6 Optimization

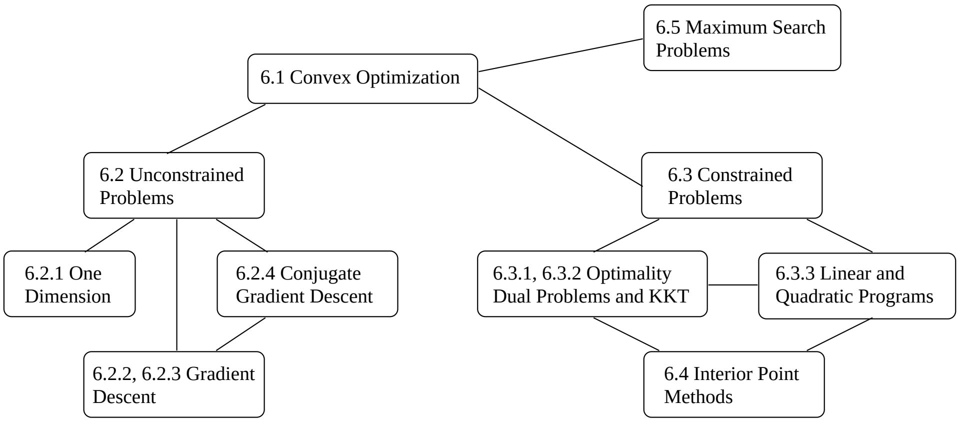
This chapter provides a self-contained overview of some of the basic tools needed to solve the optimization problems used in kernel methods. In particular, we will cover topics such as minimization of functions in one variable, convex minimization and maximization problems, duality theory, and statistical methods to solve optimization problems approximately.

The focus is noticeably different from the topics covered in works on optimization for Neural Networks, such as Backpropagation [588, 452, 317, 7] and its variants. In these cases, it is necessary to deal with non-convex problems exhibiting a large number of local minima, whereas much of the research on Kernel Methods and Mathematical Programming is focused on problems with global exact solutions. These boundaries may become less clear-cut in the future, but at the present time, methods for the solution of problems with unique optima appear to be sufficient for our purposes.

Overview

In Section 6.1, we explain general properties of convex sets and functions, and how the extreme values of such functions can be found. Next, we discuss practical algorithms to best minimize convex functions on unconstrained domains (Section 6.2). In this context, we will present techniques like interval cutting methods, Newton's method, gradient descent and conjugate gradient descent. Section 6.3 then deals with constrained optimization problems, and gives characterization results for solutions. In this context, Lagrangians, primal and dual optimization problems, and the Karush-Kuhn-Tucker (KKT) conditions are introduced. These concepts set the stage for Section 6.4, which presents an interior point algorithm for the solution of constrained convex optimization problems. In a sense, the final section (Section 6.5) is a departure from the previous topics, since it introduces the notion of randomization into the optimization procedures. The basic idea is that unless the exact solution is required, statistical tools can speed up search maximization by orders of magnitude.

For a general overview, we recommend Section 6.1, and the first parts of Section 6.3, which explain the basic ideas underlying constrained optimization. The latter section is needed to understand the calculations which lead to the dual optimization problems in Support Vector Machines (Chapters 7–9). Section 6.4 is only intended for readers interested in practical implementations of optimization algorithms. In particular, Chapter 10 will require some knowledge of this section. Finally, Section 6.5 describes novel randomization techniques, which are needed in the sparse greedy methods of Section 10.2, 15.3, 16.4, and 18.4.3. Unconstrained


6.5 Maximum Search Problems
6.1 Convex Optimization
6.2 Unconstrained Problems
6.3 Constrained Problems
6.2.1 One Dimension
6.2.4 Conjugate Gradient Descent
6.3.1, 6.3.2 Optimality Dual Problems and KKT
6.3.3 Linear and Quadratic Programs
6.2.2, 6.2.3 Gradient Descent
6.4 Interior Point Methods

optimization problems (Section 6.2) are less common in this book and will only be required in the gradient descent methods of Section 10.6.1, and the Gaussian Process implementation methods of Section 16.4.

Prerequisites

The present chapter is intended as an introduction to the basic concepts of optimization. It is relatively self-contained, and requires only basic skills in linear algebra and multivariate calculus. Section 6.3 is somewhat more technical, Section 6.4 requires some additional knowledge of numerical analysis, and Section 6.5 assumes some knowledge of probability and statistics.

6.1 Convex Optimization

In the situations considered in this book, learning (or equivalently statistical estimation) implies the minimization of some risk functional such as $R_{\text{emp}}[f]$ or $R_{\text{reg}}[f]$ (cf. Chapter 4). While minimizing an arbitrary function on a (possibly not even compact) set of arguments can be a difficult task, and will most likely exhibit many local minima, minimization of a convex objective function on a convex set exhibits exactly one *global* minimum. We now prove this property.

Definition 6.1 (Convex Set) *A set X in a vector space is called convex if for any $x, x' \in X$ and any $\lambda \in [0,1]$, we have*

$$\lambda x + (1-\lambda)x' \in X. \tag{6.1}$$

Definition and Construction of Convex Sets and Functions

Definition 6.2 (Convex Function) *A function f defined on a set X (note that X need not be convex itself) is called convex if, for any $x, x' \in X$ and any $\lambda \in [0,1]$ such that $\lambda x + (1-\lambda)x' \in X$, we have*

$$f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda)f(x'). \tag{6.2}$$

A function f is called strictly *convex if for $x \neq x'$ and $\lambda \in (0,1)$ (6.2) is a strict inequality.*

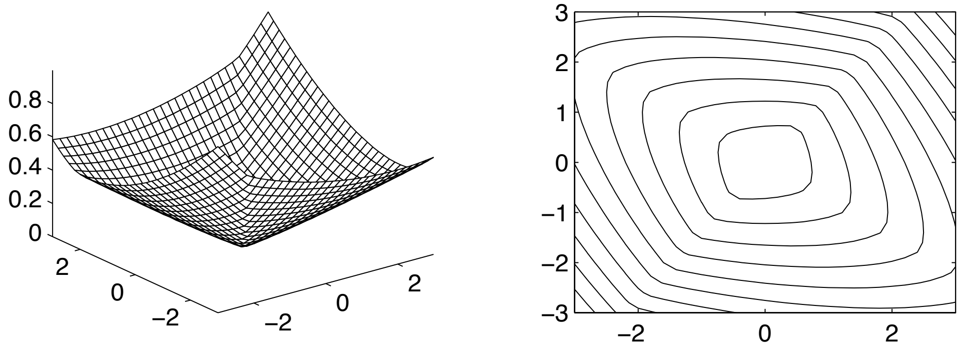


Figure 6.1 Left: Convex Function in two variables. Right: the corresponding convex level sets $\{x|f(x) \leq c\}$, for different values of c.

There exist several ways to define convex sets. A convenient method is to define them via *below sets* of convex functions, such as the sets for which $f(x) \leq c$, for instance.

Lemma 6.3 (Convex Sets as Below-Sets) *Denote by $f : \mathcal{X} \to \mathbb{R}$ a convex function on a convex set $\mathcal{X}$. Then the set*

$$X := \{x | x \in \mathcal{X} \text{ and } f(x) \leq c\}, \text{ for all } c \in \mathbb{R}, \tag{6.3}$$

is convex.

Proof We must show condition (6.1). For any $x, x' \in \mathcal{X}$, we have $f(x), f(x') \leq c$. Moreover, since f is convex, we also have

$$f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda) f(x') \leq c \text{ for all } \lambda \in [0,1]. \tag{6.4}$$

Hence, for all $\lambda \in [0,1]$, we have $(\lambda x + (1-\lambda)x') \in X$, which proves the claim. Figure 6.1 depicts this situation graphically. ■

Intersections

Lemma 6.4 (Intersection of Convex Sets) *Denote by $X, X' \subset \mathcal{X}$ two convex sets. Then $X \cap X'$ is also a convex set.*

Proof Given any $x, x' \in X \cap X'$, then for any $\lambda \in [0,1]$, the point $x_\lambda := \lambda x + (1-\lambda)x'$ satisfies $x_\lambda \in X$ and $x_\lambda \in X'$, hence also $x_\lambda \in X \cap X'$. ■

See also Figure 6.2. Now we have the tools to prove the central theorem of this section.

Theorem 6.5 (Minima on Convex Sets) *If the convex function $f : \mathcal{X} \to \mathbb{R}$ has a minimum on a convex set $X \subset \mathcal{X}$, then its arguments $x \in \mathcal{X}$, for which the minimum value is attained, form a convex set. Moreover, if f is strictly convex, then this set will contain only one element.*

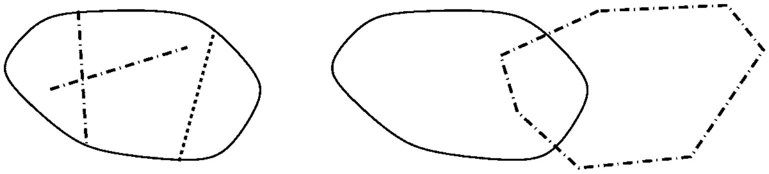


Figure 6.2 Left: a convex set; observe that lines with points in the set are fully contained inside the set. Right: the intersection of two convex sets is also a convex set.

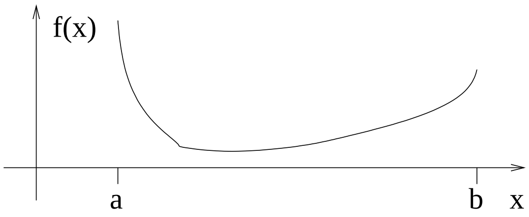


Figure 6.3 Note that the maximum of a convex function is obtained at the ends of the interval $[a, b]$.

Proof Denote by c the minimum of f on X. Then the set $X_m := \{x | x \in \mathcal{X} \text{ and } f(x) \leq c\}$ is clearly convex. In addition, $X_m \cap X$ is also convex, and $f(x) = c$ for all $x \in X_m \cap X$ (otherwise c would not be the minimum).

If f is strictly convex, then for any $x, x' \in X$, and in particular for any $x, x' \in X \cap X_m$, we have (for $x \neq x'$ and all $\lambda \in (0, 1)$),

$$f(\lambda x + (1 - \lambda)x') < \lambda f(x) + (1 - \lambda) f(x') = \lambda c + (1 - \lambda)c = c. \tag{6.5}$$

This contradicts the assumption that $X_m \cap X$ contains more then one element.

■

Global Minima

A simple application of this theorem is in constrained convex minimization. Recall that the notation $[n]$, used below, is a shorthand for $\{1, \ldots, n\}$.

Corollary 6.6 (Constrained Convex Minimization) *Given the set of convex functions $f, c_1, \ldots, c_n$ on the convex set $\mathcal{X}$, the problem*

$$\begin{aligned} &\underset{x}{\text{minimize}} && f(x), \\ &\textit{subject to} && c_i(x) \leq 0 \textit{ for all } i \in [n], \end{aligned} \tag{6.6}$$

has as its solution a convex set, if a solution exists. This solution is unique if f is strictly convex.

Many problems in Mathematical Programming or Support Vector Machines can be cast into this formulation. This means either that they all have unique solutions (if f is strictly convex), or that all solutions are equally good and form a convex set (if f is merely convex).

We might ask what can be said about convex *maximization*. Let us analyze a simple case first: convex maximization on an interval.

Lemma 6.7 (Convex Maximization on an Interval) *Denote by f a convex function on $[a, b] \in \mathbb{R}$. Then the problem of maximizing f on $[a, b]$ has $f(a)$ and $f(b)$ as solutions.*

Maxima on Extreme Points

Proof Any $x \in [a, b]$ can be written as $\frac{b-x}{b-a}a + \left(1 - \frac{b-x}{b-a}\right) b$, and hence

$$f(x) \leq \frac{b-x}{b-a} f(a) + \left(1 - \frac{b-x}{b-a}\right) f(b) \leq \max(f(a), f(b)). \tag{6.7}$$

Therefore the maximum of f on $[a, b]$ is obtained on one of the points a, b. ■

We will next show that the problem of convex *maximization* on a convex set is typically a hard problem, in the sense that the maximum can only be found at one of the extreme points of the constraining set. We must first introduce the notion of vertices of a set.

Definition 6.8 (Vertex of a Set) *A point $x \in X$ is a vertex of X if, for all $x' \in X$ with $x' \neq x$, and for all $\lambda > 1$, the point $\lambda x + (1 - \lambda)x' \notin X$.*

This definition implies, for instance, that in the case of X being an ℓ_2 ball, the vertices of X make up its surface. In the case of an ℓ_∞ ball, we have 2^n vertices in n dimensions, and for an ℓ_1 ball, we have only $2n$ of them. These differences will guide us in the choice of admissible sets of parameters for optimization problems (see, e.g., Section 14.4). In particular, there exists a connection between suprema on sets and their convex hulls. To state this link, however, we need to define the latter.

Definition 6.9 (Convex Hull) *Denote by X a set in a vector space. Then the convex hull $\mathrm{co}\, X$ is defined as*

$$\mathrm{co}\, X := \left\{ \bar{x} \,\middle|\, \bar{x} = \sum_{i=1}^{n} \alpha_i x_i \text{ where } n \in \mathbb{N}, \alpha_i \geq 0 \text{ and } \sum_{i=1}^{n} \alpha_i = 1 \right\}. \tag{6.8}$$

Theorem 6.10 (Suprema on Sets and their Convex Hulls) *Denote by X a set and by $\mathrm{co}\, X$ its convex hull. Then for a convex function f*

$$\sup\{f(x) | x \in X\} = \sup\{f(x) | x \in \mathrm{co}\, X\}. \tag{6.9}$$

Evaluating Convex Sets on Extreme Points

Proof Recall that the below set of convex functions is convex (Lemma 6.3), and that the below set of f with respect to $c = \sup\{f(x) | x \in X\}$ is by definition a superset of X. Moreover, due to its convexity, it is also a superset of $\mathrm{co}\, X$. ■

This theorem can be used to replace search operations over sets X by subsets $X' \subset X$, which are considerably smaller, if the convex hull of the latter generates X. In particular, the vertices of convex sets are sufficient to reconstruct the whole set.

Theorem 6.11 (Vertices) *A compact convex set is the convex hull of its vertices.*

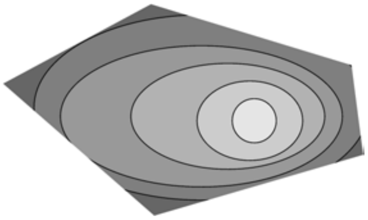


Figure 6.4 A convex function on a convex polyhedral set. Note that the minimum of this function is unique, and that the maximum can be found at one of the vertices of the constraining domain.

Reconstructing Convex Sets from Vertices

The proof is slightly technical, and not central to the understanding of kernel methods. See Rockafellar [435, Chapter 18] for details, along with further theorems on convex functions. We now proceed to the second key theorem in this section.

Theorem 6.12 (Maxima of Convex Functions on Convex Compact Sets) *Denote by X a compact convex set in $\mathcal{X}$, by $|X$ the vertices of X, and by f a convex function on X. Then*

$$\sup\{f(x)|x \in X\} = \sup\{f(x)|x \in |X\}. \tag{6.10}$$

Proof Application of Theorem 6.10 and Theorem 6.11 proves the claim, since under the assumptions made on X, we have $X = \operatorname{co}(|X)$. Figure 6.4 depicts the situation graphically. ■

6.2 Unconstrained Problems

After the characterization and uniqueness results (Theorem 6.5, Corollary 6.6, and Lemma 6.7) of the previous section, we will now study numerical techniques to obtain minima (or maxima) of convex optimization problems. While the choice of algorithms is motivated by applicability to kernel methods, the presentation here is not problem specific. For details on implementation, and descriptions of applications to learning problems, see Chapter 10.

6.2.1 Functions of One Variable

We begin with the easiest case, in which f depends on only one variable. Some of the concepts explained here, such as the interval cutting algorithm and Newton's method, can be extended to the multivariate setting (see Problem 6.5). For the sake of simplicity, however, we limit ourselves to the univariate case.

Assume we want to minimize $f : \mathbb{R} \to \mathbb{R}$ on the interval $[a, b] \subset \mathbb{R}$. If we cannot make any further assumptions regarding f, then this problem, as simple as it may seem, cannot be solved numerically.

Continuous Differentiable Functions

If f is *differentiable*, the problem can be reduced to finding $f'(x) = 0$ (see Problem 6.4 for the general case). If in addition to the previous assumptions, f is convex, then f' is nondecreasing, and we can find a fast, simple algorithm (Algorithm

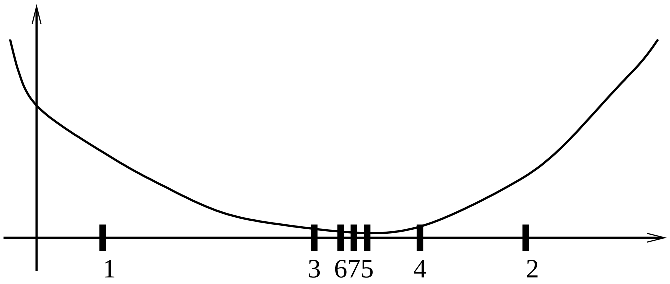


Figure 6.5 Interval Cutting Algorithm. The selection of points is ordered according to the numbers beneath (points 1 and 2 are the initial endpoints of the interval).

Algorithm 6.1 Interval Cutting

```
Require: $a, b$, Precision $\epsilon$
  Set $A = a, B = b$
  repeat
    if $f'\left(\frac{A+B}{2}\right) > 0$ then
      $B = \frac{A+B}{2}$
    else
      $A = \frac{A+B}{2}$
    end if
  until $(B - A)\min(|f'(A)|, |f'(B)|) \leq \epsilon$
Output: $x = \frac{A+B}{2}$
```

Interval Cutting

6.1) to solve our problem (see Figure 6.5).

This technique works by halving the size of the interval that contains the minimum x^* of f, since it is always guaranteed by the selection criteria for B and A that $x^* \in [A, B]$. We use the following Taylor series expansion to determine the stopping criterion.

Theorem 6.13 (Taylor Series) *Denote by $f : \mathbb{R} \to \mathbb{R}$ a function that is d times differentiable. Then for any $x, x' \in \mathbb{R}$, there exists a ξ with $|\xi| \leq |x - x'|$, such that*

$$f(x') = \sum_{i=0}^{d-1} \frac{1}{i!} f^{(i)}(x)(x' - x)^i + \frac{\xi^d}{d!} f^{(d)}(x + \xi). \tag{6.11}$$

Now we may apply (6.11) to the stopping criterion of Algorithm 6.1. We denote by x^* the minimum of $f(x)$. Expanding f around $f(x^*)$, we obtain for some $\xi_A \in [A - x^*, 0]$ that $f(A) = f(x^*) + \xi_A f'(x^* + \xi_A)$, and therefore,

$$|f(A) - f(x^*)| = |\xi_A||f'(x^* + \xi_A)| \leq (B - A)|f'(A)|.$$

Proof of Linear Convergence

Taking the minimum over $\{A, B\}$ shows that Algorithm 6.1 stops once f is ϵ-close to its minimal value. The convergence of the algorithm is *linear* with constant 0.5, since the intervals $[A, B]$ for possible x^* are halved at each iteration.

Algorithm 6.2 Newton's Method

```
Require: $x_0$, Precision $\epsilon$
  Set $x = x_0$
  repeat
    $x = x - \frac{f'(x)}{f''(x)}$
  until $|f'(x)| \leq \epsilon$
Output: $x$
```

In constructing the interval cutting algorithm, we in fact wasted most of the information obtained in evaluating f' at each point, by only making use of the sign of f. In particular, we could fit a parabola to f and thereby obtain a method that converges more rapidly. If we are only allowed to use f and f', this leads to the *Method of False Position* (see [334] or Problem 6.3).

Newton's Method

Moreover, if we may compute the second derivative as well, we can use (6.11) to obtain a quadratic approximation of f and use the latter to find the minimum of f. This is commonly referred to as *Newton's method* (see Section 16.4.1 for a practical application of the latter to classification problems). We expand $f(x)$ around x_0;

$$f(x) \approx f(x_0) + (x - x_0)f'(x_0) + \frac{(x - x_0)^2}{2} f''(x_0). \tag{6.12}$$

Minimization of the expansion (6.12) yields

$$x = x_0 - \frac{f'(x_0)}{f''(x_0)}. \tag{6.13}$$

Hence, we hope that if the approximation (6.12) is good, we will obtain an algorithm with fast convergence (Algorithm 6.2). Let us analyze the situation in more detail. For convenience, we state the result in terms of $g := f'$, since finding a zero of g is equivalent to finding a minimum of f.

Quadratic Convergence

Theorem 6.14 (Convergence of Newton Method) *Let $g : \mathbb{R} \to \mathbb{R}$ be a twice continuously differentiable function, and denote by $x^* \in \mathbb{R}$ a point with $g'(x^*) \neq 0$ and $g(x^*) = 0$. Then, provided x_0 is sufficiently close to x^*, the sequence generated by (6.13) will converge to x^* at least quadratically.*

Proof For convenience, denote by x_n the value of x at the nth iteration. As before, we apply Theorem 6.13. We now expand $g(x^*)$ around x_n. For some $\xi \in [0, x^* - x_n]$, we have

$$g(x_n) = g(x_n) - g(x^*) = g(x_n) - \left[g(x_n) + g'(x_n)(x^* - x_n) + \frac{\xi^2}{2} g''(x_n) \right], \tag{6.14}$$

and therefore by substituting (6.14) into (6.13),

$$x_{n+1} - x^* = x_n - x^* - \frac{g(x_n)}{g'(x_n)} = \xi^2 \frac{g''(x_n)}{2g'(x_n)}. \tag{6.15}$$

Since by construction $|\xi| \leq |x_n - x^*|$, we obtain a quadratically convergent algorithm in $|x_n - x^*|$, provided that $\left| (x_n - x^*) \frac{g''(x_n)}{2g'(x_n)} \right| < 1$. ■

Region of Convergence

In other words, if the Newton method converges, it converges more rapidly than interval cutting or similar methods. We cannot guarantee beforehand that we are really in the region of convergence of the algorithm. In practice, if we apply the Newton method and find that it converges, we know that the solution has converged to the minimizer of f. For more information on optimization algorithms for unconstrained problems see [173, 530, 334, 15, 159, 45].

Line Search

In some cases we will not know an upper bound on the size of the interval to be analyzed for the presence of minima. In this situation we may, for instance, start with an initial guess of an interval, and if no minimum can be found strictly *inside* the interval, enlarge it, say by doubling its size. See [334] for more information on this matter. Let us now proceed to a technique which is quite popular (albeit not always preferable) in machine learning.

6.2.2 Functions of Several Variables: Gradient Descent

Gradient descent is one of the simplest optimization techniques to implement for minimizing functions of the form $f : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ may be $\mathbb{R}^N$, or indeed any set on which a gradient may be defined and evaluated. In order to avoid further complications we assume that the gradient $f'(x)$ exists and that we are able to compute it.

Direction of Steepest Descent

The basic idea is as follows: given a location x_n at iteration n, compute the gradient $g_n := f'(x_n)$, and update

$$x_{n+1} = x_n - \gamma g_n \tag{6.16}$$

such that the decrease in f is maximal over all $\gamma > 0$. For the final step, one of the algorithms from Section 6.2.1 can be used. It is straightforward to show that $f(x_n)$ is a monotonically decreasing series, since at each step the line search updates x_{n+1} in such a way that $f(x_{n+1}) < f(x_n)$. Such a value of γ must exist, since (again by Theorem 6.13) we may expand $f(x_n + \gamma g_n)$ in terms of γ around x_n, to obtain[1]

$$f(x_n - \gamma g_n) = f(x_n) - \gamma \|g_n\|^2 + O(\gamma^2). \tag{6.17}$$

As usual $\| \cdot \|$ is the Euclidean norm. For small γ the linear contribution in the Taylor expansion will be dominant, hence for some $\gamma > 0$ we have $f(x_n - \gamma g_n) < f(x_n)$. It can be shown [334] that after a (possibly infinite) number of steps, gradient descent (see Algorithm 6.3) will converge.

Problems of Convergence

In spite of this, the performance of gradient descent is far from optimal. Depending on the shape of the landscape of values of f, gradient descent may take a long time to converge. Figure 6.6 shows two examples of possible convergence behavior of the gradient descent algorithm.

[1] To see that Theorem 6.13 applies in (6.17), note that $f(x_n + \gamma g_n)$ is a mapping $\mathbb{R} \to \mathbb{R}$ when viewed as a function of γ.

Algorithm 6.3 Gradient Descent

```
Require: $x_0$, Precision $\epsilon$
  $n = 0$
  repeat
    Compute $g = f'(x_n)$
    Perform line search on $f(x_n - \gamma g)$ for optimal $\gamma$.
    $x_{n+1} = x_n - \gamma g$
    $n = n + 1$
  until $\|f'(x_n)\| \leq \epsilon$
Output: $x_n$
```

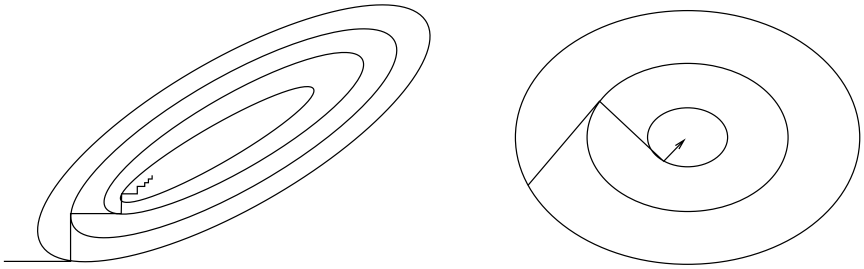


Figure 6.6 Left: Gradient descent takes a long time to converge, since the landscape of values of f forms a long and narrow valley, causing the algorithm to zig-zag along the walls of the valley. Right: due to the homogeneous structure of the minimum, the algorithm converges after very few iterations. Note that in both cases, the next direction of descent is *orthogonal* to the previous one, since line search provides the optimal step length.

6.2.3 Convergence Properties of Gradient Descent

Let us analyze the convergence properties of Algorithm 6.3 in more detail. To keep matters simple, we assume that f is a quadratic function, i.e.

$$f(x) = \frac{1}{2}(x - x^*)^\top K(x - x^*) + c_0, \tag{6.18}$$

where K is a positive definite symmetric matrix (cf. Definition 2.4) and c_0 is constant.[2] This is clearly a convex function with minimum at x^*, and $f(x^*) = c_0$. The gradient of f is given by

$$g := f'(x) = K(x - x^*). \tag{6.19}$$

To find the update of the steepest descent we have to minimize

$$f(x - \gamma g) = \frac{1}{2}(x - \gamma g - x^*)K(x - \gamma g - x^*) = \frac{1}{2}\gamma^2 g^\top K g - \gamma g^\top g. \tag{6.20}$$

[2] Note that we may rewrite (up to a constant) any convex quadratic function $f(x) = x^\top Kx + c^\top x + d$ in the form (6.18), simply by expanding f around its minimum value x^*.