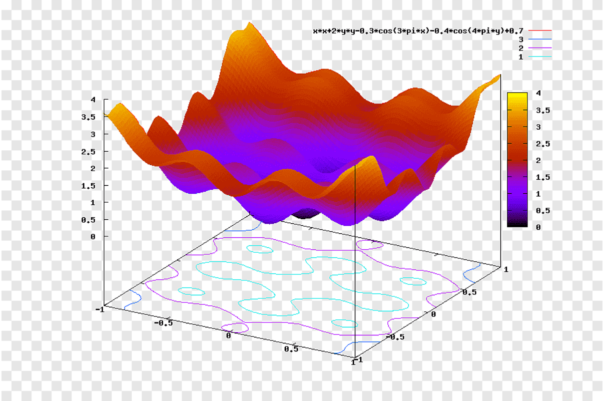

LINMA2471 Optimization models and methods II

Simon Desmidt

Academic year 2024-2025 - Q1

UCLouvain

Table des matières

1 Gradient Method **2**

1.1 Definitions 2

1.2 Complexity 3

1.3 GM with Armijo Line Search 3

1.4 Problems with convex constraints 4

1.5 Reduced gradient method 4

1.6 Proximal Gradient Method 5

1.7 Accelerated Proximal Gradient Method 5

1.8 Convexly constrained optimization problem 6

1.9 Summary 6

2 Coordinate Descent Method **7**

2.1 Randomized Coordinate Descent Method 7

2.2 Stochastic Gradient Method 7

2.3 AdaGrad 8

2.4 RMSprop 8

2.5 Adam 9

2.6 Revisiting Armijo Line Search 9

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

```
Step 0 : Given $x_0 \in \mathbb{R}^n$ and $\alpha_0 > 0$, set $k := 0$.
Step 1 : Set $\ell := 0$.
Step 1.1 : Compute $x_k^+ = x_k - (0.5)^\ell \alpha_k \nabla f(x_k)$.
Step 1.2 (Armijo Line Search) : If
    $$f(x_k) - f(x_k^+) \geq \frac{(0.5)^\ell \alpha_k}{2} \|\nabla f(x_k)\|^2 \quad (1)$$
    set $\ell_k := \ell$ and go to Step 2. Otherwise, set $\ell := \ell + 1$ and go back to Step 1.1.
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 :

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

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

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

→ N.B. : 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 :

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

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

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

→ N.B. : 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

```
Step 0 : Given $x_0 \in \mathbb{R}^n$, set $y_1 = x_0$, $t_1 = 1$ and $k = 1$.
Step 1 : Compute
    $$x_k = prox_{\frac{1}{L}\phi}\left(y_k - \frac{1}{L}\nabla f(y_k)\right) \tag{1.14}$$
Step 2 : Define
    $$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2} \tag{1.15}$$
    $$y_{k+1} = x_k + \left(\frac{t_k - 1}{t_{k+1}}\right)(x_k - x_{k-1}) \tag{1.16}$$
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

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

```
Step 0 : Given $x_0 \in \mathbb{R}^n$, $\alpha_0 > 0$, set $k := 0$.
Step 1 : Choose $i_k \in \{1, \dots, N\}$ randomly with uniform probability.
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 $i_k \in \{1, \ldots, N\}$ ranodmly with uniform probability and then set

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

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

2.6 Revisiting Armijo Line Search

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

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

slide 30.