Optimization Theory

Piotr Więcek,
Faculty of Pure and Applied Mathematics,
Wrocław University of Science and Technology

21.10.2020

Gradient methods: Part I

Gradient methods: general idea I

- We want to find a local unconstrained minimum x^* of a given function $f : \mathbb{R}^n \to \mathbb{R}$ by constructing a sequence of approximations of x^*, x_0 (given by the user), x_1, x_2, $x_3, \ldots$ converging to x^*.
- For any n, $x_n = F(x_{n-1})$, where the form of function F should depend on f, ∇f, possibly $\nabla^2 f$.
- **Desired properties:**
  1. Each iteration decreases the value of f.
  2. Decrease in the value of f big enough not to allow convergence before x^* is reached.

Gradient methods: general idea II

Gradient methods are constructed as follows:
For a given starting point x_0 next iterations are given by the formula

$$x_{k+1} := x_k - \alpha_k D_k \nabla f(x_k),$$

where:
D_k is the $n \times n$ **descent direction matrix** (It allows to choose the direction in which the value of f will decrease)
$\alpha_k \in (0, +\infty)$ is the **stepsize** (It allows to manipulate the distance between x_k and x_{k+1}, so that the method does not stop before reaching the minimizer).

Descent direction selection: general rule

Theorem

Suppose D is symmetric positive definite. Then for any $x \in \mathbb{R}^n$ such that $\nabla f(x) \neq 0$,

$$f(x - \alpha D \nabla f(x)) < f(x)$$

for α small enough.

By choosing each D_k positive definite we guarantee that in each iteration we decrease the value of f.

Descent direction selection: steepest descent I

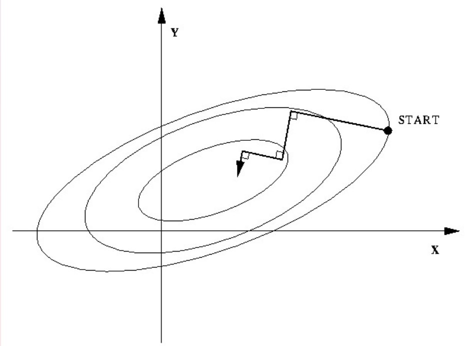
The simplest positive definite matrix that we know is the **identity matrix** I.
The method where $D_k = I$ for any k is called the steepest descent method.

Why is it called the steepest descent?
From mathematical analysis we know that $\nabla f(x)$ gives the direction in which the slope of the function is maximized.
So by taking $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$ we should decrease the value of f the fastest possible!

Descent direction selection: steepest descent II

Why do we need any other methods then?

The steepest descent method is known to **zigzag** like on this picture:



It makes its convergence to the minimum very slow.

Descent direction selection: Newton's method I

Alternative choice of the descent direction is given by the Newton (Newton-Raphson) method:

$$D_k := \left(\nabla^2 f(x_k)\right)^{-1} \quad \text{for } k = 0, 1, 2, \ldots$$

D_k defined in such a way need not be positive definite!
$\Longrightarrow$ when implementing the Newton method we should **replace** $(\nabla^2 f(x_k))^{-1}$ **by** I or a convex combination of the two **whenever the former is not positive definite**.

Descent direction selection: Newton's method II

Where does Newton descent direction form come from?

- The Taylor expansion of function f around x_k is:

$$f(x) = f(x_k)+\nabla f(x_k)^T(x-x_k)+\frac{1}{2}(x-x_k)^T\nabla^2 f(x_k)(x-x_k)+o(\|x-x_k\|^2)$$

- If we get rid of the last term, we get the quadratic approximation of f around x_k:

$$f(x) \approx f(x_k)+\nabla f(x_k)^T(x-x_k)+\frac{1}{2}(x-x_k)^T\nabla^2 f(x_k)(x-x_k).$$

- We can now look for the minimum of this polynomial:

$$\nabla f(x_k)+\nabla^2 f(x_k)(x-x_k) = 0 \Longrightarrow x = x_k-\left(\nabla^2 f(x_k)\right)^{-1}\nabla f(x_k).$$

Newton's method chooses the direction where the minimum of the quadratic approximation of f lies.

Stepsize selection I

There are several ways of choosing the stepsize α_k.

1. **Optimal stepsize (minimization rule)**:
   We choose α_k minimizing the function
   $F_k(\alpha) := f(x_k - \alpha D_k \nabla f(x_k))$ over $(0, +\infty)$
2. **Limited optimization rule**:
   We choose α_k minimizing the function
   $F_k(\alpha) := f(x_k - \alpha D_k \nabla f(x_k))$ over $(0, s]$ for some given constant s.

Usually, we are not able to apply these rules using analytical tools, thus we implement them with the aid of line search algorithms (optimization algorithms for one-dimensional problems).

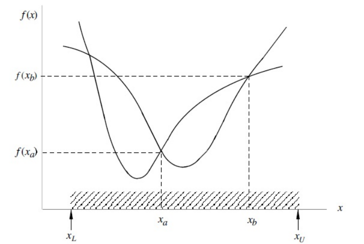
Line search algorithms I: interval reduction rules

The idea:

Suppose the function $f : \mathbb{R} \to \mathbb{R}$ is unimodal on interval $[a, b]$ with a minimum at point x^* that we want to find, and the points x_a, x_b are such that $x_L := a < x_a < x_b < b =: x_U$. Then there are 3 possibilities:

- $f(x_a) < f(x_b)$,
- $f(x_a) > f(x_b)$,
- $f(x_a) = f(x_b)$.

In each of these cases we can shrink the original search interval.



In general, the following rules can be applied:

- $f(x_a) < f(x_b) \Longrightarrow x^* \in [x_L, x_b]$,
- $f(x_a) > f(x_b) \Longrightarrow x^* \in [x_a, x_U]$,
- $f(x_a) = f(x_b) \Longrightarrow x^* \in [x_a, x_b]$.

If we decrease the interval in such a way iteratively, we may shrink it to an arbitrarily small interval containing x^*.

Line search algorithms III: golden-section search I

This can be implemented in the golden-section search method:

Input: a, b, f, $\varepsilon > 0$

$x_L := a$, $x_U := b$, $K := \frac{\sqrt{5}-1}{2}$

while $x_U - x_L > \varepsilon$ do

  $x_a := x_U - K(x_U - x_L)$, $x_b := x_L + K(x_U - x_L)$,

  $f_a = f(x_a)$, $f_b = f(x_b)$,

  if $f_a < f_b$

    $x_U := x_b$,

  elseif $f_a > f_b$

    $x_L := x_a$

  else

    $x_L := x_a$, $x_U := x_b$

$x^* := \frac{x_U + x_L}{2}$

Line search algorithms IV: golden-section search II

The same algorithm, but with less computations of function f:

$x_L := a$, $x_U := b$, $K := \frac{\sqrt{5}-1}{2}$
$x_a := x_U - K(x_U - x_L)$, $x_b := x_L + K(x_U - x_L)$,
$f_a = f(x_a)$, $f_b = f(x_b)$,
while $x_U - x_L > \varepsilon$ do
  if $f_a < f_b$
    $x_U := x_b$,
    $x_b := x_a$, $x_a := x_U - K(x_U - x_L)$
    $f_b := f_a$, $f_a := f(x_a)$
  elseif $f_a > f_b$
    $x_L := x_a$
    $x_a := x_b$, $x_b := x_L + K(x_U - x_L)$
    $f_a := f_b$, $f_b := f(x_b)$
  else
    $x_L := x_a$, $x_U := x_b$
    $x_a := x_U - K(x_U - x_L)$, $x_b := x_L + K(x_U - x_L)$,
    $f_a := f(x_a)$, $f_b := f(x_b)$
$x^* := \frac{x_U + x_L}{2}$

The idea:

- For any three points $(a, f(a))$, $(b, f(b))$, $(c, f(c))$ such that $a < c < b$ there is a unique polynomial of 2nd degree p such that $p(a) = f(a)$, $p(b) = f(b)$, $p(c) = f(c)$.
- Assuming that f behaves simialrly to p, we may expect that the minimum of f is close to the minimum of p.
- If the approximation is not good enough, we may use the minimizer $\bar{x}$ of polynomial p (together with c) to reduce the search interval.
- We may repeat the same procedure for the new interval.

Line search algorithms VI: quadratic interpolation II

Step 1: We search along the line to find initial points $a < c < b$ such that $f(a) > f(c) < f(b)$.

Step 2:

$x^* := b + \varepsilon$, $\bar{x} := c$

while $|x^* - \bar{x}| > \varepsilon$ do

  $x^* := \bar{x}$

  $\bar{x} := \frac{(b^2-c^2)f(a)+(c^2-a^2)f(b)+(a^2-b^2)f(c)}{2[(b-c)f(a)+(c-a)f(b)+(a-b)f(c)]}$

  if $\bar{x} > c$

    $d := \bar{x}$

  else

    $d := c$, $c := \bar{x}$

  if $f(c) < f(d)$

    $b := d$

  else

    $a := c$ $c := d$

$x^* := \bar{x}$

The idea:

- For any **two** points $(a, f(a))$, $(b, f(b))$ such that $a < b$ there is a unique polynomial of 3rd degree p satisfying $p(a) = f(a)$, $p(b) = f(b)$, $p'(a) = f'(a)$, $p'(b) = f'(b)$.
- We can approximate the minimum of f by a local minimum of p. A 3rd degreee polynomial usually has one local minimum and one local maximum. We want the minimum to be in $[a, b]$. For this to be sure f should satisfy:

$$f'(a) < 0 \text{ and } (f'(b) \geqslant 0 \text{ or } f(b) \geqslant f(a)).$$

- If the approximation using the minimizer of p, $\bar{x}$, is not good enough, we can divide the interval $[a, b]$ into two subintervals $[a, \bar{x}]$ and $[\bar{x}, b]$ and continue the search on one of them.

Line search algorithms VIII: cubic interpolation II

Step 1: We search along the line to find initial points $a < b$ such that $f'(a) < 0$ and $(f'(b) \geqslant 0$ or $f(b) \geqslant f(a))$.

Step 2:

$x^* := b + \varepsilon$, $\bar{x} := a - \varepsilon$

while $|x^* - \bar{x}| > \varepsilon$ do

$\quad x^* = \bar{x}$

$\quad z := \frac{3(f(a)-f(b))}{b-a} + f'(a) + f'(b)$, $w = \sqrt{z^2 - f'(a)f'(b)}$

$\quad \bar{x} := b - \frac{f'(b)+w-z}{f'(b)-f'(a)+2w}(b-a)$

$\quad$ if $f'(\bar{x}) \geqslant 0$ or $f(\bar{x}) \geqslant f(a)$

$\quad\quad b := \bar{x}$

$\quad$ else

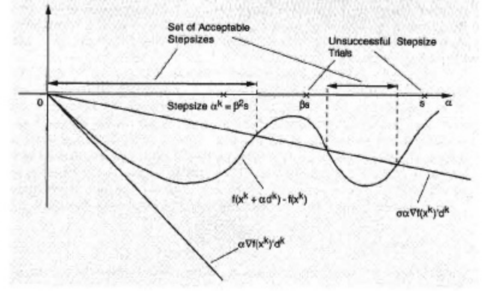
$\quad\quad a := \bar{x}$

$x^* := \bar{x}$

3. **Armijo rule (inexact line search)**
We take the constants $s > 0$ (the end of the search interval), $\beta \in (0,1)$ (the rate of reducing the search interval) and $\sigma \in (0,1)$ (slope reduction factor), and make the loop:

$\alpha := s$
while $f(x_k - \alpha D_k \nabla f(x_k)) \geqslant f(x_k) - \sigma\alpha\nabla f(x_k)^T D_k \nabla f(x_k)$ do
  $\alpha := \beta\alpha$

The final value of α is our α_k.



Stepsize selection III

4. ***n*-dimensional quadratic approximation:**
   We take the value minimizing the Taylor 2nd order approximation of $f(x_k - \alpha D_k \nabla f(x_k))$:

$$\alpha_k := \frac{\nabla f(x_k)^T \nabla f(x_k)}{\nabla f(x_k)^T \nabla^2 f(x_k) \nabla f(x_k)} \quad \text{for } k = 0, 1, 2, \ldots$$

   for the steepest descent method or

$$\alpha_k := 1 \quad \text{for } k = 0, 1, 2, \ldots$$

   for Newton's method.

5. **Diminishing stepsize:**
   Any sequence α_k such that $\alpha_k \searrow 0$ and $\sum_{k=0}^{\infty} \alpha_k = +\infty$.

6. **Constant stepsize:**

$$\alpha_k \equiv \text{const.}$$