Introduction to Optimization

Lecture 04: Strict and strong convexity. Iterative algorithms. Descent methods.

university of groningen

Characterizations of differentiable convex functions

Proposition

Let $f : D \subset \mathbb{R}^N \to \mathbb{R}$ be differentiable. The following are equivalent:

1. *f is convex;*
2. *for all $x, y \in D$, $f(y) \geq f(x) + \nabla f(x) \cdot (y - x)$;*
3. *for all $x, y \in D$, $\big(\nabla f(y) - \nabla f(x)\big) \cdot (y - x) \geq 0$.*

If f is twice differentiable, the three statements above are equivalent to

4. *for all $x \in D$, $\nabla^2 f(x)$ is positive semidefinite.*

Strict and strong convexity

A function $f : D \subset \mathbb{R}^N \to \mathbb{R}$ is strictly convex if D is convex and

$$f\big(\lambda x + (1-\lambda)y\big) < \lambda f(x) + (1-\lambda)f(y)$$

for all $x, y \in D$ and all $\lambda \in (0,1)$

Strict and strong convexity

A function $f : D \subset \mathbb{R}^N \to \mathbb{R}$ is strictly convex if D is convex and

$$f\big(\lambda x + (1-\lambda)y\big) < \lambda f(x) + (1-\lambda) f(y)$$

for all $x, y \in D$ and all $\lambda \in (0,1)$, and it is strongly convex if D is convex and

$$f\big(\lambda x + (1-\lambda)y\big) \leq \lambda f(x) + (1-\lambda) f(y) - \frac{\alpha}{2}\lambda(1-\lambda)\|x-y\|^2$$

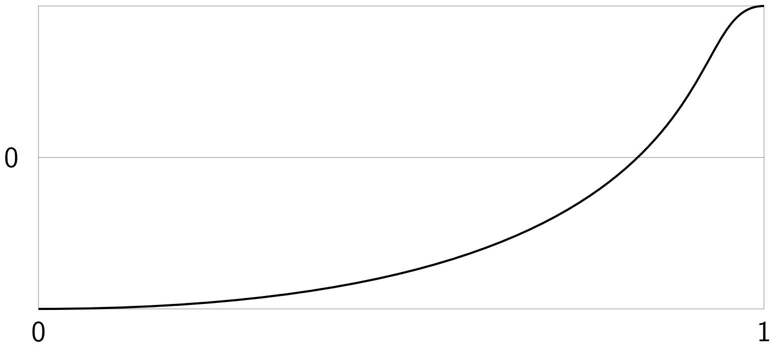
for all $x, y \in D$ and all $\lambda \in (0,1)$.

Exercises

1. Find examples of: a function that is convex, but not strictly convex; and a function that is strictly convex, but not strongly convex.
2. When is the function $f(x) = \frac{1}{2}\|Ax - y\|^2$ strictly/strongly convex?
3. Prove that every strictly convex function $f : \mathbb{R}^N \to \mathbb{R}$ has at most one minimizer, and every strongly convex function $f : \mathbb{R}^N \to \mathbb{R}$ has exactly one minimizer.
4. Can you obtain characterizations of strict and strong convexity of f in terms of properties of ∇f and $\nabla^2 f$?

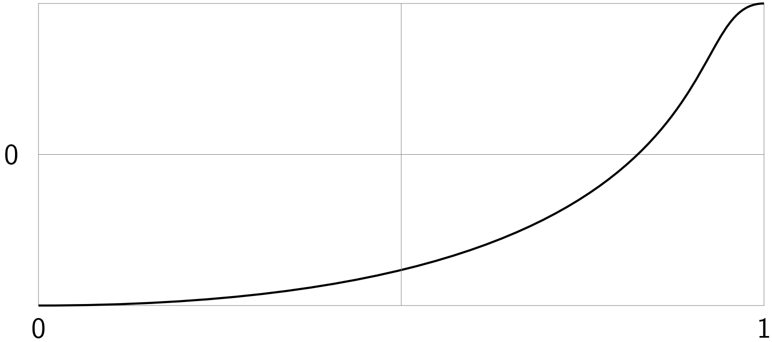
Introduction to iterative algorithms

Example: Bisection method to solve $g(x) = 0$



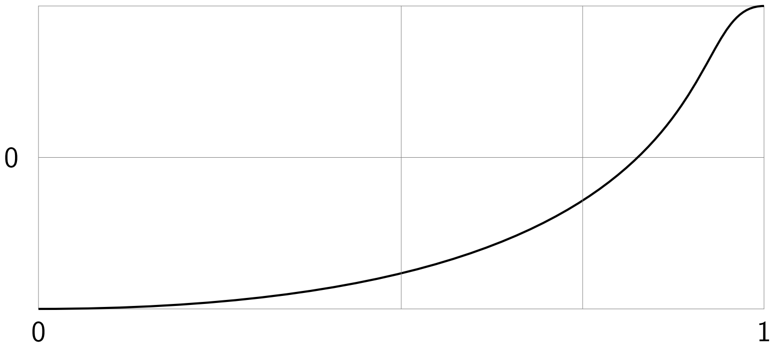
Introduction to iterative algorithms

Example: Bisection method to solve $g(x) = 0$



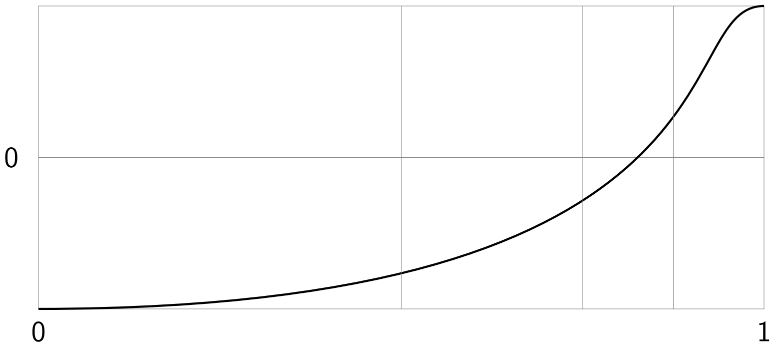
Introduction to iterative algorithms

Example: Bisection method to solve $g(x) = 0$



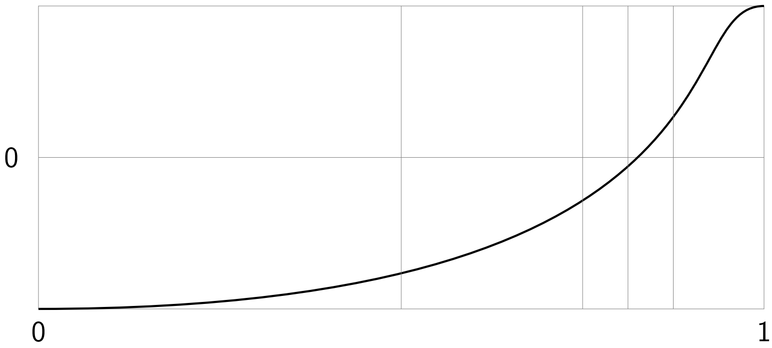
Introduction to iterative algorithms

Example: Bisection method to solve $g(x) = 0$



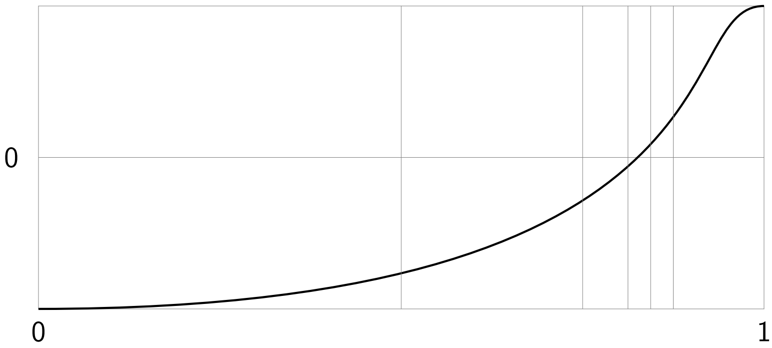
Introduction to iterative algorithms

Example: Bisection method to solve $g(x) = 0$



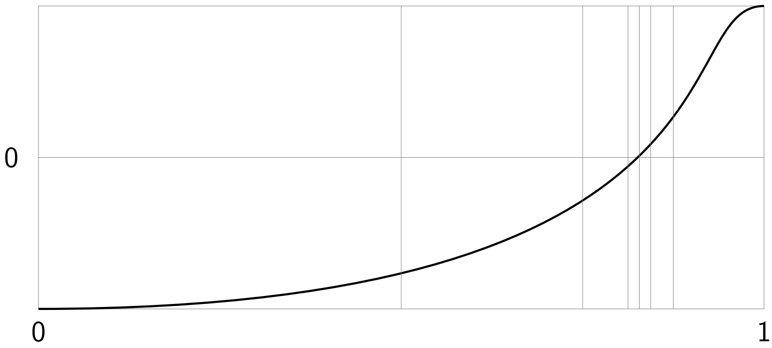
Introduction to iterative algorithms

Example: Bisection method to solve $g(x) = 0$



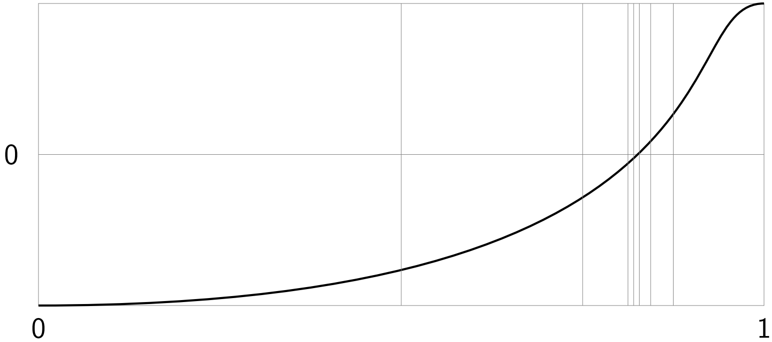
Introduction to iterative algorithms

Example: Bisection method to solve $g(x) = 0$



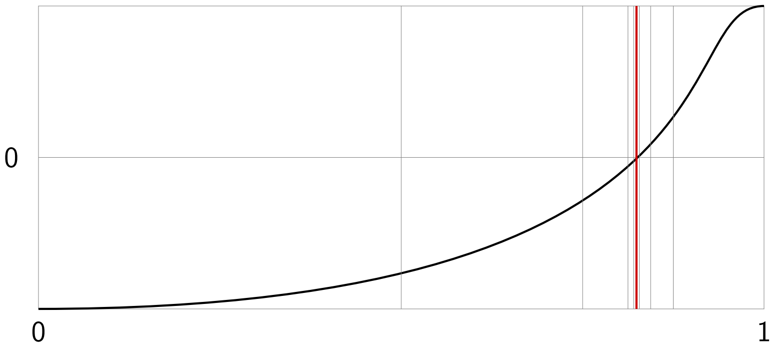
Introduction to iterative algorithms

Example: Bisection method to solve $g(x) = 0$



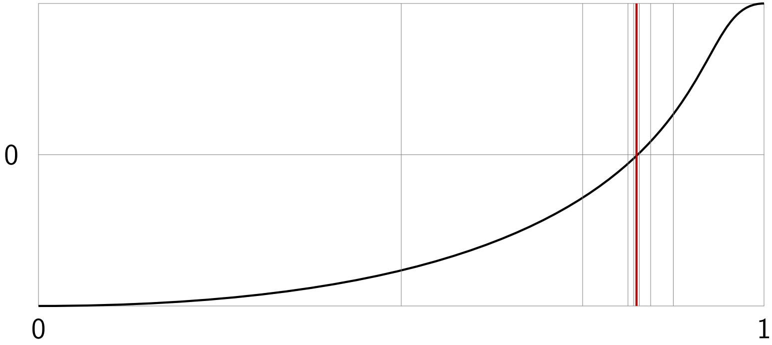
Introduction to iterative algorithms

Example: Bisection method to solve $g(x) = 0$



Introduction to iterative algorithms

Example: Bisection method to solve $g(x) = 0$



After k iterations, the distance to a solution is $|x_k - \hat{x}| \leq 2^{-k}$.

Introduction to iterative algorithms

An iterative algorithm is a procedure that computes a sequence (x_n) of points in $\mathbb{R}^N$ that approximate a solution to a problem. It requires:

- An initial guess x_0.
- A sequence (p_k) of parameters (typically $p_k \in \mathbb{R}^M$ for all $k \geq 0$).
- An operator $T : \mathbb{R}^N \times \mathbb{R}^M \to \mathbb{R}^N$ used to compute x_{k+1}, given x_k:

$$x_{k+1} = T(p_k, x_k).$$

- A stopping rule that is activated when the approximation is sufficiently good.

Introduction to iterative algorithms

An iterative algorithm is a procedure that computes a sequence (x_n) of points in $\mathbb{R}^N$ that approximate a solution to a problem. It requires:

- An initial guess x_0.
- A sequence (p_k) of parameters (typically $p_k \in \mathbb{R}^M$ for all $k \geq 0$).
- An operator $T : \mathbb{R}^N \times \mathbb{R}^M \to \mathbb{R}^N$ used to compute x_{k+1}, given x_k:

$$x_{k+1} = T(p_k, x_k).$$

- A stopping rule that is activated when the approximation is sufficiently good.

Important questions: convergence and complexity.

Stopping rules in minimization problems

Ideally, the algorithm should stop when this is when

- x_k is close to a minimizer, $\times$
- $f(x_k)$ is close to the optimal value $\inf(f)$, $\times$
- $\|\nabla f(x_k)\|$ is small. $\checkmark$

Stopping rules in minimization problems

Ideally, the algorithm should stop when this is when

- x_k is close to a minimizer, x
- $f(x_k)$ is close to the optimal value $\inf(f)$, x
- $\|\nabla f(x_k)\|$ is small. ✓

Common implementable rules:

- $\|\nabla f(x_k)\| < \varepsilon$
- $f(x_{k+1}) - f(x_k) < \varepsilon$
- $\|x_{k+1} - x_k\| < \varepsilon$

Stopping rules in minimization problems

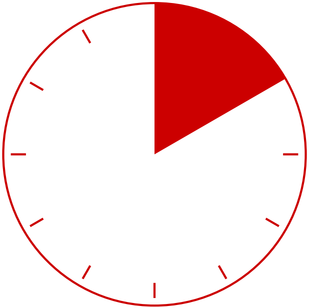
Ideally, the algorithm should stop when this is when

- x_k is close to a minimizer, $\times$
- $f(x_k)$ is close to the optimal value $\inf(f)$, $\times$
- $\|\nabla f(x_k)\|$ is small. ✓

Common implementable rules:

- $\|\nabla f(x_k)\| < \varepsilon$
- $\frac{\|\nabla f(x_k)\|}{\|\nabla f(x_0)\|} < \varepsilon$
- $f(x_{k+1}) - f(x_k) < \varepsilon$
- $\frac{f(x_{k+1}) - f(x_k)}{f(x_1) - f(x_0)} < \varepsilon$
- $\|x_{k+1} - x_k\| < \varepsilon$
- $\frac{\|x_{k+1} - x_k\|}{\|x_1 - x_0\|} < \varepsilon$

Break



Descent methods

Many algorithms are based on the idea of (sufficient) descent: given x_k, find x_{k+1} such that

$$f(x_{k+1}) \leq f(x_k) - \delta_k^2. \tag{1}$$

Descent methods

Many algorithms are based on the idea of (sufficient) descent: given x_k, find x_{k+1} such that

$$f(x_{k+1}) \leq f(x_k) - \delta_k^2. \tag{1}$$

One way is to find $d_k \in \mathbb{R}^N$ and $\alpha_k > 0$, such that (1) holds with

$$x_{k+1} = x_k - \alpha_k d_k.$$

Descent methods

Many algorithms are based on the idea of (sufficient) descent: given x_k, find x_{k+1} such that

$$f(x_{k+1}) \leq f(x_k) - \delta_k^2. \tag{1}$$

One way is to find $d_k \in \mathbb{R}^N$ and $\alpha_k > 0$, such that (1) holds with

$$x_{k+1} = x_k - \alpha_k d_k.$$

We say $-d_k$ is a descent direction, and α_k is the step size, step length or learning rate (in ML).

Motivation: 3 case studies

Let us analyze the behavior in the following cases:

1. $f(x) = x^2$

Motivation: 3 case studies

Let us analyze the behavior in the following cases:

1. $f(x) = x^2$
2. $f(x) = 1/x$, $\mathrm{dom}(f) = (0, \infty)$.

Motivation: 3 case studies

Let us analyze the behavior in the following cases:

1. $f(x) = x^2$
2. $f(x) = 1/x$, $\text{dom}(f) = (0, \infty)$.
3. $f(x) = 1/x$, $\text{dom}(f) = (-\infty, 0)$.

L-smoothness

If there is time

A differentiable function $f : A \subset \mathbb{R}^N \to \mathbb{R}$ is *L-smooth*, with $L > 0$, if

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$$

for all $x, y \in A$.

L-smoothness

If there is time

A differentiable function $f : A \subset \mathbb{R}^N \to \mathbb{R}$ is L-smooth, with $L > 0$, if

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$$

for all $x, y \in A$.

Proposition (Descent Lemma)

If f is L-smooth and A is convex, then

$$|f(y) - f(x) - \nabla f(x) \cdot (y - x)| \leq \frac{L}{2}\|x - y\|^2$$

for all $x, y \in A$.