this result is that a superlinear convergence rate can be attained **even if** the sequence of quasi-Newton matrices B_k does not converge to $\nabla^2 f(x_*)$; it suffices that the B_k become increasingly accurate approximations to $\nabla^2 f(x_*)$ along the search directions p_k.

In fact, under the assumption that f is twice continuously differentiable, we can show that a quasi-Newton method has superlinear convergence if and only if the quasi-Newton matrices B_k satisfies

$$\lim_{k\to\infty} \frac{\|(B_k - \nabla^2 f(x_*))p_k\|}{\|p_k\|} = 0\,. \tag{15}$$

(15) is called **the Dennis-Moré characterization** of superlinear convergence. We start with an equivalent condition of superlinear minimization algorithm.

§3.3 Rate of Convergence

An amazing consequence of this result is that a superlinear convergence rate can be attained **even if** the sequence of quasi-Newton matrices B_k does not converge to $\nabla^2 f(x_*)$; it suffices that the B_k become increasingly accurate approximations to $\nabla^2 f(x_*)$ along the search directions p_k.

In fact, under the assumption that f is twice continuously differentiable, we can show that a quasi-Newton method has superlinear convergence if and only if the quasi-Newton matrices B_k satisfies

$$\lim_{k\to\infty} \frac{\|(B_k - \nabla^2 f(x_*))p_k\|}{\|p_k\|} = 0\,. \tag{15}$$

(15) is called **the Dennis-Moré characterization** of superlinear convergence. We start with an equivalent condition of superlinear minimization algorithm.

§3.3 Rate of Convergence

An amazing consequence of this result is that a superlinear convergence rate can be attained **even if** the sequence of quasi-Newton matrices B_k does not converge to $\nabla^2 f(x_*)$; it suffices that the B_k become increasingly accurate approximations to $\nabla^2 f(x_*)$ along the search directions p_k.

In fact, under the assumption that f is twice continuously differentiable, we can show that a quasi-Newton method has superlinear convergence if and only if the quasi-Newton matrices B_k satisfies

$$\lim_{k\to\infty} \frac{\|(B_k - \nabla^2 f(x_*))p_k\|}{\|p_k\|} = 0\,. \tag{15}$$

(15) is called **the Dennis-Moré characterization** of superlinear convergence. We start with an equivalent condition of superlinear minimization algorithm.

§3.3 Rate of Convergence

Lemma

Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable, and $\{x_k\}$ be a sequence of iterates generated by some minimization algorithm. Assume that $\{x_k\}_{k=1}^{\infty}$ converges to a point x_* such that $(\nabla f)(x_*) = 0$ and $(\nabla^2 f)(x_*)$ is positive definite. Then $\{x_k\}_{k=1}^{\infty}$ converges superlinearly if and only if

$$\|x_{k+1} - x_k - p_k^{\mathrm{N}}\| = o(\|x_{k+1} - x_k\|)\,, \tag{16}$$

where $p_k^{\mathrm{N}} = -(\nabla^2 f_k)^{-1}\nabla f_k$ is the Newton direction.

Proof.

First we note that the remark after the quadratic converngence of Newton's method shows that under the current setting we have

$$\|x_k + p_k^{\mathrm{N}} - x_*\| = o(\|x_k - x_*\|)\,. \tag{17}$$

§3.3 Rate of Convergence

Lemma

Let $f:\mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable, and $\{x_k\}$ be a sequence of iterates generated by some minimization algorithm. Assume that $\{x_k\}_{k=1}^{\infty}$ converges to a point x_* such that $(\nabla f)(x_*) = 0$ and $(\nabla^2 f)(x_*)$ is positive definite. Then $\{x_k\}_{k=1}^{\infty}$ converges superlinearly if and only if

$$\|x_{k+1} - x_k - p_k^{\mathrm{N}}\| = o(\|x_{k+1} - x_k\|)\,, \tag{16}$$

where $p_k^{\mathrm{N}} = -(\nabla^2 f_k)^{-1}\nabla f_k$ is the Newton direction.

Proof.

First we note that the remark after the quadratic convergence of Newton's method shows that under the current setting we have

$$\|x_k + p_k^{\mathrm{N}} - x_*\| = o(\|x_k - x_*\|)\,. \tag{17}$$

§3.3 Rate of Convergence

Proof (cont'd).

Assume that

$$\|x_{k+1} - x_k - p_k^N\| = o(\|x_{k+1} - x_k\|) \tag{16}$$

holds. By the superlinear convergence of Newton's iterates (17),

$$\begin{aligned}\|x_{k+1} - x_*\| &\leqslant \|x_{k+1} - x_k - p_k^N\| + \|x_k + p_k^N - x_*\| \\ &= o(\|x_{k+1} - x_k\|) + o(\|x_k - x_*\|)\end{aligned} \tag{18}$$

Moreover, using the inequality above,

$$\begin{aligned}\|x_{k+1} - x_k\| &\leqslant \|x_{k+1} - x_*\| + \|x_k - x_*\| \\ &\leqslant o(\|x_{k+1} - x_k\|) + \mathcal{O}(\|x_k - x_*\|);\end{aligned}$$

thus $\|x_{k+1} - x_k\| = \mathcal{O}(\|x_k - x_*\|)$. Using this result back in (18), we conclude that

$$\|x_{k+1} - x_*\| = o(\|x_k - x_*\|),$$

giving the superlinear convergence result. □

§3.3 Rate of Convergence

Proof (cont'd).

Assume that

$$\|x_{k+1} - x_k - p_k^N\| = o(\|x_{k+1} - x_k\|) \tag{16}$$

holds. By the superlinear convergence of Newton's iterates (17),

$$\begin{aligned}\|x_{k+1} - x_*\| &\leqslant \|x_{k+1} - x_k - p_k^N\| + \|x_k + p_k^N - x_*\| \\ &= o(\|x_{k+1} - x_k\|) + o(\|x_k - x_*\|)\end{aligned} \tag{18}$$

Moreover, using the inequality above,

$$\begin{aligned}\|x_{k+1} - x_k\| &\leqslant \|x_{k+1} - x_*\| + \|x_k - x_*\| \\ &\leqslant o(\|x_{k+1} - x_k\|) + \mathcal{O}(\|x_k - x_*\|)\,;\end{aligned}$$

thus $\|x_{k+1} - x_k\| = \mathcal{O}(\|x_k - x_*\|)$. Using this result back in (18), we conclude that

$$\|x_{k+1} - x_*\| = o(\|x_k - x_*\|)\,,$$

giving the superlinear convergence result. □

§3.3 Rate of Convergence

Proof (cont'd).

Assume that

$$\|x_{k+1} - x_k - p_k^N\| = o(\|x_{k+1} - x_k\|) \tag{16}$$

holds. By the superlinear convergence of Newton's iterates (17),

$$\begin{aligned}\|x_{k+1} - x_*\| &\leqslant \|x_{k+1} - x_k - p_k^N\| + \|x_k + p_k^N - x_*\| \\ &= o(\|x_{k+1} - x_k\|) + o(\|x_k - x_*\|) \end{aligned} \tag{18}$$

Moreover, using the inequality above,

$$\begin{aligned}\|x_{k+1} - x_k\| &\leqslant \|x_{k+1} - x_*\| + \|x_k - x_*\| \\ &\leqslant o(\|x_{k+1} - x_k\|) + \mathcal{O}(\|x_k - x_*\|)\,;\end{aligned}$$

thus $\|x_{k+1} - x_k\| = \mathcal{O}(\|x_k - x_*\|)$. Using this result back in (18), we conclude that

$$\|x_{k+1} - x_*\| = o(\|x_k - x_*\|)\,,$$

giving the superlinear convergence result. □

§3.3 Rate of Convergence

Proof (cont'd).

On the other hand, suppose that $\{x_k\}$ converges superlinearly to x_*. Then the fact that

$$\begin{aligned} \|x_k - x_*\| &\leqslant \|x_{k+1} - x_k\| + \|x_{k+1} - x_*\| \\ &= \|x_{k+1} - x_k\| + o(\|x_k - x_*\|) \end{aligned}$$

shows that

$$\|x_k - x_*\| = \mathcal{O}(\|x_{k+1} - x_k\|)\,.$$

Therefore, using the superlinear convergence of Newton's iterates (17), we conclude that

$$\begin{aligned} \|x_{k+1} - x_k - p_k^N\| &\leqslant \|x_{k+1} - x_*\| + \|x_k + p_k^N - x_*\| \\ &\leqslant \|x_{k+1} - x_*\| + o(\|x_k - x_*\|) \\ &= o(\|x_k - x_*\|) = o(\|x_{k+1} - x_k\|)\,; \end{aligned}$$

thus condition (16) holds. □

§3.3 Rate of Convergence

Proof (cont'd).

On the other hand, suppose that $\{x_k\}$ converges superlinearly to x_*. Then the fact that

$$\begin{aligned} \|x_k - x_*\| &\leqslant \|x_{k+1} - x_k\| + \|x_{k+1} - x_*\| \\ &= \|x_{k+1} - x_k\| + o(\|x_k - x_*\|) \end{aligned}$$

shows that

$$\|x_k - x_*\| = \mathcal{O}(\|x_{k+1} - x_k\|)\,.$$

Therefore, using the superlinear convergence of Newton's iterates (17), we conclude that

$$\begin{aligned} \|x_{k+1} - x_k - p_k^{\rm N}\| &\leqslant \|x_{k+1} - x_*\| + \|x_k + p_k^{\rm N} - x_*\| \\ &\leqslant \|x_{k+1} - x_*\| + o(\|x_k - x_*\|) \\ &= o(\|x_k - x_*\|) = o(\|x_{k+1} - x_k\|)\,; \end{aligned}$$

thus condition (16) holds. □

§3.3 Rate of Convergence

If, as in Newton's method, the unit step length is taken in an algorithm, then $x_{k+1} = x_k + p_k$ and the equivalence of the superlinear convergence (18) can be rewritten as

$$\|p_k - p_k^{\mathrm{N}}\| = o(\|p_k\|) . \tag{19}$$

In other words, for an algorithm that eventually adopts unit step length, that the search direction approximates the Newton direction well enough is crucial for the superlinear convergence.

The result on the next page provides a sufficient condition for the admissibility of unite step length: if the search direction approximates the Newton direction in the sense

$$\lim_{k\to\infty} \frac{\|\nabla f_k + \nabla^2 f_k p_k\|}{\|p_k\|} = 0 , \tag{20}$$

then the unit step length will satisfy the Wolfe conditions as the iterates converge to the solution.

§3.3 Rate of Convergence

If, as in Newton's method, the unit step length is taken in an algorithm, then $x_{k+1} = x_k + p_k$ and the equivalence of the superlinear convergence (18) can be rewritten as

$$\|p_k - p_k^{\mathrm{N}}\| = o(\|p_k\|)\,. \tag{19}$$

In other words, for an algorithm that eventually adopts unit step length, that the search direction approximates the Newton direction well enough is crucial for the superlinear convergence.

The result on the next page provides a sufficient condition for the admissibility of unite step length: if the search direction approximates the Newton direction in the sense

$$\lim_{k\to\infty} \frac{\|\nabla f_k + \nabla^2 f_k p_k\|}{\|p_k\|} = 0\,, \tag{20}$$

then the unit step length will satisfy the Wolfe conditions as the iterates converge to the solution.

§3.3 Rate of Convergence

Lemma

Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable. Consider the iteration $x_{k+1} = x_k + \alpha_k p_k$, where p_k is a descent direction and α_k satisfies the Wolfe conditions

$$f(x_k + \alpha_k p_k) \leqslant f(x_k) + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \,, \tag{5a}$$

$$\nabla f(x_k + \alpha_k p_k)^{\mathrm{T}} p_k \geqslant c_2 \nabla f_k^{\mathrm{T}} p_k \,, \tag{5b}$$

with $c_1 < 1/2$. If the sequence $\{x_k\}_{k=1}^{\infty}$ converges to a point x_* such that $\nabla f(x_*) = 0$ and $\nabla^2 f(x_*)$ is positive definite, and if the search direction p_k satisfies

$$\lim_{k \to \infty} \frac{\|\nabla f_k + \nabla^2 f_k p_k\|}{\|p_k\|} = 0 \,, \tag{20}$$

then the step length $\alpha_k = 1$ is admissible for all $k \gg 1$.

§3.3 Rate of Convergence

Proof.

Note that the positive definiteness of $\nabla^2 f_*$ shows that

$$p_k^{\mathrm{T}} \nabla^2 f_* p_k \geqslant \lambda_{\min}(\nabla^2 f_*)\|p_k\|^2 \geqslant o(\|p_k\|^2) \quad \forall\, k \gg 1\,,$$

where $\lambda_{\min}(\nabla^2 f_*)$ denotes the smallest eigenvalue of $\nabla^2 f_*$. Under the assumption (20), Taylor's Theorem shows that

$$\begin{aligned}
&(\nabla f)(x_k + p_k)^{\mathrm{T}} p_k \geqslant c_2 \nabla f_k^{\mathrm{T}} p_k \\
&\Leftrightarrow \big[\nabla f_k + \nabla^2 f_k p_k\big]^{\mathrm{T}} p_k + o(\|p_k\|^2) \\
&\qquad \geqslant c_2 \big[\nabla f_k + \nabla^2 f_k p_k\big]^{\mathrm{T}} p_k - c_2 p_k^{\mathrm{T}} \nabla^2 f_k p_k \\
&\Leftrightarrow o(\|p_k\|^2) \leqslant c_2 p_k^{\mathrm{T}} \nabla^2 f_* p_k
\end{aligned}$$

so the curvature condition (5b) holds for the unit step length for $k \gg 1$. □

§3.3 Rate of Convergence

Proof.

Note that the positive definiteness of $\nabla^2 f_*$ shows that

$$p_k^{\mathrm{T}} \nabla^2 f_* p_k \geqslant \lambda_{\min}(\nabla^2 f_*) \|p_k\|^2 \geqslant o(\|p_k\|^2) \quad \forall\, k \gg 1\,,$$

where $\lambda_{\min}(\nabla^2 f_*)$ denotes the smallest eigenvalue of $\nabla^2 f_*$. Under the assumption (20), Taylor's Theorem shows that

$$\begin{aligned}
&(\nabla f)(x_k + p_k)^{\mathrm{T}} p_k \geqslant c_2 \nabla f_k^{\mathrm{T}} p_k \\
&\qquad \Leftrightarrow \left[\nabla f_k + \nabla^2 f_k p_k\right]^{\mathrm{T}} p_k + o(\|p_k\|^2) \\
&\qquad\qquad \geqslant c_2 \left[\nabla f_k + \nabla^2 f_k p_k\right]^{\mathrm{T}} p_k - c_2 p_k^{\mathrm{T}} \nabla^2 f_k p_k \\
&\qquad \Leftrightarrow o(\|p_k\|^2) \leqslant c_2 p_k^{\mathrm{T}} \nabla^2 f_* p_k
\end{aligned}$$

so the curvature condition (5b) holds for the unit step length for $k \gg 1$. □

§3.3 Rate of Convergence

Proof (cont'd).

Moreover, by the assumption (20) and Taylor's Theorem again we find that

$$
\begin{aligned}
& f(x_k + p_k) \leqslant f(x_k) + c_1 \nabla f_k^{\mathrm{T}} p_k \\
& \Leftrightarrow \nabla f_k^{\mathrm{T}} p_k + \frac{1}{2} p_k^{\mathrm{T}} \nabla^2 f_k p_k + o(\|p_k\|^2) \leqslant c_1 \nabla f_k^{\mathrm{T}} p_k \\
& \Leftrightarrow \big[\nabla f_k + \nabla^2 f_k p_k\big]^{\mathrm{T}} p_k - \frac{1}{2} p_k^{\mathrm{T}} \nabla^2 f_k p_k + o(\|p_k\|^2) \\
& \qquad \leqslant c_1 \big[\nabla f_k + \nabla^2 f_k p_k\big]^{\mathrm{T}} p_k - c_1 p_k^{\mathrm{T}} \nabla^2 f_k p_k \\
& \Leftrightarrow o(\|p_k\|^2) \leqslant \Big(\frac{1}{2} - c_1\Big) p_k^{\mathrm{T}} \nabla^2 f_* p_k ,
\end{aligned}
$$

so if $c_1 < \frac{1}{2}$ the Armijo condition (5a) holds for the unit step length for $k \gg 1$. □

§3.3 Rate of Convergence

Note that under the assumptions of previous two lemmas; that is, f is twice continuously differentiable and the sequence of iterates $\{x_k\}$ converges to x_* at which $\nabla f_* = 0$ and $\nabla^2 f_*$ is positive definite, the necessary condition for the admissibility of unit step length in the Wolfe conditions

$$\lim_{k\to\infty} \frac{\|\nabla f_k + \nabla^2 f_k p_k\|}{\|p_k\|} = 0 \tag{20}$$

is equivalent to the condition for superlinear convergence

$$\|p_k - p_k^{\mathrm{N}}\| = o(\|p_k\|) \quad \Leftrightarrow \quad \lim_{k\to\infty} \frac{\|p_k - p_k^{\mathrm{N}}\|}{\|p_k\|} = 0 \tag{19}$$

since

$$\begin{aligned} \nabla f_k + \nabla^2 f_k p_k &= (\nabla^2 f_k)(p_k - p_k^{\mathrm{N}}) \\ \Leftrightarrow p_k - p_k^{\mathrm{N}} &= (\nabla^2 f_k)^{-1}(\nabla f_k + \nabla^2 f_k p_k) \end{aligned}$$

and $\|\nabla^2 f_k\| \approx \|\nabla^2 f_*\|$ and $\|(\nabla^2 f_k)^{-1}\| \approx \|(\nabla^2 f_*)^{-1}\|$ for $k \gg 1$.

§3.3 Rate of Convergence

The observation from the previous page together with the previous two lemmas motivate the following

Theorem

Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable. Consider the iteration $x_{k+1} = x_k + p_k$ (that is, the step length α_k is uniformly 1) and that p_k is given by $p_k = -B_k^{-1}\nabla f_k$. Assume that $\{x_k\}_{k=1}^{\infty}$ converges to a point x_* such that $(\nabla f)(x_*) = 0$ and $(\nabla^2 f)(x_*)$ is positive definite. Then $\{x_k\}_{k=1}^{\infty}$ converges superlinearly if and only if

$$\lim_{k\to\infty} \frac{\|(B_k - \nabla^2 f(x_*))p_k\|}{\|p_k\|} = 0 . \tag{15}$$

Proof.

It suffices to show that (15) is equivalent to (20). □

§3.3 Rate of Convergence

The observation from the previous page together with the previous two lemmas motivate the following

Theorem

Suppose that $f:\mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable. Consider the iteration $x_{k+1} = x_k + p_k$ (that is, the step length α_k is uniformly 1) and that p_k is given by $p_k = -B_k^{-1}\nabla f_k$. Assume that $\{x_k\}_{k=1}^{\infty}$ converges to a point x_* such that $(\nabla f)(x_*) = 0$ and $(\nabla^2 f)(x_*)$ is positive definite. Then $\{x_k\}_{k=1}^{\infty}$ converges superlinearly if and only if

$$\lim_{k\to\infty} \frac{\|(B_k - \nabla^2 f(x_*))p_k\|}{\|p_k\|} = 0\,. \tag{15}$$

Proof.

It suffices to show that (15) is equivalent to (20). □

§3.3 Rate of Convergence

Proof (cont'd).

Let $\nabla^2 f_* = (\nabla^2 f)(x_*)$. Note that for $p_k = -B_k^{-1} \nabla f_k$,

$$(B_k - \nabla^2 f_*) p_k = -(\nabla f_k + \nabla^2 f_k p_k) + (\nabla^2 f_k - \nabla^2 f_*) p_k\,,$$

and the continuity of $\nabla^2 f$ implies that

$$\lim_{k \to \infty} \frac{\|(\nabla^2 f_k - \nabla^2 f_*) p_k\|}{\|p_k\|} = 0\,.$$

Therefore,

$$\lim_{k \to \infty} \frac{\|(B_k - \nabla^2 f_*) p_k\|}{\|p_k\|} = 0 \qquad (15)$$

if and only if

$$\lim_{k \to \infty} \frac{\|\nabla f_k + \nabla^2 f_k p_k\|}{\|p_k\|} = 0\,, \qquad (20)$$

giving the result. □

§3.3 Rate of Convergence

Proof (cont'd).

Let $\nabla^2 f_* = (\nabla^2 f)(x_*)$. Note that for $p_k = -B_k^{-1}\nabla f_k$,

$$(B_k - \nabla^2 f_*)p_k = -(\nabla f_k + \nabla^2 f_k p_k) + (\nabla^2 f_k - \nabla^2 f_*)p_k\,,$$

and the continuity of $\nabla^2 f$ implies that

$$\lim_{k\to\infty} \frac{\|(\nabla^2 f_k - \nabla^2 f_*)p_k\|}{\|p_k\|} = 0\,.$$

Therefore,

$$\lim_{k\to\infty} \frac{\|(B_k - \nabla^2 f_*)p_k\|}{\|p_k\|} = 0 \tag{15}$$

if and only if

$$\lim_{k\to\infty} \frac{\|\nabla f_k + \nabla^2 f_k p_k\|}{\|p_k\|} = 0\,, \tag{20}$$

giving the result. □

§3.4 Newton's Method with Hessian Modification

Away from the solution, the Hessian matrix $(\nabla^2 f)(x)$ may not be positive definite, so the Newton direction p_k^{N} defined by

$$(\nabla^2 f)(x_k) p_k^{\mathrm{N}} = -(\nabla f)(x_k) \tag{21}$$

may not be a descent direction. We now describe an approach to overcome this difficulty when a direct linear algebra technique, such as Gaussian elimination, is used to solve the Newton equations (21). This approach obtains the step p_k from a linear system identical to (21), except that the coefficient matrix is replaced with a positive definite approximation, formed before or during the solution process. The modified Hessian is obtained by adding either a positive diagonal matrix or a full matrix to the true Hessian $(\nabla^2 f)(x_k)$. A general description of this method follows.

§3.4 Newton's Method with Hessian Modification

Away from the solution, the Hessian matrix $(\nabla^2 f)(x)$ may not be positive definite, so the Newton direction p_k^{N} defined by

$$(\nabla^2 f)(x_k)p_k^{\mathrm{N}} = -(\nabla f)(x_k) \tag{21}$$

may not be a descent direction. We now describe an approach to overcome this difficulty when a direct linear algebra technique, such as Gaussian elimination, is used to solve the Newton equations (21). This approach obtains the step p_k from a linear system identical to (21), except that the coefficient matrix is replaced with a positive definite approximation, formed before or during the solution process. The modified Hessian is obtained by adding either a positive diagonal matrix or a full matrix to the true Hessian $(\nabla^2 f)(x_k)$. A general description of this method follows.

§3.4 Newton's Method with Hessian Modification

Away from the solution, the Hessian matrix $(\nabla^2 f)(x)$ may not be positive definite, so the Newton direction $p_k^{\rm N}$ defined by

$$(\nabla^2 f)(x_k)p_k^{\rm N} = -(\nabla f)(x_k) \tag{21}$$

may not be a descent direction. We now describe an approach to overcome this difficulty when a direct linear algebra technique, such as Gaussian elimination, is used to solve the Newton equations (21). This approach obtains the step p_k from a linear system identical to (21), except that the coefficient matrix is replaced with a positive definite approximation, formed before or during the solution process. The modified Hessian is obtained by adding either a positive diagonal matrix or a full matrix to the true Hessian $(\nabla^2 f)(x_k)$. A general description of this method follows.

§3.4 Newton's Method with Hessian Modification

Algorithm 3.2 (Line Search Newton with Modification):

Given initial point x_0;

for $k = 0, 1, 2, \cdots$

Factorize the matrix $B_k = (\nabla^2 f)(x_k) + E_k$, where $E_k = 0$ if $(\nabla^2 f)(x_k)$ is sufficiently positive definite; otherwise, E_k is chosen to ensure that B_k is sufficiently positive definite;

Solve $B_k p_k = -(\nabla f)(x_k)$;

Set $x_{k+1} \leftarrow x_k + \alpha_k p_k$, where α_k satisfies the Wolfe, Goldstein, or Armijo backtracking conditions;

end

§3.4 Newton's Method with Hessian Modification

Algorithm 3.2 is a practical Newton method that can be applied from any starting point. We can establish fairly satisfactory global convergence results for it, provided that the strategy for choosing E_k (and hence B_k) satisfies the **bounded modified factorization property**. This property is that the matrices in the sequence $\{B_k\}_{k=1}^{\infty}$ have bounded condition number whenever the sequence of Hessians $\{(\nabla^2 f)(x_k)\}_{k=1}^{\infty}$ is bounded; that is, there exists $C > 0$ such that

$$\kappa(B_k) \equiv \|B_k\|\|B_k^{-1}\| \leqslant C \qquad \forall\, k \in \mathbb{N}\,. \tag{22}$$

If this property holds, global convergence of the modified line search Newton method follows from the results of Section 3.2 (page 73 of this slide).

§3.4 Newton's Method with Hessian Modification

Theorem

Let f be twice continuously differentiable on an open set $\mathcal{D}$, and assume that the starting point x_0 of Algorithm 3.2 is such that the level set $\{x \in \mathcal{D} \,|\, f(x) \leqslant f(x_0)\}$ is compact. Then if the bounded modified factorization property holds, we have that

$$\lim_{k\to\infty} (\nabla f)(x_k) = 0\,.$$

Note that since the level set $\{x \in \mathcal{D} \,|\, f(x) \leqslant f(x_0)\}$ is indeed $f^{-1}\big((-\infty, f(x_0))]\big)$ which is closed by the continuity of f, by the Heine-Borel Theorem this level set is compact if and only if it is bounded.

§3.4 Newton's Method with Hessian Modification

We now consider the convergence rate of Algorithm 3.2. Suppose that the sequence of iterates x_k converges to a point x_* where $(\nabla^2 f)(x_*)$ is sufficiently positive definite in the sense that the modification strategies described in the next section return the modification $E_k = 0$ for all sufficiently large k. By one of the previous theorem, we have that $\alpha_k = 1$ for all sufficiently large k, so that Algorithm 3.2 reduces to a pure Newton method, and the rate of convergence is quadratic.

§3.4 Newton's Method with Hessian Modification

For problems in which $\nabla^2 f_*$ is close to singular, there is no guarantee that the modification E_k will eventually vanish, and the convergence rate may be only linear. Besides requiring the modified matrix B_k to be well conditioned (so that the previous theorem holds), we would like the modification to be as small as possible, so that the second-order information in the Hessian is preserved as far as possible. Naturally, we would also like the modified factorization to be computable at moderate cost.

§3.4 Newton's Method with Hessian Modification

To set the stage for the matrix factorization techniques that will be used in Algorithm 3.2, we will begin by assuming that the eigenvalue decomposition of $(\nabla^2 f)(x_k)$ is available. This is not realistic for large-scale problems because this decomposition is generally too expensive to compute, but it will motivate several practical modification strategies.

- **Eigenvalue modification**

Consider a problem in which, at the current iterate x_k, $(\nabla f)(x_k) = (1, -3, 2)^{\mathrm{T}}$ and $(\nabla^2 f)(x_k) = \mathrm{diag}(10, 3, -1)$, which is clearly indefinite. By the spectral decomposition theorem we can define $Q = \mathrm{I}$ and $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)$, and write

$$(\nabla^2 f)(x_k) = Q\Lambda Q^{\mathrm{T}} = \sum_{i=1}^{3} \lambda_i \boldsymbol{q}_i \boldsymbol{q}_i^{\mathrm{T}} . \tag{23}$$

§3.4 Newton's Method with Hessian Modification

To set the stage for the matrix factorization techniques that will be used in Algorithm 3.2, we will begin by assuming that the eigenvalue decomposition of $(\nabla^2 f)(x_k)$ is available. This is not realistic for large-scale problems because this decomposition is generally too expensive to compute, but it will motivate several practical modification strategies.

- **Eigenvalue modification**

Consider a problem in which, at the current iterate x_k, $(\nabla f)(x_k) = (1, -3, 2)^{\mathrm{T}}$ and $(\nabla^2 f)(x_k) = \operatorname{diag}(10, 3, -1)$, which is clearly indefinite. By the spectral decomposition theorem we can define $Q = \mathrm{I}$ and $\Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \lambda_3)$, and write

$$(\nabla^2 f)(x_k) = Q\Lambda Q^{\mathrm{T}} = \sum_{i=1}^{3} \lambda_i \boldsymbol{q}_i \boldsymbol{q}_i^{\mathrm{T}} . \tag{23}$$

§3.4 Newton's Method with Hessian Modification

The pure Newton step – the solution of (21) – is $p_k^{\mathrm{N}} = (-0.1, 1, 2)^{\mathrm{T}}$, which is not a descent direction, since $\nabla f(x_k)^{\mathrm{T}} p_k^{\mathrm{N}} > 0$. One might suggest a modified strategy in which we replace $(\nabla^2 f)(x_k)$ by a positive definite approximation B_k, in which all negative eigenvalues in $(\nabla^2 f)(x_k)$ are replaced by a small positive number δ that is somewhat larger than machine precision $\boldsymbol{u}$; say $\delta = \sqrt{\boldsymbol{u}}$. For a machine precision of 10^{-16}, the resulting matrix in our example is

$$B_k = \sum_{i=1}^{2} \lambda_i \boldsymbol{q}_i \boldsymbol{q}_i^{\mathrm{T}} + \delta \boldsymbol{q}_3 \boldsymbol{q}_3^{\mathrm{T}} = \mathrm{diag}(10, 3, 10^{-8}), \qquad (24)$$

which is numerically positive definite and whose curvature along the eigenvectors q_1 and q_2 has been preserved. Note, however, that the search direction based on this modified Hessian is

$$p_k = -B_k^{-1} \nabla f_k = -\sum_{i=1}^{2} \frac{1}{\lambda_i} \boldsymbol{q}_i (\boldsymbol{q}_i^{\mathrm{T}} \nabla f_k) - \frac{1}{\delta} \boldsymbol{q}_3 (\boldsymbol{q}_3^{\mathrm{T}} \nabla f_k). \qquad (25)$$

§3.4 Newton's Method with Hessian Modification

For small δ, this step is nearly parallel to $\boldsymbol{q}_3$ and quite long. Although f decreases along the direction p_k, its extreme length violates the spirit of Newton's method, which relies on a quadratic approximation of the objective function in a neighborhood of the current iterate x_k. It is therefore not clear that this search direction is effective.

Various other modification strategies are possible. We could

1. flip the signs of the negative eigenvalues in (23), which amounts to setting $\delta = 1$ in our example, or
2. set the last term in (25) to zero, so that the search direction has no components along the negative curvature directions, or
3. adapt the choice of δ to ensure that the length of the step is not excessive, a strategy with the flavor of trust-region methods.

There is a great deal of freedom in devising modification strategies, and there is currently no agreement on which strategy is best.

§3.4 Newton's Method with Hessian Modification

For small δ, this step is nearly parallel to $\boldsymbol{q}_3$ and quite long. Although f decreases along the direction p_k, its extreme length violates the spirit of Newton's method, which relies on a quadratic approximation of the objective function in a neighborhood of the current iterate x_k. It is therefore not clear that this search direction is effective.

Various other modification strategies are possible. We could

1. flip the signs of the negative eigenvalues in (23), which amounts to setting $\delta = 1$ in our example, or
2. set the last term in (25) to zero, so that the search direction has no components along the negative curvature directions, or
3. adapt the choice of δ to ensure that the length of the step is not excessive, a strategy with the flavor of trust-region methods.

There is a great deal of freedom in devising modification strategies, and there is currently no agreement on which strategy is best.

§3.4 Newton's Method with Hessian Modification

For small δ, this step is nearly parallel to $\boldsymbol{q}_3$ and quite long. Although f decreases along the direction p_k, its extreme length violates the spirit of Newton's method, which relies on a quadratic approximation of the objective function in a neighborhood of the current iterate x_k. It is therefore not clear that this search direction is effective.

Various other modification strategies are possible. We could

1. flip the signs of the negative eigenvalues in (23), which amounts to setting $\delta = 1$ in our example, or
2. set the last term in (25) to zero, so that the search direction has no components along the negative curvature directions, or
3. adapt the choice of δ to ensure that the length of the step is not excessive, a strategy with the flavor of trust-region methods.

There is a great deal of freedom in devising modification strategies, and there is currently no agreement on which strategy is best.

§3.4 Newton's Method with Hessian Modification

Setting the issue of the choice of δ aside for the moment, let us look more closely at the process of modifying a matrix so that it becomes positive definite. The modification (24) to the example matrix (23) can be shown to be optimal in the following sense: if A is a symmetric matrix with spectral decomposition $A = Q\Lambda Q^{\mathrm{T}}$, then the correction matrix ΔA of **minimum Frobenius norm** that ensures that $\lambda_{\min}(A + \Delta A) \geqslant \delta$ is given by

$$\Delta A = Q\mathrm{diag}(\tau_1, \cdots, \tau_n)Q^{\mathrm{T}}, \quad \tau_i = \begin{cases} 0 & \text{if } \lambda_i \geqslant \delta, \\ \delta - \lambda_i & \text{if } \lambda_i < \delta. \end{cases} \tag{26}$$

Here, $\lambda_{\min}(A)$ denotes the smallest eigenvalue of A, and the Frobenius norm of a matrix A is defined as $\|A\|_F^2 = \mathrm{tr}(AA^{\mathrm{T}})$. Note that ΔA is not diagonal in general, and that the modified matrix is

$$A + \Delta A = Q(\Lambda + \mathrm{diag}(\tau))Q^{\mathrm{T}}.$$

§3.4 Newton's Method with Hessian Modification

Setting the issue of the choice of δ aside for the moment, let us look more closely at the process of modifying a matrix so that it becomes positive definite. The modification (24) to the example matrix (23) can be shown to be optimal in the following sense: if A is a symmetric matrix with spectral decomposition $A = Q\Lambda Q^{\mathrm{T}}$, then the correction matrix ΔA of **minimum Frobenius norm** that ensures that $\lambda_{\min}(A + \Delta A) \geqslant \delta$ is given by

$$\Delta A = Q\operatorname{diag}(\tau_1, \cdots, \tau_n)Q^{\mathrm{T}}, \quad \tau_i = \begin{cases} 0 & \text{if } \lambda_i \geqslant \delta\,, \\ \delta - \lambda_i & \text{if } \lambda_i < \delta\,. \end{cases} \tag{26}$$

Here, $\lambda_{\min}(A)$ denotes the smallest eigenvalue of A, and the Frobenius norm of a matrix A is defined as $\|A\|_F^2 = \operatorname{tr}(AA^{\mathrm{T}})$. Note that ΔA is not diagonal in general, and that the modified matrix is

$$A + \Delta A = Q(\Lambda + \operatorname{diag}(\boldsymbol{\tau}))Q^{\mathrm{T}}\,.$$

§3.4 Newton's Method with Hessian Modification

By using a different norm we can obtain a diagonal modification. Suppose again that A is a symmetric matrix with spectral decomposition $A = Q\Lambda Q^{\mathrm{T}}$. A correction matrix ΔA with **minimum Euclidean norm** that satisfies $\lambda_{\min}(A + \Delta A) \geqslant \delta$ is given by

$$\Delta A = \tau \mathrm{I} \quad \text{with} \quad \tau = \max\left\{0, \delta - \lambda_{\min}(A)\right\}. \tag{27}$$

All the eigenvalues of $A + \Delta A$ have thus been shifted, and all are greater than δ. The modified matrix now has the form $A + \tau\mathrm{I}$ which happens to have the same form as the matrix occurring in (unscaled) trust–region methods (see Chapter 4).

§3.4 Newton's Method with Hessian Modification

These results suggest that both diagonal and non-diagonal modifications can be considered. Even though we have not answered the question of what constitutes a good modification, various practical diagonal and non-diagonal modifications have been proposed and implemented in software. They do not make use of the spectral decomposition of the Hessian, since it is generally too expensive to compute. Instead, they use Gaussian elimination, choosing the modifications indirectly and hoping that somehow they will produce good steps. Numerical experience indicates that the strategies described next often (but not always) produce good search directions.

§3.4 Newton's Method with Hessian Modification

- **Adding a multiple of the identity**

Perhaps the simplest idea is to find a scalar $\tau > 0$ such that $\nabla^2 f(x_k) + \tau \mathrm{I}$ is sufficiently positive definite. From the previous discussion we know that τ must satisfy (27), but a good estimate of the smallest eigenvalue of the Hessian is normally not available. The following algorithm describes a method that tries successively larger values of τ.

§3.4 Newton's Method with Hessian Modification

Algorithm 3.3 (Cholesky with Added Multiple of the Identity):

Choose $\beta > 0$;

if $\min_j a_{jj} > 0$

  set $\tau_0 \leftarrow 0$;

else

  $\tau_0 = -\min_j a_{jj} + \beta$;

end (if)

for $k = 0, 1, 2, \cdots$

  Try to apply the Cholesky algorithm to obtain $LL^{\mathrm{T}} = A + \tau_k \mathrm{I}$;

  if the factorization is completed successfully

    stop and return L;

  else

    $\tau_{k+1} = \max\{2\tau_k, \beta\}$;

  end (if)

end (for)

§3.4 Newton's Method with Hessian Modification

The choice of β is heuristic; a typical value is $\beta = 10^{-3}$. ~~We could choose the first nonzero shift τ_0 to be proportional to be the final value of τ used in the latest Hessian modification; see also Algorithm B.1.~~ The strategy implemented in Algorithm 3.3 is quite simple and may be preferable to the modified factorization techniques described next, but it suffers from one drawback: every value of τ_k requires a new factorization of $A + \tau_k \mathrm{I}$, and the algorithm can be quite expensive if several trial values are generated. Therefore it may be advantageous to increase τ more rapidly, say by a factor of 10 instead of 2 in the last else clause.

§3.4 Newton's Method with Hessian Modification

- **Modified Cholesky factorization**

Another approach for modifying a Hessian matrix that is not positive definite is to perform a **Cholesky factorization** of $(\nabla^2 f)(x_k)$, but to increase the diagonal elements encountered during the factorization (where necessary) to ensure that they are sufficiently positive. This modified Cholesky approach is designed to accomplish two goals: It guarantees that the modified Cholesky factors exist and are bounded relative to the norm of the actual Hessian, and it does not modify the Hessian if it is sufficiently positive definite.

§3.4 Newton's Method with Hessian Modification

We begin our description of this approach by briefly reviewing the Cholesky factorization. Every symmetric positive definite matrix A can be written as

$$A = LDL^{\mathrm{T}}, \tag{28}$$

where L is a lower triangular matrix with **unit** diagonal elements and D is a diagonal matrix with positive elements on the diagonal. By equating the elements in (28), column by column, it is easy to derive formulas for computing L and D.

§3.4 Newton's Method with Hessian Modification

Example

Consider the case $n = 3$. Suppose the symmetric matrix $A = [a_{ij}]$ is factorized into

$$\begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ \ell_{21} & 1 & 0 \\ \ell_{31} & \ell_{32} & 1 \end{bmatrix} \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{bmatrix} \begin{bmatrix} 1 & \ell_{21} & \ell_{31} \\ 0 & 1 & \ell_{32} \\ 0 & 0 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 0 & 0 \\ \ell_{21} & 1 & 0 \\ \ell_{31} & \ell_{32} & 1 \end{bmatrix} \begin{bmatrix} d_1 & d_1\ell_{21} & d_1\ell_{31} \\ 0 & d_2 & d_2\ell_{32} \\ 0 & 0 & d_3 \end{bmatrix}$$
$$= \begin{bmatrix} d_1 & d_1\ell_{21} & d_1\ell_{31} \\ d_1\ell_{21} & d_1\ell_{21}^2 + d_2 & d_1\ell_{31}\ell_{21} + d_2\ell_{32} \\ d_1\ell_{31} & d_1\ell_{31}\ell_{21} + d_2\ell_{32} & d_1\ell_{31}^2 + d_2\ell_{32}^2 + d_3 \end{bmatrix}.$$

§3.4 Newton's Method with Hessian Modification

Example (cont'd)

By equating the elements of the first column, we have

$$\begin{aligned} a_{11} &= d_1 &\Rightarrow\quad d_1 &= a_{11}\,, \\ a_{21} &= d_1\ell_{21} &\Rightarrow\quad \ell_{21} &= \frac{a_{21}}{d_1}\,, \\ a_{31} &= d_1\ell_{31} &\Rightarrow\quad \ell_{31} &= \frac{a_{31}}{d_1}\,. \end{aligned}$$

Proceeding with the next two columns, we obtain

$$\begin{aligned} a_{22} &= d_1\ell_{21}^2 + d_2 &\Rightarrow\quad d_2 &= a_{22} - d_1\ell_{21}^2\,, \\ a_{32} &= d_1\ell_{31}\ell_{21} + d_2\ell_{32} &\Rightarrow\quad \ell_{32} &= \frac{a_{32} - d_1\ell_{31}\ell_{21}}{d_2}\,, \\ a_{33} &= d_1\ell_{31}^2 + d_2\ell_{32}^2 + d_3 &\Rightarrow\quad d_3 &= a_{33} - d_1\ell_{31}^2 + d_2\ell_{32}^2\,. \end{aligned}$$

§3.4 Newton's Method with Hessian Modification

In general, for symmetric $n \times n$ matrix A, we want to have the following decomposition

$$A = LDL^{\mathrm{T}}, \quad A = [a_{ij}], L = [\ell_{ij}], D = [d_{ij}] ,$$

where L is lower triangular matrix with unit diagonal elements, and D is a diagonal matrix. Writing d_{jj} as d_j, we have

$$a_{ij} = \sum_{r,s=1}^{n} \ell_{ir} d_{rs} \ell_{js} = \sum_{s=1}^{n} d_s \ell_{is} \ell_{js} .$$

Assuming $i \geqslant j$, the identity above shows that

$$a_{ij} = \sum_{s=1}^{j} d_s \ell_{is} \ell_{js} = d_j \ell_{ij} + \sum_{s=1}^{j-1} d_s \ell_{is} \ell_{js}$$

or

$$d_j \ell_{ij} = c_{ij} \equiv a_{ij} - \sum_{s=1}^{j-1} d_s \ell_{is} \ell_{js} .$$

§3.4 Newton's Method with Hessian Modification

In general, for symmetric $n \times n$ matrix A, we want to have the following decomposition

$$A = LDL^{\mathrm{T}}, \quad A = [a_{ij}], L = [\ell_{ij}], D = [d_{ij}],$$

where L is lower triangular matrix with unit diagonal elements, and D is a diagonal matrix. Writing d_{jj} as d_j, we have

$$a_{ij} = \sum_{r,s=1}^{n} \ell_{ir} d_{rs} \ell_{js} = \sum_{s=1}^{n} d_s \ell_{is} \ell_{js}.$$

Assuming $i \geqslant j$, the identity above shows that

$$a_{ij} = \sum_{s=1}^{j} d_s \ell_{is} \ell_{js} = d_j \ell_{ij} + \sum_{s=1}^{j-1} d_s \ell_{is} \ell_{js}$$

or

$$d_j \ell_{ij} = c_{ij} \equiv a_{ij} - \sum_{s=1}^{j-1} d_s \ell_{is} \ell_{js}.$$

§3.4 Newton's Method with Hessian Modification

Algorithm 3.4 (Cholesky Factorization, LDL^{T} Form).

for $j = 1, 2, \cdots, n$

for $i = j, j+1, \cdots, n$

$$c_{ij} \leftarrow a_{ij} - \sum_{s=1}^{j-1} d_s \ell_{is} \ell_{js};$$

$d_j \leftarrow c_{jj};$

$\ell_{ij} \leftarrow c_{ij}/d_j;$

end

end

§3.4 Newton's Method with Hessian Modification

One can show that the diagonal elements d_j are all positive whenever A is positive definite. The scalars c_{ij} have been introduced only to facilitate the description of the modified factorization discussed below. We should note that Algorithm 3.4 differs a little from the standard form of the Cholesky factorization, which produces a **lower triangular matrix** M such that

$$A = MM^{\mathrm{T}}. \tag{29}$$

In fact, we can make the identification $M = LD^{1/2}$ to relate M to the factors L and D computed in Algorithm 3.4. The technique for computing M appears as Algorithm A.2 in Appendix A.

§3.4 Newton's Method with Hessian Modification

One can show that the diagonal elements d_j are all positive whenever A is positive definite. The scalars c_{ij} have been introduced only to facilitate the description of the modified factorization discussed below. We should note that Algorithm 3.4 differs a little from the standard form of the Cholesky factorization, which produces a **lower triangular matrix** M such that

$$A = MM^{\mathrm{T}} . \tag{29}$$

In fact, we can make the identification $M = LD^{1/2}$ to relate M to the factors L and D computed in Algorithm 3.4. The technique for computing M appears as Algorithm A.2 in Appendix A.

§3.4 Newton's Method with Hessian Modification

If A is indefinite, the factorization $A = LDL^{\mathrm{T}}$ may not exist. Even if it does exist, Algorithm 3.4 is **numerically unstable** when applied to such matrices, in the sense that the elements of L and D can become arbitrarily large. It follows that a strategy of computing the LDL^{T} factorization and then modifying the diagonal after the fact to force its elements to be positive may break down, or may result in a matrix that is drastically different from A.

§3.4 Newton's Method with Hessian Modification

Instead, we can modify the matrix A during the course of the factorization in such a way that all elements in D are sufficiently positive, and so that the elements of D and L are not too large. To control the quality of the modification, we choose two positive parameters δ and β, and require that during the computation of the j-th columns of L and D in Algorithm 3.4 (that is, for each j in the outer loop of the algorithm) the following bounds be satisfied:

$$d_j \geqslant \delta, \quad |m_{ij}| \leqslant \beta \text{ for } i = \quad j+1, j+2, \cdots, n, \tag{30}$$

where $m_{ij} = \ell_{ij}\sqrt{d_j}$. To satisfy these bounds we only need to change one step in Algorithm 3.4: The formula for computing the diagonal element d_j in Algorithm 3.4 is replaced by

$$d_j = \max\left\{|c_{jj}|, \left(\frac{\theta_j}{\beta}\right)^2, \delta\right\} \quad \text{with } \theta_j = \max_{j<i\leqslant n} |c_{ij}|. \tag{31}$$

§3.4 Newton's Method with Hessian Modification

Instead, we can modify the matrix A during the course of the factorization in such a way that all elements in D are sufficiently positive, and so that the elements of D and L are not too large. To control the quality of the modification, we choose two positive parameters δ and β, and require that during the computation of the j-th columns of L and D in Algorithm 3.4 (that is, for each j in the outer loop of the algorithm) the following bounds be satisfied:

$$d_j \geqslant \delta, \quad |m_{ij}| \leqslant \beta \text{ for } i = \quad j+1, j+2, \cdots, n, \tag{30}$$

where $m_{ij} = \ell_{ij}\sqrt{d_j}$. To satisfy these bounds we only need to change one step in Algorithm 3.4: The formula for computing the diagonal element d_j in Algorithm 3.4 is replaced by

$$d_j = \max\left\{|c_{jj}|, \left(\frac{\theta_j}{\beta}\right)^2, \delta\right\} \quad \text{with } \theta_j = \max_{j<i\leqslant n} |c_{ij}|. \tag{31}$$

§3.4 Newton's Method with Hessian Modification

Instead, we can modify the matrix A during the course of the factorization in such a way that all elements in D are sufficiently positive, and so that the elements of D and L are not too large. To control the quality of the modification, we choose two positive parameters δ and β, and require that during the computation of the j-th columns of L and D in Algorithm 3.4 (that is, for each j in the outer loop of the algorithm) the following bounds be satisfied:

$$d_j \geqslant \delta, \quad |m_{ij}| \leqslant \beta \text{ for } i = j, j+1, j+2, \cdots, n, \tag{30}$$

where $m_{ij} = \ell_{ij}\sqrt{d_j}$. To satisfy these bounds we only need to change one step in Algorithm 3.4: The formula for computing the diagonal element d_j in Algorithm 3.4 is replaced by

$$d_j = \max\left\{|c_{jj}|, \left(\frac{\theta_j}{\beta}\right)^2, \delta\right\} \quad \text{with } \theta_j = \max_{j \leqslant i \leqslant n} |c_{ij}|. \tag{31}$$

§3.4 Newton's Method with Hessian Modification

Algorithm 3.4 (Cholesky Factorization, LDL^{T} Form).

for $j = 1, 2, \cdots, n$

  for $i = j, j+1, \cdots, n$

$$c_{ij} \leftarrow a_{ij} - \sum_{s=1}^{j-1} d_s \ell_{is} \ell_{js};$$

$$\theta_j \leftarrow \max_{j<i\leqslant n} |c_{ij}| \ \Big(\text{or } \max_{j\leqslant i\leqslant n} |c_{ij}|\Big);$$

$$d_j \leftarrow \max\left\{ |c_{jj}|, \left(\frac{\theta_j}{\beta}\right)^2, \delta \right\};$$

$$\ell_{ij} \leftarrow c_{ij}/d_j;$$

  end

end

§3.4 Newton's Method with Hessian Modification

To verify that (30) holds, we note from Algorithm 3.4 that $c_{ij} = \ell_{ij} d_j$, and therefore

$$|m_{ij}| = |\ell_{ij}\sqrt{d_j}| = \frac{|c_{ij}|}{\sqrt{d_j}} \leqslant \frac{|c_{ij}|\beta}{\theta_j} \leqslant \beta \quad \text{for all } i > (\text{or } \geqslant)\ j.$$

We note that θ_j can be computed prior to d_j because the elements c_{ij} in the second for loop of Algorithm 3.4 do not involve d_j. In fact, this is the reason for introducing the quantities c_{ij} into the algorithm.

§3.4 Newton's Method with Hessian Modification

These observations are the basis of the modified Cholesky algorithm described in detail in Gill, Murray, and Wright [130], which introduces symmetric interchanges of rows and columns to try to reduce the size of the modification. If P denotes the permutation matrix associated with the row and column interchanges, the algorithm produces the Cholesky factorization of the permuted, modified matrix $PAP^{\mathrm{T}} + E$; that is,

$$PAP^{\mathrm{T}} + E = LDL^{\mathrm{T}} = MM^{\mathrm{T}}, \tag{32}$$

where E is a non-negative diagonal matrix that is zero if A is sufficiently positive definite. One can show that the matrices B_k obtained by this modified Cholesky algorithm to the exact Hessians $(\nabla^2 f)(x_k)$ have bounded condition numbers; that is, the bound (22) holds for some value of C.

§3.4 Newton's Method with Hessian Modification

- **Modified symmetric indefinite factorization**

Another strategy for modifying an indefinite Hessian is to use a procedure based on a symmetric indefinite factorization. Any symmetric matrix A, whether positive definite or not, can be written as

$$PAP^{\mathrm{T}} = LBL^{\mathrm{T}} \tag{33}$$

where L is **unit** lower triangular, B is a block diagonal matrix with blocks of dimension 1 or 2, and P is a permutation matrix (see our discussion in Appendix A and also Golub and Van Loan [136, Section 4.4]). By using the block diagonal matrix B, which allows 2×2 blocks as well as 1×1 blocks on the diagonal, we can guarantee that the factorization (33) always exists and can be computed by a numerically stable process.

§3.4 Newton's Method with Hessian Modification

Example

The matrix $A = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 2 & 2 & 2 \\ 2 & 2 & 3 & 3 \\ 3 & 2 & 3 & 4 \end{bmatrix}$ can be written in the form (33) with $P = [\mathbf{e}_1, \mathbf{e}_4, \mathbf{e}_3, \mathbf{e}_2]$,

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \frac{1}{9} & \frac{2}{3} & 1 & 0 \\ \frac{2}{9} & \frac{1}{3} & 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 3 & 0 & 0 \\ 3 & 4 & 0 & 0 \\ 0 & 0 & \frac{7}{9} & \frac{5}{9} \\ 0 & 0 & \frac{5}{9} & \frac{10}{9} \end{bmatrix}.$$

Note that both diagonal blocks in B are 2×2. Several algorithms for computing symmetric indefinite factorizations are discussed in Section A.1 of Appendix A.

§3.4 Newton's Method with Hessian Modification

The symmetric indefinite factorization allows us to determine the **inertia of a matrix**; that is, the number of positive, zero, and negative eigenvalues. One can show that the inertia of B equals the inertia of A. Moreover, the 2×2 blocks in B are always constructed to have one positive and one negative eigenvalue; thus the number of positive eigenvalues in A equals the number of positive 1×1 blocks plus the number of 2×2 blocks.

§3.4 Newton's Method with Hessian Modification

As for the Cholesky factorization, an indefinite symmetric factorization algorithm can be modified to ensure that the modified factors are the factors of a positive definite matrix. The strategy is first to compute the factorization (33), as well as the spectral decomposition $B = Q\Lambda Q^{\mathrm{T}}$, which is inexpensive to compute because B is block diagonal. We then construct a modification matrix F such that

$$L(B+F)L^{\mathrm{T}}$$

is sufficiently positive definite. Motivated by the modified spectral decomposition (26), we choose a parameter $\delta > 0$ and define F to be

$$F = Q\mathrm{diag}(\tau_i)Q^{\mathrm{T}}, \ \tau_i = \begin{cases} 0 & \text{if } \lambda_i \geqslant \delta, \\ \delta - \lambda_i & \text{if } \lambda_i < \delta, i = 1, 2, \cdots, n, \end{cases} \tag{34}$$

where λ_i are the eigenvalues of B.

§3.4 Newton's Method with Hessian Modification

The matrix F is thus the modification of minimum Frobenius norm that ensures that all eigenvalues of the modified matrix $B+F$ are no less than δ. This strategy therefore modifies the factorization (33) as follows:

$$P(A+E)P^{\mathrm{T}} = L(B+F)L^{\mathrm{T}}, \quad \text{where } E = P^{\mathrm{T}} LFL^{\mathrm{T}} P.$$

Note that in general E will not be diagonal; thus in contrast to the modified Cholesky approach, this modification strategy changes the entire matrix A, not just its diagonal. The aim of strategy (34) is that the modified matrix satisfies $\lambda_{\min}(A+E) \approx \delta$ whenever the original matrix A has $\lambda_{\min}(A) < \delta$. It is not clear; however, whether it always comes close to attaining this goal.

§3.4 Newton's Method with Hessian Modification

The matrix F is thus the modification of minimum Frobenius norm that ensures that all eigenvalues of the modified matrix $B+F$ are no less than δ. This strategy therefore modifies the factorization (33) as follows:

$$P(A+E)P^{\mathrm{T}} = L(B+F)L^{\mathrm{T}}, \quad \text{where } E = P^{\mathrm{T}}LFL^{\mathrm{T}}P.$$

Note that in general E will not be diagonal; thus in contrast to the modified Cholesky approach, this modification strategy changes the entire matrix A, not just its diagonal. The aim of strategy (34) is that the modified matrix satisfies $\lambda_{\min}(A+E) \approx \delta$ whenever the original matrix A has $\lambda_{\min}(A) < \delta$. It is not clear; however, whether it always comes close to attaining this goal.

§3.5 Step-Length Selection Algorithms

We now consider techniques for finding a minimum of the one-dimensional function

$$\varphi(\alpha) = f(x_k + \alpha p_k)\,, \tag{35}$$

or for simply finding a step length α_k satisfying one of the termination conditions such as the Wolfe conditions and the Goldstein conditions in Section 3.1. We assume that p_k is a descent direction; that is, $\varphi'(0) < 0$, so that our search can be confined to positive values of α.

§3.5 Step-Length Selection Algorithms

If f is a convex quadratic given by

$$f(x) = \frac{1}{2}x^{\mathrm{T}}Qx - b^{\mathrm{T}}x,$$

its one-dimensional minimizer along the ray $x_k + \alpha p_k$ can be computed analytically and is given by

$$\alpha_k = -\frac{\nabla f_k^{\mathrm{T}} p_k}{p_k^{\mathrm{T}} Q p_k}\,. \tag{36}$$

For general nonlinear functions, it is necessary to use an iterative procedure. The line search procedure deserves particular attention because it has a major impact on the robustness and efficiency of all nonlinear optimization methods.

§3.5 Step-Length Selection Algorithms

Line search procedures can be classified according to the type of derivative information they use. Algorithms that use only function values can be inefficient since, to be theoretically sound, they need to continue iterating until the search for the minimizer is narrowed down to a small interval. In contrast, knowledge of gradient information allows us to determine whether a suitable step length has been located, as stipulated, for example, by the Wolfe conditions

$$f(x_k + \alpha_k p_k) \leqslant f(x_k) + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \,, \tag{5a}$$

$$\nabla f(x_k + \alpha_k p_k)^{\mathrm{T}} p_k \geqslant c_2 \nabla f_k^{\mathrm{T}} p_k \,, \tag{5b}$$

with $0 < c_1 < c_2 < 1$ or Goldstein conditions

$$f(x_k) + (1-c)\alpha_k \nabla f_k^{\mathrm{T}} p_k \leqslant f(x_k + \alpha_k p_k) \leqslant f(x_k) + c\alpha_k \nabla f_k^{\mathrm{T}} p_k \tag{8}$$

with $0 < c < 1/2$.

§3.5 Step-Length Selection Algorithms

All line search procedures require an initial estimate α_0 and generate a sequence $\{\alpha_i\}$ that either terminates with a step length satisfying the conditions specified by the user (for example, the Wolfe conditions) or determines that such a step length does not exist. Typical procedures consist of two phases: a bracketing phase that finds an interval $[\bar{a}, \bar{b}]$ containing acceptable step lengths, and a selection phase that zooms in to locate the final step length.

In the following discussion we let α_k and α_{k-1} denote the step lengths used at iterations k and $k-1$ of the optimization algorithm, respectively. On the other hand, we denote the trial step lengths generated during the line search by α_i and α_{i-1} and also α_j. We use α_0 to denote the initial guess.

§3.5 Step-Length Selection Algorithms

All line search procedures require an initial estimate α_0 and generate a sequence $\{\alpha_i\}$ that either terminates with a step length satisfying the conditions specified by the user (for example, the Wolfe conditions) or determines that such a step length does not exist. Typical procedures consist of two phases: a bracketing phase that finds an interval $[\bar{a}, \bar{b}]$ containing acceptable step lengths, and a selection phase that zooms in to locate the final step length.

In the following discussion we let α_k and α_{k-1} denote the step lengths used at iterations k and $k-1$ of the optimization algorithm, respectively. On the other hand, we denote the trial step lengths generated during the line search by α_i and α_{i-1} and also α_j. We use α_0 to denote the initial guess.

§3.5 Step-Length Selection Algorithms

- **Interpolation**

We begin by describing a line search procedure based on interpolation of known function and derivative values of the function φ. This procedure can be viewed as an enhancement of Algorithm 3.1, the **Backtracking Line Search** algorithm. The aim is to find a value of α that satisfies the sufficient decrease condition (5a), without being "too small". Accordingly, the procedures here generate a decreasing sequence of values α_i such that each value α_i is not too much smaller than its predecessor α_{i-1}.

§3.5 Step-Length Selection Algorithms

Note that we can write the sufficient decrease condition, in the notation of $\varphi(\alpha) = f(x_k + \alpha p_k)$, as

$$\varphi(\alpha_k) \leqslant \varphi(0) + c_1 \alpha_k \varphi'(0)\,, \tag{37}$$

and that since the constant c_1 is usually chosen to be small in practice ($c_1 = 10^{-4}$, say), this condition asks for little more than descent in f. We design the procedure to be "efficient" in the sense that it computes the derivative $\nabla f(x)$ as few times as possible.

§3.5 Step-Length Selection Algorithms

Suppose that the initial guess α_0 is given. If we have

$$\varphi(\alpha_0) \leqslant \varphi(0) + c_1\alpha_0\varphi'(0)\,,$$

this step length satisfies the condition, and we terminate the search. Otherwise, we know that the interval $[0, \alpha_0]$ contains acceptable step lengths. We form a quadratic approximation $\varphi_q(\alpha)$ to φ by interpolating the three pieces of information available – $\varphi(0)$, $\varphi'(0)$, and $\varphi(\alpha_0)$ – to obtain

$$\varphi_q(\alpha) = \Big(\frac{\varphi(\alpha_0) - \varphi(0) - \alpha_0\varphi'(0)}{\alpha_0^2}\Big)\alpha^2 + \varphi'(0)\alpha + \varphi(0)\,. \qquad (38)$$

Note that this function is constructed so that it satisfies the interpolation conditions $\varphi_q(0) = \varphi(0), \varphi_q'(0) = \varphi'(0)$, and $\varphi_q(\alpha_0) = \varphi(\alpha_0)$.

§3.5 Step-Length Selection Algorithms

Suppose that the initial guess α_0 is given. If we have

$$\varphi(\alpha_0) \leqslant \varphi(0) + c_1\alpha_0\varphi'(0)\,,$$

this step length satisfies the condition, and we terminate the search. Otherwise, we know that the interval $[0, \alpha_0]$ contains acceptable step lengths. We form a quadratic approximation $\varphi_q(\alpha)$ to φ by interpolating the three pieces of information available – $\varphi(0)$, $\varphi'(0)$, and $\varphi(\alpha_0)$ – to obtain

$$\varphi_q(\alpha) = \Big(\frac{\varphi(\alpha_0) - \varphi(0) - \alpha_0\varphi'(0)}{\alpha_0^2}\Big)\alpha^2 + \varphi'(0)\alpha + \varphi(0)\,. \tag{38}$$

Note that this function is constructed so that it satisfies the interpolation conditions $\varphi_q(0) = \varphi(0), \varphi_q'(0) = \varphi'(0)$, and $\varphi_q(\alpha_0) = \varphi(\alpha_0)$.

§3.5 Step-Length Selection Algorithms

The new trial value α_1 is defined as the minimizer of this quadratic; that is, we obtain

$$\alpha_1 = -\frac{\varphi'(0)\alpha_0^2}{2[\varphi(\alpha_0) - \varphi(0) - \varphi'(0)\alpha_0]} . \tag{39}$$

We note that $0 < c_1 < \frac{1}{2}$ if and only if $\alpha_1 \in (0, \alpha_0)$.

If the sufficient decrease condition (37) is satisfied at α_1, we terminate the search. Otherwise, we construct a cubic function φ_c that interpolates the four pieces of information $\varphi(0)$, $\varphi'(0)$, $\varphi(\alpha_0)$, and $\varphi(\alpha_1)$, obtaining $\varphi_c(\alpha) = a\alpha^3 + b\alpha^2 + \alpha\varphi'(0) + \varphi(0)$, where

$$\begin{bmatrix} a \\ b \end{bmatrix} = \frac{1}{\alpha_0^2\alpha_1^2(\alpha_1 - \alpha_0)} \begin{bmatrix} \alpha_0^2 & -\alpha_1^2 \\ -\alpha_0^3 & \alpha_1^3 \end{bmatrix} \begin{bmatrix} \varphi(\alpha_1) - \varphi(0) - \alpha_1\varphi'(0) \\ \varphi(\alpha_0) - \varphi(0) - \alpha_0\varphi'(0) \end{bmatrix} .$$

§3.5 Step-Length Selection Algorithms

The new trial value α_1 is defined as the minimizer of this quadratic; that is, we obtain

$$\alpha_1 = -\frac{\varphi'(0)\alpha_0^2}{2[\varphi(\alpha_0) - \varphi(0) - \varphi'(0)\alpha_0]}\,. \tag{39}$$

We note that $0 < c_1 < \frac{1}{2}$ if and only if $\alpha_1 \in (0, \alpha_0)$.

If the sufficient decrease condition (37) is satisfied at α_1, we terminate the search. Otherwise, we construct a cubic function φ_c that interpolates the four pieces of information $\varphi(0)$, $\varphi'(0)$, $\varphi(\alpha_0)$, and $\varphi(\alpha_1)$, obtaining $\varphi_c(\alpha) = a\alpha^3 + b\alpha^2 + \alpha\varphi'(0) + \varphi(0)$, where

$$\begin{bmatrix} a \\ b \end{bmatrix} = \frac{1}{\alpha_0^2\alpha_1^2(\alpha_1 - \alpha_0)} \begin{bmatrix} \alpha_0^2 & -\alpha_1^2 \\ -\alpha_0^3 & \alpha_1^3 \end{bmatrix} \begin{bmatrix} \varphi(\alpha_0) - \varphi(0) - \alpha_0\varphi'(0) \\ \varphi(\alpha_1) - \varphi(0) - \alpha_1\varphi'(0) \end{bmatrix}.$$

§3.5 Step-Length Selection Algorithms

By differentiating $\varphi_c(x)$, we see that the minimizer α_2 of φ_c lies in the interval $[0, \alpha_1]$ and is given by

$$\alpha_2 = \frac{-b + \sqrt{b^2 - 3a\varphi'(0)}}{3a}.$$

If necessary, this process is repeated, using a cubic interpolant of $\varphi(0)$, $\varphi'(0)$ and the two most recent values of φ, until an α that satisfies (37) is located. If any α_i is either too close to its predecessor α_{i-1} or else too much smaller than α_{i-1}, we reset $\alpha_i = \alpha_{i-1}/2$. This safeguard procedure ensures that we make reasonable progress on each iteration and that the final α is not too small.

§3.5 Step-Length Selection Algorithms

By differentiating $\varphi_c(x)$, we see that the minimizer α_2 of φ_c lies in the interval $[0, \alpha_1]$ and is given by

$$\alpha_2 = \frac{-b + \sqrt{b^2 - 3a\varphi'(0)}}{3a} .$$

If necessary, this process is repeated, using a cubic interpolant of $\varphi(0)$, $\varphi'(0)$ and the two most recent values of φ, until an α that satisfies (37) is located. If any α_i is either too close to its predecessor α_{i-1} or else too much smaller than α_{i-1}, we reset $\alpha_i = \alpha_{i-1}/2$. This safeguard procedure ensures that we make reasonable progress on each iteration and that the final α is not too small.

§3.5 Step-Length Selection Algorithms

The strategy just described assumes that derivative values are significantly more expensive to compute than function values. It is often possible, however, to compute the directional derivative simultaneously with the function, at little additional cost; see Chapter 8. Accordingly, we can design an alternative strategy based on cubic interpolation of the values of φ and φ' at the two most recent values of α. Cubic interpolation provides a good model for functions with significant changes of curvature. Suppose we have an interval $[\bar{a}, \bar{b}]$ known to contain desirable step lengths, and two previous step length estimates α_{i-1} and α_i in this interval. We use a cubic function to interpolate $\varphi(\alpha_{i-1})$, $\varphi'(\alpha_{i-1})$, $\varphi(\alpha_i)$, and $\varphi'(\alpha_i)$. (This cubic function always exists and is unique; see, for example, Bulirsch and Stoer [41, p. 52].)

§3.5 Step-Length Selection Algorithms

The strategy just described assumes that derivative values are significantly more expensive to compute than function values. It is often possible, however, to compute the directional derivative simultaneously with the function, at little additional cost; see Chapter 8. Accordingly, we can design an alternative strategy based on cubic interpolation of the values of φ and φ' at the two most recent values of α. Cubic interpolation provides a good model for functions with significant changes of curvature. Suppose we have an interval $[\bar{a}, \bar{b}]$ known to contain desirable step lengths, and two previous step length estimates α_{i-1} and α_i in this interval. We use a cubic function to interpolate $\varphi(\alpha_{i-1})$, $\varphi'(\alpha_{i-1})$, $\varphi(\alpha_i)$, and $\varphi'(\alpha_i)$. (This cubic function always exists and is unique; see, for example, Bulirsch and Stoer [41, p. 52].)

§3.5 Step-Length Selection Algorithms

The minimizer of this cubic function in $[\bar{a}, \bar{b}]$ is either at one of the endpoints or else in the interior, in which case it is given by

$$\alpha_{i+1} = \alpha_i - (\alpha_i - \alpha_{i-1}) \left[\frac{\varphi'(\alpha_i) + d_2 - d_1}{\varphi'(\alpha_i) - \varphi'(\alpha_{i-1}) + 2d_2} \right], \qquad (40)$$

with

$$d_1 = \varphi'(\alpha_{i-1}) + \varphi'(\alpha_i) - 3\,\frac{\varphi(\alpha_{i-1}) - \varphi(\alpha_i)}{\alpha_{i-1} - \alpha_i},$$

$$d_2 = \mathrm{sign}(\alpha_i - \alpha_{i-1}) \sqrt{d_1^2 - \varphi'(\alpha_{i-1})\varphi'(\alpha_i)}\,.$$

§3.5 Step-Length Selection Algorithms

The interpolation process can be repeated by discarding the data at one of the step lengths α_{i-1} or α_i and replacing it by $\varphi(\alpha_{i+1})$ and $\varphi'(\alpha_{i+1})$. The decision on which of α_{i-1} and α_i should be kept and which discarded depends on the specific conditions used to terminate the line search; we discuss this issue further below in the context of the Wolfe conditions. Cubic interpolation is a powerful strategy, since it usually produces a quadratic rate of convergence of the iteration (40) to the minimizing value of α.

§3.5 Step-Length Selection Algorithms

The interpolation process can be repeated by discarding the data at one of the step lengths α_{i-1} or α_i and replacing it by $\varphi(\alpha_{i+1})$ and $\varphi'(\alpha_{i+1})$. The decision on which of α_{i-1} and α_i should be kept and which discarded depends on the specific conditions used to terminate the line search; we discuss this issue further below in the context of the Wolfe conditions. Cubic interpolation is a powerful strategy, since it usually produces a quadratic rate of convergence of the iteration (40) to the minimizing value of α.

§3.5 Step-Length Selection Algorithms

- **Initial Step Length**

For Newton and quasi-Newton methods, the step $\alpha_0 = 1$ should always be used as the initial trial step length. This choice ensures that unit step lengths are taken whenever they satisfy the termination conditions and allows the rapid rate-of-convergence properties of these methods to take effect. For methods that do not produce well scaled search directions, such as the steepest descent and conjugate gradient methods, it is important to use current information about the problem and the algorithm to make the initial guess. A popular strategy is to assume that the first-order change in the function at iterate x_k will be the same as that obtained at the previous step. In other words, we choose the initial guess α_0 so that $\alpha_0 \nabla f_k^{\mathrm{T}} p_k = \alpha_{k-1} \nabla f_{k-1}^{\mathrm{T}} p_{k-1}$; that is, $\alpha_0 = \alpha_{k-1} \dfrac{\nabla f_{k-1}^{\mathrm{T}} p_{k-1}}{\nabla f_k^{\mathrm{T}} p_k}$.

§3.5 Step-Length Selection Algorithms

- **Initial Step Length**

For Newton and quasi-Newton methods, the step $\alpha_0 = 1$ should always be used as the initial trial step length. This choice ensures that unit step lengths are taken whenever they satisfy the termination conditions and allows the rapid rate-of-convergence properties of these methods to take effect. For methods that do not produce well scaled search directions, such as the steepest descent and conjugate gradient methods, it is important to use current information about the problem and the algorithm to make the initial guess. A popular strategy is to assume that the first-order change in the function at iterate x_k will be the same as that obtained at the previous step. In other words, we choose the initial guess α_0 so that $\alpha_0 \nabla f_k^{\mathrm{T}} p_k = \alpha_{k-1} \nabla f_{k-1}^{\mathrm{T}} p_{k-1}$; that is, $\alpha_0 = \alpha_{k-1} \dfrac{\nabla f_{k-1}^{\mathrm{T}} p_{k-1}}{\nabla f_k^{\mathrm{T}} p_k}$.

§3.5 Step-Length Selection Algorithms

- **Initial Step Length**

For Newton and quasi-Newton methods, the step $\alpha_0 = 1$ should always be used as the initial trial step length. This choice ensures that unit step lengths are taken whenever they satisfy the termination conditions and allows the rapid rate-of-convergence properties of these methods to take effect. For methods that do not produce well scaled search directions, such as the steepest descent and conjugate gradient methods, it is important to use current information about the problem and the algorithm to make the initial guess. A popular strategy is to assume that the first-order change in the function at iterate x_k will be the same as that obtained at the previous step. In other words, we choose the initial guess α_0 so that $\alpha_0 \nabla f_k^{\mathrm{T}} p_k = \alpha_{k-1} \nabla f_{k-1}^{\mathrm{T}} p_{k-1}$; that is, $\alpha_0 = \alpha_{k-1} \dfrac{\nabla f_{k-1}^{\mathrm{T}} p_{k-1}}{\nabla f_k^{\mathrm{T}} p_k}$.

§3.5 Step-Length Selection Algorithms

Don't know what this slide is about!!!

Another useful strategy is to interpolate a quadratic to the data $f(x_{k-1})$, $f(x_k)$, and $\nabla f_{k-1}^{\mathrm{T}} p_{k-1}$ and to define α_0 to be its minimizer. This strategy yields

$$\alpha_0 = \frac{2(f_k - f_{k-1})}{\varphi'(0)}. \tag{41}$$

It can be shown that if $x_k \to x_*$ superlinearly, then the ratio in this expression converges to 1. If we adjust the choice (41) by setting $\alpha_0 \leftarrow \min(1, 1.01\alpha_0)$, we find that the unit step length $\alpha_0 = 1$ will eventually always be tried and accepted, and the superlinear convergence properties of Newton and quasi-Newton methods will be observed.

§3.5 Step-Length Selection Algorithms

- **A Line Search Algorithm for the Wolfe Conditions**

The Wolfe (or strong Wolfe) conditions are among the most widely applicable and useful termination conditions. We now describe in some detail a one-dimensional search procedure that is guaranteed to find a step length satisfying the **strong** Wolfe conditions

$$f(x_k + \alpha_k p_k) \leqslant f(x_k) + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \,, \tag{6a}$$

$$\left|\nabla f(x_k + \alpha_k p_k)^{\mathrm{T}} p_k\right| \geqslant c_2 \left|\nabla f_k^{\mathrm{T}} p_k\right| \,, \tag{6b}$$

for any parameters c_1 and c_2 satisfying $0 < c_1 < c_2 < 1$. As before, we assume that p is a descent direction and that f is bounded from below along the direction p.

§3.5 Step-Length Selection Algorithms

The algorithm has two stages. This first stage begins with a trial estimate α_1, and keeps increasing it until it finds either an acceptable step length or an interval that brackets the desired step lengths. In the latter case, the second stage is invoked by calling a function called zoom (Algorithm 3.6, below), which successively decreases the size of the interval until an acceptable step length is identified.

A formal specification of the line search algorithm follows. We refer to (6a) as the sufficient decrease condition and to (6b) as the curvature condition. The parameter $\alpha_{\max}$ is a user-supplied bound on the maximum step length allowed. The line search algorithm terminates with α_* set to a step length that satisfies the strong Wolfe conditions.

§3.5 Step-Length Selection Algorithms

The algorithm has two stages. This first stage begins with a trial estimate α_1, and keeps increasing it until it finds either an acceptable step length or an interval that brackets the desired step lengths. In the latter case, the second stage is invoked by calling a function called zoom (Algorithm 3.6, below), which successively decreases the size of the interval until an acceptable step length is identified.

A formal specification of the line search algorithm follows. We refer to (6a) as the sufficient decrease condition and to (6b) as the curvature condition. The parameter α_{max} is a user-supplied bound on the maximum step length allowed. The line search algorithm terminates with α_* set to a step length that satisfies the strong Wolfe conditions.

§3.5 Step-Length Selection Algorithms

Algorithm 3.5 (Line Search Algorithm).

Set $\alpha_0 \leftarrow 0$, choose $\alpha_{\max} > 0$ and $\alpha_1 \in (0, \alpha_{\max})$;

$i \leftarrow 1$;

repeat

  Evaluate $\varphi(\alpha_i)$;

  if $[\varphi(\alpha_i) > \varphi(0) + c_1 \alpha_i \varphi'(0)]$ or $[\varphi(\alpha_i) \geqslant \varphi(\alpha_{i-1})$ and $i > 1]$

    $\alpha_* \leftarrow \text{zoom}(\alpha_{i-1}, \alpha_i)$ and stop;

  Evaluate $\varphi'(\alpha_i)$;

  if $|\varphi'(\alpha_i)| \leqslant -c_2 \varphi'(0)$

    set $\alpha_* \leftarrow \alpha_i$ and stop;

  if $\varphi'(\alpha_i) \geqslant 0$

    set $\alpha_* \leftarrow \text{zoom}(\alpha_i, \alpha_{i-1})$ and stop;

  Choose $\alpha_{i+1} \in (\alpha_i, \alpha_{\max})$;

  $i \leftarrow i + 1$;

end (repeat)

§3.5 Step-Length Selection Algorithms

Note that the sequence of trial step lengths $\{\alpha_i\}$ is monotonically increasing, but that the order of the arguments supplied to the zoom function may vary. The procedure uses the knowledge that the interval (α_{i-1}, α_i) contains step lengths satisfying the strong Wolfe conditions if "one of the following three conditions is satisfied":

1. α_i violates the sufficient decrease condition;
2. $\varphi(\alpha_i) \geqslant \varphi(\alpha_{i-1})$;
3. α_i violates the curvature condition and $\varphi'(\alpha_i) \geqslant 0$.

The last step of the algorithm performs extrapolation to find the next trial value α_{i+1}. To implement this step we can use approaches like the interpolation procedures above, or we can simply set α_{i+1} to some constant multiple of α_i. Whichever strategy we use, it is important that the successive steps increase quickly enough to reach the upper limit $\alpha_{\max}$ in a finite number of iterations.

§3.5 Step-Length Selection Algorithms

Note that the sequence of trial step lengths $\{\alpha_i\}$ is monotonically increasing, but that the order of the arguments supplied to the zoom function may vary. The procedure uses the knowledge that the interval (α_{i-1}, α_i) contains step lengths satisfying the strong Wolfe conditions if "one of the following three conditions is satisfied":

1. α_i violates the sufficient decrease condition;
2. $\varphi(\alpha_i) \geqslant \varphi(\alpha_{i-1})$;
3. α_i violates the curvature condition and $\varphi'(\alpha_i) \geqslant 0$.

The last step of the algorithm performs extrapolation to find the next trial value α_{i+1}. To implement this step we can use approaches like the interpolation procedures above, or we can simply set α_{i+1} to some constant multiple of α_i. Whichever strategy we use, it is important that the successive steps increase quickly enough to reach the upper limit $\alpha_{\max}$ in a finite number of iterations.

§3.5 Step-Length Selection Algorithms

Note that the sequence of trial step lengths $\{\alpha_i\}$ is monotonically increasing, but that the order of the arguments supplied to the zoom function may vary. The procedure uses the knowledge that the interval (α_{i-1}, α_i) contains step lengths satisfying the strong Wolfe conditions if "one of the following three conditions is satisfied":

1. α_i violates the sufficient decrease condition;
2. $\varphi(\alpha_i) \geqslant \varphi(\alpha_{i-1})$;
3. α_i violates the curvature condition and $\varphi'(\alpha_i) \geqslant 0$.

The last step of the algorithm performs extrapolation to find the next trial value α_{i+1}. To implement this step we can use approaches like the interpolation procedures above, or we can simply set α_{i+1} to some constant multiple of α_i. Whichever strategy we use, it is important that the successive steps increase quickly enough to reach the upper limit $\alpha_{\max}$ in a finite number of iterations.

§3.5 Step-Length Selection Algorithms

We now specify the function zoom, which requires a little explanation. The order of its input arguments is such that each call has the form zoom$(\alpha_{\text{lo}}, \alpha_{\text{hi}})$, where

ⓐ the interval bounded by α_{lo} and α_{hi} contains step lengths that satisfy the strong Wolfe conditions;

ⓑ α_{lo} is, among all step lengths generated so far and satisfying the sufficient decrease condition, the one giving the smallest function value; and

ⓒ α_{hi} is chosen so that $\varphi'(\alpha_{\text{lo}})(\alpha_{\text{hi}} - \alpha_{\text{lo}}) < 0$.

Each iteration of zoom generates an iterate α_j between α_{lo} and α_{hi}, and then replaces one of these endpoints by α_j in such a way that the properties ⓐ, ⓑ, and ⓒ continue to hold.

§3.5 Step-Length Selection Algorithms

Algorithm 3.6 (zoom).

repeat

Interpolate (using quadratic, cubic, or bisection) to find a trial step length α_j between α_{lo} and α_{hi};

Evaluate $\varphi(\alpha_j)$;

if $[\varphi(\alpha_j) > \varphi(0) + c_1\alpha_j\varphi'(0)]$ or $[\varphi(\alpha_j) \geqslant \varphi(\alpha_{\text{lo}})]$

$\alpha_{\text{hi}} \leftarrow \alpha_j$;

else

Evaluate $\varphi'(\alpha_j)$;

if $|\varphi'(\alpha_j)| \leqslant -c_2\varphi'(0)$

Set $\alpha_* \leftarrow \alpha_j$ and stop;

if $\varphi'(\alpha_j)(\alpha_{\text{hi}} - \alpha_{\text{lo}}) \geqslant 0$

$\alpha_{\text{hi}} \leftarrow \alpha_{\text{lo}}$

$\alpha_{\text{lo}} \leftarrow \alpha_j$;

end (repeat)

§3.5 Step-Length Selection Algorithms

If the new estimate α_j happens to satisfy the strong Wolfe conditions, then zoom has served its purpose of identifying such a point, so it terminates with $\alpha_* = \alpha_j$. Otherwise, if α_j satisfies the sufficient decrease condition and has a lower function value than $\alpha_{\rm lo}$, then we set $\alpha_{\rm lo} \leftarrow \alpha_j$ to maintain condition ⓑ. If this setting results in a violation of condition ⓒ, we remedy the situation by setting $\alpha_{\rm hi}$ to the old value of $\alpha_{\rm lo}$. Readers should sketch some graphs to see for themselves how zoom works!

§3.5 Step-Length Selection Algorithms

One may ask how much more expensive it is to require the strong Wolfe conditions instead of the regular Wolfe conditions. Our experience suggests that for a "loose" line search (with parameters such as $c_1 = 10^{-4}$ and $c_2 = 0.9$), both strategies require a similar amount of work. The strong Wolfe conditions have the advantage that by decreasing c_2 we can directly control the quality of the search, by forcing the accepted value of α to lie closer to a local minimum. This feature is important in steepest descent or nonlinear conjugate gradient methods, and therefore a step selection routine that enforces the strong Wolfe conditions has wide applicability.

§3.5 Step-Length Selection Algorithms

One may ask how much more expensive it is to require the strong Wolfe conditions instead of the regular Wolfe conditions. Our experience suggests that for a "loose" line search (with parameters such as $c_1 = 10^{-4}$ and $c_2 = 0.9$), both strategies require a similar amount of work. The strong Wolfe conditions have the advantage that by decreasing c_2 we can directly control the quality of the search, by forcing the accepted value of α to lie closer to a local minimum. This feature is important in steepest descent or nonlinear conjugate gradient methods, and therefore a step selection routine that enforces the strong Wolfe conditions has wide applicability.