Université de Limoges **Academic Year 2023-2024**

Master ACSYON, Final Exam

Splitting Methods in Convex Optimization

December 12, 2023 - 3 hours

Documents are not allowed.

1 Rate for the Proximal Point Algorithm (Theoretical part)

1.1 Preliminaries

We consider a maximally monotone operator $A : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ defined on $\mathbb{R}^n$. For simplicity, we assume that the domain of A is the whole space $\mathbb{R}^n$. The space $\mathbb{R}^n$ is endowed with the Euclidean scalar product $\langle \cdot, \cdot \rangle$ and the corresponding Euclidean norm $\| \cdot \|$. The identity map on $\mathbb{R}^n$ is denoted by Id, and the resolvent operator associated with A is denoted by $\mathrm{J}_A = (\mathrm{Id} + A)^{-1}$. We also assume that the set of zeros of A, denoted by S, is nonempty; that is, $S = A^{-1}(0) \neq \emptyset$. We recall the following properties, taken from the course:

(i) A nonexpansive operator is an operator which is 1-Lipschitz continuous.

(ii) An operator $T : \mathbb{R}^n \to \mathbb{R}^n$ is said to be firmly nonexpansive if

$$\|T(x_2) - T(x_1)\|^2 + \|(\mathrm{Id} - T)(x_2) - (\mathrm{Id} - T)(x_1)\|^2 \leq \|x_2 - x_1\|^2, \ \forall x_1, x_2 \in \mathbb{R}^n.$$

(iii) An operator $T : \mathbb{R}^n \to \mathbb{R}^n$ is said to be α-averaged, with $0 < \alpha < 1$, if there exists a nonexpansive operator $R : \mathbb{R}^n \to \mathbb{R}^n$ such that $T = (1 - \alpha)\mathrm{Id} + \alpha R$.

(iv) An operator $T : \mathbb{R}^n \to \mathbb{R}^n$ is firmly nonexpansive if and only if it is $\dfrac{1}{2}$-averaged.

(v) A mapping T is said to be quasinonexpansive if for all $(x_1, x_2) \in \mathrm{Fix}(T) \times \mathbb{R}^n$, the following inequality holds:

$$\|T(x_2) - x_1\| \leq \|x_2 - x_1\|.$$

(vi) The resolvent operator to a maximally monotone operator is firmly nonexpansive.

(vii) firmly nonexpansive $\implies$ nonexpansive $\implies$ quasinonexpansive,

The following Lemma taken from the course can be useful.

Lemma 1. *Let $T : D \to D$ be a continuous and quasinonexpansive operator such that D is closed and Fix$(T) \neq \emptyset$. Consider the fixed-point algorithm given by*

$$x_0 \in D \text{ and } x_{\nu+1} := T(x_\nu), \ \forall \nu \in \mathbb{N}.$$

If $x_\nu - T(x_\nu) \to 0$ in $\mathbb{R}^d$ when $\nu \to \infty$, then the sequence $(x_\nu)_{\nu \in \mathbb{N}}$ converges to some point in Fix(T).

1.2 Part I

Let x^* be an element of S, which implies $0 \in A(x^*)$. Given an initial point $x_0 \in \mathbb{R}^n$, we define a sequence $(x_\nu)_{\nu\in\mathbb{N}}$ by the following iterative process:

$$x_{\nu+1} = \mathrm{J}_A(x_\nu). \tag{1}$$

This defines the iterations of the Proximal-Point Algorithm (PPA).

1. Show that $x^* = \mathrm{J}_A(x^*)$ and that for every $\nu \in \mathbb{N}$, we have

   $$\|x_{\nu+1} - x^*\|^2 + \|x_\nu - x_{\nu+1}\|^2 \leq \|x_\nu - x^*\|^2. \tag{2}$$

   Deduce that for every $\nu \in \mathbb{N}$, we have

   $$\|x_\nu - x^*\|^2 \leq \|x_0 - x^*\|^2.$$

   Hint. To prove (2), use the fact that J_A is firmly nonexpansive.

2. Show that $\nu \in \mathbb{N}^* \mapsto \varphi(\nu) := \|x_{\nu+1} - x_\nu\|^2$ is a decreasing function of $\nu \in \mathbb{N}^*$.

3. Deduce from the previous question that, we have for every $\nu \in \mathbb{N}$

   $$(\nu+1)\|x_{\nu+1} - x_\nu\|^2 \leq \sum_{k=0}^{\nu} \|x_{k+1} - x_k\|^2. \tag{3}$$

4. Using (2), show that for every $\nu \in \mathbb{N}$

   $$\sum_{k=0}^{\nu} \|x_{k+1} - x^*\|^2 + \sum_{k=0}^{\nu} \|x_{k+1} - x_k\|^2 \leq \sum_{k=0}^{\nu} \|x_k - x^*\|^2.$$

   Deduce that

   $$\|x_{\nu+1} - x^*\|^2 + \sum_{k=0}^{\nu} \|x_{k+1} - x_k\|^2 \leq \|x_0 - x^*\|^2. \tag{4}$$

5. Deduce from (3) and (4) that for every $\nu \in \mathbb{N}$

   $$\|x_{\nu+1} - x_\nu\| \leq \frac{1}{\sqrt{\nu+1}} \mathrm{dist}(x_0, S). \tag{5}$$

   Hint. Start by showing that $(\nu+1)\|x_{\nu+1} - x_\nu\|^2 \leq \|x_0 - x^*\|^2$.

6. Use Lemma 1 to deduce that the sequence $(x_\nu)_{\nu\in\mathbb{N}}$ is convergent to some point in $S = A^{-1}(0)$.

1.3 Part II

7. Let $T : \mathbb{R}^n \to \mathbb{R}^n$ be a nonexpansive operator. For $\theta \in]0,1[$, we set $T_\theta = (1-\theta)\mathrm{Id} + \theta T$. Let x^* be a fixed point of T.

   (a) Show that T_θ is also nonexpansive and that $\mathrm{Fix}(T) = \mathrm{Fix}(T_\theta)$.

(b) We now consider the algorithm given by

$$x_{\nu+1} = T_\theta(x_\nu),\ x_0 \in \mathbb{R}^n,\ \nu = 0, 1, 2 \ldots.$$

Show that:

$$\|x_{\nu+1} - x^*\|^2 \le \|x_\nu - x^*\|^2 - \theta(1-\theta)\|T(x_\nu) - x_\nu\|^2.$$

Hint. Use the fact that for every $x,\ y \in \mathbb{R}^n$ and $\theta \in]0,1[$, we have

$$\|(1-\theta)x + \theta y\|^2 = (1-\theta)\|x\|^2 + \theta\|y\|^2 - \theta(1-\theta)\|x-y\|^2.$$

8. (a) Use the same technique as in Part I to deduce that for every $\nu \in \mathbb{N}$

$$\|T(x_\nu) - x_\nu\| \le \frac{1}{\sqrt{\theta(1-\theta)(\nu+1)}}\|x_0 - x^*\|. \tag{6}$$

(b) Deduce from (6) that for every $\nu \in \mathbb{N}$, we have

$$\|T_\theta(x_\nu) - x_\nu\| \le \frac{\sqrt{\theta}}{\sqrt{(1-\theta)(\nu+1)}}\|x_0 - x^*\|. \tag{7}$$

9. Consider the over-relaxed proximal point algorithm:

$$\text{(ORPPA)} \begin{cases} y_\nu = \mathrm{J}_A(x_\nu),\ \nu \in \mathbb{N} \\ x_{\nu+1} = x_\nu + \omega(y_\nu - x_\nu). \end{cases}$$

for $0 < \omega < 2$.

(a) Using properties (iv) and (vi) in the preliminaries subsection, show that there exists a nonexpansive map $R : \mathbb{R}^n \to \mathbb{R}^n$ such that $\mathrm{J}_A = \frac{1}{2}\mathrm{Id} + \frac{1}{2}R$.

(b) Show that the sequence $(x_\nu)_{\nu\in\mathbb{N}}$ defined by (ORPPA) satisfies

$$x_{\nu+1} = (1 - \frac{\omega}{2})x_\nu + \frac{\omega}{2}Rx_\nu.$$

(c) Use (7) and Lemma 1 to conclude that the sequence $(x_\nu)_{\nu\in\mathbb{N}}$, defined in the over-relaxed proximal point algorithm (ORPPA), converges to some point in $S = A^{-1}(0)$.

2 Coding Exercise (Practical part)

In this section, students who are more proficient in Python are welcome to use it, even if the questions are specifically designed for Matlab.

Objective: Implement two versions of the proximal point algorithm in MATLAB for the case where A is the subdifferential of the ℓ_1 norm on $\mathbb{R}^n$. The ℓ_1 norm is defined as $\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|$, where $\mathbf{x}$ is a vector in $\mathbb{R}^n$.

1. **Basic Proximal Point Algorithm:**

   Implement the basic proximal point algorithm where the update rule is given by:

   $$x_{\nu+1} = (\mathrm{Id} + \lambda \partial \| \cdot \|_1)^{-1} x_\nu.$$

   Here, $\lambda > 0$ is a regularization parameter, and x_0 is your initial guess (you can use a random vector in $\mathbb{R}^n$). The proximal operator of $\lambda \| \cdot \|_1$ can be computed element-wise.

   Instructions:

   - Create a MATLAB function that accepts the initial vector x_0, the regularization parameter λ, the tolerance for convergence, and the maximum iteration limit. It outputs the final iterate x_ν, the optimal value, the actual number of iterations executed, and the vector '$iter\_diff\_norm$'. This last vector tracks the norm of the difference between consecutive iterates, aiding in evaluating the algorithm's convergence speed under various λ values and initial conditions.
   - Test the function with diverse λ values and starting vectors.
   - Visualize convergence by plotting $\|x_{\nu+1} - x_\nu\|$ against the iteration count.

2. **Over-Relaxed Proximal Point Algorithm:**

   Implement the over-relaxed proximal point algorithm with the update rules:

   $$y_\nu = (\mathrm{Id} + \lambda \partial \| \cdot \|_1)^{-1} x_\nu, \quad x_{\nu+1} = x_\nu + \omega(y_\nu - x_\nu).$$

   Here, $0 < \omega < 2$ is the relaxation parameter. Use the same proximal operator as in the first part.

   Instructions:

   - Create a MATLAB function similar to the first part, but include the relaxation parameter ω as an additional input.
   - Test your implementations with different values of λ, ω, and initial vectors.
   - Plot the norm of the difference between consecutive iterates ($\|x_{\nu+1} - x_\nu\|$) against the iteration number to observe convergence behavior.
   - Plot ω against the iteration number and find the optimal value $\omega^* \in]0, 2[$.
   - Comment on the effects of different values of λ and ω on the convergence speed.

3. **Basic Proximal Point Algorithm for the LASSO:**

   Implement the basic proximal point algorithm for the LASSO problem to minimize $h(x) = f(x) + g(x)$ with $f(x) = \frac{1}{2}\|Ax - b\|_2^2$ and $g(x) = \lambda \|x\|_1$ with $\lambda > 0$. The update rule is given by:

   $$x_{\nu+1} = \mathrm{prox}_g \left(x_\nu - s \nabla f(x_\nu) \right). \tag{8}$$

   Here, $s > 0$ is a step size parameter. The proximal operator for the ℓ_1 norm term $\lambda \|x\|_1$ can be computed element-wise.

   Instructions:

- Create a MATLAB function that inputs the matrix $A \in \mathbb{R}^{m \times n}$, the vector $b \in \mathbb{R}^m$, the regularization parameter λ, the step size s, the initial vector x_0, and the number of iterations. The function should output the same as before.
- Test your implementations with different values of λ, s, and initial vectors x_0.
- Plot the norm of the difference between consecutive iterates ($\|x_{\nu+1} - x_\nu\|$) against the iteration number to observe convergence behavior.
- Comment on the effects of different values of λ and s on the convergence speed and solution quality.

You can use the following Matlab code as a starting point for your own coding (this is just for reference, and you may choose a different coding approach).

```
% Test parameters
    lambdas = [0.1, 0.5, 1]; % Different values of lambda
    num_tests = length(lambdas);
    m = 50; % Number of rows in A
    n = 100; % Number of columns in A
    max_iter = 1000; % Maximum number of iterations
    b = randn(m, 1); % Random vector b
    A = randn(m, n); % Random matrix A
    step_sizes = [0.0001, 0.001 0.005]; % Different values of step size s
```

4. **Over-Relaxed Proximal Point Algorithm for LASSO:**

Implement the over-relaxed proximal point algorithm for the LASSO problem with the following update rules:

Basic Update Step:

$$y_\nu = \operatorname{prox}_g \left(x_\nu - s\nabla f(x_\nu) \right), \; x_{\nu+1} = x_\nu + \omega(y_\nu - x_\nu). \tag{9}$$

Here, s is the step size, and prox_g is the proximal operator for the ℓ_1 norm scaled by λ, and where $0 < \omega < 2$ is the over-relaxation parameter.

Instructions:

- Test your implementations with different values of λ, ω, s, and initial vectors x_0.
- Plot the norm of the difference between consecutive iterates ($\|x_{\nu+1} - x_\nu\|$) against the iteration number to observe convergence behavior.
- Plot ω against the iteration number and find the optimal value $\omega^* \in]0, 2[$.
- Comment on the effects of different values of λ, ω, and s on the convergence speed and solution quality.