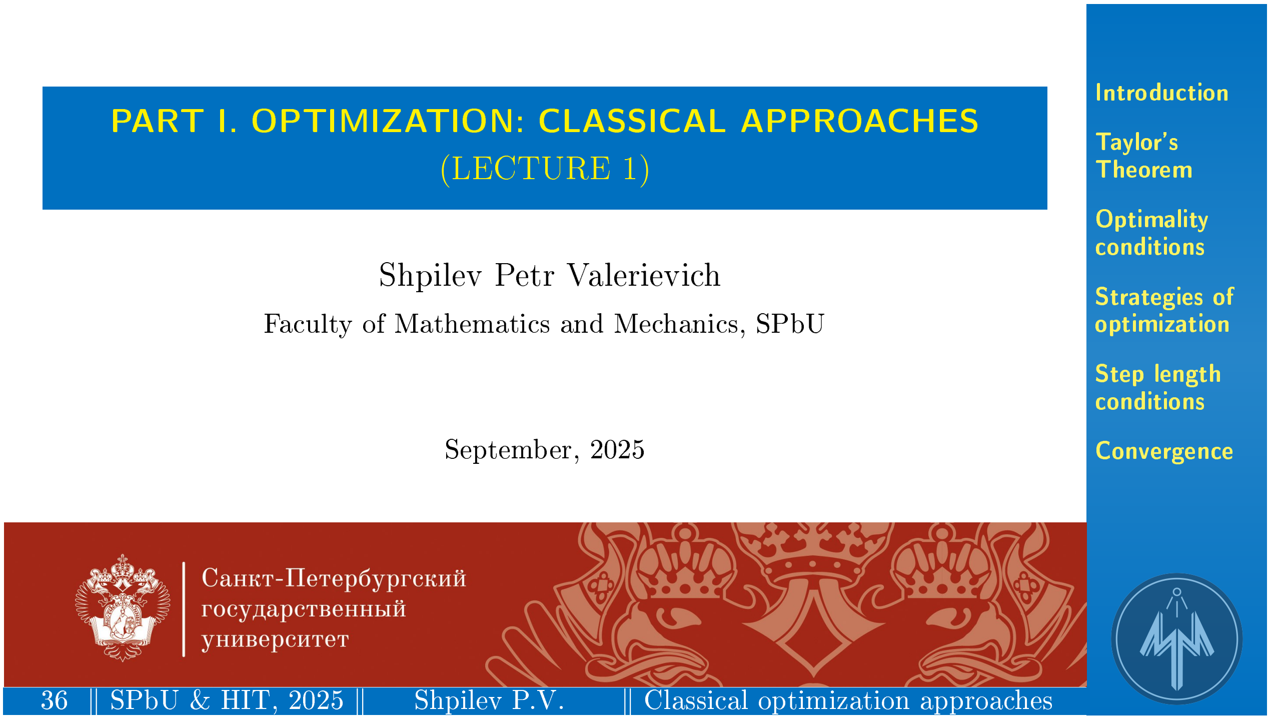


PART I. OPTIMIZATION: CLASSICAL APPROACHES (LECTURE 1)

Shpilev Petr Valerievich

Faculty of Mathematics and Mechanics, SPbU

September, 2025

Санкт-Петербургский
государственный
университет

Introduction

Taylor's Theorem

Optimality conditions

Strategies of optimization

Step length conditions

Convergence

Comments

This is the first lecture of the first part of the our course, which is called **"Optimization: classical approaches"**. We will look at the basic concepts and results, as well as the key principles of approaches to solving unconstrained optimization problems.

Unconstrained Optimization Problem

Introduction
Taylor's Theorem
Optimality conditions
Strategies of optimization
Step length conditions
Convergence

Standard Formulation

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{or} \quad f(x) \to \min_{x \in \mathbb{R}^n}$$

- $f : \mathbb{R}^n \to \mathbb{R}$ — smooth (continuously differentiable) scalar function.
- No constraints on the variable x: the feasible set is $\mathbb{R}^n$.
- Such problems arise in physics (energy minimization), economics (utility optimization), machine learning (loss minimization), etc.

Objective

Find a point x^* such that $f(x^*) \leq f(x)$ for all x in R^n.

Comments

Let's begin with the most basic formulation of an optimization problem.

On the slide, you see the standard mathematical model for an unconstrained optimization problem. We are looking to minimize a scalar function of several variables, which we denote as $f(x)$, where x is a vector in $\mathbb{R}^n$.

The key point here is that there are no constraints on the variable x. That means we are free to search for the minimum of the function over the entire real space, without any additional conditions or restrictions.

The function f is assumed to be continuously differentiable — in other words, it has derivatives that are continuous — because these derivatives will be crucial when we discuss optimality conditions and develop numerical algorithms.

Problems of this type appear frequently in applications. For example, in physics, we often want to minimize the potential energy of a system; in economics, we may want to maximize utility, which we can convert to a minimization problem; and in machine learning, we typically minimize a loss function.

Our goal is to find a point — let's call it x^* — such that $f(x^*) \leq f(x)$ for all x in R^n.

Local vs Global Minimum

Global Minimum

A point $x^* \in \mathbb{R}^n$ is a global minimum of f if

$$f(x^*) \leq f(x) \quad \text{for all } x \in \mathbb{R}^n$$

Local Minimum

A point $x^* \in \mathbb{R}^n$ is a local minimum of f if there exists $\varepsilon > 0$ such that

$$f(x^*) \leq f(x) \quad \text{for all } x \text{ with } \|x - x^*\| < \varepsilon$$

- Every global minimum is local, but not vice versa.
- Global minimization is significantly more challenging in general.

Comments

For the sake of completeness we will give a rigorous mathematical definition of global and local minimums and begin discussing the problems associated with finding them.

A point x^* is called a global minimum of the function f if $f(x^*) \leq f(x)$ for all $x \in \mathbb{R}^n$. By contrast, a local minimum requires that this condition holds only within some neighborhood of x^*, i.e. for all x with $\|x - x^*\| < \varepsilon$.

Every global minimum is also a local minimum, but not every local minimum is global — especially in the case of non-convex functions. In the general case, the task of finding the global extremum turns out to be significantly more difficult than the task of finding the local one. Global optimization methods, as a rule, require additional assumptions or the presence of structural properties of the function under study, such as convexity, for example.

Taylor's Theorem

Theorem 1 (Taylor's Theorem)

Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable and $p \in \mathbb{R}^n$. Then:

$$f(x+p) = f(x) + \nabla f(x+tp)^T p \quad \text{for some } t \in (0,1).$$

If f is twice continuously differentiable, then:

$$\nabla f(x+tp) = \nabla f(x) + \int_0^1 \nabla^2 f(x+tp)p\,dt$$

and then

$$f(x+p) = f(x) + \nabla f(x)^T p + \frac{1}{2} p^T \nabla^2 f(x+tp) p \quad \text{for some } t \in (0,1).$$

- ∇f is the gradient; $\nabla^2 f$ is the Hessian matrix.
- The remainder is expressed via evaluation at an intermediate point.

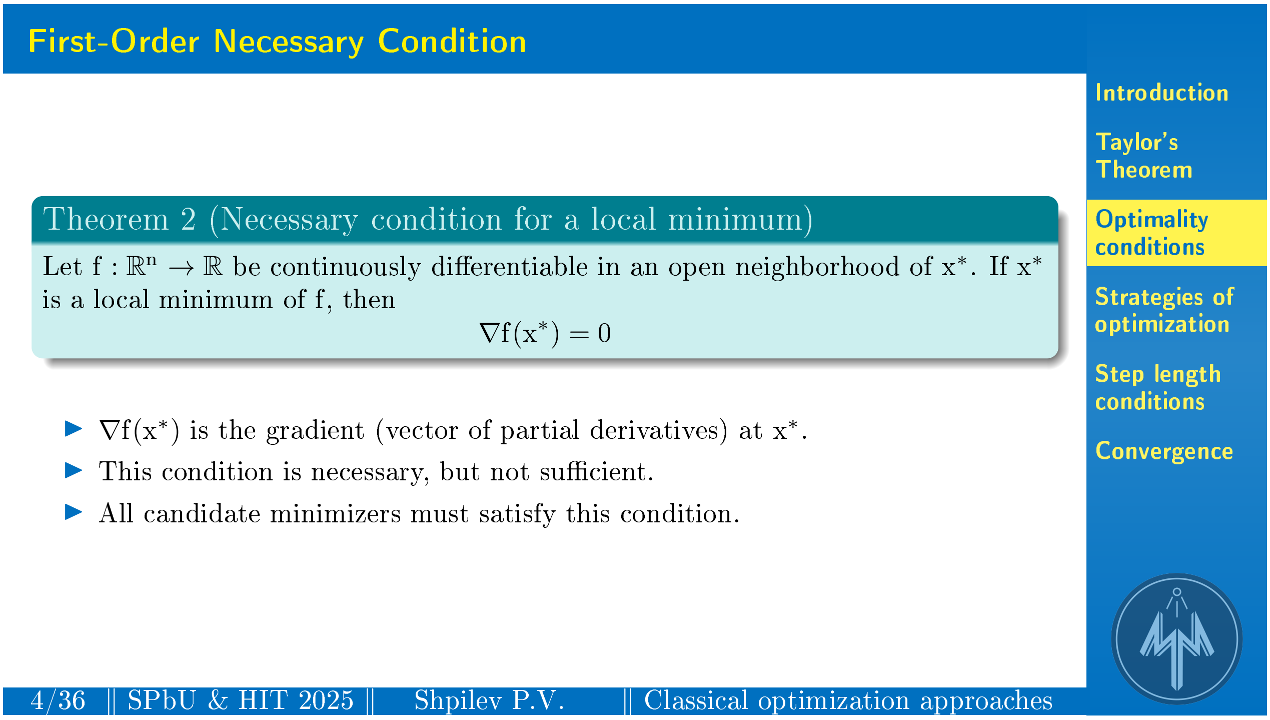
Comments

Let us now recall Taylor's theorem in a form convenient for analyzing optimality conditions. Let the function f be continuously differentiable, and let p be an arbitrary vector in $\mathbb{R}^n$.

Then, for some $t \in (0,1)$, we can write: $f(x+p) = f(x) + \nabla f(x+tp)^T p$.

If the function f is twice continuously differentiable, this expansion can be extended by adding a second-order term. Specifically, for some $t \in (0,1)$: $f(x+p) = f(x) + \nabla f(x)^T p + \frac{1}{2} p^T \nabla^2 f(x+tp) p$.

This is a form of Taylor's theorem with remainder in Lagrange form. It is this form that we will use in several upcoming proofs — in particular when establishing first and second-order conditions for optimality. The proof of this theorem can be found in any good textbook on mathematical analysis.



First-Order Necessary Condition

Theorem 2 (Necessary condition for a local minimum)

Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable in an open neighborhood of x^*. If x^* is a local minimum of f, then

$$\nabla f(x^*) = 0$$

- $\nabla f(x^*)$ is the gradient (vector of partial derivatives) at x^*.
- This condition is necessary, but not sufficient.
- All candidate minimizers must satisfy this condition.

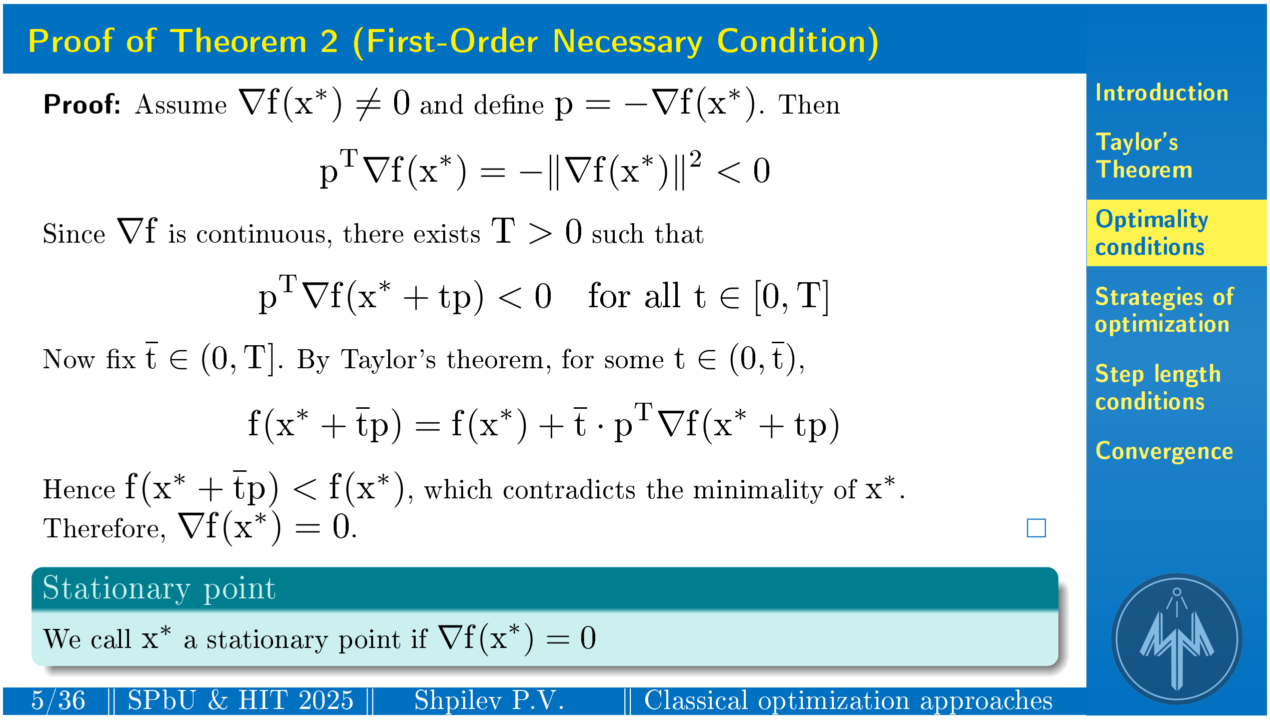
Comments

We now proceed to one of the key analytical tools in optimization — the first-order necessary condition for a local minimum.

Suppose the function f is continuously differentiable. Then, if a point x^* is a local minimum, the gradient of f at that point must be equal to zero: $\nabla f(x^*) = 0$. That is, all partial derivatives of f vanish at x^*.

Intuitively, this means the following: when we expand the function in a neighborhood of x^*, using a first-order approximation, there is no direction in which the value of the function decreases. More precisely, the directional derivative of f at x^* in any direction is zero.

This condition is called necessary because any local minimum must satisfy it. However, it is not sufficient: there may be points where the gradient is zero, but which are not minima — for example, maxima or saddle points. In practice, this condition allows us to identify candidate points for minima, which are then subject to further analysis — for instance, using second-order conditions.



Proof of Theorem 2 (First-Order Necessary Condition)

Proof: Assume $\nabla f(x^*) \neq 0$ and define $p = -\nabla f(x^*)$. Then

$$p^T \nabla f(x^*) = -\|\nabla f(x^*)\|^2 < 0$$

Since ∇f is continuous, there exists $T > 0$ such that

$$p^T \nabla f(x^* + tp) < 0 \quad \text{for all } t \in [0, T]$$

Now fix $\bar{t} \in (0, T]$. By Taylor's theorem, for some $t \in (0, \bar{t})$,

$$f(x^* + \bar{t}p) = f(x^*) + \bar{t} \cdot p^T \nabla f(x^* + tp)$$

Hence $f(x^* + \bar{t}p) < f(x^*)$, which contradicts the minimality of x^*.
Therefore, $\nabla f(x^*) = 0$. □

Stationary point

We call x^* a stationary point if $\nabla f(x^*) = 0$

Comments

Suppose the function f is continuously differentiable in an open neighborhood of a point x^*, and that x^* is a local minimizer.

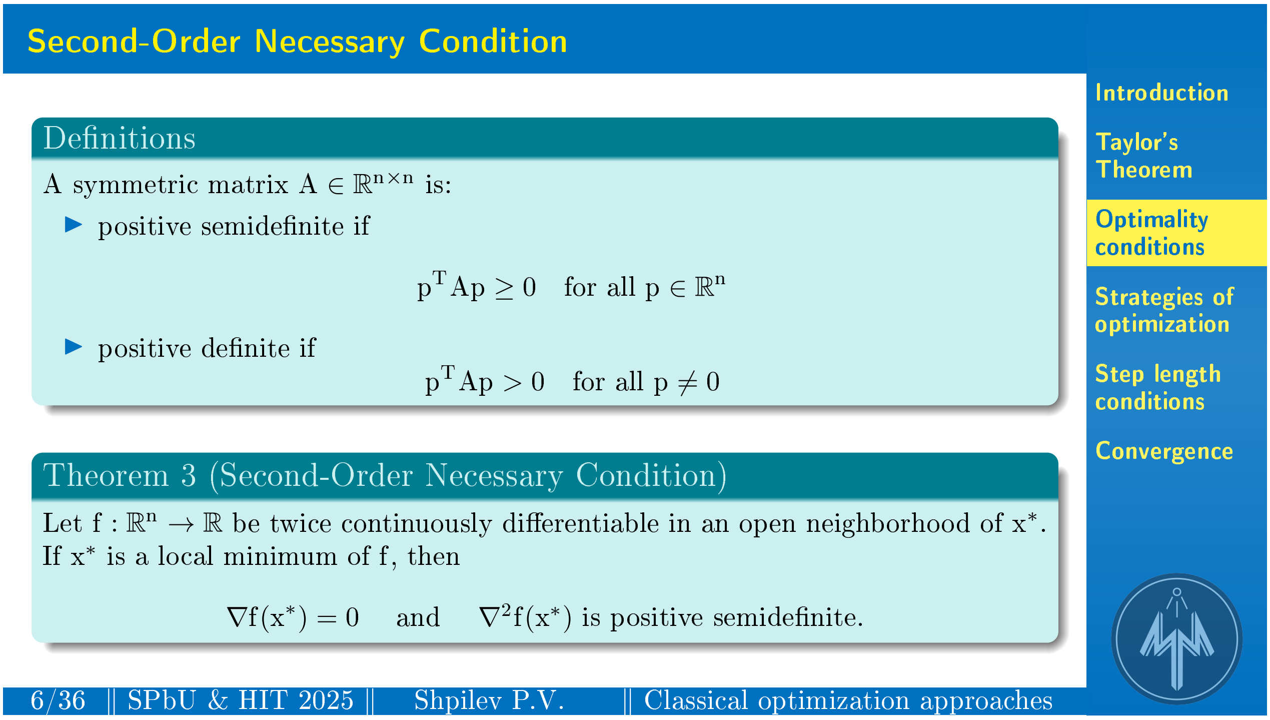
We argue by contradiction. Assume that $\nabla f(x^*) \neq 0$. Then we define the vector $p = -\nabla f(x^*)$. The scalar product is then $p^T \nabla f(x^*) = -\|\nabla f(x^*)\|^2 < 0$.

Since the gradient is continuous, we can choose a small enough $T > 0$ such that for all $t \in [0, T]$, the scalar product $p^T \nabla f(x^* + tp) < 0$.

Now, take any $\bar{t} \in (0, T]$. Applying Taylor's theorem we can write $f(x^* + \bar{t}p) = f(x^*) + \bar{t} \cdot p^T \nabla f(x^* + tp)$ for some $t \in (0, \bar{t})$.

Since $p^T \nabla f(x^* + tp)$ is negative, we get $f(x^* + \bar{t}p) < f(x^*)$ — contradicting the assumption that x^* is a local minimum. Hence, the assumption was false, and the gradient at x^* must be zero: $\nabla f(x^*) = 0$.

We call x^* a stationary point if $\nabla f(x^*) = 0$.



Second-Order Necessary Condition

Definitions

A symmetric matrix $A \in \mathbb{R}^{n \times n}$ is:

- positive semidefinite if

$$p^T A p \geq 0 \quad \text{for all } p \in \mathbb{R}^n$$

- positive definite if

$$p^T A p > 0 \quad \text{for all } p \neq 0$$

Theorem 3 (Second-Order Necessary Condition)

Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable in an open neighborhood of x^*. If x^* is a local minimum of f, then

$$\nabla f(x^*) = 0 \quad \text{and} \quad \nabla^2 f(x^*) \text{ is positive semidefinite.}$$

Comments

Before stating the theorem, let us recall the standard definitions of positive definite and positive semidefinite matrices. A symmetric matrix A is called positive semidefinite if $p^T A p \geq 0$ for all $p \in \mathbb{R}^n$. The matrix is positive definite if $p^T A p > 0$ for all $p \neq 0$.

Now we can formulate the second-order necessary condition for a local minimum. Suppose that the function f is twice continuously differentiable in an open neighborhood of a point x^*, and that x^* is a local minimum. Then two conditions must hold: $\nabla f(x^*) = 0$ and the Hessian matrix $\nabla^2 f(x^*)$ is positive semidefinite.

These two conditions — vanishing of the gradient and nonnegativity of the second-order term — together form the second-order necessary condition.

Proof of Theorem 3 (Second-Order Necessary Condition)

Proof. Let x^* be a local minimum, and let f be twice continuously differentiable in an open neighborhood of x^*. Then $\nabla f(x^*) = 0$ (by the first-order condition). Fix any $p \in \mathbb{R}^n$ and define $\varphi(\alpha) = f(x^* + \alpha p)$. From Taylor's theorem:

$$\varphi(\alpha) = f(x^*) + \alpha \nabla f(x^*)^T p + \frac{\alpha^2}{2} p^T \nabla^2 f(x^* + \theta\alpha p)p$$

for some $\theta \in (0, 1)$. Since $\nabla f(x^*) = 0$ and x^* is a minimum, we have

$$\varphi(\alpha) \geq \varphi(0) \quad \text{for small } \alpha > 0$$

This implies:

$$p^T \nabla^2 f(x^* + \theta\alpha p)p \geq 0$$

Taking the limit as $\alpha \to 0$, continuity of $\nabla^2 f$ gives:

$$p^T \nabla^2 f(x^*)p \geq 0 \quad \text{for all } p \in \mathbb{R}^n$$

i.e. the Hessian at x^* ($\nabla^2 f(x^*)$) is positive semidefinite. □

Comments

Let us now prove the second-order necessary condition. Suppose that x^* is a local minimum, and that the function f is twice continuously differentiable in an open neighborhood of x^*.

From the first-order condition, we already know that $\nabla f(x^*) = 0$. Now fix an arbitrary direction p, and define an auxiliary function $\varphi(\alpha) = f(x^* + \alpha p)$.

From Taylor's theorem we have $\varphi(\alpha) = f(x^*) + \alpha \nabla f(x^*)^T p + \frac{\alpha^2}{2} p^T \nabla^2 f(x^* + \theta\alpha p)p$.

Since $\nabla f(x^*) = 0$ and x^* is a minimum, we must have $\varphi(\alpha) \geq \varphi(0)$ for small positive α. This implies that the quadratic term $p^T \nabla^2 f(x^* + \theta\alpha p)p \geq 0$. Passing to the limit as $\alpha \to 0$ and using the continuity of the Hessian, we conclude that $p^T \nabla^2 f(x^*)p \geq 0$ for all $p \in \mathbb{R}^n$.

This proves that the Hessian $\nabla^2 f(x^*)$ is positive semidefinite at the local minimum.

Second-Order Sufficient Condition

Theorem 4 (Second-Order Sufficient Condition)

Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable in an open neighborhood of x^*. Suppose $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite. Then x^* is a strict local minimum of f.

- Positive definiteness of the Hessian strengthens the second-order condition.
- A strict minimum means: $f(x) > f(x^*)$ for all x sufficiently close to x^* with $x \neq x^*$.

Comments

We now formulate the second-order sufficient condition for a strict local minimum. Let the function f be twice continuously differentiable in an open neighborhood of a point x^*.

Suppose that two conditions hold: $\nabla f(x^*) = 0$ and the Hessian matrix $\nabla^2 f(x^*)$ is positive definite. Then x^* is a strict local minimum of the function f.

In other words, $f(x) > f(x^*)$ for all points x sufficiently close to x^* (but not equal to it). This result strengthens the necessary condition from the previous slide. Positive definiteness of the Hessian guarantees that the second-order term in the Taylor expansion is strictly positive in all nonzero directions.

Proof of Theorem 4 (Second-Order Sufficient Condition)

Proof. Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable in an open neighborhood of x^*. Assume $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite. Fix any $p \in \mathbb{R}^n$ and consider $\varphi(\alpha) = f(x^* + \alpha p)$. By Taylor's theorem:

$$f(x^* + \alpha p) = f(x^*) + \frac{\alpha^2}{2} p^T \nabla^2 f(x^* + \theta\alpha p)p$$

for some $\theta \in (0, 1)$. Since $\nabla^2 f$ is continuous and positive definite at x^*, we have:

$$p^T \nabla^2 f(x^* + \theta\alpha p)p > 0 \quad \text{for all small } \alpha > 0$$

Hence,

$$f(x^* + \alpha p) > f(x^*) \quad \text{for all small } \alpha > 0$$

Thus x^* is a strict local minimum. □

Comments

Let us now prove the second-order sufficient condition for a strict local minimum. Suppose that the function f is twice continuously differentiable in an open neighborhood of a point x^*, and that $\nabla f(x^*) = 0$. Suppose also that the Hessian matrix $\nabla^2 f(x^*)$ is positive definite.

Fix any unit vector p, and define the function $\varphi(\alpha) = f(x^* + \alpha p)$. By Taylor's theorem, we have the expansion: $f(x^* + \alpha p) = f(x^*) + \frac{\alpha^2}{2} p^T \nabla^2 f(x^* + \theta\alpha p)p$.

Since the Hessian is continuous and positive definite at x^*, the quadratic form $p^T \nabla^2 f(x^* + \theta\alpha p)p > 0$ for all small positive α. Therefore, $f(x^* + \alpha p) > f(x^*)$ for all small positive α. This proves that x^* is a strict local minimum.

Global Minimizer in Convex Case

Theorem 5 (Global Minimum for Convex Functions)

Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex. Then any local minimizer x^* is also a global minimizer. If in addition f is differentiable, then any stationary point x^* with

$$\nabla f(x^*) = 0$$

is a global minimizer.

Proof. Suppose x^* is a local but not global minimizer. Then there exists $z \in \mathbb{R}^n$ such that

$$f(z) < f(x^*)$$

Define $x = \lambda z + (1 - \lambda)x^*$ for $\lambda \in (0, 1]$.
By convexity of f:

$$f(x) \leq \lambda f(z) + (1 - \lambda)f(x^*) < f(x^*)$$

But every neighborhood of x^* contains such x, contradicting the local minimality of x^*.

Comments

Let us now state an important result connecting convexity and global optimality. Suppose the function f is convex. Then any local minimizer of f is automatically a global minimizer.

Moreover, if f is differentiable, then any stationary point — that is, any point where $\nabla f(x^*) = 0$ — is a global minimizer. This result shows how convexity allows us to go beyond local analysis. The proof is by contradiction. Suppose that x^* is a local minimum but not a global one. Then there exists a point z such that $f(z) < f(x^*)$.

Now consider the convex combination $x = \lambda z + (1 - \lambda)x^*$, for some $\lambda \in (0, 1]$. By convexity of the function f, we have $f(x) \leq \lambda f(z) + (1 - \lambda)f(x^*) < f(x^*)$. But such a point x lies arbitrarily close to x^* — contradicting the assumption that x^* is a local minimum.

Therefore, x^* must be a global minimizer.

Proof of Theorem 5 (Stationary Point in Convex Case)

Proof. For the second part of the theorem, suppose that x^* is not a global minimizer and choose z as above. From convexity,

$$\nabla f(x^*)^T(z - x^*) = \frac{d}{d\lambda} f(x^* + \lambda(z - x^*))\Big|_{\lambda=0} = \lim_{\lambda \downarrow 0} \frac{f(x^* + \lambda(z - x^*)) - f(x^*)}{\lambda}$$

$$\leq \lim_{\lambda \downarrow 0} \frac{\lambda f(z) + (1 - \lambda) f(x^*) - f(x^*)}{\lambda} = f(z) - f(x^*) < 0$$

Hence, $\nabla f(x^*) \neq 0$, and so x^* is not a stationary point. □

Comments

Let us now prove the second part of the theorem: that is, we want to show that if the function f is convex and differentiable, then any stationary point is a global minimizer.

We proceed by contradiction. Suppose that x^* is not a global minimizer. Consider the point z introduced in the first part of the proof, such that $f(z) < f(x^*)$.

Now take the directional derivative of f at x^* in the direction of $(z - x^*)$. Since f is convex and differentiable, this directional derivative, $\nabla f(x^*)^T(z - x^*) \leq f(z) - f(x^*)$.

And since $f(z) - f(x^*) < 0$, the inner product $\nabla f(x^*)^T(z - x^*)$ is strictly less than zero. Therefore, the gradient $\nabla f(x^*) \neq 0$.

This contradicts the assumption that x^* is a stationary point, and the statement is proved. This result has a fundamental implication for optimization methods. In the convex and differentiable case, we do not need to distinguish between local and global minimizers — any point where the gradient vanishes is guaranteed to be a global minimum. This is why many algorithms for unconstrained optimization are built around the search for stationary points:under the right structural assumptions, such as convexity, they correspond to global solutions.

Overview of Algorithms

- All unconstrained optimization algorithms generate a sequence $\{x_k\}$ starting from an initial guess x_0 (or some set of initial points).
- At each iteration, they attempt to reduce the objective function $f(x)$ by using information at x_k and possibly also information from earlier iterates $x_0, \ldots, x_{k-1}$.
- Most classical algorithms follow one of two main strategies:
  - Line Search: Choose a direction p_k and a step length α_k to move along p_k.
  - Trust Region: Build a local model of $f(x)$ near x_k and minimize it within a restricted region.
- The search direction p_k may depend only on the current point or on previous iterates as well.

Comments

In the future, we will consider various algorithmic approaches to finding the minimum (or maximum) of a function. Let us briefly summarize how unconstrained optimization algorithms work in general. Every algorithm starts from an initial point (or a population of initial points), usually called x_0, and generates a sequence of iterates x_k. At each step, the algorithm uses information about the function at the current point — such as the value of f and possibly the gradient or Hessian — to decide where to move next. Most classical algorithms are based on one of two general ideas. The first is line search: we pick a direction p_k and move in that direction by choosing a step length α_k. The second is the trust-region approach: here, we build a local approximation of the function near the current point, and try to minimize this model inside a region — typically a ball or an ellipsoid — centered at x_k. The way we choose the direction p_k can depend either only on the current point, or also on earlier iterates. We'll look at examples of both in upcoming lectures.

Two Strategies: Line Search and Trust Region

Line Search Methods

- Choose a search direction p_k.
- Select a step length $\alpha_k > 0$ to reduce $f(x)$ along p_k:

$$\min_{\alpha > 0} f(x_k + \alpha p_k)$$

- Simpler step computation, but step size must be carefully chosen.

Trust Region Methods

- Construct a model $m_k(p)$ approximating $f(x_k + p)$.
- Solve a constrained subproblem:

$$\min_{p:\|p\| \leq \Delta_k} m_k(p)$$

- Step is accepted if it gives sufficient reduction; otherwise shrink Δ_k.

Comments

Let us now compare the two fundamental strategies for unconstrained optimization:line search methods and trust region methods. In line search methods, we first fix a direction — typically a descent direction — and then look for a suitable step length that decreases the function along this direction. The step length is found by solving, either exactly or approximately, a one-dimensional minimization problem. Then we select a new direction and repeat the process. Trust region methods, on the other hand, follow a different logic. We first define a neighborhood — the trust region — around the current point, and within this region we minimize a model function $m_k(p)$ that approximates f. The model is usually quadratic and based on local information such as the gradient and Hessian or their approximations. If the step computed from the model does not reduce the original function sufficiently, we shrink the trust region and try again. These two strategies differ not only in mechanics, but also in how they balance local approximation and global behavior.

Search Directions in Line Search Methods

Let's denote by f_k and ∇f_k the function and gradient values at the point x_k. In practice, several options are used for p_k:

- Steepest Descent: $p_k = -\nabla f_k$
  - Requires only gradient computation.
  - Simple but often slow.
- Newton Direction: $p_k = -[\nabla^2 f_k]^{-1} \nabla f_k$
  - Uses second-order information.
  - Fast near solution if $\nabla^2 f_k$ is positive definite.
- Quasi-Newton Direction: $p_k = -B_k^{-1} \nabla f_k$
  - B_k approximates Hessian at point x_k.
  - No need to compute second derivatives.
- Nonlinear Conjugate Gradient: $p_k = -\nabla f_k + \beta_k p_{k-1}$
  - Requires no matrices.
  - Efficient for large-scale problems.

Comments

Let us now look at different ways to choose the search direction p_k in line search methods. The most basic choice is the steepest descent direction, $p_k = -\nabla f_k$. It is easy to compute but usually converges slowly, especially in ill-conditioned problems. A more powerful alternative is the Newton direction, which uses the inverse of the Hessian matrix. This can lead to rapid convergence, especially near the solution, provided the Hessian $\nabla^2 f_k$ is positive definite. To avoid computing the full Hessian, quasi-Newton methods approximate it with a matrix B_k that is updated at each step. These methods offer a good compromise between speed and computational cost. Finally, the nonlinear conjugate gradient direction is often used in large-scale optimization.It combines the current gradient with the previous direction and avoids storing or inverting matrices altogether. All of these directions are used within the same basic line search framework, where the next iterate is computed by moving from x_k along p_k with some step length α_k.

Models for Trust-Region Methods

Trust-region methods solve:

$$\min_{p} m_k(p) = f_k + p^T \nabla f_k + \frac{1}{2} p^T B_k p \quad \text{s.t. } \|p\| \le \Delta_k$$

Typical model choices:

- Steepest descent model: $B_k = 0$, gives

$$p_k = -\Delta_k \frac{\nabla f_k}{\|\nabla f_k\|}$$

- Newton model: $B_k = \nabla^2 f_k$ (exact Hessian)
- Quasi-Newton model: B_k approximates $\nabla^2 f_k$

The trust-region constraint ensures existence of a solution, even if B_k is not positive definite.

Comments

Let us now take a closer look at the types of models used in trust-region methods. At each iteration, the algorithm builds a local quadratic model $m_k(p)$ of the objective function near the current point. This model includes a constant term f_k, a linear term $p^T \nabla f_k$, and a quadratic term $\frac{1}{2} p^T B_k p$ defined by a symmetric matrix B_k. We then minimize this model subject to a constraint: the step must remain within a region of radius Δ_k, called the trust region. When $B_k = 0$, we obtain a step that points in the direction of steepest descent, scaled to match the trust-region radius. More powerful models use second-order information. If B_k is the exact Hessian $\nabla^2 f_k$, we get the trust-region Newton. If B_k is built via quasi-Newton updates, we get the trust-region quasi-Newton method. Unlike line search, the trust-region formulation guarantees that a solution to the subproblem always exists (due to the constraint), even if the Hessian is not positive definite.

The Scaling Problem

Optimization performance can be sensitive to variable scaling.

- A poorly scaled problem has variables with very different magnitudes.
- This can distort level sets and hinder convergence of gradient-based methods.

Example

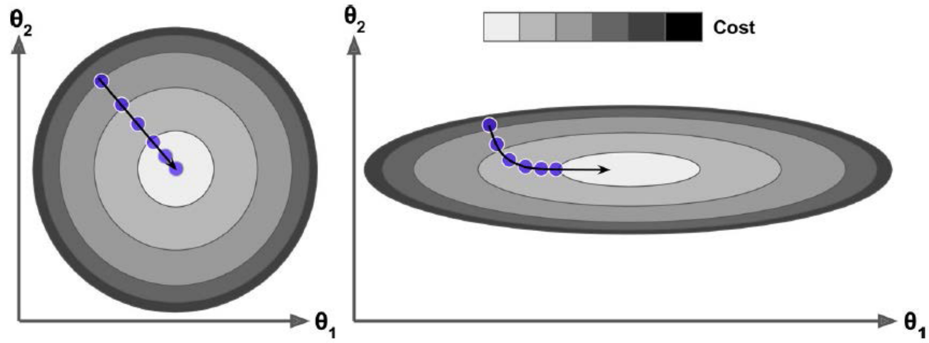
Minimize $f(x) = 100x_1^2 + x_2^2$ Level sets are highly elongated ellipses. Steepest descent zigzags and converges slowly.

Solution: Rescale variables so that typical values are of similar order. This can dramatically improve algorithm performance.

Comments

Let us now briefly discuss the scaling problem. Optimization algorithms can behave very differently depending on how the variables are scaled. In poorly scaled problems, some variables may take values that are orders of magnitude larger than others. This stretches and distorts the level sets of the function, turning circles into narrow ellipses. In such cases, gradient-based methods like steepest descent may perform very poorly. The algorithm tends to zigzag and make slow progress toward the solution. A typical example is the function $f(x) = 100x_1^2 + x_2^2$. Its level sets are narrow ellipses, and steepest descent struggles with it. The solution is to rescale the variables so that their magnitudes are more uniform. This simple change can make a big difference in convergence speed and reliability.

Effect of Scaling



- Balanced scaling yields nearly spherical level sets.
- Poor scaling results in elongated level sets.
- Gradient methods are sensitive to this distortion.

Proper scaling improves conditioning and accelerates convergence.

Comments

This diagram illustrates how variable scaling affects the geometry of the objective function. On the left, we see a well-scaled problem: the level sets are nearly circular, and the gradient consistently points toward the minimum. On the right, one variable is scaled much more than the other, causing the level sets to stretch into narrow ellipses. In such cases, gradient-based methods tend to make slow progress or zigzag across the valley. Intuitively, the objective function becomes distorted — instead of a well-shaped bowl, we get a flat dish with a shallow bottom. This effect is especially pronounced in methods like steepest descent and nonlinear conjugate gradient, which rely solely on the gradient direction. Newton-type methods, on the other hand, use the Hessian matrix and are generally more tolerant to poor scaling, since the curvature information helps adjust the step direction and length. Overall, optimization methods that are less sensitive to variable scaling tend to be more reliable in practice, as they can handle poorly scaled problems more robustly. When designing algorithms, one should aim to preserve scale invariance not just in the search direction, but also in components like the line search or trust-region strategy and stopping criteria. In general, achieving scale invariance is easier in line search methods than in trust-region methods.

Line Search Methods

Basic update:

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

Direction p_k is usually fixed (e.g., $-\nabla f_k$), but step length α_k must be carefully chosen:

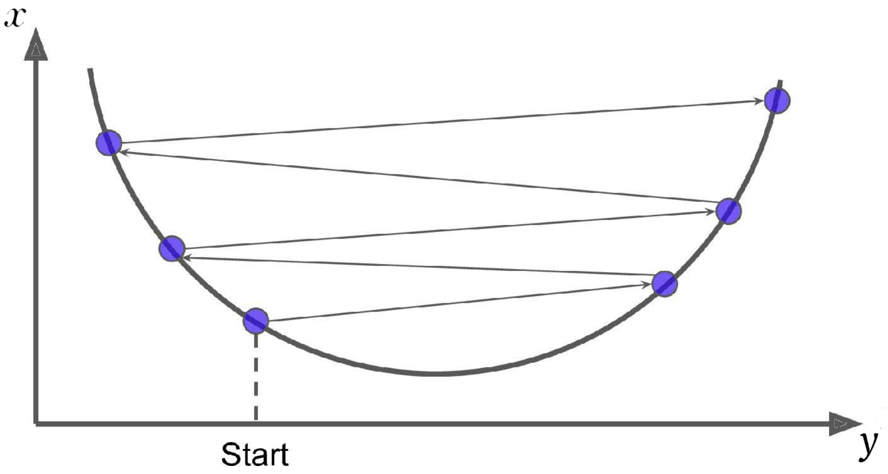
- Too large: the algorithm may overshoot the minimum or diverge.
- Too small: the progress becomes negligible and convergence slows down.
- Inappropriate steps can lead to instability, even with a good search direction.

A good step length balances sufficient decrease with stability.

Comments

Let us now focus on the role of the step length in line search methods. The search direction p_k is often determined by a specific rule, for example $p_k = -\nabla f_k$. But the choice of step length — that is, how far to move in that direction — is just as important. If the step is too large, the method may overshoot the region where the function decreases and fail to converge. If the step is too small, we make very little progress and waste many iterations. Even if the direction is well chosen, poor step sizes can lead to instability or stagnation. The goal is to strike a balance: to take steps that are not too small and not too aggressive,ensuring stable and consistent progress toward the solution.

Example: Divergence Due to Poor Step Length



Even when a descent direction is used, a poorly chosen step length may lead to divergence.

Comments

This figure illustrates a case where the optimization algorithm diverges. Although the search direction is a descent direction — for example, the negative gradient —the step length is chosen poorly: it is too large to stay within the region where the function decreases. As a result, the iterates move away from the solution. The function values may even start increasing. In some cases, this behavior leads to complete divergence — the algorithm never returns to the vicinity of the minimizer. This example emphasizes that it is not enough to choose a good direction — the step length must also be carefully controlled to ensure convergence.

Desirable Properties of Step Length

We aim to choose α_k such that:

- $f(x_k + \alpha_k p_k) < f(x_k)$ (descent)
- The decrease is not too small
- The search direction remains effective
- Convergence is ensured

However: Simple decrease is not sufficient for convergence. Counterexample:

Suppose the minimum value of $f(x)$ is -1, and define $f(x_k) = \frac{1}{k}$. Then $f(x_{k+1}) < f(x_k)$ for all k, but $f(x_k) \to 0 > -1$ as $k \to \infty$. Conclusion: To avoid this

behavior, we must require that the decrease in f is substantial enough, i.e. we need to enforce a sufficient decrease condition, which we will discuss next.

Comments

Let us now clarify what we expect from a good step length in line search methods. First and foremost, the step must produce a decrease in the objective function — that is, $f(x_k + \alpha_k p_k) < f(x_k)$. But decrease alone is not enough. We must also make sure that the decrease is not too small. Otherwise, we might waste many iterations making almost no progress. Furthermore, the choice of step length should support convergence in the long run. In particular, it must allow the iterates to approach an actual minimizer, not just wander around points with slightly decreasing values. These principles guide the design of practical step length rules. To illustrate why such care is necessary, let us look at a simple example. Suppose the minimum value of f is -1, and we define the value of f at each step as $f(x_k) = 1/k$. Then $f(x_{k+1}) < f(x_k)$ — the function decreases at every step. But the values converge to zero, which is still strictly greater than the minimum. So, the method never reaches the minimizer, even though the function seems to decrease. To avoid this behavior, we must require that the decrease in f is substantial enough, i.e., we need to enforce a sufficient decrease condition, which we will discuss next.

Sufficient Decrease Condition

We say that a step length $\alpha > 0$ satisfies the sufficient decrease condition if

$$f(x + \alpha p) \leq f(x) + c_1 \alpha \nabla f(x)^T p, \quad c_1 \in (0, 1)$$

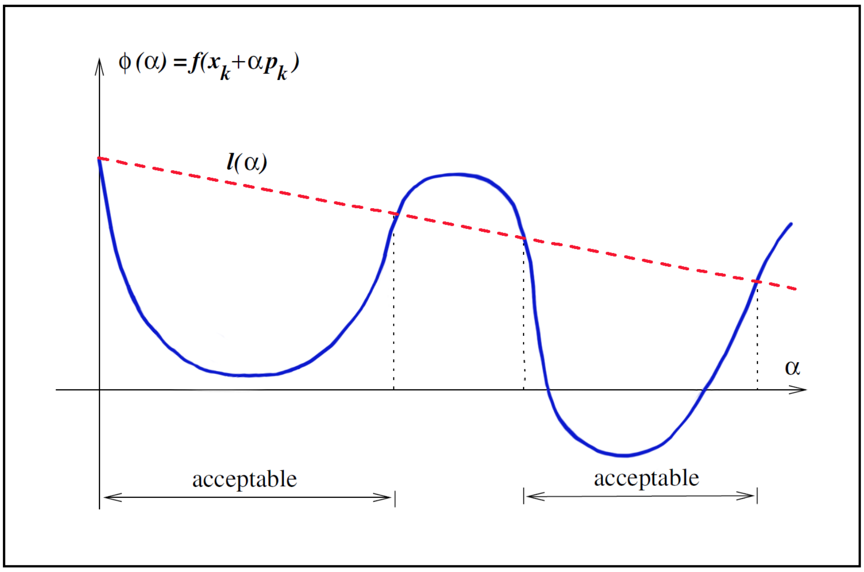
Interpretation:

- Ensures f decreases by at least a fixed fraction of the directional derivative.
- Prevents steps that produce only negligible improvement.
- Also known as the Armijo condition.

Comments

We now formally introduce the sufficient decrease condition, which ensures that the step we take leads to a meaningful reduction in the objective function. A step length α is said to satisfy this condition if $f(x + \alpha p) \leq f(x) + c_1 \alpha \nabla f(x)^T p$. The constant c_1 is fixed in the interval $(0, 1)$ and is typically chosen close to zero in practice — for example, 10^{-4}. This means that the function must decrease by at least a small but fixed portion of how much we expect it to decrease in the direction p. This condition is often referred to as the Armijo condition, and it provides the foundation for practical step length selection.

Illustration of the Sufficient Decrease Condition



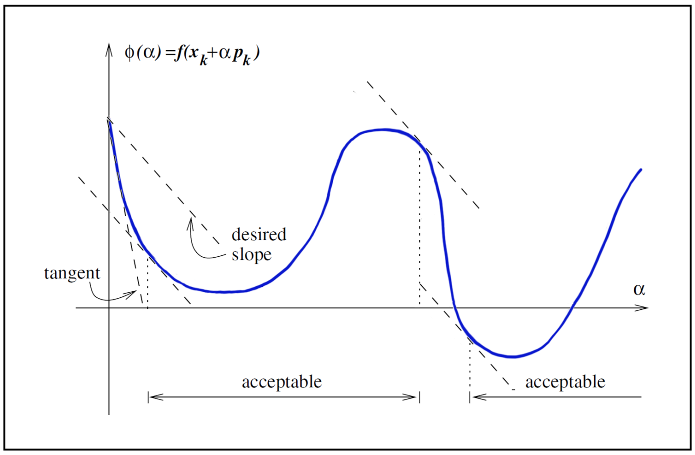
Comments

Let us now interpret the sufficient decrease condition geometrically. We consider the function $\phi(\alpha) = f(x_k + \alpha p_k)$. It is shown in blue on the graph. The dashed line represents the value $f(x_k) + c_1 \alpha \nabla f(x_k)^T p_k$. For small values of c_1 — which are typically chosen in practice — this line decreases slowly and remains close to horizontal. Thus, in the neighborhood of $\alpha = 0$, the function ϕ decreases more rapidly than this line. Therefore, the dashed line lies above the graph of ϕ, which corresponds to the requirement of the sufficient decrease condition. In essence, the condition means that the actual reduction in the function must be no less than a fixed fraction of the linear approximation to f along the direction p_k. In the figure, the regions of acceptable values of α — where this inequality is satisfied — are marked explicitly.

The Curvature Condition

- To prevent overly small steps, we impose an additional constraint: the curvature condition:

$$\nabla f(x_k + \alpha p_k)^T p_k \geq c_2 \nabla f(x_k)^T p_k, \text{ where } c_2 \in (c_1, 1).$$

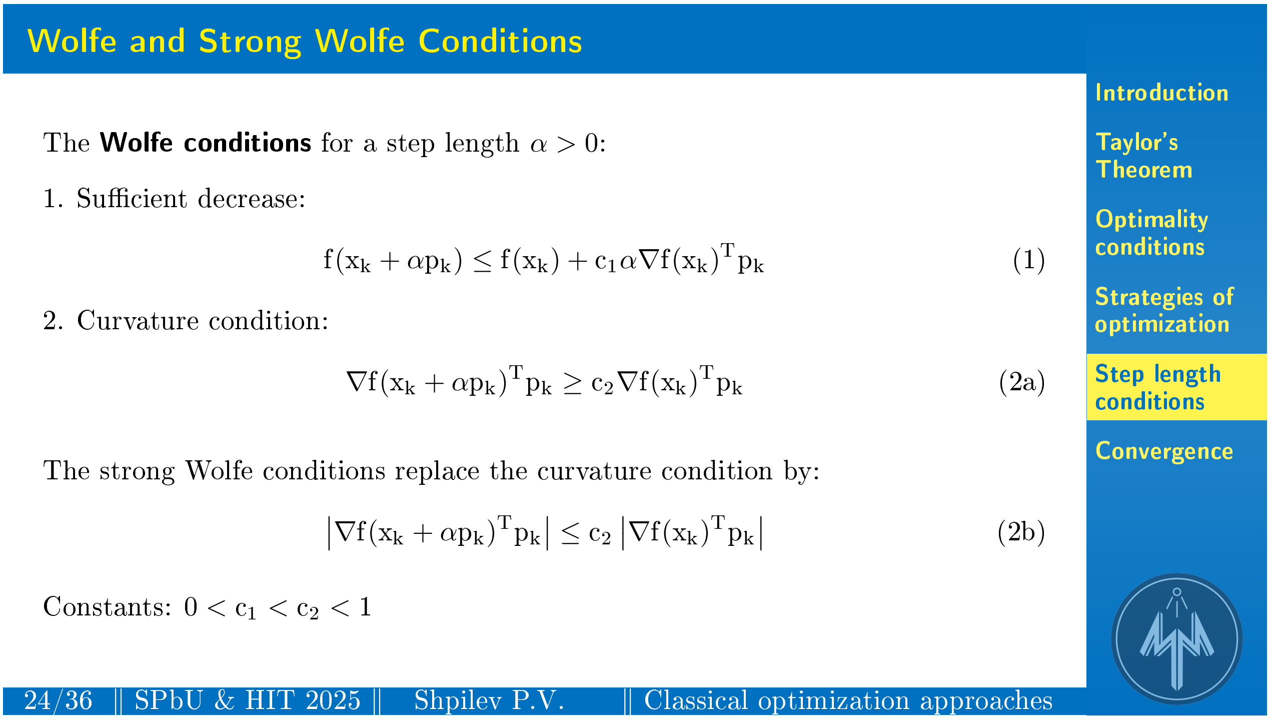


Typical values: $c_2 \approx 0.9$ for Newton-type methods, $c_2 \approx 0.1$ for conjugate gradient.

Comments

Let us now introduce the second condition used in line search algorithms — the curvature condition. As we have seen, the sufficient decrease condition alone is not enough. It may allow very small steps that formally reduce the function value but do not ensure real progress. The curvature condition aims to exclude such steps.

It requires that the directional derivative of the function f at the new point — that is, at $x_k + \alpha p_k$ — is greater than or equal to c_2 times the directional derivative of f at x_k. In other words, the slope of the function at the new point must not be too steep — it must be at least a fixed fraction of the initial slope. This ensures that the method does not stop while the function is still decreasing rapidly. Note that the left-hand side of the curvature condition is simply the derivative of the function ϕ at the current step length α — that is, $\phi'(\alpha)$. So this condition ensures that the slope of ϕ at α is no more negative than a fixed fraction of the initial slope at zero. This makes sense: if the slope is still sharply negative, the function may continue to decrease and a longer step would be preferable. On the other hand, if the slope is only mildly negative or has become positive, it indicates that no substantial decrease remains in that direction, and the line search should terminate. As we can see in the figure, the tangent at zero has a greater slope than the dashed line. Typical values for the constant c_2 are close to one in Newton-type methods, and much smaller — around 0.1 — in conjugate gradient methods.



Wolfe and Strong Wolfe Conditions

Introduction
Taylor's Theorem
Optimality conditions
Strategies of optimization
Step length conditions
Convergence

The **Wolfe conditions** for a step length $\alpha > 0$:

1. Sufficient decrease:

$$f(x_k + \alpha p_k) \leq f(x_k) + c_1 \alpha \nabla f(x_k)^T p_k \tag{1}$$

2. Curvature condition:

$$\nabla f(x_k + \alpha p_k)^T p_k \geq c_2 \nabla f(x_k)^T p_k \tag{2a}$$

The strong Wolfe conditions replace the curvature condition by:

$$\left|\nabla f(x_k + \alpha p_k)^T p_k\right| \leq c_2 \left|\nabla f(x_k)^T p_k\right| \tag{2b}$$

Constants: $0 < c_1 < c_2 < 1$

Comments

The sufficient decrease and curvature conditions are known collectively as the Wolfe conditions. A step length α satisfies the Wolfe conditions if, first, the value of the function at $x_k+\alpha p_k$ is less than or equal to the value at x_k plus $c_1\alpha$ times the directional derivative at x_k. This ensures a meaningful decrease in the objective function. Second, the directional derivative at the new point is greater than or equal to c_2 times the directional derivative at the current point. This prevents the step from being too short.

In many algorithms, especially those that do not assume convexity, it is more appropriate to use the strong Wolfe conditions. In that case, the curvature condition is replaced by requiring that the absolute value of the directional derivative at the new point is less than or equal to c_2 times the absolute value of the initial directional derivative. This allows the derivative to change sign near a local minimizer or stationary point, while still limiting how steep it can be. Hence, we exclude points that are far from stationary points of ϕ.

Inexact Line Search and Wolfe Conditions in Practice

Exact line search is rarely used in practice:

- Too expensive to minimize $f(x_k + \alpha p_k)$ exactly.
- Noisy or costly function evaluations in real-world problems.

Instead, inexact line search is used:

- Satisfy Wolfe or strong Wolfe conditions.
- Provide balance between decrease and progress.

Typical implementation:

- Start with $\alpha = 1$.
- Use backtracking or bracketing strategy.
- Adjust until Wolfe conditions are met.

Comments

In theory, one could choose the step length α by exactly minimizing the function $f(x_k + \alpha p_k)$. But in practice, this is almost never done. Exact line search is too computationally expensive, and in real-world applications, function evaluations can be noisy or expensive. Instead, most practical algorithms rely on inexact line search strategies. These strategies look for a step length α that satisfies the Wolfe or strong Wolfe conditions. This approach is more efficient and still guarantees convergence under reasonable assumptions. A typical implementation works as follows: we start with $\alpha = 1$, then reduce it using backtracking or expand and shrink a bracketed interval, adjusting α until both the sufficient decrease and the curvature conditions are satisfied. In the backtracking strategy, we repeatedly multiply α by a factor less than one —for example, one-half — until the conditions are met. This ensures that we quickly reduce the step size if the function is not decreasing enough. The result is a compromise: we don't insist on finding the best possible α, but we ensure that the function is decreasing enough, and that the step is not too short. This method is particularly effective in Newton-type methods, where the initial step is often close to acceptable. It is less suited for quasi-Newton and conjugate gradient methods, where step control requires more subtle handling.

Existence of Step lengths that satisfy the Wolfe conditions

Lemma 1 (Step lengths Existence)

Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable. Suppose p_k is a descent direction at x_k, and f is bounded below along the ray $\{x_k + \alpha p_k \mid \alpha > 0\}$. Then, for any

$0 < c_1 < c_2 < 1$, there exists an interval of step lengths α such that the Wolfe and strong Wolfe conditions are satisfied.

Proof. Define the function $\phi(\alpha) = f(x_k + \alpha p_k)$. Then $\phi(0) = f(x_k)$ and $\phi'(0) = \nabla f(x_k)^T p_k < 0$. We seek a value of $\alpha > 0$ that satisfies:

$$\phi(\alpha) \leq \phi(0) + c_1 \alpha \phi'(0), \quad \phi'(\alpha) \geq c_2 \phi'(0)$$

Such α exists by analysis of the function ϕ along the descent direction.

Comments

Let us now justify the use of Wolfe conditions in line search algorithms by proving that suitable step lengths actually exist under mild assumptions. The lemma on this slide guarantees that for any constants c_1 and c_2 between zero and one, we can always find a value of α that satisfies the Wolfe or strong Wolfe conditions — provided that the search direction is indeed a descent direction, and the function f is continuously differentiable and bounded below along the chosen ray. This result is important: it shows that the criteria we impose during line search are not overly restrictive and do not rule out the existence of a valid step. Now we begin the proof. We define a one-dimensional function $\phi(\alpha)$ as $f(x_k + \alpha p_k)$. Since p_k is a descent direction $\phi'(0) = \nabla f(x_k)^T p_k < 0$. This tells us that the function ϕ initially decreases, and we now seek a value of $\alpha > 0$ such that $\phi(\alpha)$ satisfies the sufficient decrease and curvature conditions. These two conditions can be rewritten in terms of ϕ, and we will now study the behavior of this function to show that they can be simultaneously satisfied.

Proof of Lemma 1 (Existence of Step Lengths)

Since ϕ is bounded below by assumption, while $\ell(\alpha) \to -\infty$ as $\alpha \to \infty$ (because $\phi'(0) < 0$ and $c_1 > 0$), the graphs of ϕ and ℓ must intersect: there exists $\alpha^* > 0$ such that

$$\phi(\alpha^*) = \ell(\alpha^*) \quad \text{and} \quad \phi(\alpha) < \ell(\alpha) \quad \text{for all } \alpha < \alpha^*.$$

By the mean value theorem, there exists $\alpha' \in (0, \alpha^*)$ such that

$$\phi(\alpha^*) = \phi(0) + \alpha^* \phi'(\alpha').$$

Combining this with $\ell(\alpha^*) = \phi(0) + c_1 \alpha^* \phi'(0)$, we get:

$$\phi'(\alpha') = c_1 \phi'(0).$$

Since $\phi'(0) < 0$ and $c_1 < c_2$, it follows that:

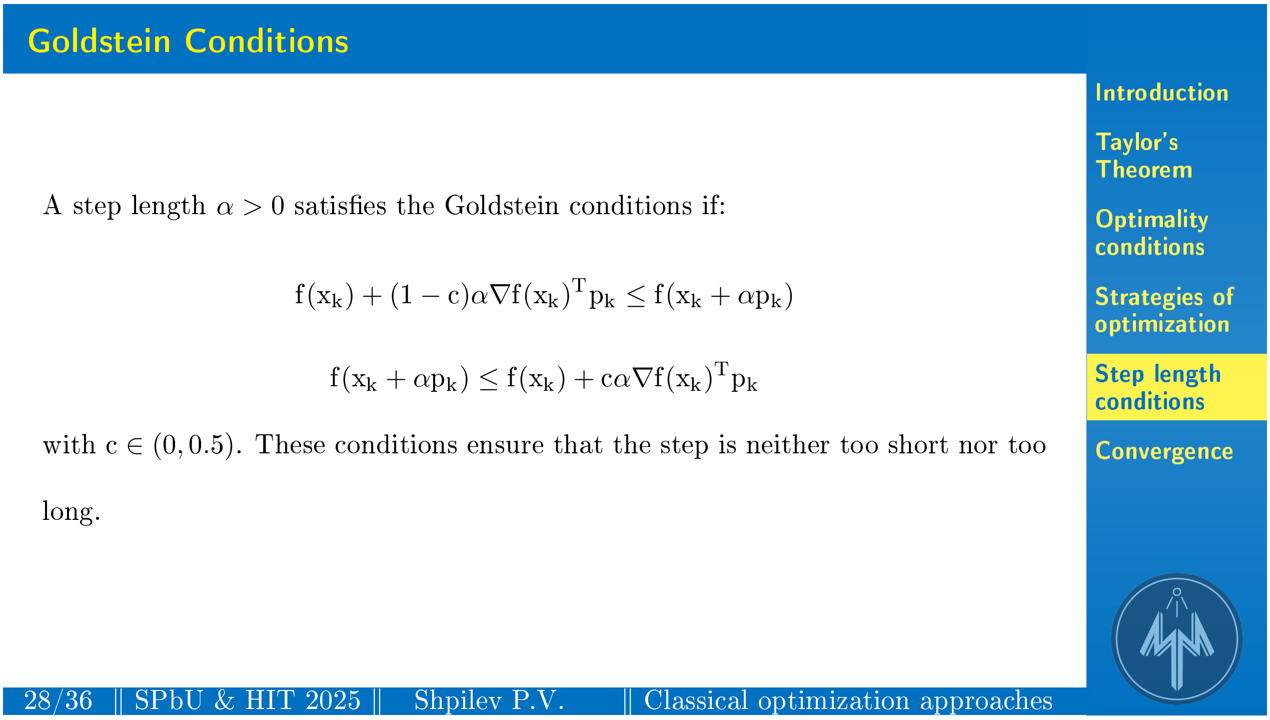
$$\phi'(\alpha') > c_2 \phi'(0).$$

Thus, α' satisfies both Wolfe conditions.

Comments

Now we will complete the proof of the lemma using the mean value theorem.

Since the function ϕ is bounded below, but the line ℓ tends to minus infinity due to the negative slope $\phi'(0)$, the two graphs must intersect at some point α^*. For all smaller α, ϕ lies strictly below ℓ. Applying the mean value theorem to the function ϕ on the interval from zero to α^*, we obtain some intermediate point α' where the derivative matches the slope of the chord. Because the line ℓ intersects ϕ at α^*, we equate the two expressions and find that the directional derivative of ϕ at α' is equal to $c_1\phi'(0)$. Since $c_1 < c_2$, and $\phi'(0)$ is negative, this derivative is strictly greater than $c_2\phi'(0)$. Therefore, α' satisfies both Wolfe conditions: sufficient decrease and the curvature condition. Since the derivative ϕ at α' is negative and $c_1 < c_2$, its absolute value is strictly less than c_2 times the absolute value of the initial derivative. Therefore, the strong Wolfe condition holds in the same interval as well.



Goldstein Conditions

A step length $\alpha > 0$ satisfies the Goldstein conditions if:

$$f(x_k) + (1-c)\alpha \nabla f(x_k)^T p_k \leq f(x_k + \alpha p_k)$$

$$f(x_k + \alpha p_k) \leq f(x_k) + c\alpha \nabla f(x_k)^T p_k$$

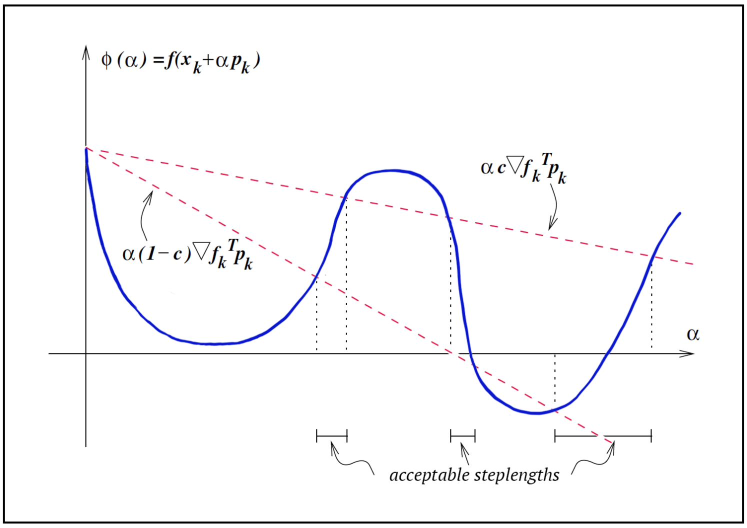
with $c \in (0, 0.5)$. These conditions ensure that the step is neither too short nor too long.

Comments

The Goldstein conditions define another pair of inequalities used to guide the choice of step lengths in line search. A step α satisfies these conditions if the value of the function at the trial point lies between two linear boundaries based on the initial value of the function and the directional derivative. The constant c must lie strictly between zero and one-half. The first inequality ensures that the step is not too short — that is, the function has decreased enough. The second inequality ensures that the step is not too long — meaning we have not overshot the region of useful decrease. Together, these bounds define an interval of acceptable step lengths centered around the optimal value predicted by linear approximation.

Goldstein Conditions — Illustration

Introduction
Taylor's Theorem
Optimality conditions
Strategies of optimization
Step length conditions
Convergence


$\phi(\alpha) = f(x_k + \alpha p_k)$
$\alpha c \nabla f_k^T p_k$
$\alpha(1-c)\nabla f_k^T p_k$
α
acceptable steplengths

The Goldstein conditions define an interval that avoids too-short and too-long steps, but may exclude points where the directional derivative vanishes.

Comments

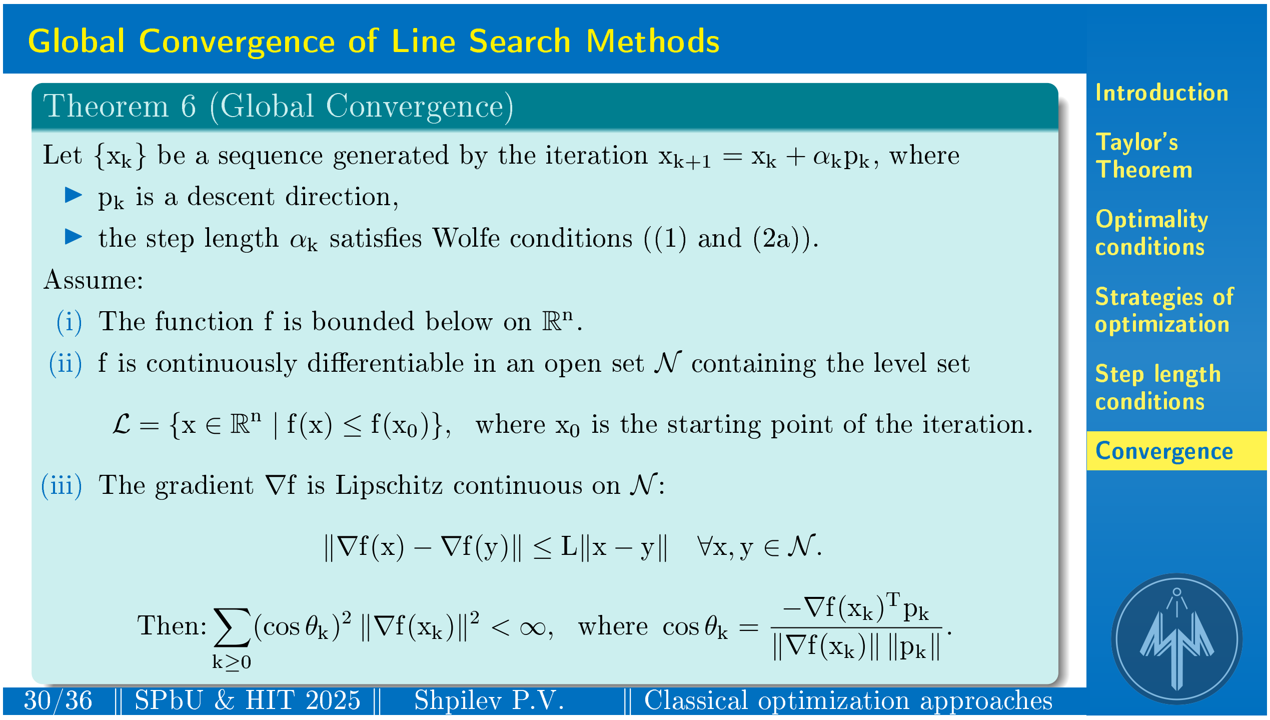
This figure illustrates the idea behind the Goldstein conditions. The step length α must be chosen so that the function value lies between two lines defined by the initial value and the directional derivative.

The resulting interval excludes steps that are either too short or too long. However, a known drawback is that this condition can eliminate points where the directional derivative is zero — that is, local minimizers of the function ϕ.

In contrast, the Wolfe conditions explicitly allow such points and are therefore more suitable for general-purpose optimization, especially in quasi-Newton methods where maintaining a positive definite Hessian approximation is essential.

Nonetheless, both Goldstein and Wolfe conditions share similar theoretical properties, and their convergence analysis is largely parallel.

The Goldstein conditions are often preferred in Newton-type methods where the curvature of the function is accurately captured by the Hessian.



Global Convergence of Line Search Methods

Theorem 6 (Global Convergence)

Let $\{x_k\}$ be a sequence generated by the iteration $x_{k+1} = x_k + \alpha_k p_k$, where

- p_k is a descent direction,
- the step length α_k satisfies Wolfe conditions ((1) and (2a)).

Assume:

(i) The function f is bounded below on $\mathbb{R}^n$.

(ii) f is continuously differentiable in an open set $\mathcal{N}$ containing the level set

$$\mathcal{L} = \{x \in \mathbb{R}^n \mid f(x) \le f(x_0)\}, \quad \text{where } x_0 \text{ is the starting point of the iteration.}$$

(iii) The gradient ∇f is Lipschitz continuous on $\mathcal{N}$:

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

$$\text{Then: } \sum_{k \ge 0} (\cos\theta_k)^2 \, \|\nabla f(x_k)\|^2 < \infty, \quad \text{where } \cos\theta_k = \frac{-\nabla f(x_k)^T p_k}{\|\nabla f(x_k)\| \, \|p_k\|}.$$

Comments

We now state a result that concerns the convergence of line search methods in a global sense. That is, it guarantees that, regardless of the starting point x_0, and under certain general conditions on the function, its gradient, and the step lengths, the method will converge toward a stationary point.

More precisely, let's consider a sequence of iterates defined by the standard scheme: each new point is obtained by moving from the previous one along some search direction, scaled by a step length.

We assume that every direction is a descent direction, and that the step length satisfies Wolfe conditions, i.e., conditions (1) and (2a) from earlier.

In addition, the function is assumed to be bounded below on the whole space and continuously differentiable in an open set that contains the lower-level set of points where the function value is less than or equal to the initial value.

We also require that the gradient be Lipschitz continuous in that region. Under these assumptions, the theorem states that the sum of the squared gradients, weighted by the square of the cosine of the angle between the gradient and the search direction, is finite.

This result does not immediately imply that the gradient tends to zero, but it provides a crucial foundation for proving convergence under additional assumptions — for example, when the angle between the directions and gradients stays away from ninety degrees.

Proof of Theorem 6 (part 1)

Proof. From the curvature condition (Wolfe's second condition), we have

$$(\nabla f(x_{k+1}) - \nabla f(x_k))^T p_k \geq (c_2 - 1)\, \nabla f(x_k)^T p_k$$

From Lipschitz continuity of ∇f, we obtain

$$(\nabla f(x_{k+1}) - \nabla f(x_k))^T p_k \leq \|\nabla f(x_{k+1}) - \nabla f(x_k)\|\, \|p_k\| \leq \alpha_k L \|p_k\|^2$$

Combining both estimates:

$$\alpha_k \geq \frac{(c_2 - 1)\, \nabla f(x_k)^T p_k}{L\|p_k\|^2}$$

Substituting into the sufficient decrease condition (Armijo rule):

$$f(x_{k+1}) \leq f(x_k) - \frac{c_1(1 - c_2)}{L} \cdot \frac{(\nabla f(x_k)^T p_k)^2}{\|p_k\|^2}$$

Comments

We begin the proof. According to the curvature condition, which is Wolfe's second condition, the scalar product of $\nabla f(x_{k+1}) - \nabla f(x_k)$ with the direction p_k is greater than or equal to $(c_2 - 1)$ times the scalar product of $\nabla f(x_k)$ with the same direction p_k.

On the other hand, because the gradient is Lipschitz continuous, the same scalar product is less than or equal to α_k times the Lipschitz constant L times the square of the norm of p_k.

Combining the two gives a lower bound for α_k. It must be greater than or equal to the product of $(c_2 - 1)$ and the scalar product of the gradient and p_k, divided by L times the squared norm of p_k.

Now we substitute this into the first Wolfe condition, also known as the Armijo condition. It states that the value of the function at x_{k+1} is less than or equal to the value at x_k plus α_k times c_1 times the directional derivative.

Substituting our estimate for α_k, we get that the function decreases by at least a constant times the square of the scalar product of the gradient and p_k, divided by the squared norm of p_k.

Proof of Theorem 6 (part 2)

Introduction

Taylor's Theorem

Optimality conditions

Strategies of optimization

Step length conditions

Convergence

Proof (continued). Using

$$\cos\theta_k = \frac{-\nabla f(x_k)^T p_k}{\|\nabla f(x_k)\| \cdot \|p_k\|}$$

we get

$$f(x_{k+1}) \le f(x_k) - c\,(\cos\theta_k)^2 \|\nabla f(x_k)\|^2$$

where $c = \frac{c_1(1-c_2)}{L} > 0$. Summing over k:

$$c \sum_{k=0}^{N} (\cos\theta_k)^2 \|\nabla f(x_k)\|^2 \le f(x_0) - f(x_{N+1})$$

Since f is bounded below by assumption, the sum must converge:

$$\sum_{k\ge 0} (\cos\theta_k)^2 \|\nabla f(x_k)\|^2 < \infty \quad \square$$

Comments

We now continue the proof. To express everything in terms of the angle between the gradient and the direction, we use the definition of the cosine.

The cosine of the angle θ_k is equal to minus the scalar product of $\nabla f(x_k)$ and p_k, divided by the product of their norms.

Substituting the expression for cosine into the inequality, we get that $f(x_{k+1}) \le f(x_k) - c\,(\cos\theta_k)^2 \|\nabla f(x_k)\|^2$

Now we sum over all k from zero to N. On the left, we have a constant times the sum of $(\cos\theta_k)^2$ times the squared gradient norm.

On the right, we have the total decrease in the function value, from x_0 to x_{N+1}. Since the function is bounded below, the decrease is bounded above. Therefore, the sum must be finite.

This completes the proof.

It is worth noting that similar results can be obtained when different line search conditions are used.

In particular, if we replace the Wolfe conditions with the Goldstein conditions or the strong Wolfe conditions, the conclusion remains valid.

This shows that the convergence mechanism is robust with respect to the specific form of the step-size rules.

Global Convergence

- Zoutendijk condition:

$$\sum_{k\geq 0}(\cos\theta_k)^2\|\nabla f(x_k)\|^2 < \infty$$

- If $(\cos\theta_k)^2$ is bounded away from zero, then $\|\nabla f(x_k)\| \to 0$.
- For Newton-type steps (when $p_k = -B_k^{-1}\nabla f_k$), we have the estimate:

$$\cos\theta_k \geq \frac{1}{\kappa(B_k)}, \quad \text{where } \kappa(B_k) \text{ is the condition number of } B_k.$$

  This follows from the inequality (if we substitute $B = B_k,\ a = \nabla f_k$):

$$\frac{a^TBa}{\|a\|\|Ba\|} \geq \frac{a^TBa}{\|a\|^2\|B\|} \geq \frac{\lambda_{\min}}{\lambda_{\max}} = \frac{1}{\kappa(B_k)} \quad \forall\, a \in \mathbb{R}^n,\ \text{if } B \succ 0,$$

  where $\lambda_{\max}, \lambda_{\min}$ are the minimum and maximum eigenvalues of matrix B.
- Similar convergence holds for:
  - Newton and quasi-Newton methods with globalization
  - Conjugate gradient methods (under assumptions)
- Global convergence requires: descent direction + proper line search.

Introduction
Taylor's Theorem
Optimality conditions
Strategies of optimization
Step length conditions
Convergence

Comments

We have completed the discussion of global convergence for optimization methods using line search.

The main theoretical result in this part is the Zoutendijk condition, which holds for a wide class of methods that satisfy two key requirements: each step must use a descent direction, and the step size must satisfy Wolfe-type conditions.

The Zoutendijk condition states that the sum over all iterations of the squared cosine of the angle between the search direction and the negative gradient, multiplied by the squared norm of the gradient, is finite.

This is crucial, because if the angle between the direction and the gradient remains uniformly bounded away from ninety degrees, then it follows that the gradient norm must go to zero.

In other words, the iterates converge to a point satisfying the first-order optimality condition.

It is important to understand that global convergence means convergence from an arbitrary starting point to a stationary point.

This cannot be guaranteed by the core method alone — for example, Newton's method may diverge if used naively.

Therefore, additional mechanisms are introduced: the direction must be a descent direction, and the step size must ensure sufficient decrease. These elements are implemented through line search strategies satisfying conditions such as Armijo, Wolfe, or Goldstein.

In Newton or quasi-Newton methods, such as BFGS, we can estimate the angle between the gradient and the search direction: if the matrix B_k in Newton-type methods is symmetric and positive definite, then the cosine of this angle at iteration k is at least $1/\kappa(B_k)$, where $\kappa(B_k)$ is the condition number of B_k.

This follows directly from the well-known inequality for a symmetric positive definite matrix.

Global Convergence

Introduction
Taylor's Theorem
Optimality conditions
Strategies of optimization
Step length conditions
Convergence

- Zoutendijk condition:

$$\sum_{k\geq 0}(\cos\theta_k)^2\|\nabla f(x_k)\|^2 < \infty$$

- If $(\cos\theta_k)^2$ is bounded away from zero, then $\|\nabla f(x_k)\| \to 0$.
- For Newton-type steps (when $p_k = -B_k^{-1}\nabla f_k$), we have the estimate:

$$\cos\theta_k \geq \frac{1}{\kappa(B_k)}, \quad \text{where } \kappa(B_k) \text{ is the condition number of } B_k.$$

  This follows from the inequality (if we substitute $B = B_k,\ a = \nabla f_k$):

$$\frac{a^T Ba}{\|a\|\|Ba\|} \geq \frac{a^T Ba}{\|a\|^2\|B\|} \geq \frac{\lambda_{\min}}{\lambda_{\max}} = \frac{1}{\kappa(B_k)} \quad \forall\, a \in \mathbb{R}^n,\ \text{if } B \succ 0,$$

  where $\lambda_{\max}, \lambda_{\min}$ are the minimum and maximum eigenvalues of matrix B.
- Similar convergence holds for:
  - Newton and quasi-Newton methods with globalization
  - Conjugate gradient methods (under assumptions)
- Global convergence requires: descent direction + proper line search.

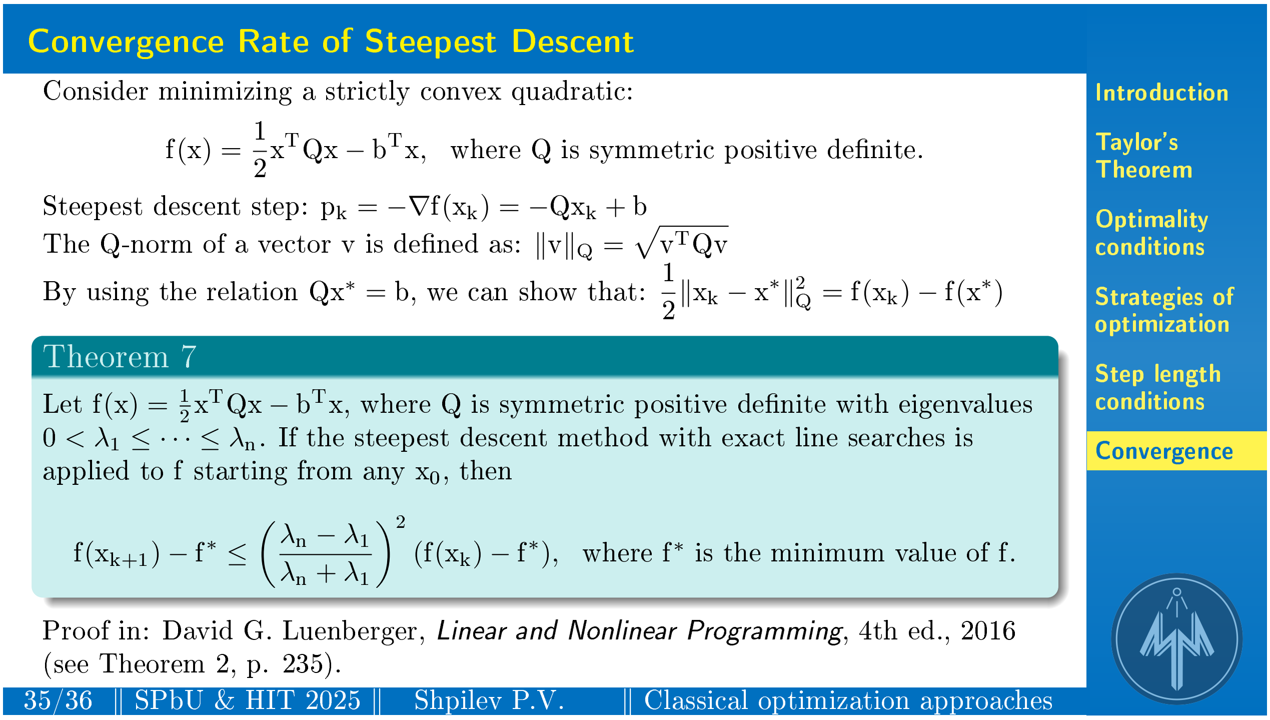
Comments

This means that if B_k is well-conditioned, the direction cannot be nearly orthogonal to the gradient, and the Zoutendijk condition implies convergence. Besides gradient-based and quasi-Newton methods, global convergence theory also applies to conjugate gradient methods, which we will discuss further. These are especially effective for solving large-scale quadratic problems with sparse structure.

Under certain assumptions, such as convexity and exact line search, one can prove global convergence results for these methods as well.

Although more general settings require stronger assumptions, the overall logic remains similar: descent plus control over the direction leads to convergence. In conclusion, we now have a complete view of the conditions that guarantee convergence to stationary points with minimal assumptions.

The next step in our analysis is to study the rate of convergence, where we shift from the question "Will it converge?" to "How fast will it converge?" — a key concern for algorithmic efficiency.



Convergence Rate of Steepest Descent

Consider minimizing a strictly convex quadratic:

$$f(x) = \frac{1}{2}x^T Qx - b^T x, \quad \text{where } Q \text{ is symmetric positive definite.}$$

Steepest descent step: $p_k = -\nabla f(x_k) = -Qx_k + b$

The Q-norm of a vector v is defined as: $\|v\|_Q = \sqrt{v^T Qv}$

By using the relation $Qx^* = b$, we can show that: $\frac{1}{2}\|x_k - x^*\|_Q^2 = f(x_k) - f(x^*)$

Theorem 7

Let $f(x) = \frac{1}{2}x^T Qx - b^T x$, where Q is symmetric positive definite with eigenvalues $0 < \lambda_1 \leq \cdots \leq \lambda_n$. If the steepest descent method with exact line searches is applied to f starting from any x_0, then

$$f(x_{k+1}) - f^* \leq \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^2 (f(x_k) - f^*), \quad \text{where } f^* \text{ is the minimum value of } f.$$

Proof in: David G. Luenberger, *Linear and Nonlinear Programming*, 4th ed., 2016 (see Theorem 2, p. 235).

Comments

Before proceeding to the analysis of the convergence rate of the steepest descent method, let us highlight some important conceptual points.

Convergence rate is not merely a technical concern. Algorithmic strategies often have to balance between achieving fast local convergence and maintaining global reliability. For example, the steepest descent method is the quintessential example of a globally convergent algorithm: it reliably finds a stationary point when proper step sizes are used. However, its practical rate of convergence is often quite slow.

Conversely, the pure Newton iteration achieves rapid convergence near a solution, but away from the solution its steps may not even be descent directions, which endangers global convergence.

In the following, we will examine the convergence rate for different line search strategies. We begin with the simplest case — the steepest descent method.

To better understand the nature of the steepest descent method, we consider the ideal case — minimizing a strictly convex quadratic function.

It has the form: $f(x) = \frac{1}{2}x^T Qx - b^T x$. Here, the matrix Q is symmetric and positive definite.

At each iteration, the steepest descent method selects the search direction as minus the gradient of the function at the current point.

Since the function is quadratic, the gradient is $\nabla f(x_k) = Qx_k - b$. Therefore, the search direction is $p_k = -Qx_k + b$.

To estimate the convergence rate for a quadratic function, we first need to introduce the concept of the Q-norm.

For a vector v, the Q-norm, denoted as $\|v\|_Q$, is the square root of $v^T Qv$. Here, the matrix Q is symmetric and positive definite, and it reflects the curvature of the quadratic function.

For a quadratic function f, there is a useful formula: $\frac{1}{2}\|x_k - x^*\|_Q^2 = f(x_k) - f(x^*)$. This relation is obtained by using the optimality condition that $Qx^* = b$.

Convergence Rate of Steepest Descent

Consider minimizing a strictly convex quadratic:

$$f(x) = \frac{1}{2}x^TQx - b^Tx, \quad \text{where } Q \text{ is symmetric positive definite.}$$

Steepest descent step: $p_k = -\nabla f(x_k) = -Qx_k + b$
The Q-norm of a vector v is defined as: $\|v\|_Q = \sqrt{v^TQv}$
By using the relation $Qx^* = b$, we can show that: $\frac{1}{2}\|x_k - x^*\|_Q^2 = f(x_k) - f(x^*)$

Theorem 7

Let $f(x) = \frac{1}{2}x^TQx - b^Tx$, where Q is symmetric positive definite with eigenvalues $0 < \lambda_1 \leq \cdots \leq \lambda_n$. If the steepest descent method with exact line searches is applied to f starting from any x_0, then

$$f(x_{k+1}) - f^* \leq \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^2 (f(x_k) - f^*), \quad \text{where } f^* \text{ is the minimum value of f.}$$

Proof in: David G. Luenberger, *Linear and Nonlinear Programming*, 4th ed., 2016 (see Theorem 2, p. 235).

Comments

Now, let us state a theorem that describes the convergence rate of the steepest descent method with exact line searches when optimizing a quadratic function.

Suppose the function f has the form: $f(x) = \frac{1}{2}x^TQx - b^Tx$.

The matrix Q is symmetric and positive definite, and its eigenvalues are ordered as follows: $0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. Then, for any initial point x_0, the steepest descent method with exact line searches satisfies the following bound. The difference $f(x_{k+1}) - f^*$ is at most $\left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right)^2$ multiplied by the difference $f(x_k) - f^*$, where f^* is the minimum value of f.

Thus, the convergence rate is determined by the condition number of the matrix Q, which is the ratio of its largest eigenvalue λ_n to its smallest eigenvalue λ_1. The smaller the condition number, the faster the convergence.

In the extreme case where all eigenvalues are equal, the method achieves the minimum in a single iteration.

The proof of this theorem can be found in Luenberger's book Linear and Nonlinear Programming, fourth edition, 2016, Theorem 2 on page 235.