The Proximal-proximal Gradient Algorithm

Ting Kei Pong
PIMS Postdoctoral Fellow
Department of Computer Science
University of British Columbia
Vancouver

WCOM, Autumn
Oct 2013
Outline

- Motivations.
- The proximal-proximal gradient algorithm.
- Convergence and complexity results.
- Other related algorithms.
- Numerical results.
Motivations

- System realization problem: (Liu, Vandenberghe ’08)

\[
\min_{z} \frac{1}{2} \| w \circ z - w \circ \bar{z} \|_F^2 + \mu \| H(z) \|_*,
\]

where \( z = (z_0 \cdots z_{j+k-1}) \in \mathbb{R}^{m \times n(j+k)} \), \( w \) is zero-one matrix,

\[
H(z) = \begin{pmatrix}
    z_0 & z_1 & \cdots & z_{k-1} \\
    z_1 & \cdots & \cdots & z_k \\
    \vdots & \vdots & \ddots & \vdots \\
    z_{j-1} & z_j & \cdots & z_{j+k-2}
\end{pmatrix} \in \mathbb{R}^{mj \times nk}.
\]

- Logistic fused LASSO: (Ma, Zhang ’13)

\[
\min_{z \in \mathbb{R}^n, t \in \mathbb{R}} \sum_{i=1}^{m} \log(1 + \exp(-b_i(a_i^Tz + t))) + \lambda_1 \| z \|_1 + \lambda_2 \sum_{i=1}^{n-1} |z_{i+1} - z_i|.
\]
General Problem

\[ \min_z \quad h(z) + P(Mz), \]

where:

- \( h \) is smooth, \( \nabla h \) is Lipschitz continuous with modulus \( L \);
- \( P \) is proper closed convex, with “easy” proximal operator;
- \( M \) is nonzero linear map;
- Assume the solution set is nonempty and

\[ \text{Range}(M) \cap \text{ri(dom}(P)) \neq \emptyset. \]
Proximal Operator

For a proper closed convex function $P$,

$$\text{prox}_P(y) := \arg \min_z \left\{ P(z) + \frac{1}{2} \| z - y \|^2 \right\}.$$ 

This is well-defined for all $y$. 
Proximal Operator

For a proper closed convex function $P$,

$$\text{prox}_P(y) := \arg \min_z \left\{ P(z) + \frac{1}{2}\|z - y\|^2 \right\}.$$ 

This is well-defined for all $y$.

Some prox are “easy”:

- If $P(z) = \mu \|z\|_1$, then $\text{prox}_P(y) = \text{sign}(y) \circ \max\{|y| - \mu, 0\}$.

- If $P(Z) = \mu \|Z\|_*$, then $\text{prox}_P(Y) = U \text{Diag}(\max\{\sigma(Y) - \mu, 0\}) V^T$, where $Y = U \text{Diag}(\sigma(Y)) V^T$ is an SVD of $Y$. 
Proximal Gradient Algorithm

$$\min_z h(z) + P(Mz),$$

Replace the smooth part with a quadratic approximation:
The Proximal-proximal Gradient Algorithm

Proximal Gradient Algorithm

\[
\min_z \quad h(z) + P(Mz),
\]

Replace the smooth part with a quadratic approximation:

For \( t = 0, 1, 2, \ldots \), update

\[
\begin{align*}
    z^{t+1} &= \arg\min_z \left\{ \langle \nabla h(z^t), z - z^t \rangle + \frac{L}{2} \| z - z^t \|^2 + P(Mz) \right\} \\
    &= \arg\min_z \left\{ \frac{L}{2} \left\| z - \left( z^t - \frac{1}{L} \nabla h(z^t) \right) \right\|^2 + P(Mz) \right\}.
\end{align*}
\]
The Proximal-proximal Gradient Algorithm

**Proximal Gradient Algorithm**

\[
\min_z \quad h(z) + P(Mz),
\]

Replace the smooth part with a quadratic approximation:

For \( t = 0, 1, 2, \ldots \), update

\[
z^{t+1} = \arg \min_z \left\{ \langle \nabla h(z^t), z - z^t \rangle + \frac{L}{2}\|z - z^t\|^2 + P(Mz) \right\}
\]

\[
= \arg \min_z \left\{ \frac{L}{2}\left\|z - \left( z^t - \frac{1}{L} \nabla h(z^t) \right) \right\|^2 + P(Mz) \right\}.
\]

Essentially computing \( \text{prox} \) for \( \frac{1}{L}P \circ M \): NOT necessarily easy...
Inexact Proximal Gradient

One solution: use iterative method to solve subproblem.

\[
\min_z \left\{ \frac{L}{2} \left\| z - \left( z^t - \frac{1}{L} \nabla h(z^t) \right) \right\|^2 + P(Mz) \right\}
\]

\[
= \max_y \left\{ -\frac{1}{2L} \| M^*y \|^2 + \langle M^*y, z^t - \frac{1}{L} \nabla h(z^t) \rangle - P^*(y) \right\}.
\]

Moreover, if \( \tilde{y}^{t+1} \) solves the maximization problem, then

\[
z^{t+1} = z^t - \frac{1}{L} \left( \nabla h(z^t) + M^*\tilde{y}^{t+1} \right)
\]

solves the minimization problem. This will be the \( z \)-update.
Inexact Proximal Gradient

Solve the subproblem also using proximal gradient algorithm:

- Initialize \( z^0, y^0 \). Set \( \beta = \frac{1}{L} \) and \( \tau \geq \beta \| \mathcal{M} \|^2 \).

- For \( t = 0, 1, 2, \ldots \)
  - For \( s = 0, 1, 2, \ldots \), starting with \( u^0 = y^t \), (warm start)
    \[
    u^{s+1} = \text{prox}_{\tau^{-1} P_*} \left( u^s - \frac{1}{\tau} \left( \beta \mathcal{M} \mathcal{M}^* u^s - \mathcal{M} (z^t - \beta \nabla h(z^t)) \right) \right).
    \]
  - Get approximate solution \( y^{t+1} = u^{s+1} \).

Update \( z^{t+1} = z^t - \beta (\nabla h(z^t) + \mathcal{M}^* y^{t+1}) \).
Proximal-proximal Gradient Algorithm

- Initialize $z^0, y^0$. Set $\beta \in (0, \frac{2}{L})$, $\gamma \in (0, 1 + \min\{\frac{1}{2}, \frac{1}{\beta L} - \frac{1}{2}\})$ and $\tau \geq \beta \|M\|^2$.

- For $t = 0, 1, 2, \ldots$

\[
\begin{align*}
    y^{t+1} &= \text{prox}_{\tau^{-1}P^*} \left( y^t - \frac{1}{\tau} \left( \beta MM^* y^t - M (z^t - \beta \nabla h(z^t)) \right) \right), \\
    z^{t+1} &= z^t - \gamma \beta (\nabla h(z^t) + M^* y^{t+1}).
\end{align*}
\]
The Proximal-proximal Gradient Algorithm

Proximal-proximal Gradient Algorithm

- Initialize $z^0, y^0$. Set $\beta \in (0, \frac{2}{L})$, $\gamma \in (0, 1 + \min\{\frac{1}{2}, \frac{1}{\beta L} - \frac{1}{2}\})$ and $\tau \geq \beta \|\mathcal{M}\|^2$.

- For $t = 0, 1, 2, \ldots$

\[
\begin{align*}
    y^{t+1} &= \text{prox}_{\frac{\tau}{1}} P^* \left( y^t - \frac{1}{\tau} \left( \beta \mathcal{M} \mathcal{M}^* y^t - \mathcal{M} (z^t - \beta \nabla h(z^t)) \right) \right), \\
    z^{t+1} &= z^t - \gamma \beta (\nabla h(z^t) + \mathcal{M}^* y^{t+1}).
\end{align*}
\]

Remarks:

- This is basically a very inexact proximal gradient algorithm. No need to worry about inner loop accuracy ($s = 0$).

- Computing the prox of $P^*$ is easy, by Moreau’s identity.
Convergence

Fenchel dual problem:

\[ v_{\text{opt}} := \min_{x,y} h^*(x) + P^*(y) \]
\[ \text{s.t. } x + M^*y = 0, \]

where \( f^*(u) = \sup_z \{ \langle u, z \rangle - f(z) \} \).

Fact 1 (P'13): Let \( \{(y^t, z^t)\} \) be generated from the PPG algorithm, and set \( x^{t+1} = \nabla h(z^t) \). Then \( \{z^t\} \) converges to a primal optimal solution, and \( \{(x^t, y^t)\} \) converges to a dual optimal solution. Moreover

\[-\frac{C_2}{\sqrt{N}} \leq h^*(\bar{x}^N) + P^*(\bar{y}^N) - v_{\text{opt}} \leq \frac{C_1}{N}, \quad \|\bar{x}^N + M^*\bar{y}^N\| \leq \frac{C_3}{\sqrt{N}},\]

where \((\bar{x}^N, \bar{y}^N) = \frac{1}{N} \sum_{t=1}^{N} (x^t, y^t)\).
Intuition Behind Convergence

The proximal gradient algorithm (with $\beta \in \left(0, \frac{2}{L}\right)$ in place of $\frac{1}{L}$) is the same as the alternating minimization algorithm applied to the Fenchel dual (Tseng ’91):

- Initialize $z^0, y^0$. Set $\beta \in \left(0, \frac{2}{L}\right)$.
- For $t = 0, 1, 2, \ldots$

\[
\begin{align*}
x^{t+1} &= \arg \min_x \left\{ h^*(x) - \langle z^t, x \rangle \right\}, \\
y^{t+1} &\in \text{Arg min}_y \left\{ P^*(y) - \langle z^t, \mathcal{M}^*y \rangle + \frac{\beta}{2} \| x^{t+1} + \mathcal{M}^*y \|^2 \right\}, \\
z^{t+1} &= z^t - \beta(x^{t+1} + \mathcal{M}^*y^{t+1}).
\end{align*}
\]
THE PROXIMAL-PROXIMAL GRADIENT ALGORITHM

Intuition Behind Convergence

Adding “proximal term”, we get the PPG algorithm with \( \gamma = 1 \):

- Initialize \( z^0, y^0 \). Set \( \beta \in (0, \frac{2}{L}) \) and \( \tau \geq \beta \|M\|^2 \).
- For \( t = 0, 1, 2, \ldots \)

\[
\begin{align*}
x^{t+1} &= \arg \min_x \{ h^*(x) - \langle z^t, x \rangle \}, \\
y^{t+1} &\in \text{Arg min}_y \left\{ P^*(y) - \langle z^t, M^* y \rangle + \frac{\beta}{2} \|x^{t+1} + M^* y\|^2 \\
&\quad+ \frac{1}{2} \langle y - y^t, [\tau I - \beta M M^*](y - y^t) \rangle \right\}, \\
z^{t+1} &= z^t - \beta(x^{t+1} + M^* y^{t+1}).
\end{align*}
\]
Facts about PPG

- The PPG algorithm reduces to the proximal gradient algorithm if \( \tau = \beta = \frac{1}{L} \), \( \gamma = 1 \) and \( \mathcal{M} = \mathcal{I} \); or, more generally, \( \mathcal{M}\mathcal{M}^* = \mathcal{I} \).

- If \( h(z) = \frac{1}{2}\|z - a\|^2 \), \( \beta = 1 = \frac{1}{L} \) and \( \gamma = 1 \), then the PPG algorithm reduces to the proximal gradient algorithm applied to the Fenchel dual directly.

- Problems in the form of

\[
\min_z h(z) + \sum_{i=1}^m P_i(z),
\]

can be solved by setting

\[
P(z_1, \ldots, z_m) = \sum_{i=1}^m P_i(z_i) \text{ and } \mathcal{M}z = (z, \ldots, z)_m.
\]
Other Approaches for the General Problem

- A recent algorithm by (Condat ’13) and (Vu ’13) is closely related but yet different from the PPG algorithm.

- Convex-concave minimization maximization:

\[ \min_{z} \max_{y} h(z) + \langle M^* y, z \rangle - P^*(y). \]

A lot of algorithms for solving this type of problem and its variants; see the textbook by (Bauschke, Combettes ’08). Most of them reduce to the modified forward-backward splitting method (Tseng ’01).
Numerical Simulations

• Compare the PPG algorithm and the MFBS method on system realization problem.

\[
\min_z \frac{1}{2} \| w \circ (z - \bar{z}) \|_F^2 + \mu \| \mathcal{H}(z) \|_\star.
\]

Instances randomly generated. Results averaged over 10 instances.

• Terminate when relative duality gap and relative dual infeasibility are below \(10^{-4}\) and \(2 \times 10^{-5}\), respectively.

• Two 2.4 GHz quad-core Intel E5620 Xeon 64-bit CPUs, 48 GB RAM, Matlab 7.14 (R2012a).
The Proximal-proximal Gradient Algorithm

Numerical Simulations

- Matrix size: $210 \times 10^k$.

- Set $\beta = 0.05/L$ for $\mu \geq 0.1$, and $\beta = 1/L$ else. Set $\tau = \beta \|H\|^2$ and $\gamma = 1 + 0.95 \min\left\{\frac{1}{2}, \frac{1}{\beta L} - \frac{1}{2}\right\}$.

<table>
<thead>
<tr>
<th>$k$</th>
<th>$\mu$</th>
<th>PPG</th>
<th>MFBS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>iter</td>
<td>cpu</td>
</tr>
<tr>
<td>100</td>
<td>0.05</td>
<td>123</td>
<td>7.4</td>
</tr>
<tr>
<td>100</td>
<td>0.10</td>
<td>82</td>
<td>4.7</td>
</tr>
<tr>
<td>100</td>
<td>0.50</td>
<td>58</td>
<td>3.8</td>
</tr>
<tr>
<td>200</td>
<td>0.05</td>
<td>41</td>
<td>4.8</td>
</tr>
<tr>
<td>200</td>
<td>0.10</td>
<td>100</td>
<td>11.7</td>
</tr>
<tr>
<td>200</td>
<td>0.50</td>
<td>51</td>
<td>5.9</td>
</tr>
<tr>
<td>300</td>
<td>0.05</td>
<td>30</td>
<td>5.0</td>
</tr>
<tr>
<td>300</td>
<td>0.10</td>
<td>156</td>
<td>28.7</td>
</tr>
<tr>
<td>300</td>
<td>0.50</td>
<td>53</td>
<td>8.8</td>
</tr>
</tbody>
</table>
Conclusion and Future Directions

- The PPG algorithm admits easy subproblems per iteration
- The algorithm is an “inexact” proximal gradient algorithm, and can also be viewed as a proximal alternating minimization algorithm.
- Acceleration of the PPG algorithm?
- Consider other “inexact” algorithms?

Thanks for coming! 😊