Optimization in higher dimensions

- Theoretical aspects
- Gradient descent methods
- Newton's method
- Other methods

Higher dimensions

⋆ we consider functions f defined on $K = \overline{O}$ where $O \subset \mathbb{R}^n$ is open, smooth and connected.

⋆ the objective is to solve problems of the form

$$\min_{x \in K} f(x)$$

⋆ most of the theoretical aspects regarding existence and uniqueness of minimizers are similar to the one dimensional case: however, all partial derivatives need to be taken into account, and the notions of gradient and Hessian are essential

⋆ once a descent direction is found, we come back to one-dimensional algorithms when looking along this direction in order to decrease f

Optimization in higher dimensions

- Theoretical aspects
- Gradient descent methods
- Newton's method
- Other methods

Partial derivatives

$\star$ for simplicity, some results are stated for $f : \mathbb{R}^n \to \mathbb{R}$, but they apply to f defined on more restricted "nice" domains

$\star$ as usual, we denote by $e_i, i = 1, ..., n$ the canonical basis of $\mathbb{R}^n$

$$e_i = (..., 0, 1, 0, ...) \text{ only component } i \text{ is non-zero equal to } 1$$

Definition 1 (Partial derivatives, gradient, Hessian)

Consider a function $f : \mathbb{R}^n \to \mathbb{R}$. The partial derivative with respect to x_i is

$$\frac{\partial f}{\partial x_i}(x) = \lim_{t \to 0} \frac{f(x + te_i) - f(x)}{t}$$

In practice, $\frac{\partial f}{\partial x_i}$ is computed by differentiating f w.r.t x_i, supposing that the other coordinates are constant.

The gradient vector contains all partial derivatives: $\nabla f(x) = (\frac{\partial f}{\partial x_i}(x))_{i=1,...,n}$.

The Hessian matrix contains all combinations of two successive partial derivatives: $\mathcal{D}^2 f(x) = (\frac{\partial^2 f}{\partial x_i \partial x_j})_{i,j=1,...,n}$.

$\star$ note that f is of class C^2 then $D^2 f(x)$ is a symmetric matrix (result known as Schwarz's theorem)

Basic examples

1. $f(x) = \|x\|^2 = x_1^2 + ... + x_n^2$

$$\nabla f(x) = 2x, \quad D^2 f(x) = 2\,\mathrm{Id}$$

where Id is the identity matrix.

2. $f(x) = \frac{1}{2}x^T Ax - b^T x$

$$\nabla f(x) = Ax - b, \quad D^2 f(x) = A$$

Directional and Fréchet derivatives

Definition 2 (Directional (Gateaux) derivative)

$f : \mathbb{R}^n \to \mathbb{R}$ is differentiable at x in direction d if the one dimensional function $t \mapsto f(x + td)$ is differentiable at $t = 0$.

Definition 3 (Fréchet derivative)

$f : \mathbb{R}^n \to \mathbb{R}$ is Fréchet differentiable at x if there exists a bounded linear mapping $L : \mathbb{R}^n \to \mathbb{R}$ such that for $h \in \mathbb{R}^n$ with $|h|$ small enough we have

$$f(x + h) = f(x) + Lh + o(h)$$

$\star$ the application L is denoted by $f'(x)$. When f is C^1 we simply have $f'(x)(h) = \nabla f(x) \cdot h$.

$\star$ in general Fréchet differentiability implies the existence of directional derivatives, but the converse is false

$\star$ if the partial derivatives exist and are continuous then the function is Fréchet differentiable

$\star$ for more subtle differences and implications consult a real analysis course: e.g. [Differential Calculus, by Henri Cartan]

Taylor expansion in higher dimensions

Consider $f : \mathbb{R}^n \to \mathbb{R}$. Then

- if f is of class C^1

$$f(x+h) = f(x) + f'(x)(h) + o(|h|) \text{ as } |h| \to 0$$

$$f(x+h) = f(x) + \nabla f(x) \cdot h + o(|h|) \text{ as } |h| \to 0$$

- if f is of class C^2

$$f(x+h) = f(x) + f'(x)(h) + \frac{1}{2!} f''(x)(h,h) + o(|h|^2) \text{ as } |h| \to 0$$

$$f(x+h) = f(x) + \nabla f(x) \cdot h + \frac{1}{2} h^T D^2 f(x) h + o(|h|^2) \text{ as } |h| \to 0$$

⋆ again it is possible to write the remainder in Lagrange form
⋆ recall that the second derivative (in the sense of Fréchet) of a function is a bilinear form. Why? For each differentiation you need to choose a direction...

compute first $f'(x)(h_1)$ and then $(f'(x)(h_1))'(h_2) \longrightarrow f''(x)(h_1, h_2)$

In the same way as in dimension one we have the following

Proposition 4

⋆ *If f is continuous it attains its extremal values on compact sets.*
⋆ *If $f : \mathbb{R}^n \to \mathbb{R}$ is continuous and "infinite at infinity" i.e.*

$$|f(x)| \to \infty \text{ as } |x| \to \infty$$

then f admits minimizers on $\mathbb{R}^n$.

Definition 5

A matrix $A \in \mathcal{M}_n(\mathbb{R})$ is called:

- **positive definite** if for every vector $x \in \mathbb{R}^n \setminus \{0\}$
$$x^T Ax > 0$$
- **positive semi-definite** if for every vector $x \in \mathbb{R}^n$
$$x^T Ax \geq 0$$

$\star$ these notions are often useful when dealing with optimization problems

$\star$ when A is also symmetric, it is possible to give a characterization of the above definition in terms of the eigenvalues of A:

- A is positive definite if all its eigenvalues are positive
- A is positive semi-definite if all its eigenvalues are non-negative

$\star$ recall that symmetric matrices are diagonalizable and there exists an orthonormal basis made of eigenvectors

Basic optimality conditions

Proposition 6

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a C^1 function. If x^* is a local minimum (maximum) of f then $\nabla f(x^*) = 0$. Moreover, if f is of class C^2 then the Hessian matrix $D^2 f(x^*)$ is positive (negative) semi-definite.

Conversely, if f is of class C^2, $\nabla f(x^*) = 0$ and $D^2 f$ is positive semi-definite in a neighborhood of x^* then x^* is a local minimum of f.
As a consequence, if f is of class C^2, $\nabla f(x^*) = 0$ and $D^2 f(x^*)$ is positive definite then x^* is a local minimum of f.

$\star$ The proof comes immediately from the Taylor expansion formulas.

Euler inequalities

⋆ what happens when we minimize on a closed convex set $K \subset \mathbb{R}^d$?

Proposition 7

Let K be a convex set and x^* be a minimum of f on K. Suppose that J is differentiable at x^*. Then for every $x \in K$ we have

$$\nabla f(x^*) \cdot (x - x^*) \geq 0.$$

⋆ Proof: just write the directional derivative at x^* in the direction $x - x^*$.

⋆ compare with the 1D case!

Convex functions again...

$\star$ In higher dimensions convex functions give the same advantages regarding the existence, unicity and convergence of algorithms as in dimension one.

Definition 8 (Convex functions)

A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be convex if for every $x, y \in \mathbb{R}^n$ and for every $t \in (0,1)$ we have

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y)$$

$\star$ for strict convexity the inequality is strict.

Equivalent definitions: f is convex iff

- f is below any affine section
- f is above its tangent planes
- any 1D "slice" is a convex 1D function

Useful characterizations

Proposition 9

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a C^1 function. The following statements are equivalent:

1. *f is convex*
2. $f(y) \geq f(x) + \nabla f(x) \cdot (y - x),\ \forall x, y \in \mathbb{R}^n$
3. $(\nabla f(x) - \nabla f(y)) \cdot (x - y) \geq 0,\ \forall x, y \in \mathbb{R}^n$

Proof: Exercise!

Proposition 10

Let $f : \mathbb{R} \to \mathbb{R}$ be a C^2 function. Then f is convex if and only if the Hessian matrix $\mathcal{D}^2 f$ is positive semi-definite everywhere.

$\star$ we say that f is α-convex for some $\alpha > 0$ if the Hessian matrix has eigenvalues $\geq \alpha > 0$.

Optimality conditions

$\star$ for convex functions, the usual necessary optimality conditions are also sufficient

Proposition 11

$\star$ *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function and x^* be a point such that $\nabla f(x^*) = 0$. Then x^* is a global minimum of f.*

$\star$ *Let $f : K \to \mathbb{R}$ be a convex function defined on a convex subset K of $\mathbb{R}^n$. Then if $x^* \in K$ verifies*

$$\nabla f(x^*) \cdot (x - x^*) \geq 0$$

for every $x \in K$ then x^* is a global minimum of f on K.

Proof: $f(x) \geq f(x^*) + \nabla f(x^*) \cdot (x - x^*), \ \forall x \in K$

Optimization without Calculus

[Charles L. Byrne, *A first Course in Optimization*]
[Niven, I. *Maxima and Minima Without Calculus*]
$\star$ sometimes, solutions can be found without the need of calculus or algorithms
Basic ingredients.

- $x^2 \geq 0$: the most basic inequality
- **AM-GM**:
$$x_i \geq 0 \Rightarrow \frac{x_1 + ... + x_n}{n} \geq (x_1 ... x_n)^{1/n}$$
- **Generalized AM-GM** (or just convexity of the $-\log$ function):
$$x_i > 0, a_i \geq 0, \sum_{i=1}^n a_i = 1 \Longrightarrow x_1^{a_1} ... x_n^{a_n} \leq a_1 x_1 + ... + a_n x_n$$
- **Cauchy-Schwarz**: $a_i, b_i \in \mathbb{R}$
$$\left(\sum_{i=1}^n a_i b_i \right)^2 \leq \left(\sum_{i=1}^n a_i^2 \right) \left(\sum_{i=1}^n b_i^2 \right) \text{ or } |\mathbf{a} \cdot \mathbf{b}| \leq |\mathbf{a}||\mathbf{b}|$$

Examples

1. minimize $f(x,y) = \dfrac{12}{x} + \dfrac{18}{y} + xy$ on $(0,\infty)^2$
2. maximize $f(x,y) = xy(72-3x-4y)$
3. minimize $f(x,y) = 4x + \dfrac{x}{y^2} + \dfrac{4y}{x}$ on $(0,\infty)^2$
4. maximize $f(x,y,z) = 2x+3y+6z$ when $x^2+y^2+z^2=1$
5. maximize $f(x,y,z) = 2x+3y+6z$ when $x^p+y^p+z^p=1$, $p>1$.

Example 1

⋆ minimize $f(x,y) = \dfrac{12}{x} + \dfrac{18}{y} + xy$ on $(0,\infty)^2$

Since we are dealing with positive numbers apply AM-GM:

$$\frac{12}{x} + \frac{18}{y} + xy \geq 3 \cdot \left(\frac{12}{x}\frac{18}{y}xy\right)^{1/3} = 3 \cdot 6 = 18.$$

⋆ Therefore the lower bound of the above expression is 18

⋆ it is attained when $\frac{12}{x} = \frac{18}{y} = xy$ leading to $x = 2, y = 3$.

⋆ the same technique can be applied for Examples 2 and 3

Example 4

$\star$ maximize $f(x, y, z) = 2x + 3y + 6z$ when $x^2 + y^2 + z^2 = 1$
Here it is possible to use Cauchy-Schwarz:

$$(2x + 3y + 6z)^2 \leq (2^2 + 3^2 + 6^2)(x^2 + y^2 + z^2) = 49$$

with equality of (x, y, z) and $(2, 3, 6)$ are colinear.

$\star$ recognize cases when the solution can be found explicitly.
$\star$ provide examples on which to test numerical algorithms!

Optimization in higher dimensions

- Theoretical aspects
- Gradient descent methods
- Newton's method
- Other methods

Basic idea

Suppose that f is C^1 (at least). Then the Taylor expansion says

$$f(x+h) = f(x) + \nabla f(x) \cdot h + o(|h|), |h| \to 0$$

Basic idea

Suppose that f is C^1 (at least). Then the Taylor expansion says

$$f(x+h) \approx f(x) + \nabla f(x) \cdot h$$

With this in mind, the following definition is natural

Definition 12 (Descent direction)

A direction $d \in \mathbb{R}^n$ is called a descent direction for f at x if $\nabla f(x) \cdot d < 0$

This gives the following natural result

Proposition 13

If d is a descent direction for f at x, then going from x along d with a small step increment decreases the value of f.
Equivalently, if $q(t) = f(x+td)$ then $q'(0) < 0$.

Indeed, by the chain rule, $q'(0) = \nabla f(x) \cdot d < 0$.

Gradient descent algorithm

$\star$ the direction which gives (asymptotically) the steepest descent is the opposite of the gradient

Indeed, if $|d| = |\nabla f|$ then by the Cauchy-Schwarz inequality

$$|d \cdot \nabla f| \leq |d||\nabla f| = |\nabla f|^2$$

Therefore

$$d \cdot \nabla f \geq -|\nabla f|^2$$

and the minimum is attained for $d = -\nabla f$

Algorithm 1 (Generic gradient descent)

Initialization: *Choose a starting point* x_0 *and set* $i = 0$

Step *i:*

- *compute* $f(x_i)$ *and* $\nabla f(x_i)$
- *choose a step size* t *and set*

$$x_{i+1} = x_i - t\nabla f(x_i)$$

Simplest algorithm: fixed step

⋆ fix the descent step $t = t_0$, the tolerance $\varepsilon > 0$ and run the algorithm

Algorithm 2 (GD with fixed step)

Initialization: *Choose a starting point* x_0 *and set* $i = 0$

Step i:

- *compute* $f(x_i)$ *and* $\nabla f(x_i)$
- *set*

$$x_{i+1} = x_i - t_0 \nabla f(x_i)$$

- *check convergence*
  - $|\nabla f(x_i)| < \varepsilon$ *(the gradient is too small)*
  - $|x_{i+1} - x_i| < \varepsilon$ *(the position of the optimum does not change much)*
  - $|f(x_{i+1}) - f(x_i)| < \varepsilon$ *(the objective function does not change much)*

⋆ the algorithm is stopped in one of the following situations

- convergence is reached
- maximum number of iterations/function evaluations is reached

⋆ the choice of t_0 is essential

Quadratic case

⋆ simple example in where the solution is known
⋆ easy to visualize in 2D

$$f(x) = \frac{1}{2}x^T Ax - b \cdot x$$

with A symmetric positive definite
⋆ recall that A is positive semi-definite if $Ax \cdot x \geq 0$ for every x
⋆ recall that A is positive definite if $Ax \cdot x \geq 0$ and $Ax \cdot x = 0 \Rightarrow x = 0$.

Compute the gradient: two options

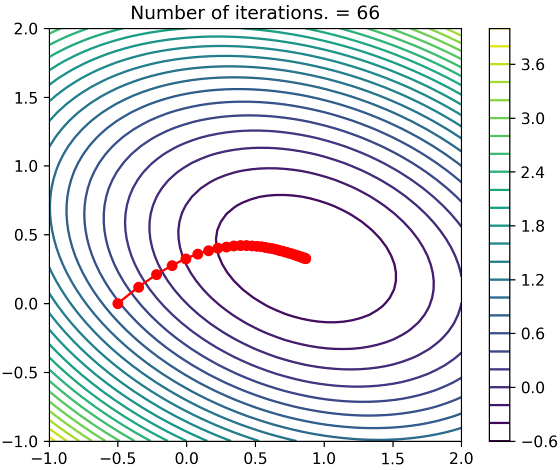
- write down the formulas in terms of $x = (x_1, ..., x_N)$ and compute the partial derivatives (a bit long)
- write $f(x+h)$ for h small and identify the derivative from there as the linear part of the decomposition, proving that what remains is $o(h)$ as $|h| \to 0$

⋆ in the end $\nabla f(x) = Ax - b$
⋆ note that minimizing f amounts to solving the system $Ax = b$

Concrete quadratic example

$A = \begin{pmatrix} 1 & 0.4 \\ 0.4 & 2 \end{pmatrix}, b = (1,1), x_0 = (-0.5, 0)$

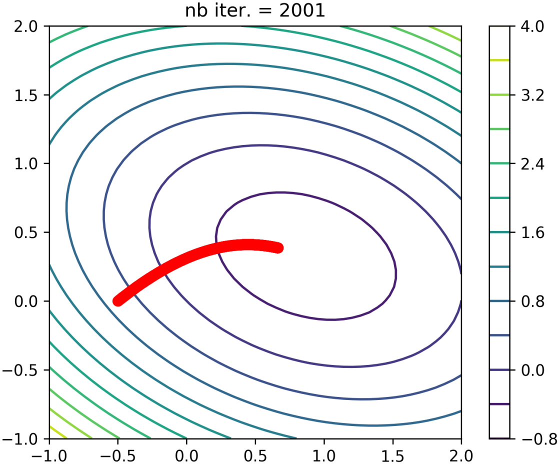
Step size $t = 0.1$: the algorithm converges



Concrete quadratic example

$A = \begin{pmatrix} 1 & 0.4 \\ 0.4 & 2 \end{pmatrix}, b = (1,1), x_0 = (-0.5, 0)$

Step size $t = 0.001$: no convergence before reaching max number of iterations...



For which steps we have convergence?

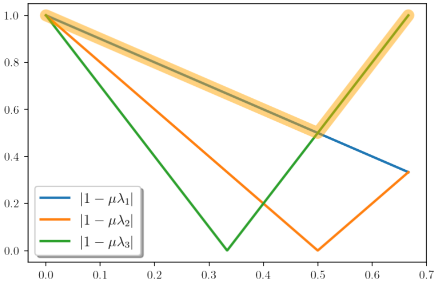
- $\star$ In the quadratic case the GD algorithm is
$$x_{k+1} = x_k - t(Ax_k - b)$$
- $\star$ subtracting the solution x^* and using $Ax^* = b$ we get
$$(x_{k+1} - x^*) = (I - tA)(x_k - x^*) = (I - tA)^k(x_0 - x^*).$$
- $\star$ it is well known that $B^k \to 0$ if and only if $\rho(B) < 1$, where
$$\rho(B) = \max_{i=1,\dots,n} \lambda_i(B) \text{ is the spectral radius of } B.$$
- $\star$ the GD algorithm converges if and only if $\max_{i=1,\dots,n} |1 - t\lambda_i(A)| < 1$
- $\star$ a simple computation shows that GD converges if and only if $t \in (0, 2/\lambda_n(A))$

The best convergence ratio

⋆ the ratio of convergence is $\rho(I - tA)$

Question: Minimize this ratio for $t \in (0, 2/\lambda_n)$

⋆ minimize the maximum of $|1 - t\lambda_i|$, $i = 1, ..., n$



⋆ a brief graphical argument shows that

$$\rho(I - tA) = \max\{|1 - t\lambda_1|, |1 - t\lambda_n|\}$$

⋆ the spectral radius is minimized when $t = 2/(\lambda_1 + \lambda_n)$.

Steepest Descent

⋆ In an ideal world, one would like to minimize $q(t) = f(x_i - t\nabla f(x_i))$

Algorithm 3 (GD with Steepest Descent)

Initialization: *Choose a starting point* x_0 *and set* $i = 0$

Step i:

- *compute* $f(x_i)$ *and* $\nabla f(x_i)$
- *choose the step size* t_{opt} *which minimizes the (one-dimensional) function* $q(t) = f(x_i - t\nabla f(x_i))$ *and set*

$$x_{i+1} = x_i - t_{opt}\nabla f(x_i)$$

⋆ note that the second step is an optimization problem in itself: if this cannot be solved explicitly, this algorithm is not too efficient.

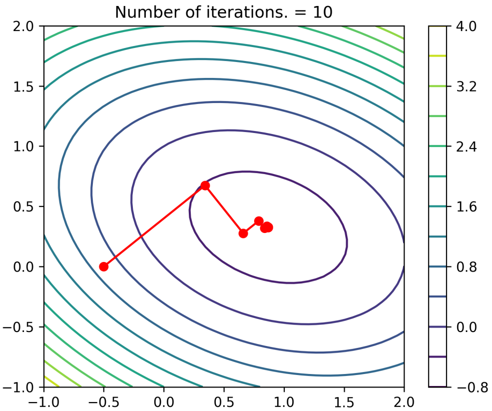
Back to the quadratic function

⋆ $f(x) = \frac{1}{2}x^T Ax - b \cdot x,\ \nabla f(x) = Ax - b$
⋆ in the following denote $g_i = \nabla f(x_i)$
⋆ $q(t) = f(x_i - tg_i)$ is a quadratic function of t
⋆ $q'(t) = \nabla f(x_i - tg_i) \cdot (-g_i) = -g_i^T(Ax_i - b) + tg_i^T Ag_i$
⋆ a simple computation yields

$$q'(t) = 0 \Longrightarrow t_{opt} = \frac{g_i^T g_i}{g_i^T Ag_i}$$

⋆ in particular the gradient at the next point $x_i - t_{opt}g_i$ is orthogonal to the actual gradient g_i
⋆ note that the knowledge of the optimal descent step is strictly related to the objective function

What happens in practice



Proposition 14

When using the Gradient Descent algorithm with optimal descent step, any two consecutive descent directions are orthogonal.

Orthogonality of consecutive descent directions

Two ideas of proof:

1. $q'(t) = 0 \iff \nabla f(x_i - t\nabla f(x_i)) \cdot \nabla f(x_i) = 0$

2. Let $d_i = \nabla f(x_i)$ be the ith gradient descent direction. If $d_i \cdot d_{i+1} \neq 0$ then the previous step was not optimal!

- $d_i \cdot d_{i+1} > 0$: then $-d_i$ is still a descent direction
- $d_i \cdot d_{i+1} < 0$: then d_i is still a descent direction

$\star$ this brings us to one important idea

Other descent directions

The opposite of the gradient is not the only descent direction! For example, every symmetric positive definite matrix A generates a descent direction

$$d = -A\nabla f(x).$$

but more on this fact later on in the course...

Accelerate convergence: variable step

⋆ modify the step at each iteration, making sure that the obj. function decreases
⋆ trivial line-search algorithm

Algorithm 4 (GD with variable step)

Initialization: *Choose a starting point* x_0, *starting step* $t = t_0$, *maximum step* t_M, $\eta_+ > 1$, $\eta_- < 1$ *and set* $i = 0$
Step i:

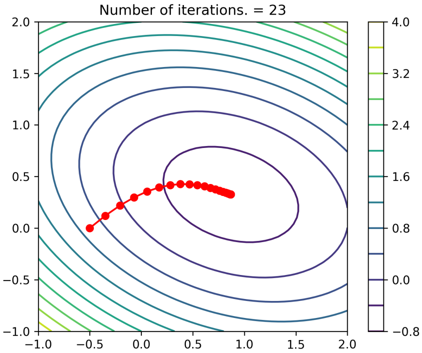
- *compute* $f(x_i)$ *and* $\nabla f(x_i)$
- *set a temporary new point*

$$x_{temp} = x_i - t\nabla f(x_i)$$

- **If** $f(x_{i+1}) < f(x_i)$
  - **Accept the iteration**: $x_{i+1} = x_{temp}$
  - **increase the step size**: $t = \min\{t \cdot \eta_+, t_M\}$
- **Else**
  - **Refuse the iteration**
  - **decrease the step size**: $t = t \cdot \eta_-$
- *check convergence (additionally you may check if* t *is too small)*

Back to the quadratic example

Step size $t = 0.5, t_M = 10, \eta_+ = 1.1, \eta_- = 0.8, \varepsilon = 10^{-6}$: the algorithm converges faster



⋆ a simple trick accelerates the convergence

GD with Armijo line-search

Algorithm 5 (GD with Armijo line-search)

Initialization: *Choose a starting point* x_0, *an initial step* $t = t_0$, $\eta > 1$, $m_1 \in (0, 0.5)$ *and set* $i = 0$

Step i:

- *compute* $f(x_i)$ *and* $\nabla f(x_i)$
- **line-search**: $q(t) = f(x_i - t\nabla f(x_i))$, *set* $t = t_0$
- **while**: $m_1 q'(0) < (q(t) - q(0))/t$ **do** $t \leftarrow t/\eta$
- *set*

$$x_{i+1} = x_i - t\nabla f(x_i)$$

⋆ the above algorithm is similar to the GD with adaptive step, but is somewhat stronger since it imposes a quantified descent condition

⋆ note that $q'(0) < 0$ so in the end

$$\frac{q(t) - q(0)}{t} \leq m_1 q'(0) < 0$$

which guarantees that $q(t) < q(0)$

⋆ as in the lectures regarding the 1D case it is also possible to formulate GD algorithms with Goldstein-Price or Wolfe line-search routines

Convergence of the GD algorithm

Proposition 15

For a given C^1 function f denote by Γ_f the set of its critical points

$$\Gamma_f = \{x \in \mathbb{R}^n : \nabla f(x) = 0\}$$

and suppose that f admits minimizers on $\mathbb{R}^n$. Furthermore, suppose that the set $\mathcal{S} = \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$ is bounded.

The trajectory (x_n) of a GD algorithm with Steepest-Descent (Armijo, Goldstein-Price, ...) line-search possesses limiting points and any such limiting point belongs to the set of critical points Γ_f.

Proof idea for Steepest Descent:

$\star$ we have $\min f \leq f(x_{k+1}) \leq f(x_k)$. Therefore $(x_k) \subset \mathcal{S}$

$\star$ suppose that $\nabla f(x_k)$ does not converge to zero and arrive at a contradiction

$\star$ this kind of argument could be made rigorous using a point to set definition of the optimization algorithm also in the case where line-search is used

Limiting points of GD

Consider the ODE $\dfrac{d}{dt}x(t) = -\nabla f(x(t))$: the trajectory dictated by the gradient

$\star$ Note that the gradient descent is just a discretization for this ODE!

$\star$ $\nabla f(x(t)) = \nabla f(x(t)) - \nabla f(x^*) \approx D^2 f(x^*)(x(t) - x^*)$

$$-\nabla f(x(t)) \cdot (x^* - x(t)) \approx (x(t) - x^*)^T D^2 f(x^*)(x(t) - x^*).$$

We have the following situations:

A $D^2 f(x^*)$ is positive definite: then x^* can be a limiting point for GD as it is a local minimum

B $D^2 f(x^*)$ is negative definite: then the trajectory $x(t)$ will never get close to x^* provided it does not start there.

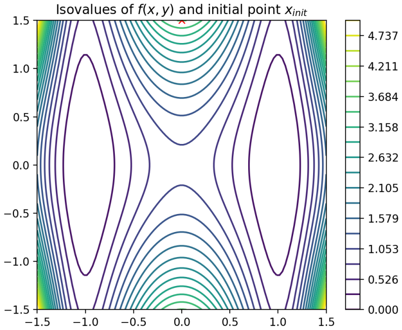
C $D^2 f(x^*)$ is indefinite: then x^* is a saddle point of f. In order to reach x^* you need to start in a particular set S of dimension less than n: practically, this is extremely unlikely.

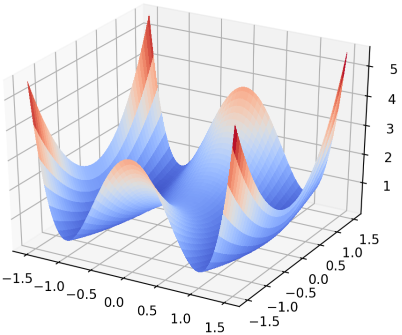
Example: Saddle point

$f(x,y) = (x^2-1)^2(y^2+1) + 0.2y^2$

$\star$ $f \geq 0$ and f attains its minimum for $(\pm 1, 0)$

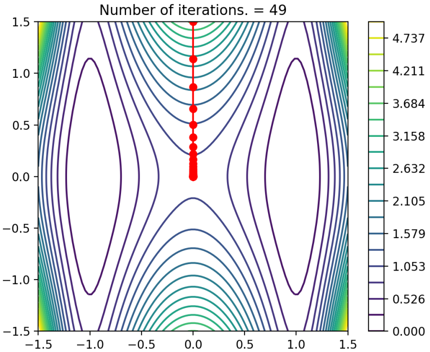
$\star$ $(0,0)$ is a saddle point: $\nabla f(0,0) = (0,0)$, $D^2 f(0,0) = \begin{pmatrix} -4 & 0 \\ 0 & 2.4 \end{pmatrix}$





Behavior of GD with different initializations

$\star$ Initializing on the "ridge" that passes through the saddle point: $x_0 = (0, 1.5)$

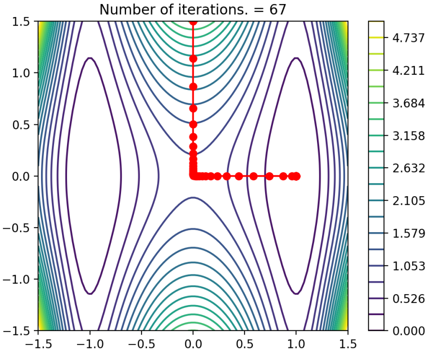


$\star$ the algorithm converges to the saddle point

$\star$ the gradient information "does not see" that there are regions where the value of f is lower

Behavior of GD with different initializations (2)

⋆ A slightly perturbed initialization: $x_0 = (10^{-6}, 1.5)$



⋆ the algorithm converges to a local minimum and avoids the saddle point

⋆ Remember: avoid initializations that may be biased with respect to the function f (e.g. $x_0 = 0$, etc...). You may use a random number generator to add some random noise to your initial condition. Also, repeat simulation with multiple initializations in order to avoid saddle points and local minima

Convergence of GD for quadratic functionals

⋆ Consider $f(x) = \frac{1}{2}x^T Ax - b^T x$ with A symmetric positive-definite and denote by $0 < \lambda_{\min} < \lambda_{\max}$ the smallest and largest of its eigenvalues

⋆ the gradient is $\nabla f(x) = Ax - b$ and x^* verifies $Ax^* = b$

⋆ inaccuracy in terms of the objective:

$$E(x) = f(x) - f(x^*) = \frac{1}{2}(x - x^*)^T A(x - x^*) = \frac{1}{2}\|x - x^*\|_A^2$$

⋆ denoting $g_i = Ax_i - b$ (the gradient at iteration i) we previously found that the optimal step for the Steepest descent is

$$t_i = \frac{g_i \cdot g_i}{g_i^T A g_i}, \text{ which gives } x_{i+1} = x_i - \frac{g_i \cdot g_i}{g_i^T A g_i} g_i$$

⋆ explicit computation gives

$$E(x_{i+1}) = \left(1 - \frac{(g_i \cdot g_i)^2}{[g_i^T A g_i][g_i^T A^{-1} g_i]}\right) E(x_i)$$

Lemma: (Kantorovich) if Q is the condition number of a positive definite and symmetric matrix A (ratio largest/smallest eigenvalues) then

$$\frac{(x \cdot x)^2}{[x^T Ax][x^T A^{-1} x]} \geq \frac{4Q}{(1+Q)^2}.$$

GD with steepest descent

⋆ Consider the norm given by A: $\|x\|_A^2 = x^T Ax$.

Proposition 16 (Convergence ratio: Steepest Descent, quadratic case)

The Steepest Descent algorithm applied to a strongly convex quadratic form f with condition number Q converges linearly with the convergence ratio at most

$$1 - \frac{4Q}{(1+Q)^2} = \left(\frac{Q-1}{Q+1}\right)^2.$$

More precisely, we have

$$f(x_N) - \min f \leq \left(\frac{Q-1}{Q+1}\right)^{2N} [f(x_0) - \min f].$$

Another interpretation is:

$$\|x_N - x^*\|_A \leq \left(\frac{Q-1}{Q+1}\right)^N \|x_0 - x^*\|_A.$$

⋆ note that if Q is large then the convergence is slow: this is observed in practice

Convergence rate: α-convex case

Proposition 17

Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is α-convex, i.e.

$$f(y) \geq f(x) + \nabla f(x) \cdot (y - x) + \frac{\alpha}{2}|x - y|^2$$

for some $\alpha > 0$. Moreover, suppose that ∇f is Lipschitz, i.e. there exists a constant $L > 0$ such that

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

Then, if t_0 is small enough, then the Gradient Descent algorithm with fixed step $t = t_0$ converges linearly to the global optimum.

Proof: As in the one dimensional case, simply define the fixed-point application

$$\mathcal{F}_t(x) = x - t\nabla f(x),$$

which is a contraction for t small enough.

⋆ therefore, the recurrence $x_{n+1} = \mathcal{F}_t(x_n)$ converges to the fixed point x^* which verifies $\nabla f(x^*) = 0$ and is thus the global minimum.

⋆ the hypotheses could be somewhat relaxed, but the theoretical proof gets more involved

Interpretation

⋆ it is possible to prove that

$$|\mathcal{F}_t(x) - \mathcal{F}_t(y)| \leq (1 - 2\alpha t + L^2 t^2)^{1/2} |x - y|$$

⋆ for $t \in (0, 2\alpha/L^2)$ we have $(1 - 2\alpha t + L^2 t^2) \in (0, 1)$ so $\mathcal{F}_t$ is a contraction

⋆ in particular $|x_{n+1} - x^*| \leq (1 - 2\alpha t + L^2 t^2)^{1/2} |x_n - x^*|$

⋆ for $t = \alpha/L^2$ the contraction factor is $(1 - \alpha^2/L^2)^{1/2}$

⋆ the eigenvalues of $D^2 f(x)$ are in $[\alpha, L]$ so the condition number verifies

$$1 \leq Q = \frac{\lambda_{\max}}{\lambda_{\min}} \leq \frac{L}{\alpha}.$$

⋆ the convergence is linear, but the ratio of convergence is (roughly) dictated by the condition number of the Hessian $D^2 f(x)$ at x^*

Important observation

Note that in the convergence estimates for the Gradient descent the condition number Q is important for evaluating the speed of convergence!

Optimal convergence speed

Proposition 18

Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is α-convex, i.e.

$$f(y) \geq f(x) + \nabla f(x) \cdot (y - x) + \frac{\alpha}{2}|x - y|^2$$

for some $\alpha > 0$. Moreover, suppose that ∇f is Lipschitz, i.e. there exists a constant $L > 0$ such that

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

Then, then the Gradient Descent algorithm with fixed step t converges linearly to the optimum for all initalizations x_0 if and only if $t \in (0, 2/L)$.
Moreover, the optimal convergence speed is attained for the step $t_{\mathrm{opt}} = 2/(L + \alpha)$ and the optimal convergence ratio γ_{opt} verifies

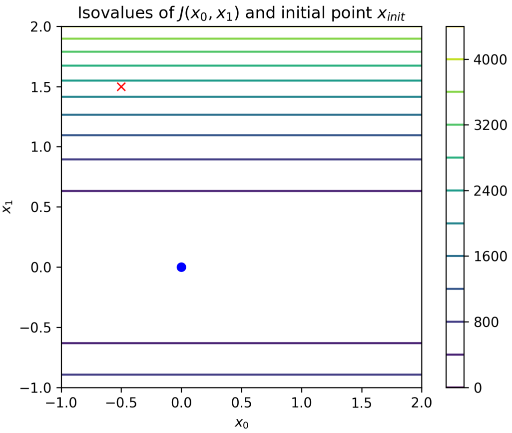
$$\gamma_{\mathrm{opt}} = \frac{1 - \alpha/L}{1 + \alpha/L}, \|x_{n+1} - x^*\| \leq \gamma_{\mathrm{opt}}\|x_n - x^*\|.$$

- ⋆ the proof uses the Taylor remainder theorem with exact remainder
- ⋆ see the course MAP435 by G. Allaire!
- ⋆ the optimal convergence speed is still bad if the condition number L/α is big.

Quadratic ill-conditioned problem

$f(x) = x^T A x,\;\; A = \begin{pmatrix} 0.1 & 0 \\ 0 & 2000 \end{pmatrix}, x_0 = (-0.5, 1.5), Q = 20000$

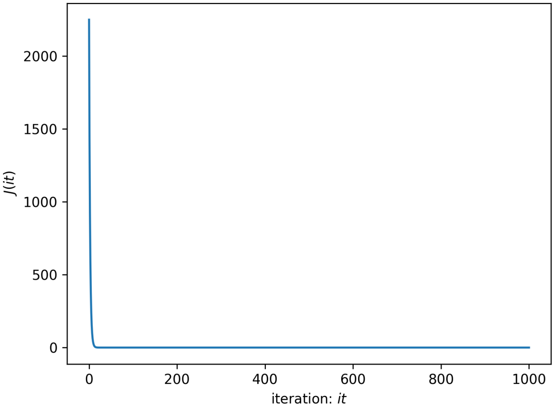
Geometry and Initialization:



Quadratic ill-conditioned problem

$f(x) = x^T Ax,\ \ A = \begin{pmatrix} 0.1 & 0 \\ 0 & 2000 \end{pmatrix}, x_0 = (-0.5, 1.5), Q = 20000$

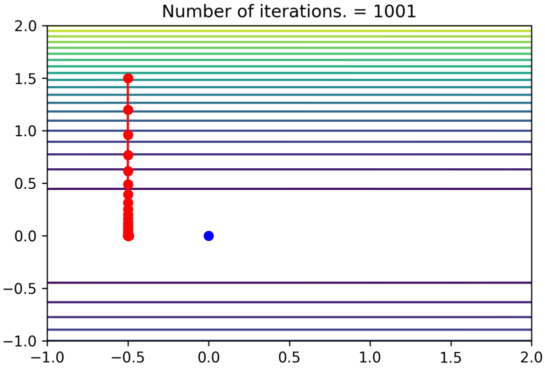
Fixed step, 1000 **iterations:** **algorithm seems to converge**



Quadratic ill-conditioned problem

$$f(x) = x^T A x,\ \ A = \begin{pmatrix} 0.1 & 0 \\ 0 & 2000 \end{pmatrix}, x_0 = (-0.5, 1.5), Q = 20000$$

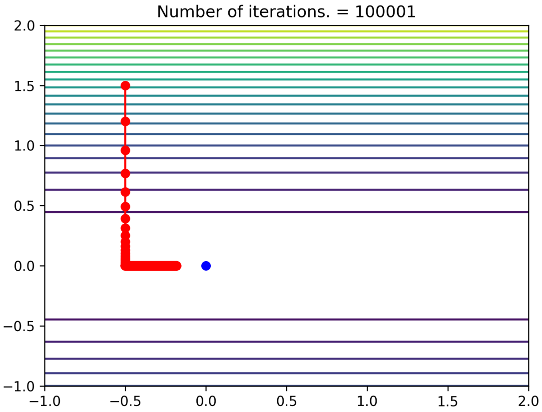
Fixed step, 1000 **iterations:**



Quadratic ill-conditioned problem

$$f(x) = x^T A x,\ \ A = \begin{pmatrix} 0.1 & 0 \\ 0 & 2000 \end{pmatrix}, x_0 = (-0.5, 1.5), Q = 20000$$

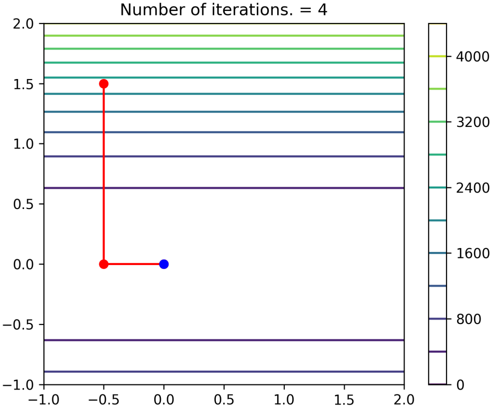
Fixed step, 10^5 iterations:



Quadratic ill-conditioned problem

$$f(x) = x^T A x, \;\; A = \begin{pmatrix} 0.1 & 0 \\ 0 & 2000 \end{pmatrix}, x_0 = (-0.5, 1.5), Q = 20000$$

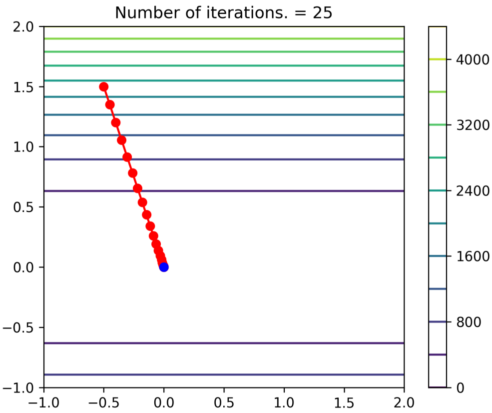
Optimal step: good, but not applicable to general functions



Quadratic ill-conditioned problem

$$f(x) = x^T A x,\ \ A = \begin{pmatrix} 0.1 & 0 \\ 0 & 2000 \end{pmatrix}, x_0 = (-0.5, 1.5), Q = 20000$$

Rescale using the Hessian: look at the function in the right coordinates



Conclusions for GD

- the GD algorithms usually converge to local minimizers under very weak hypothesis
- in the strongly convex case we can prove that the rate of convergence is linear
- the speed of convergence is dictated by the condition number of f: in cases where this condition number is large, the GD algorithm may fail to converge rapidly enough
- when the problem is ill-conditioned GD algorithms look at the optimization path in the wrong coordinates: the key to accelerating the convergence is to modify the geometry by rescaling some directions with respect to others!
- source of ill conditioning in practice: components of the gradients are orders of magnitude apart, different units of measure for different variables, etc.

Before going further: constraints

⋆ often the minimization is subject to some constraints

$$\min_{x \in K} f(x)$$

where K is defined via some analytic relations or inequalities

⋆ the theory of Lagrange multipliers is presented further on in the course, but there is a simple way to handle basic constraints: projection

⋆ suppose that K is closed and convex. Then for every $y \in \mathbb{R}^n$ the projection $P_K y$ is well defined and solves the problem

$$P_K(y) \leftarrow \min_{x \in K} |x - y|$$

Algorithm 6 (Projected GD)

Consider K a closed and convex set in $\mathbb{R}^n$ and let $x_0 \in K$ be an initial point. The solution of the problem

$$\min_{x \in K} f(x)$$

may be approximated using the iterative algorithm

$$x_{i+1} = P_K(x_i - t\nabla f(x_i))$$

Proposition 19 (Convergence of Projected GD)

Suppose that f is α-convex, differentiable and f' is L-Lipschitz. Then if the step t verifies $t \in (0, 2\alpha/L^2)$ then the GD algorithm with fixed step and projection on K converges to the unique solution.

Proof: The same as for the GD algorithm using the fact that the projection is a weak-contraction

$$|P_K x - P_K y| \leq |x - y|$$

⋆ Projected GD may seem good, but is of limited practical use: the main difficulty is how to compute P_K which is in itself an optimization problem

⋆ particular cases which are easy:

- $K = \prod_{i=1}^n [a_i, b_i]$: P_K is just the truncation operator on each coordinate
- $K = B(c, r)$ is a ball in $\mathbb{R}^d$: $P_K(x) = c + r(x - c)/|x - c|$
- $K = \{x : \sum_{i=1}^n v_i x_i = c\}$: affine hyperplanes - projection can be computed analytically

Projection on affine constraints

Suppose $K = \{x : Ax = b\}$ where A is an $m \times n$ matrix of rank m and $b \in \mathbb{R}^m$. We are interested in solving

$$P_K(y) = \text{argmin}_{x \in K} |x - y|^2$$

- Existence, uniqueness: $x \mapsto |x - y|^2$ is "∞ at infinity" and strictly convex, K is convex
- Euler inequality: $\langle \nabla_x |x^* - y|^2, v \rangle \geq 0$ for every $v \in \ker A$
- $x^* - y \in (\ker A)^\perp = \text{Im} A^T$ (**Exercise!**)
- $x^* = y + A^T \lambda \quad (\lambda \in \mathbb{R}^m$ contains the Lagrange multipliers)
- $Ax = b \Rightarrow b = Ax^* = Ay + AA^T \lambda$ so finally $\lambda = (AA^T)^{-1}(b - Ay)$
- In the end, use λ to find x^*:

$$x^* = y + A^T (AA^T)^{-1}(b - Ay).$$

Constraints: second method

$\star$ we can eliminate the constraints by including them into the function to be minimized

$$\min_{C(x)=0} f(x) \text{ becomes } \min_{x\in\mathbb{R}^n} f(x) + \frac{1}{\varepsilon}|C(x)|^2 \quad (\varepsilon > 0)$$

$\star$ we obtain an optimization problem without constraints for which classical algorithms can be applied

Proposition 20 (Constraints via Penalization)

Consider the problem (P) defined by $\min_{C(x)=0} f(x)$, where C is a continuous function $C : \mathbb{R}^n \to \mathbb{R}^p$ defining the constraints. Suppose that f is convex, continuous and ∞ at infinity.
Define now for $\varepsilon > 0$ the problems (P_ε) by $\min_{x\in\mathbb{R}^n} f(x) + \frac{1}{\varepsilon}|C(x)|^2$. The problems (P_ε) admit minimizers denoted by x_ε. Then every limit point of x_ε as $\varepsilon \to 0$ converges to a solution of (P).

Proof: **Exercise!**

Conclusion: constraints

- for simple constraints: projected gradient algorithm works fine
- it is possible to eliminate the constraints using a penalization
  - simple to implement in practice if f and C are smooth
  - theoretical convergence is valid for $\varepsilon \to 0$: in practice we never get to 0...
  - as ε grows, the constraint term $\frac{1}{\varepsilon}|C(x)|^2$ may dominate in (P_ε) so we no longer advance in a direction which minimizes (P)
  - in practice we often start with ε large and solve the problem multiple times, diminishing ε and starting from the previous solution.
- we will come back later to the optimality conditions related to constraints related to the Lagrange multipliers

Optimization in higher dimensions

- Theoretical aspects
- Gradient descent methods
- Newton's method
- Other methods

⋆ the anti-gradient direction $d = -\nabla f(x)$: the best asymptotic descent direction
⋆ that does not mean it is the best choice in all applications!
⋆ other descent directions exist: any direction such that $d \cdot \nabla f(x) < 0$ is a descent direction.

Examples:

- $d = -\frac{\partial f}{\partial x_i}(x) e_i$
- $d = -D\nabla f(x)$, where D is a diagonal matrix with positive entries
- $d = -A\nabla f(x)$ (or $-A^{-1}\nabla f(x)$) where A is a positive-definite matrix

Why these work?

$$f(x+td) = f(x) + t\nabla f(x) \cdot d + o(t) = f(x) - t\underbrace{(\nabla f(x))^T A \nabla f(x)}_{\geq 0} + o(t)$$

Recall Wolfe's condition

⋆ $m_1, m_2 \in (0,1)$ are chosen constants
⋆ d is a descent direction at x: $d \cdot \nabla f(x) < 0$, $q(t) = f(x+td)$
⋆ recall that $q'(0) = \nabla f(x) \cdot d < 0$

a) $\frac{q(t)-q(0)}{t} \leq m_1 q'(0)$ and $q'(t) \geq m_2 q'(0)$ (then we have a good t)

b) $\frac{q(t)-q(0)}{t} > m_1 q'(0)$ (then t is too big)

c) $\frac{q(t)-q(0)}{t} \leq m_1 q'(0)$ and $q'(t) < m_2 q'(0)$ (then t is too small)

⋆ Interpretation of $q'(t) \geq m_2 q'(0)$: the slope should be "less negative" at the next point
⋆ If $x_{i+1} = x_i + t_i d_i$ with t_i verifying the above then:

$$\nabla f(x_{k+1}) \cdot d_k \geq m_2 \nabla f(x_k) \cdot d_k.$$

⋆ define θ_k as the angle between d_k and $-\nabla f(x_k)$:

$$\cos \theta_k = \frac{-\nabla f(x_k) \cdot d_k}{|\nabla f(x_k)||d_k|}.$$

Zoutendijk condition

Theorem 21

Consider the iteration $x_{i+1} = x_i + t_i d_i$ where $d_i \cdot \nabla f(x_i) < 0$ and t_i verifies the Wolfe conditions. Suppose that f is of class C^1 on $\mathbb{R}^n$ and is bounded from below. Assume also that ∇f is L-Lipschitz, i.e.

$$|\nabla f(x) - \nabla f(y)| \leq L|x-y|, \text{ for all } x, y \in \mathbb{R}^n.$$

Then

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

⋆ the proof is rather straightforward (in the Notes)
⋆ Immediate consequence: if $d_i = -\nabla f(x_i)$ then $\theta_i = 0$ and $|\nabla f(x_i)| \to 0$.
⋆ if the descent direction is chosen such that θ_k is bounded away from 90°, i.e. $\cos \theta_k \geq \delta > 0$ then $|\nabla f_k| \to 0$.

The basic Newton Method

$\star$ as in the 1D case, look at the second order Taylor expansion

$$f(x+h) = f(x) + \nabla f(x) \cdot h + \frac{1}{2} h^T D^2 f(x) h + o(|h|^2)$$

The basic Newton Method

⋆ as in the 1D case, look at the second order Taylor expansion

$$f(x+h) \approx f(x) + \nabla f(x) \cdot h + \frac{1}{2} h^T D^2 f(x) h$$

⋆ then minimize the quadratic function in order to find the new iterate

$$\min_h \left(f(x) + \nabla f(x) \cdot h + \frac{1}{2} h^T D^2 f(x) h \right)$$

$$D^2 f(x) h + \nabla f(x) = 0 \Longrightarrow h = -[D^2 f(x)]^{-1} \nabla f(x)$$

Algorithm 7 (Newton's method)

Given a starting point x_0 run the recurrence

$$x_{i+1} = x_i - [D^2 f(x_i)]^{-1} \nabla f(x_i).$$

Remarks

Inconvenients:

- the method is not necessarily well-defined: is $D^2 f(x_i)$ invertible at x_i?
- the Taylor expansion is local: are we sure that $[D^2 f(x_i)]^{-1}\nabla f(x_i)$ is small?
- is the value of the function decreasing: $f(x_{i+1}) < f(x_i)$?
- is $d = [D^2 f(x_i)]^{-1}\nabla f(x_i)$ a descent direction? Yes, if $D^2 f(x_i)$ is positive-definite!
- note that $[D^2 f(x_i)]^{-1}\nabla f(x_i)$ implies the resolution of a linear system (recall that for large matrices we NEVER compute inverses!) - this might be costly if the number of variables is large

Advantage: when the method converges, the convergence is quadratic!

Theorem 22 (Quadratic convergence: Newton method)

If x^* is a non-degenerate minimizer for the function $f : \mathbb{R}^n \to \mathbb{R}$, i.e. $D^2 f(x^*)$ is positive definite, and the starting point x_0 is close enough to the optimum x^* then Newton's algorithm converges quadratically to x^*.

Newton-Rhapson Method

⋆ another point of view: solve nonlinear systems

$$\begin{cases} g_1(x_1, ..., x_n) & = & 0 \\ \vdots & \ddots & \vdots \\ g_n(x_1, ..., x_n) & = & 0 \end{cases}$$

⋆ denote $g(x) = (g_1(x), ..., g_n(x))$ and $Dg(x) = (\frac{\partial g_i}{\partial x_j})$ (the Jacobian matrix)
⋆ the Newton iteration

$$x_{n+1} = x_n - (Dg(x_n))^{-1} g(x)$$

converges to a zero x^* of g quadratically provided that x_0 is close to x^* and $Dg(x^*)$ is non-degenerate.

⋆ note that the Newton method corresponds to the Newton-Rhapson method applied for finding the zeros of $g = \nabla f$

Fixing Newton's method

1. Use a line-search procedure. If $D^2 f(x)$ is positive definite then the Newton direction $d = -(D^2 f(x))^{-1} \nabla f(x)$ is a descent direction.

Proposition 23 (Newton with line-search)

Let f be a C^2 function and α-convex function. Let x_0 be such that the level set $S = \{x : f(x) \leq f(x_0)\}$ is bounded. Then the Newton method with Wolfe line-search converges to the unique global minimizer of f.

Proof: A lower bound for $\cos \theta_k$ can be found in terms of the eigenvalues of $D^2 f(x)$. The sequence of iterates converges to a critical point. Convergence is not quadratic if the step t is smaller than 1!

2. Variable metric methods. Any positive definite matrix A defines a new metric. There are choices of A for which convergence towards the minimum may be faster.

Discussion

⋆ gradient descent direction as the minimizer of a quadratic function

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

⋆ the quadratic approximation is minimized by

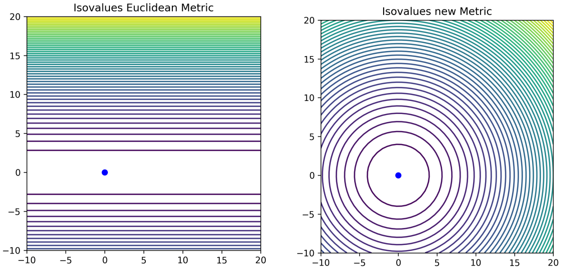
$$d^* = -\nabla f(x)$$

Remarks:

⋆ Note that the gradient method is the same as the Newton method when the Hessian $D^2 f(x)$ is the identity matrix.

⋆ This is bad, especially if the Hessian matrix is ill conditioned

⋆ The current gradient does not necessarily point towards the minimizer



- ⋆ change the metric: change the coordinate system around x
- ⋆ let A be a symmetric positive-definite matrix

$$f(x+d) \approx f(x) + d^T \nabla f(x) + \frac{1}{2} d^T A d$$

- ⋆ the quadratic approximation is minimized by

$$d = -A^{-1} \nabla f(x)$$

- ⋆ how to choose A?

What metric to choose?

⋆ For $f(x) = \frac{1}{2}x^T Ax - b^T x$ change the variable to $\xi = A^{1/2}x$

⋆ Recall that $A^{1/2} = P^{-1}\sqrt{D}P$ where $A = P^{-1}DP$ is a diagonalization of A.

⋆ Then denote $g(\xi) = f(x) = f(A^{-1/2}\xi) = \frac{1}{2}\xi^T\xi - b^T A^{-1/2}\xi$ and note that this function is well conditioned

⋆ Write the GD algorithm for $\xi \mapsto f(A^{-1/2}\xi)$:

$$\xi_{n+1} = \xi_n - t\nabla g(\xi_n)$$

$$\xi_{n+1} = \xi_n - tA^{-1/2}\nabla f(A^{-1/2}\xi_n)$$

Then multiplying by $A^{-1/2}$ we get

$$x_{n+1} = x_n - tA^{-1}\nabla f(x_n).$$

⋆ Choosing the descent direction $-A^{-1}\nabla f(x)$ is equivalent to performing a GD step in the new metric (coordinate system)!

Practical remark: the optimal metric given by $A^{1/2}$ is not known! Finding it may require more computational effort than the optimization problem

⋆ in practice the metric A is changed iteratively (see the next course)

General algorithm

incorporating all previous algorithms...

Algorithm 8 (Generic Variable Metric method)

Choose the starting point x_0

Iteration i:

- *compute* $f(x_i), \nabla f(x_i)$ *and eventually* $D^2 f(x_i)$
- *choose a symmetric positive-definite matrix* A_i*: compute the new direction*
$$d_i = -A_i^{-1} \nabla f(x_i)$$
- *perform a line-search from* x_i *in the direction* d_i *giving a new iterate*
$$x_{i+1} = x_i + t_i d_i = x_i - t_i A_i^{-1} \nabla f(x_i).$$

$\star$ $A_i = \mathrm{Id}$ gives the Gradient Descent method

$\star$ $A_i = D^2 f(x_i)$ gives the Newton method with line search (only when $D^2 f(x_i)$ is positive-definite)

$\star$ such an algorithm will converge to a critical point provided the set $\{f(x) \leq f(x_0)\}$ is bounded. The key point is that line-search guarantees descent: $f(x_{i+1}) < f(x_i)$ when not at a critical point

Modified Newton method

Idea: Choose A_i based on $D^2f(x_i)$ by eventually changing the Hessian matrix to make it positive definite

1. Choose a threshold $\delta > 0$ and compute the spectral decomposition
$$D^2f(x_i) = U_iD_iU_i^T.$$
If a diagonal value of D_i is smaller than δ then replace it with δ.
$\longrightarrow$ Large arithmetic cost: $2n^3$ to $4n^3$ arithmetic operations
2. Levenberg-Marquardt modification: $A_i = D^2f(x_i) + \varepsilon Id$. Choose ε such that A_i is positive definite by using a bisection scheme.

   Test the positive-definiteness using the Cholesky Factorization: $A_i = LDL^T$ - arithmetic cost: $n^3/6$
3. Use a modified Cholesky factorization so that the resulting diagonal matrix has entries bigger than $\delta > 0$.

$\star$ all these techniques are too costly for large n
$\star$ we lose quadratic convergence as soon as $A_i \neq D^2f(x_i)$ or the corresponding line-search step is smaller than 1

Conclusion: Newton's method

- quadratic convergence when we start close to a non-degenerate minimizer
- in order to guarantee convergence in general a line-search procedure should be used
- if $D^2 f(x_i)$ is not positive-definite then multiple ways exist to "correct the algorithm" but they are all costly: $O(n^3)$
- a linear system should be solved at each iteration
- the cost becomes too big if n is very large
- even the RAM memory usage is too heavy for large n: $O(n^2)$ when the Hessian is full

Optimization in higher dimensions

- Theoretical aspects
- Gradient descent methods
- Newton's method
- Other methods

Gauss-Newton Method

⋆ non-linear least squares: assume $m \geq n$

$$f(x) = \sum_{j=1}^{m} r_j(x)^2$$

⋆ define the Jacobian matrix

$$J(x) = \begin{pmatrix} \frac{\partial r_1}{\partial x_1} & \cdots & \frac{\partial r_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_m}{\partial x_1} & \cdots & \frac{\partial r_m}{\partial x_n} \end{pmatrix}$$

⋆ note that $\nabla f(x) = 2(J(x))^T r$ where $r = (r_1, ..., r_m)$

⋆ Hessian computation: $D^2 f(x) = 2J(x)^T J(x) +$ something small...

⋆ choose to approximate the Hessian by $2J(x)^T J(x)$ which is positive definite when J is of maximal rank

⋆ Therefore we get the Gauss-Newton method

$$x_{i+1} = x_i - \gamma_i (J(x_i)^T J(x_i))^{-1} J^T(x_i) r(x_i)$$

where either $\gamma_i = 1$ or a line-search is performed

⋆ as before one must check if $-(J(x_i)^T J(x_i))^{-1} J^T(x_i) r(x_i)$ is a descent direction

Example 1

⋆ the Rosenbrock function: $f(x) = 100(y - x^2)^2 + (1-x)^2 \Longrightarrow$
$r_1 = 10(y-x)^2,\ r_2 = (1-x)$

⋆ $J(x) = \begin{pmatrix} -20x & 10 \\ -1 & 0 \end{pmatrix}$

⋆ true Hessian vs Gauss-Newton approx:

$$H(x) = \begin{pmatrix} 1200x^2 - 400y + 2 & -400x \\ -400x & 200 \end{pmatrix}$$

$$2J^T J = \begin{pmatrix} 800x^2 + 2 & -400x \\ -400x & 200 \end{pmatrix}$$

⋆ Numerically this converges very fast, using only gradient information

Example 2: Triangulations

Suppose you know the coordinates (x_i, y_i) of three antennas and the distances d_i of a cellphone to these antennas, find the coordinates (x_0, y_0) of the cellphone.

⋆ least squares formulation:

$$f(x,y) = \sum_{i=1}^{3} r_i^2, \quad r_i(x,y) = d_i - \sqrt{(x-x_i)^2 + (y-y_i)^2}.$$

⋆ Gauss-Newton generally converges faster than GD here

Further examples

$\star$ Other important applications: least squares are often used when fitting models to data

$$f(x) = \sum_{i=1}^{m} r_i(x)^2 = \sum_{i=1}^{m} (y(s_i, x) - y_i)^2$$

where $y(s, x)$ is a non-linear function

Practical session:

$\star$ find parameters of a population model: exponential model, logistic model

$\star$ find parameters for a temperature model: $T(t) = A\sin(wt + \phi) + C$

Nelder-Mead method

⋆ gradient free

Algorithm 9 (Nelder-Mead method)

Current test points $x_1, ..., x_{n+1} \in \mathbb{R}^n$

1. **Order**: *relabel points such that* $f(x_1) \leq ... \leq f(x_{n+1})$
2. *Compute centroid* x_0 *of points* $x_1, ..., x_n$
3. **Reflection**: *compute* $x_r = x_0 + \alpha(x_0 - x_{n+1})$ *with* $\alpha > 0$. *If* $f(x_1) \leq f(x_r) < f(x_n)$ *then replace* x_{n+1} *by* x_r *and go to Step 1*
4. **Expansion**: *if* $f(x_r) < f(x_1)$ *compute* $x_e = x_0 + \gamma(x_r - x_0)$ *with* $\gamma > 1$.
   If $f(x_e) < f(x_r)$ *replace* x_{n+1} *by* x_e *and go to Step 1*
   Else *replace* x_{n+1} *by* x_r *and go to Step 1*
5. **Contraction**: *If* $f(x_r) \geq f(x_n)$ *then compute* $x_c = x_0 + \rho(x_{n+1} - x_0)$ *with* $\rho \in (0, 0.5]$. *If* $f(x_c) < f(x_{n+1})$ *then replace* x_{n+1} *by* x_c *and go to Step 1*
6. **Shrink:** *Replace all points except* x_1 *by* $x_i = x_1 + \sigma(x_i - x_1)$. *Go to Step 1*

⋆ Standard parameters: $\alpha = 1, \gamma = 2, \rho = 1/2, \sigma = 1/2$.

⋆ Termination criterion: Simplex too small, variation of f small, etc.