Nesterov加速算法

宋晓良

大连理工大学数学科学学院

教材《最优化：建模、算法与理论》配套电子教案

http://bicmr.pku.edu.cn/~wenzw/optbook.html

提纲

典型问题形式

考虑如下复合优化问题：

$$\min_{x\in\mathbb{R}^n} \quad \psi(x) = f(x) + h(x) \tag{1}$$

- $f(x)$是连续可微的凸函数，且梯度是利普西茨连续的：

$$\|\nabla f(x) - \nabla f(y)\| \le L\|x-y\|;$$

- $h(x)$是适当的闭凸函数，且临近算子

$$\mathrm{prox}_h(x) = \operatorname*{argmin}_{u\in \mathrm{dom} h}\left\{h(u) + \frac{1}{2}\|x-u\|^2\right\}$$

  容易计算.

- 对于上述问题，近似点梯度法

$$x^{k+1} = \mathrm{prox}_{t_k h}(x^k - t_k\nabla f(x^k))$$

  在步长取常数$t_k = 1/L$时，收敛速度为$\mathcal{O}(1/k)$.

Nesterov加速算法简史

- 一个自然的问题是如果仅用梯度信息，我们能不能取得更快的收敛速度．
- Nesterov分别在1983年、1988年和2005年提出了三种改进的一阶算法，收敛速度能达到$\mathcal{O}\left(\frac{1}{k^2}\right)$．实际上，这三种算法都可以应用到近似点梯度算法上．
- 在Nesterov加速算法刚提出的时候，由于牛顿算法有更快的收敛速度，Nesterov加速算法在当时并没有引起太多的关注．但近年来，随着数据量的增大，牛顿型方法由于其过大的计算复杂度，不便于有效地应用到实际中，Nesterov加速算法作为一种快速的一阶算法重新被挖掘出来并迅速流行起来．
- Beck和Teboulle就在2008年给出了Nesterov在1983年提出的算法的近似点梯度法版本——FISTA．

FISTA

- FISTA算法由两步组成：第一步沿着前两步的计算方向计算一个新点，第二步在该新点处做一步近似点梯度迭代（如图所示）.

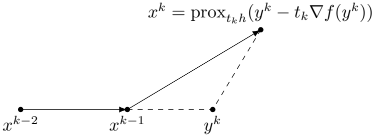

Figure: FISTA算法图示

- 完整的FISTA见算法2：

$$y^k = x^{k-1} + \frac{k-2}{k+1}(x^{k-1} - x^{k-2})$$
$$x^k = \mathrm{prox}_{t_k h}(y^k - t_k \nabla f(y^k)) \tag{2}$$

FISTA的等价形式

- 算法3给出了FISTA的一个等价变形：

$$\begin{aligned} y^k &= (1-\gamma_k)x^{k-1} + \gamma_k v^{k-1} \\ x^k &= \mathrm{prox}_{t_k h}(y^k - t_k \nabla f(y^k)) \\ v^k &= x^{k-1} + \frac{1}{\gamma_k}(x^k - x^{k-1}) \end{aligned} \tag{3}$$

- 当$\gamma_k = \frac{2}{k+1}$时，并且取固定步长时，两个算法是等价的；
- 但是当γ_k采用别的取法时，算法3 将给出另一个版本的加速算法.
- 也就是说，算法2中$\frac{k-2}{k+1}$可以取其他值.

FISTA算法小结

- 总的来说，固定步长的FISTA算法对于步长的选取是较为保守的，为了保证收敛，有时不得不选取一个很小的步长，这使得固定步长的FISTA算法收敛较慢.
- 如果采用线搜索，则在算法执行过程中会有很大机会选择符合条件的较大步长，因此线搜索可能加快算法的收敛，但代价就是每一步迭代的复杂度变高.
- 在实际的FISTA算法中，需要权衡固定步长和线搜索算法的利弊，从而选择针对特定问题的高效算法.

下降FISTA算法

- 原始的FISTA算法不是一个下降算法，这里给出一个FISTA的下降算法变形．
- 只需要对算法3的第2步进行修改．在计算邻近算子之后，我们并不立即选取此点作为新的迭代点，而是检查函数值在当前点处是否下降，只有当函数值下降时才更新迭代点．
- 假设经过近似点映射之后的点为u，则对当前点x^k做如下更新：

$$x^k = \begin{cases} u, & \psi(u) \leq \psi(x^{k-1}), \\ x^{k-1}, & \psi(u) > \psi(x^{k-1}). \end{cases} \tag{4}$$

- 由于步长或γ_k会随着k变化，(4)式中的$\psi(u) > \psi(x^{k-1})$不会一直成立，即算法不会停留在某个x^{k-1}而不进行更新．
- 步长和γ_k的选取只需使用固定步长$t_k \leq \frac{1}{L}$，$\gamma_k = \frac{2}{k+1}$或者使用前述的任意一种线搜索方法均可．

提纲

第二类Nesterov加速算法

- 对于复合优化问题(1)，我们给出第二类Nesterov加速算法：

$$\begin{aligned} z^k &= (1-\gamma_k)x^{k-1} + \gamma_k y^{k-1} \\ y^k &= \operatorname{prox}_{(t_k/\gamma_k)h}\left(y^{k-1} - \frac{t_k}{\gamma_k}\nabla f(z^k)\right) \\ x^k &= (1-\gamma_k)x^{k-1} + \gamma_k y^k \end{aligned} \tag{5}$$

- 和经典FISTA 算法的一个重要区别在于，第二类Nesterov 加速算法中的三个序列$\{x^k\}$，$\{y^k\}$和$\{z^k\}$都可以保证在定义域内．而FISTA 算法中的序列$\{y^k\}$不一定在定义域内．

第二类Nesterov加速算法

- 第二类Nesterov 加速算法的一步迭代可参考下图.

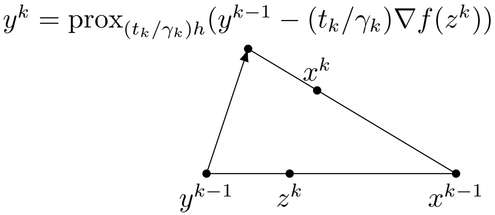

Figure: 第二类Nesterov加速算法的一步迭代

第三类Nesterov加速算法

- 针对问题(1)的第三类Nesterov加速算法框架为：

$$
\begin{aligned}
z^k &= (1-\gamma_k)x^{k-1} + \gamma_k y^{k-1} \\
y^k &= \mathrm{prox}_{(t_k \sum_{i=1}^{k} 1/\gamma_i)h}\left(-t_k \sum_{i=1}^{k} \frac{1}{\gamma_i} \nabla f(z^i)\right) \\
x^k &= (1-\gamma_k)x^{k-1} + \gamma_k y^k
\end{aligned} \tag{6}
$$

- 该算法和第二类Nesterov加速算法（算法5）的区别仅仅在于y^k的更新：第三类Nesterov加速算法计算y^k时需要利用全部已有的$\{\nabla f(z^i)\}, i = 1, 2, \cdots, k$.
- 同样地，该算法取$\gamma_k = \frac{2}{k+1}$，$t_k = \frac{1}{L}$时，也有$\mathcal{O}\left(\frac{1}{k^2}\right)$的收敛速度.

针对非凸问题的Nesterov加速算法

- 仍然考虑问题(1)的形式，这里并不要求f是凸的，但是要求其是可微的且梯度是利普希茨连续的，h与之前的要求相同.
- 算法7给出非凸复合优化问题的加速梯度法框架.

$$\begin{aligned} z^k &= \gamma_k y^{k-1} + (1-\gamma_k)x^{k-1} \\ y^k &= \mathrm{prox}_{\lambda_k h}\left(y^{k-1} - \lambda_k \nabla f(z^k)\right) \\ x^k &= \mathrm{prox}_{t_k h}\left(z^k - t_k \nabla f(z^k)\right) \end{aligned} \tag{7}$$

提纲

- LASSO问题为

$$\min_x \quad \frac{1}{2}\|Ax-b\|_2^2+\mu\|x\|_1 \tag{8}$$

- 求解LASSO问题(8)的FISTA算法可以由下面的迭代格式给出：

$$\begin{aligned} y^k &= x^{k-1}+\frac{k-2}{k+1}(x^{k-1}-x^{k-2}), \\ w^k &= y^k - t_k A^T(Ay^k-b), \\ x^k &= \operatorname{sign}(w^k)\max\{|w^k|-t_k\mu, 0\}. \end{aligned}$$

- 与近似点梯度算法相同，由于最后一步将w^k中绝对值小于$t_k\mu$的分量置零，该算法能够保证迭代过程中解具有稀疏结构．

- 我们也给出第二类Nesterov加速算法：

$$
\begin{aligned}
z^k &= (1-\gamma_k)x^{k-1} + \gamma_k y^{k-1}, \\
w^k &= y^{k-1} - \frac{t_k}{\gamma_k} A^T(Az^k - b), \\
y^k &= \operatorname{sign}(w^k) \max\left\{ |w^k| - \frac{t_k}{\gamma_k}\mu, 0 \right\}, \\
x^k &= (1-\gamma_k)x^{k-1} + \gamma_k y^k,
\end{aligned}
$$

- 和第三类Nesterov加速算法：

$$
\begin{aligned}
z^k &= (1-\gamma_k)x^{k-1} + \gamma_k y^{k-1}, \\
w^k &= -t_k \sum_{i=1}^{k} \frac{1}{\gamma_i} A^T (Az^i - b), \\
y^k &= \operatorname{sign}(w^k) \max\left\{ |w^k| - t_k \sum_{i=1}^{k} \frac{1}{\gamma_i}\mu, 0 \right\}, \\
x^k &= (1-\gamma_k)x^{k-1} + \gamma_k y^k.
\end{aligned}
$$

LASSO问题求解（续）

- 取$\mu = 10^{-3}$，分别利用连续化近似点梯度法、连续化FISTA加速算法、连续化第二类Nesterov算法来求解问题
- 分别取固定步长$t = \frac{1}{L}$，这里$L = \lambda_{\max}(A^TA)$，和结合线搜索的BB步长.
- 结果如下图：

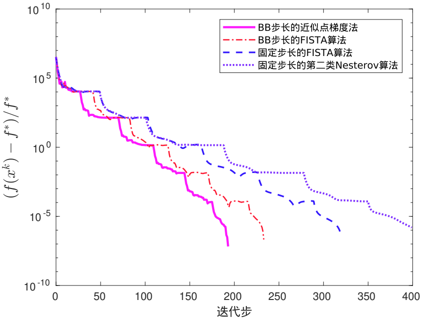

Figure: 使用近似点梯度法以及不同的加速算法求解LASSO 问题

LASSO问题求解（续）

可以看到：

- 就固定步长而言，FISTA算法相较于第二类Nesterov加速算法收敛得略快一些；
- 注意到FISTA算法是非单调算法．
- BB步长和线搜索技巧可以加速算法的收敛速度．
- 带线搜索的近似点梯度法可以比带线搜索的FISTA算法更快收敛．

提纲

收敛性假设

- f 在其定义域$\textbf{dom}\, f = \mathbb{R}^n$内为凸的，$\nabla f$在常数$L$意义下利普西茨连续，即
$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \qquad \forall x, y;$$
- h是适当的闭凸函数；
- $\psi(x)$的最小值ψ^*是有限的，并且在点x^*处可以取到．

固定步长近似点梯度法的收敛速度

首先回顾固定步长近似点梯度法的收敛速度：

定理

在上述收敛性假设的条件下，取定步长$t_k = t \in (0, 1/L]$. 设$\{x^k\}$是由近似点梯度法迭代产生的序列，则

$$\psi(x^k) - \psi^* \le \frac{1}{2kt}\|x^0 - x^*\|^2 \tag{9}$$

因此，近似点梯度法的收敛速度为$\mathcal{O}(1/k)$；而固定步长FISTA算法则可以加速到$\mathcal{O}(1/k^2)$.

固定步长FISTA算法收敛速度

定理 (固定步长FISTA算法收敛速度)

在上述收敛性假设的条件下，当用算法3求解凸复合优化问题(1)时，若取固定步长$t_k = \frac{1}{L}$，则

$$\psi(x^k) - \psi(x^*) \leq \frac{2L}{(k+1)^2}\|x^0 - x^*\|^2. \tag{10}$$