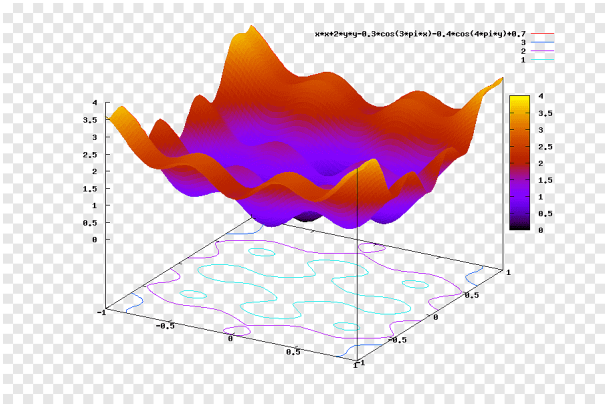


LINMA2471 Optimization models and methods II

Simon Desmidt

Academic year 2024-2025 - Q1

UCLouvain

Table des matières

Gradient Method

An optimization problem is defined as

$$\min_{x \in \mathbb{R}^n} f(x) \tag{1.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function.

1.1 Definitions

— A function $F : \mathbb{R}^n \to \mathbb{R}^n$ is L-Lipschitz continuous when

$$\|F(y) - F(x)\| \leq L\|y - x\| \qquad \forall x, y \in \mathbb{R}^n$$

where we use the euclidian norm.

— If ∇f is L-Lipschitz then, given $x \in \mathbb{R}^n$,

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2}\|y - x\|^2 = m_x(y) \qquad \forall y \in \mathbb{R}^n$$

and f is said to be a L-smooth function.

— We say that a differentiable function $\Psi : \mathbb{R}^n \to \mathbb{R}$ is L-smooth for some $L \geq 0$ when, given $x \in \mathbb{R}^n$,

$$\Psi(y) \leq \Psi(x) + \langle \nabla \Psi(x), y - x \rangle + \frac{L}{2}\|y - x\|^2 \qquad \forall y \in \mathbb{R}^n$$

— A convex function $f : \mathbb{R}^n \to \mathbb{R}$ is convex when, given $x, y \in \mathbb{R}^n$ and $\lambda \in [0, 1]$, we have

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda) f(y)$$

— Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex. If f is differentiable, then

$$f(y) \geq f(x) + \nabla f(x)^T (x - y) \qquad \forall x, y \in \mathbb{R}^n$$

— A differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is μ-strongly convex ($\mu > 0$) if, given $x \in \mathbb{R}^n$,

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{\mu}{2}\|y - x\|^2 \qquad \forall y \in \mathbb{R}^n$$

— PL inequality for a μ-strongly convex function[1] :

$$f(x) - f(x^*) \leq \frac{1}{2\mu}\|\nabla f(x)\|^2 \qquad \forall x \in \mathbb{R}^n$$

1. x^* is the minimizer of f

1.2 Complexity

The demonstration of the final results here obtained is in the notes, but not explained here.

1.2.1 Hypotheses

— f is convex and differentiable;

— ∇f is L-Lipschitz;

— we start from a $x_0 \in \mathbb{R}^n$ that is not a minimizer of f;

1.2.2 Results

We use the sequence $\{x_k\}_{k\geq 0}$, given a $x_0 \in \mathbb{R}^n$, such that

$$x_{k+1} = x_k - \frac{1}{L}\nabla f(x_k)$$

Problem class	Goal	Complexity bound
Non-convex f	$\|\nabla f(x_k)\| \leq \varepsilon$	$\mathcal{O}(\varepsilon^{-2})$
Convex f	$f(x_k) - f(x^*) \leq \varepsilon$	$\mathcal{O}(\varepsilon^{-1})$
μ-strongly-convex f	$f(x_k) - f(x^*) \leq \varepsilon$	$\mathcal{O}(\log(\varepsilon^{-1}))$

1.3 GM with Armijo Line Search

The Armijo Line Search consists of changing the constant in the GM in order to be more efficient and be able to make bigger steps in some directions where it is possible.

$$x_{k+1} = x_k - \alpha\nabla f(x_k) \qquad \alpha > 0 \tag{1.2}$$

Algorithm 1 Gradient Method with Armijo Line Search

1: **Step 0 :** Given $x_0 \in \mathbb{R}^n$ and $\alpha_0 > 0$, set $k := 0$.
2: **Step 1 :** Set $\ell := 0$.
3: **Step 1.1 :** Compute $x_k^+ = x_k - (0.5)^\ell \alpha_k \nabla f(x_k)$.
4: **Step 1.2 (Armijo Line Search) :** If

$$f(x_k) - f(x_k^+) \geq \frac{(0.5)^\ell \alpha_k}{2}\|\nabla f(x_k)\|^2 \tag{1}$$

set $\ell_k := \ell$ and go to Step 2. Otherwise, set $\ell := \ell + 1$ and go back to Step 1.1.
5: **Step 3 :** Define $x_{k+1} = x_k^+$, $\alpha_{k+1} = (0.5)^{\ell_k - 1}\alpha_k$, set $k := k + 1$ and go back to Step 1.

1.4 Problems with convex constraints

Consider the problem

$$\min_{x\in\mathbb{R}^n} f(x) \text{ such that } x \in \Omega \tag{1.3}$$

where f is L-smooth, and $\Omega \subseteq \mathbb{R}^n$ is nonempty, closed and convex. Given an approximation $x_k \in \Omega$ for a solution of 1.3, a possible generalization of the Gradient Method is to define

$$x_{k+1} = P_\Omega\left(x_k - \frac{1}{L}\nabla f(x_k)\right) \tag{1.4}$$

where P_Ω is the projection of z onto Ω, and we call this method the Projected Gradient Method.
If $\Omega = [a,b]^n$, then the projection of an element z onto Ω is such that its element i is given by :

$$[P_\Omega(z)]_i = \begin{cases} z_i \text{ if } a \le z_i \le b \\ a \text{ if } z_i < a \\ b \text{ if } z_i > b \end{cases} \qquad \forall i = 1,\dots,n \tag{1.5}$$

If x^* is a solution of (1.3), then

$$\langle \nabla f(x^*), z - x^* \rangle \ge 0 \qquad \forall z \in \Omega$$

→ Note : if $\Omega = \mathbb{R}^n$, then this lemma is true, in particular for $z = x^* - \nabla f(x^*)$. Then it is straightforward that we must have $\nabla f(x^*) = 0$.

1.5 Reduced gradient method

For a L-smooth function for the problem (1.3), we define

$$G_L(x_k) = L(x_k - x_{k+1}) \tag{1.6}$$

where x_{k+1} is given by the general formula [2]

$$x_{k+1} = \arg\min_{y\in\Omega} f(x_k) + \langle \nabla f(x_k), y - x_k\rangle + \frac{L}{2}|y - x_k\|_2^2 \tag{1.7}$$

From this, we can show as we did in the previous sections that there is a lower bound for the method :

$$f(x_k) - f(x_{k+1}) \ge \frac{1}{2L}\|G_L(x_k)\|_2^2 \tag{1.8}$$

This is the same result we found for the unconstrained gradient method, but with a different gradient definition. This is thus a generalization of the first cases. Furtheremore, by the same process we used before, we can show that the complexity of this Reduced Gradient Method is the same as in the table 1.2.2.

2. This definition of x_{k+1} is true for any type of gradient method, the first case seen being with $\Omega = \mathbb{R}^n$.

1.6 Proximal Gradient Method

We will here consider problems of the form

$$\min_{x\in\mathbb{R}^n} F(x) \equiv f(x) + \phi(x) \tag{1.9}$$

where $f(\cdot)$ is L-smooth and $\phi : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is convex, possibly nonsmooth. In this case, the formula for x_{k+1} is

$$x_{k+1} = \arg\min_{y\in\mathbb{R}^n} f(x_k) + \langle \nabla f(x_k), y - x_k\rangle + \frac{L}{2}|y - x_k|_2^2 + l(y) \tag{1.10}$$

which can be re-expressed as

$$x_{k+1} = \arg\min_{y\in\mathbb{R}^n} \frac{1}{2}\|y - (x_k - \frac{1}{L}\nabla f(x_k))\|^2 + \frac{1}{L}l(y) \tag{1.11}$$

Given a convex function h, we define the proximal operator $prox_h : \mathbb{R}^n \to \mathbb{R}^n$ by

$$prox_h(z) = \arg\min_{y\in\mathbb{R}^n} \frac{1}{2}\|y - z\|^2 + h(y) \tag{1.12}$$

Then, we can write

$$x_{k+1} = prox_{\frac{1}{L}\phi}\left(x_k - \frac{1}{L}\nabla f(x_k)\right) \tag{1.13}$$

$\to$ Note : if the ϕ function is the indicator function, i.e. $\phi = i_\Omega = \begin{cases} 0 \text{ if } x \in \Omega \\ \infty \text{ otherwise} \end{cases}$, then $prox_{\frac{1}{L}i_\Omega}(z) = P_\Omega(z)$.

1.7 Accelerated Proximal Gradient Method

This method's goal is to take into account the history of the method, so that the convergence is faster. This method still makes the hypothesis that the function f is convex.

Algorithm 2 Accelerated Proximal Gradient Method

1: **Step 0 :** Given $x_0 \in \mathbb{R}^n$, set $y_1 = x_0$, $t_1 = 1$ and $k = 1$.
2: **Step 1 :** Compute

$$x_k = prox_{\frac{1}{L}\phi}\left(y_k - \frac{1}{L}\nabla f(y_k)\right) \tag{1.14}$$

3: **Step 2 :** Define

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2} \tag{1.15}$$

$$y_{k+1} = x_k + \left(\frac{t_k - 1}{t_{k+1}}\right)(x_k - x_{k-1}) \tag{1.16}$$

4: **Step 3 :** Set $k = k + 1$ and go back to Step 1.

This method takes at most $\mathcal{O}(\varepsilon^{-1/2})$ iterations to generate x_k such that $f(x_k) - f(x^*) \leq \varepsilon$.

1.8 Convexly constrained optimization problem

Consider the problem

$$\min f(x) \text{ such that } x \in \Omega \tag{1.17}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function possibly nonsmooth, and Ω is convex, closed and nonempty.

Definition 1.1. A subgradient of the convex, non differentiable function f at x is a function $g : \mathbb{R}^n \to \mathbb{R}^n : x \to g(x)$ such that

$$f(y) \geq f(x) + \langle g(x), y - x \rangle \qquad \forall y \in \mathbb{R}^n \tag{1.18}$$

The set of all subgradients of f at point x is called subdifferential of f at x and is denoted by $\partial f(x)$.

A generalization of PGM to non smooth functions is

$$x_{k+1} = P_\Omega(x_k - \alpha g(x_k)) \qquad g(x_k) \in \partial f(x_k), \alpha_k > 0, \forall k \geq 0 \tag{1.19}$$

— If we take $\alpha_k = \alpha$, $\forall k \geq 0$, then we need at most $\mathcal{O}(\varepsilon^{-2})$ iterations.

— If we assume that $\|g(x_k)\| \leq M$ for all $k \geq 0$, then we can take $\alpha_k = \frac{\varepsilon}{\|g(x_k)\|^2}$, and the convergence is in $\mathcal{O}(\varepsilon^{-2})$ too. However, this is a good example of a dynamic step (changes with $g(x_k)$).

1.9 Summary

Method	Goal	Complexity
PGM	$F(x_k) - F(x^*) \leq \varepsilon$	$\mathcal{O}(\varepsilon^{-1})$
Accelerated PGM	$F(x_k) - F(x^*) \leq \varepsilon$	$\mathcal{O}(\varepsilon^{-1/2})$
PSG	$F(x_k) - F(x^*) \leq \varepsilon$	$\mathcal{O}(\varepsilon^{-2})$

Coordinate Descent Method

The goal here is to solve the problem

$$\min_{x \in \mathbb{R}^n} f(x) \tag{2.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is L-smooth and bounded from below by f_{low}.

The cost of computing the gradient at each step can require a lot of operations : e.g. the gradient of a quadratic function is calculated in $\mathcal{O}(n^2)$. In this section, we consider the setting in which n is huge to such an extent that $\mathcal{O}(n^p)$ operations to get $\nabla f(x)$ is not acceptable.

2.1 Randomized Coordinate Descent Method

This algorithm randomly chooses a single component of the gradient to compute the next iterate, for a L-smooth function. This algorithm converges in $\mathcal{O}(n\varepsilon^{-2})$.

Algorithm 3 Randomized Coordinate Descent Method

1: **Step 0 :** Given $x_0 \in \mathbb{R}^n$ and $L > 0$, set $k := 0$.
2: **Step 1 :** Choose $i_k \in \{1, \dots, n\}$ randomly with uniform probability.
3: **Step 2 :** Compute $x_{k+1} = x_k - \frac{1}{L} \left(\nabla f(x_k)\right)_{i_k} e_{i_k}$.
4: **Step 3 :** Set $k := k + 1$, and go back to step 1.

2.2 Stochastic Gradient Method

Consider a dataset $\{(a^{(i)}, b^{(i)})\}_{i=1}^N \subset \mathbb{R}^p \times \mathbb{R}$. Let $m_X : \mathbb{R}^p \to \mathbb{R}$ be defined by a parameter $x \in \mathbb{R}^n$. In ML, we want to find x^* that solves the optimization problem

$$\min_{x \in \mathbb{R}^n} \frac{1}{N} \sum_{i=1}^{N} \underbrace{\left(m_x\left(a^{(i)}\right) - b\right)^2}_{=f_i(x)} \tag{2.2}$$

The cost to compute $\nabla f(x)$ is thus $\mathcal{O}(Nn^p)$. We will use the SGD method when N is big.

$$x_{k+1} = x_k - \alpha_k \nabla f_{i_k}(x_k) \tag{2.3}$$

Algorithm 4 Stochastic Gradient Descent Method

1: **Step 0 :** Given $x_0 \in \mathbb{R}^n$, $\alpha_0 > 0$, set $k := 0$.
2: **Step 1 :** Choose $i_k \in \{1, \dots, N\}$ randomly with uniform probability.
3: **Step 2 :** Compute $x_{k+1} = x_k - \alpha_k \theta \nabla f_{i_k}(x_k)$.

Suppose that $f(\cdot)$ is L-smooth and bounded from below by f_{low}, and that $\|\nabla f_i(x)\| \leq G \ \forall i \in \{1, \dots, n\}$ and $\forall x \in \mathbb{R}^n$. Let us take $\alpha_k = \alpha = \frac{\varepsilon^2}{LG^2}$, the ideal case if we want α_k to be constant. The SGD converges in $\mathcal{O}(\varepsilon^{-4})$, which is very bad. The advantages of this method resides in the easy calculations at each step.

2.2.1 Momentum trick

The idea is to take into account the previous iterations :

$$x_{k+1} = x_k - \alpha \left(\sum_{i=0}^{k} \beta^{k-i} \nabla f(x_i) \right) \tag{2.4}$$

where $\beta \in (0, 1)$ is a discount factor. To get this, we can define (using $m_0 = 0$) :

$$\begin{aligned} m_{k+1} &= \beta m_k + (1 - \beta) \nabla f(x_k) \\ x_{k+1} &= x_k - \gamma m_{k+1} \end{aligned} \tag{2.5}$$

and $\alpha = \gamma(1 - \beta)$.
This trick can be used with SGD to improve its efficiency. Pushing this to its extremity, we get the AdaGrad method.

2.3 AdaGrad

At the beginning of the kth iteration, we choose $i_k \in \{1, \dots, N\}$ ranodmly with uniform probability and then set

$$\begin{aligned} [v_{k+1}]_j &= [v_k]_j + [\nabla f_{i_k}(x_k)]_j^2 \qquad j = 1, \dots, n \\ [x_{k+1}]_j &= [x_k]_j - \frac{\eta}{\delta + \sqrt{[v_{k+1}]_j}} [\nabla f_{i_k}(x_k)]_j \qquad j = 1, \dots, n \end{aligned} \tag{2.6}$$

with $v_0 = 0$ and $\eta, \delta > 0$. We can now mix the Momentum trick with AdaGrad.

2.4 RMSprop

At the beginning of the kth iteration, we choose $i_k \in \{1, \dots, N\}$ ranodmly with uniform probability and then set

$$\begin{aligned} [v_{k+1}]_j &= \beta [v_k]_j + (1 - \beta) [\nabla f_{i_k}(x_k)]_j^2 \qquad j = 1, \dots, n \\ [x_{k+1}]_j &= [x_k]_j - \frac{\eta}{\delta + \sqrt{[v_{k+1}]_j}} [\nabla f_{i_k}(x_k)]_j \qquad j = 1, \dots, n \end{aligned} \tag{2.7}$$

with $v_0 = 0$ and $\eta, \delta > 0$.

2.5 Adam

Even more extreme is the Adam method : RMSprop + Momentum trick. At the beginning of the kth iteration, we choose $i_k \in \{1, \dots, N\}$ ranodmly with uniform probability and then set

$$\begin{aligned} m_{k+1} &= \beta_1 m_k + (1-\beta_1)\nabla f_{i_k}(x_k) \\ [v_{k+1}]_j &= \beta_2 [v_k]_j + (1-\beta_2)[\nabla f_{i_k}(x_k)]_j^2 \qquad j = 1, \dots, n \\ \hat{m}_{k+1} &= m_{k+1} / \left(1 - \beta_1^{k+1}\right) \\ \hat{v}_{k+1} &= v_{k+1} / \left(1 - \beta_2^{k+1}\right) \\ [x_{k+1}]_j &= [x_k]_j - \frac{\eta}{\delta + \sqrt{[v_{k+1}]_j}} [\hat{m}_{k+1}]_j \qquad j = 1, \dots, n \end{aligned} \tag{2.8}$$

with $m_0 = 0$, $v_0 = 0$, $\beta_1, \beta_2 \in (0,1)$ and $\eta, \delta > 0$.

2.6 Revisiting Armijo Line Search - Cf 1.3

Lemma 2.1. Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable at $x \in \mathbb{R}^n$. If $\nabla f(x)^T d < 0$ and $\eta \in (0,1)$, then there exists $\delta > 0$ such that

$$(x + \alpha d) \leq f(x) + \eta \alpha \nabla f(x)^T d \qquad \forall \alpha \in [0, \delta) \tag{2.9}$$

We start from the following algorithm :

Algorithm 5 General Descent Method with Armijo Line Search

1: **Step 0 :** Given $x_0 \in \mathbb{R}^n$, $\alpha_0 > 0$ and $\eta \in (0,1)$, set $k := 0$.
2: **Step 1 :** If $\nabla f(x_k) = 0$, stop.
3: **Step 2 :** Compute $d_k \in \mathbb{R}^n$ such that $\langle \nabla f(x_k), d_k \rangle < 0$.
4: **Step 2.1 :** Find the smallest integer $i_k \in \{0, \dots, n\}$ such that $\alpha_k = 2^{-i_k}$ satisfies

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \eta \alpha_k \langle \nabla f(x_k), d_k \rangle \tag{2.10}$$

5: **Step 3 :** Define $x_{k+1} = x_k + \alpha_k d_k$, set $k := k+1$ and go to **Step 1**.

Lemma 2.2. Let $\{x_k\}_{k \geq 0}$ and $\{\alpha_k\}_{k \geq 0}$ be sequences generared by algorithm 5. If f is L-smooth, and if there exist $c_1, c_2 > 0$ such that

$$\begin{cases} \langle \nabla f(x_k), d_k \rangle \leq -c_1 \|\nabla f(x_k)\|^2 \\ \|d_k\| \leq c_2 \|\nabla f(x_k)\| \end{cases} \qquad \forall k \geq 0 \tag{2.11}$$

then

$$\alpha_k \geq \min \left\{ 1, \frac{(1-\eta)c_1}{L c_2^2} \right\} \equiv \alpha_{\min} \tag{2.12}$$

Properties :

— $f(x_k) - f(x_{k+1}) \geq \eta \alpha_{\min} c_1 \|\nabla f(x_k)\|^2$
— The complexity of 5 is described by table 1.2.2

2.6.1 Choosing the search direction

If we choose $d_k = -B_k \nabla f(x_k)$ with $c_1 I \preceq B_k \preceq c_2$, $\forall k \geq 0$, then the sequence $\{d_k\}_{k \geq 0}$ of search directions verifies equations (2.11).
Here are some examples for B_k :

— $B_k = I : d_k = -\nabla f(x_k) \Longrightarrow$ Gradient Direction ;

— $B_k = (\nabla^2 f(x_k))^{-1}$: Newton Direction ;

— $B_k \approx (\nabla^2 f(x_k))^{-1}$: Quasi-Newton Direction ;

Second order methods

3.1 Newton Method

Consider the unconstrained optimization

$$\min_{x \in \mathbb{R}^n} f(x) \tag{3.1}$$

To fing the basic Newton step, we do a Second order Taylor expansion of f around x_k : and minimize that quantity. This gives

$$0 = \nabla f(x_k) + \nabla^2 f(x_k)h \iff \nabla^2 f(x_k)h = -\nabla f(x_k) \tag{3.2}$$

Assuming the Hessian to be invertible,

$$h = -\nabla^2 f(x_k)^{-1} \nabla f(x_k) \tag{3.3}$$

We call h the Newton step $n(x) = -\nabla^2 f(x)^{-1} \nabla f(x)$.

$\rightarrow$ Note : In practice, we never compute $\nabla^2 f(x_k)^{-1}$ as it is not needed by itself.

Algorithm 6 Newton Method

1: **Step 0 :** Given $x_0 \in \mathbb{R}^n$, f, ∇f, $\nabla^2 f$ invertible, set $k := 0$.
2: **Step 1 :** If $\nabla f(x_k) = 0$, stop.
3: **Step 2 :** Define $x_{k+1} = x_k - \nabla^2 f(x_k)^{-1} \nabla f(x_k)$, set $k := k + 1$ and go to **Step 1**.

The problem with this method is that it does not find a minimizer, it only computes the solution of $\nabla m_{x_k}(h) = 0$. It is not always well defined, with not convex functions, and the computational cost of one iteration is high.

Theorem 3.1. Let $f \in \mathcal{C}^2$. If $\nabla^2 f$ is M-Lipschitz and x^* is a minimum of f such that $\nabla^2 f(x^*) \succeq \mu I$, with $\mu > 0$, then for any x such that $\|x - x^*\| \leq \frac{\mu}{2M}$, we have

$$|x^+ - x^*\| \leq \frac{M}{\mu} \|x - x^*\|^2 \tag{3.4}$$

where $x^+ = x - \nabla^2 f(x)^{-1} \nabla f(x)$ is well-defined.

Newton's method is invariant with respect to linear changes of variables, while gradient/first-order methods are not. However, it does not always converge.

3.2 Self-concordance

Definition 3.2. Given an open domain $X \subseteq \mathbb{R}^n$, a function $f : X \to \mathbb{R}$ is called self-concordant iff

— $f \in \mathcal{C}^3$ is convex;

— f is closed, i.e. its epigraph is a closed set;

— $\nabla^3 f(x)[h,h,h] \leq 2\nabla^2 f(x)[h,h]^{3/2}$, $\forall x \in X$, $\forall h \in \mathbb{R}^n$.

with

— $\nabla f(x)[h] = \nabla f(x) \cdot h$

— $\nabla^2 f(x)[h,h] = h^T \nabla f(x) h$

— $\nabla^3 f(x)[h,h,h] = \sum_i \sum_j \sum_k \frac{\partial^3 f}{\partial x_i \partial x_j \partial x_k} h_i h_j h_k$

For univariate functions, the conditions are simpler :

— $f \in \mathcal{C}^3$ is convex;

— $|f'''(x)| \leq 2f''(x)^{3/2}$, $\forall x \in X$

Property 3.3. The self-concordance property is conserved by sum and by linear changes of variables, and is nondegenerate :
Let X be an open set containing no line. Then,

— Any self concordant function defined over X satisfies $\nabla^2 f(x) \succ 0$;

— $f(x) \to \infty$ as $x \to \partial X$, where ∂X is the boundary of X.

3.3 Local norms

Optimality measure $\|\nabla f(x)\|$ is not suitable, as it is not affine-invariant. This is why we define local norms :

Definition 3.4. Given a self-concordant function f, the local norm at x is

$$\|z\|_x = (z^T \nabla^2 f(x) z)^1 / 2 \tag{3.5}$$

The corresponding dual norml is given by

$$\|z\|_x^* = (z^T \nabla^2 f(x)^{-1} z)^{1/2} \tag{3.6}$$

3.4 Optimality measure

Using the dual local norm defined in the last section, we define the optimality measure :

$$\delta(x) := \|\nabla f(x)\|_x^* = \|n(x)\|_x \tag{3.7}$$

Because of convexity, x is optimal iff $\nabla f(x) = 0 \iff \delta(x) = 0$.

3.5 Improving Newton

Given an open convex domain X and a self-concordnat function $f : X \to \mathbb{R}$, we want $\min_{x \in X} f(x)$.
Let $x \in X$ such that $\delta(x) < 1$:

— A global minimum x^* of f exists ;

— $f(x) \leq f(x^*) - \delta(x) - \log(1 - \delta(x)) \Longrightarrow f(x) - f(x^*) = \mathcal{O}(\delta(x)^2)$;

— $\|x - x^*\|_x \leq \frac{\delta}{1-\delta} = \mathcal{O}(\delta(x))$;

— Newton step $x^+ = x + n(x)$ is feasible, i.e. $x^+ \in X$, meaning the method is well-defined ;

— $\delta(x^+) \leq \left(\frac{\delta}{1-\delta}\right)^2$ meaning quadratic convergence.

If the method starts close to the minimizer, converges in less than 10 iterations.
If $x \in X$ and $\delta(x) \geq 1$, the good behaviour of the method is no longer guaranteed. To fix this, we introduce a damping of the steps :

For any $x \in X$ and thus any $\delta(x)$,

— The damped Newton step $x^+ = x + \left(\frac{1}{1+\delta(x)}\right) n(x)$ is feasible ;

— The decrease is guaranteed : $f(x) - f(x^+) \geq \delta(x) - \log(1 + \delta(x)) \geq 0$.

3.6 Globally convergent Newton's method

Suppose $\delta(x_0) > \frac{1}{\sqrt{2}}$. While $\delta(x_i) > \frac{1}{\sqrt{2}}$,

$$f(x_i) - f(x_{i+1}) > \frac{1}{\sqrt{2}} - \log(1 + \frac{1}{\sqrt{2}}) > \frac{1}{6} \tag{3.8}$$

Hence, after at most $k \leq \lceil 6(f(x_0) - f(x^*)) \rceil$, we must have $\delta(x_k) \leq \frac{1}{\sqrt{2}}$. Applying pure Newton steps once this stage is reached, we obtain a ϵ-solution after

$$\mathcal{O}(f(x_0) - f(x^*)) + \mathcal{O}(\log\log\frac{1}{\epsilon}) \text{ iterations} \tag{3.9}$$

Interior-point methods

Definition 4.1. Given a convex set C and a self-concordant function $F : int(C) \to \mathbb{R}$, the analytic center of C is the unique minimizer of F. It always exists if C is bounded, and depends on the choice of F as much as C.

Let $f : \mathbb{R}^n \mapsto \mathbb{R}$ be a convex function, $C \subseteq \mathbb{R}^n$ be a closed convex set. The objective if optimizing a vector $x \in \mathbb{R}^n$:

$$\inf_{x \in \mathbb{R}^n} c^T x \text{ s.t. } x \in C \tag{4.1}$$

The interior-point method consists in approximating this constrained problem by a family of unconstrained problems, using a barrier function F to replace the inclusion $x \in C$:

— F is smooth ;

— F is strictly convex on $int(C)$;

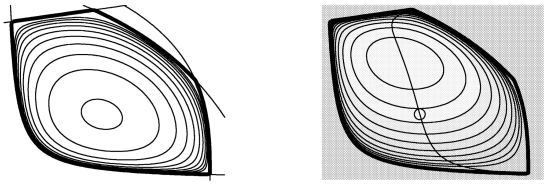
— $F(x) \to +\infty$ when $x \to \partial C$.

4.1 Central path method

Let $\mu > 0$ be a scalar parameter and consider the problem

$$\inf_{x \in \mathbb{R}^n} \frac{c^T x}{\mu} + F(x) = f_\mu(x) \tag{4.2}$$

Let us call this problem (P_μ)



The optimum value x^*_μ of this problem tends to x^* a solution of the original problem when μ decreases to zero, following the central path.

→ Note : When C is defined with convex constraints $g_i(x) \leq 0$, we can use the barrier function $F(x) = -\sum_i \log(-g_i(x))$.

4.1.1 Self-concordant barriers

$F : int(C) \mapsto \mathbb{R}$ is called a ν-self-concordant barrier for C iff

— $F \in \mathcal{C}^3$ is convex;

— $F(x) \xrightarrow{x \to \partial C} +\infty$;

— The following two conditions hold :

$$\nabla^3 F(x)[h,h,h] \leq 2(\nabla F(x)[h,h])^{3/2} \tag{4.3}$$

$$\nabla F(x)^T (\nabla F(x))^{-1} \nabla F(x) \leq \nu \tag{4.4}$$

$$\tag{4.5}$$

for all $x \in int(C)$ and $h \in \mathbb{R}^n$.

Properties

— If f is ν_1-s.c. for C_1 and G is ν_2-s.c. for C_2, then $f + G$ is $(\nu_1 + \nu_2)$-s.c. for $C_1 \cap C_2$ (if nonempty);

— If $x \mapsto F(x)$ is a ν-s.c. barrier for C, then $y \mapsto F(c - A^T y)$ is a ν-s.c. barrier for the set $\{y \text{ such that } c - A^T y \in C\}$.

4.1.2 Computing x_μ

By definition,

$$x_\mu = \arg_{x \in int(C)} \min \frac{c^T x}{\mu} + F(x) \tag{4.6}$$

and thus satisfies the condition

$$\frac{c}{\mu} + \nabla F(x) = 0 \tag{4.7}$$

To compute x_μ, we use Newton steps $n_\mu(x)$:

$$n_\mu(x) = -\nabla^2 f_\mu(x) \nabla f_\mu(x) = -\nabla^2 F(x)(\frac{c}{\mu} + \nabla F(x)) \tag{4.8}$$

In practice, this method is not efficient, as it takes a long time to only compute a starting point.

Theorem 4.2. Assume x is such that $\delta_\mu(x) \leq \tau < 1$. Then,

$$c^T x - c^T x^* < \frac{\nu \mu}{1 - \tau} \tag{4.9}$$

This means that choosing μ_{final} as the solution to $\frac{\nu\mu}{1-\tau} = \epsilon$ guarantees a final ϵ accuracy on the linear objective.

4.1.3 Short-step path-following method

Given a starting iterate x_0 not too far from a point on the central path $x_0 \approx x_{\mu_0}$,

Algorithm 7 Short-step path-following method

Find a ν-s.c. barrier F such that $dom(F) = int(C)$;
Choose $0 < \tau < 1$ and $0 < \theta < 1$;
Given a starting point $x_0 \in int(C)$ and target accuracy ϵ :
Find initial value μ_0 such that $\delta_{\mu_0}(x_0) \leq \tau$;
Compute final value $\mu_f = \epsilon \frac{1-\tau}{\nu}$;
Set $k \leftarrow 0$, perform the main loop :
while $\mu_k > \mu_f$ **do**
    Step 1 : $\mu_{k+1} \leftarrow \mu_k(1-\theta)$;
    Step 2 : $x_{k+1} \leftarrow x_k + n_{\mu_{k+1}}(x_k)$;
    Step 3 : $k \leftarrow k+1$;
end while

This algorithm takes polynomial time.
As the distance to the central path is impossible to compute in practice, we approximate it to $\delta_{\mu_k}(x) = \|n_\mu(x)\|_x$.
This method reaches an accuracy ϵ on the objective within $\mathcal{O}\left(\sqrt{\nu}\log\frac{1}{\epsilon}\right)$ iterations. Therefore, it takes polynomial time if computing $F, \nabla F, \nabla^2 F$ takes polynomial time.

4.2 Practical details

— The barrier function is very hard to compute, but there exists universal ones for some specific domains (e.g. cones).

4.2.1 Initialisation procedure

1. Pick a reasonable starting iterate $\bar{x}$ that belongs to $int(C)$;
2. Find a reasonable starting μ_0 :
   — Any value $\mu_0 > 0$ works, but if too small, it will lead to longer initialization;
   — $\mu_0 = \min_\mu \delta_\mu(\bar{x})$ works well.
3. Compute x_0 by minimizing this problem starting from $\bar{x}$ and using damped Newton steps :
$$\min \frac{c^T x}{\mu_0} + F(x) \tag{4.10}$$
4. Stop the procedure when $\delta_{\mu_0}(x_0) \leq \tau$.