Sparse Optimization
Lecture: Proximal Operator/Algorithm and Lagrange Dual

Instructor: Wotao Yin

July 2013

online discussions on piazza.com

Those who complete this lecture will know

- learn the proximal operator and its basic properties
- the proximal algorithm
- the proximal algorithm applied to the Lagrange dual
Gradient descent / forward Euler

- Assume function $f$ is convex, differentiable
- Consider
  $$\min f(x)$$
- Gradient descent iteration (with step size $c$):
  $$x^{k+1} = x^k - c \nabla f(x^k)$$
  \(x^{k+1}\) minimizes the following local quadratic approximation of $f$:
  $$f(x^k) + \langle \nabla f(x^k), x - x^k \rangle + \frac{1}{2c} \|x - x^k\|^2_2$$
- Compare with forward Euler iteration, a.k.a. the explicit update:
  $$x(t + 1) = x(t) - \Delta t \cdot \nabla f(x(t))$$
• backward Euler iteration, also known as the implicit update:

\[ x(t + 1) \xleftarrow{\text{solve}} x(t + 1) = x(t) - \Delta t \cdot \nabla f(x(t + 1)). \]

• equivalent to:

1. \( u(t + 1) \xleftarrow{\text{solve}} u = \nabla f(x(t) - \Delta t \cdot u), \)
2. \( x(t + 1) = x(t) - \Delta t \cdot u(t + 1). \)

• we can view it as the “implicit gradient” descent:

\( (x^{k+1}, u^{k+1}) \xleftarrow{\text{solve}} x = x_k - cu, \quad u = \nabla f(x). \)

• \( c \) is the “step size”, very different from a standard step size.

• explicit (implicit) update uses the gradient at the start (end) point
Implicit gradient step = proximal operation

- **proximal update:**

\[
\text{prox}_{c,f}(z) := \arg\min_x f(x) + \frac{1}{2c} \|x - z\|^2.
\]

- **optimality condition:**

\[0 = c \nabla f(x^*) + (x^* - z).\]

- **given input** \(z\),
  - \(\text{prox}_{c,f}(z)\) returns solution \(x^*\)
  - \(\nabla f(\text{prox}_{c,f}(z))\) returns \(u^*\)

\[
(x^*, u^*) \overset{\text{solve}}{\leftarrow} x = z - cu, \ u = \nabla f(x).
\]

**Proposition**

*Proximal operator is equivalent to an implicit gradient (or backward Euler) step.*
Proximal operator handles sub-differentiable $f$

- assume that $f$ is closed, proper, sub-differentiable convex function
- $\partial f(x)$ is denoted as the subdifferential of $f$ at $x$.
  Recall $u \in \partial f(x)$ if
  \[
  f(x') \geq f(x) + \langle u, x' - x \rangle, \quad \forall x' \in \mathbb{R}^n.
  \]
- $\partial f(x)$ is point-to-set, neither direction is unique
- prox is well-defined for sub-differentiable $f$; it is point-to-point, prox maps any input to a unique point
Proximal operator

\[
\text{prox}_{cf}(z) := \arg \min_x f(x) + \frac{1}{2c} \|x - z\|^2.
\]

- since objective is strongly convex, solution \( \text{prox}_{cf}(z) \) is unique
- since \( f \) is proper, \( \text{dom \ prox}_{cf} = \mathbb{R}^n \)
- the followings are equivalent

\[
\text{prox}_{cf} z = x^* = \arg \min_x f(x) + \frac{1}{2c} \|x - z\|^2,
\]

\[
x^* \overset{\text{solve}}{\leftarrow} 0 \in c \partial f(x) + (x - z),
\]

\[
(x^*, u^*) \overset{\text{solve}}{\leftarrow} x = z - cu, \quad u \in \partial f(x).
\]

- point \( x^* \) minimizes \( f \) if and only if \( x^* = \text{prox}_f(x^*) \).
Examples

- $f(x) = \lambda_{x \in \mathcal{C}}$
- $f(x) = \frac{\lambda}{2} \|x\|_2^2$
- $f(x) = \|x\|_1$
- $f(x) = \sum_i \|x_{g_i}\|_2$
- $f(X) = \|X\|_*$
given $\text{prox}_f$ for function $f$, it is easy to derive $\text{prox}_g$ for

- $g(x) = \alpha f(x) + \beta$
- $g(x) = f(\alpha x + b)$
- $g(x) = f(x) + a^T x + \beta$
- $g(x) = f(x) + (\rho/2)\|x - a\|^2$
Resolvent of $\partial f$

- $\partial f$ is a point-to-set mapping, so is $I + c \partial f$
- in general, $(I + c \partial f)^{-1}$ is a point-to-set mapping
- however, we claim
  $$\text{prox}_{c_f} = (I + c \partial f)^{-1}$$
- since $\text{prox}_{c_f}(z)$ is always unique, $(I + c \partial f)^{-1}$ is a point-to-point mapping
- $(I + c \partial f)^{-1}$ is known as the resolvent of $\partial f$ with parameter $c$.
- by the way, $\nabla f$ is the gradient operator, and $(I - c \nabla f)$ is the gradient-descent operator.
Moreau envelope

• **idea**: to smooth a closed, proper, *nonsmooth* convex function $f$

• definition:

$$M_{cf}(x) = \inf_{y} f(y) + \frac{1}{2c} \|y - x\|^2.$$ 

• $\text{dom } M_{cf} = \mathbb{R}^n$ even if $f$ is not

• $M_{cf} \in C^1$ even if $f$ is not; in fact,

$$M_{cf} = ((cf)^* + (1/2)\| \cdot \|^2)^*$$

the dual of strongly convex function is differentiable (with Lipschitz gradient)

• relation with $\text{prox}_{cf}$
  - $\nabla M_{cf}(x) = (1/c)(x - \text{prox}_{cf}(x))$
  - $\text{prox}_{cf}(x) = x - c\nabla M_{cf}(x)$, explicit gradient step of $M_{cf}$
  - $\text{prox}_{f}(x) = \nabla M_{f^*}(x)$

• example: the Huber function is $M_f$ where $f(x) = \|x\|_1$

• $c$ is not a usual step size. As $c \to \infty$, $(\text{prox}_{cf}(x) - x) \to (x^* - x)$. 
**Proximal algorithm**

Assume that $f$ has a minimizer, then iterate

$$x^{k+1} = \text{prox}_{c_k f}(x^k)$$

\text{prox} is *firmly nonexpansive*

$$\|\text{prox}_f(x) - \text{prox}_f(y)\|^2 \leq \langle \text{prox}_f(x) - \text{prox}_f(y), x - y \rangle, \ \forall x, y \in \mathbb{R}^n$$

It converges to the minimizer as long as $c_k > 0$ and

$$\sum_{k=1}^{\infty} c_k = \infty.$$ 

For example, one can fix $c_k \equiv c$

Step-sized iteration: fix $c > 0$ and pick $\alpha_k \in (0, 2)$ uniformly away from 0 and 2:

$$x^{k+1} = \alpha_k \text{prox}_{c f}(x^k) + (1 - \alpha^k)x^k.$$

The convergence takes a *finite* number of iterations if $f$ is polyhedral (i.e. piece-wise linear)
Proximal algorithm

Diminishing regularization

\[ x^{k+1} = \arg\min f(x) + \frac{1}{2c} \|x - x^k\|^2 \]

As \( x^k \to x^* \), \( \|\partial f(x^k)\| \to 0 \) and thus \( f(x) \) becomes “weaker.” Hence, \( x^{k+1} - x^k \) tends to be smaller.

Many algorithms use \( \text{prox}_{c_k}f(x^k) \), either entirely or as a part (but most of them were motivated through other means).

Although \( \text{prox}_{c_k}f \) can sometimes be difficult to compute, it simplifies computation

- for some sub-differentiable functions
- for those rising in duality (our next focus)
Lagrange duality

Convex problem

\[
\min f(x) \quad \text{s.t. } Ax = b.
\]

Relax the constraints and price their violation (pay a price if violated one way; get paid if violated the other way; payment is linear to the violation)

\[
\mathcal{L}(x; y) := f(x) + y^T(Ax - b)
\]

For later use, define the augmented Lagrangian

\[
\mathcal{L}_A(x; y, c) := f(x) + y^T(Ax - b) + \frac{c}{2}\|Ax - b\|^2
\]

Minimize \( \mathcal{L} \) for fixed price \( y \):

\[
d(y) := -\min_x \mathcal{L}(x; y).
\]

Always, \( d(y) \) is convex

The Lagrange dual problem

\[
\min d(y)
\]

Given dual solution \( y^* \), recover \( x^* = \min_x \mathcal{L}(x; y^*) \) (under which conditions?)

Question: how to compute the explicit/implicit gradients of \( d(y) \)?
Dual explicit gradient (ascent) algorithm

Assume $d(y)$ is differentiable (true if $f(x)$ is strictly convex. Is this if-and-only-if?)

Gradient descent iteration (if the maximizing dual is used, it is called gradient ascent):

$$y^{k+1} = y^k - c \nabla f(y^k).$$

It turns out to be relatively easy to compute $\nabla d$, via an unstrained subproblem:

$$\nabla d(y) = b - A\bar{x}, \quad \text{where } \bar{x} = \arg\min_x L(x; y).$$

Dual gradient iteration

1. $x^k \xleftarrow{\text{solve}} \min_x L(x; y^k)$;
2. $y^{k+1} = y^k - c(b - Ax^k)$. 
Sub-gradient of $d(y)$

Assume $d(y)$ is sub-differentiable (which condition on primal can guarantee this?)

**Lemma**

*Given dual point $y$ and $\bar{x} = \arg\min_x \mathcal{L}(x; y)$, we have $b - A\bar{x} \in \partial d(y)$.***

**Proof.**

Recall

- $u \in \partial d(y)$ if $d(y') \geq d(y) + \langle u, y' - y \rangle$ for all $y'$;
- $d(y) := -\min_x \mathcal{L}(x; y)$.

From (ii) and definition of $\bar{x}$,

\[
\begin{align*}
d(y) + \langle b - A\bar{x}, y' - y \rangle &= -\mathcal{L}(\bar{x}; y) + (b - A\bar{x})^T (y - y') \\
&= -[f(\bar{x} + y^T (A\bar{x} - b))] + (b - A\bar{x})^T (y - y') \\
&= -[f(\bar{x}) + (y')^T (A\bar{x} - b)] \\
&= -\mathcal{L}(\bar{x}; y') \leq d(y').
\end{align*}
\]

From (i), $b - A\bar{x} \in \partial d(y)$. □
Dual explicit (sub)gradient iteration

The iteration:

1. $x_k^{\text{solve}} \leftarrow \min_x \mathcal{L}(x; y^k)$;
2. $y^{k+1} = y^k - c_k(b - Ax^k)$;

Notes:

- $(b - Ax^k) \in \partial d(y^k)$ as shown in the last slide
- it does not require $d(y)$ to be differentiable
- convergence might require a careful choice of $c_k$ (e.g., a diminishing sequence) if $d(y)$ is only sub-differentiable (or lacking Lipschitz continuous gradient)
Dual implicit gradient

**Goal**: to descend using the (sub)gradient of $d$ at the *next point* $y^{k+1}$:

Following from the Lemma, we have

$$b - Ax^{k+1} \in \partial d(y^{k+1})$$

where $x^{k+1} = \arg\min_x \mathcal{L}(x; y^{k+1})$

Since the implicit step is $y^{k+1} = y^k - c(b - Ax^{k+1})$, we can derive

$$x^{k+1} = \arg\min_x \mathcal{L}(x; y^{k+1}) \iff \quad 0 \in \partial_x \mathcal{L}(x^{k+1}; y^{k+1}) = \partial f(x^{k+1}) + A^T y^{k+1}$$

$$= \partial f(x^{k+1}) + A^T (y^k - c(b - Ax^{k+1})).$$

Therefore, while $x^{k+1}$ is a solution to $\min_x \mathcal{L}(x; y^{k+1})$; it is also a solution to

$$\min_x \mathcal{L}_A(x; y^k, c) = f(x) + (y^k)^T (Ax - b) + \frac{c}{2} \|Ax - b\|^2,$$

which is *independent* of $y^{k+1}$. 
Dual implicit gradient

Proposition

Assuming \( y' = y - c(b - Ax') \), the followings are equivalent

1. \( x' \xleftarrow{\text{solve}} \min_x \mathcal{L}(x; y') \),
2. \( x' \xleftarrow{\text{solve}} \min_x \mathcal{L}_A(x; y, c) \).
Dual implicit gradient iteration

The iteration

\[ y^{k+1} = \text{prox}_{cd}(y^k) \]

is commonly known as the augmented Lagrangian method or the method of multipliers.

Implementation:

1. \( x^{k+1} \xleftarrow{\text{solve}} \min_x \mathcal{L}_A(x; y^k, c) \);
2. \( y^{k+1} = y^k - c(b - Ax^{k+1}) \).

Proposition

The followings are equivalent

1. the augmented Lagrangian iteration;
2. the implicit gradient iteration of \( d(y) \);
3. the proximal iteration \( y^{k+1} = \text{prox}_{cd}(y^k) \).
Dual explicit/implicit (sub)gradient computation

Definitions:

- $\mathcal{L}(x; y) = f(x) + y^T(Ax - b)$
- $\mathcal{L}_A(x; y, c) = \mathcal{L}(x; y) + \frac{c}{2} ||Ax - b||^2$

Objective:

$$d(y) = -\min_x \mathcal{L}(x; y).$$

Explicit (sub)gradient iteration: $y^{k+1} = y^k - c \nabla d(y^k)$ or use a subgradient $\partial d(y^k)$

1. $x^{k+1} = \arg \min_x \mathcal{L}(x; y^k)$;
2. $y^{k+1} = y^k - c(b - Ax^{k+1})$.

Implicit (sub)gradient step: $y^{k+1} = \text{prox}_{cd}y^k$

1. $x^{k+1} = \arg \min_x \mathcal{L}_A(x; y^k, c)$;
2. $y^{k+1} = y^k - c(b - Ax^{k+1})$.

The implicit iteration is more stable; “step size” $c$ does not need to diminish.