最优化实验报告

左一泓 PB22000116

日期：June 21, 2024

摘　要

本文研究了稀疏二分类逻辑回归问题，采用近似点梯度法和 FISTA（Fast Iterative Shrinkage Thresholding Algorithm）算法进行求解，并比较了两者的收敛速率及参数选择对收敛性和解的影响。实验结果表明，FISTA 算法在迭代次数和收敛速度上显著优于近似点梯度法，且能更有效地控制解的稀疏性。本文还探讨了正则化参数 μ 对解稀疏性的影响，发现较大的 μ 值可以显著增加解的稀疏性并减少迭代次数。最后，对未来可能的研究方向进行了展望，提出了进一步扩展和优化的建议。

1　理论背景

1.1　近似点梯度法

1.1.1　历史发展

近似点梯度法（Proximal Gradient Method）是一种用于处理具有非光滑正则项的优化问题的方法。其起源可以追溯到凸优化领域中的早期工作，例如由 Moreau 在 1962 年提出的近似点映射（Proximal Mapping）概念。该方法最初主要用于凸优化问题，但随着机器学习和统计领域的发展，尤其是稀疏学习（Sparse Learning）的兴起，近似点梯度法得到了广泛应用。

1.1.2　理论基础

近似点梯度法的核心思想是将优化问题分解为两个部分：一个光滑部分和一个非光滑部分。对于光滑部分，采用梯度下降法进行优化；对于非光滑部分，使用近似点映射来处理。具体来说，对于目标函数：

$$\min_{x} \ f(x) + g(x)$$

其中 $f(x)$ 是光滑且可微的，而 $g(x)$ 是可能非光滑的凸函数。每次迭代更新的过程如下：

$$x^{k+1} = \mathrm{Prox}_{t^k g}(x^k - t^k \nabla f(x^k))$$

其中，$\mathrm{Prox}_{t^k g}(v)$ 表示近似点映射，定义为：

$$\mathrm{Prox}_{t^k g}(v) = \arg\min_u \left(g(u) + \frac{1}{2t^k} \|u - v\|^2 \right)$$

1.1.3 与其他优化算法的对比

与传统的梯度下降法相比，近似点梯度法可以处理非光滑项，因此在稀疏优化问题中表现优异。相比于坐标下降法（Coordinate Descent），近似点梯度法可以同时更新所有变量，通常在高维问题上更有效。

1.2 FISTA 算法

1.2.1 历史发展

FISTA（Fast Iterative Shrinkage-Thresholding Algorithm）由 Beck 和 Teboulle 在 2009 年提出，是对近似点梯度法的加速版本。其灵感来自于 Nesterov 在 1983 年提出的加速梯度方法，用于提高优化算法的收敛速度。FISTA 特别适用于大规模稀疏优化问题，在理论上和实践中都显示了显著的性能提升。

1.2.2 理论基础

FISTA 在每次迭代中引入了一个动量项，使得算法能够更快地收敛到最优解。具体而言，FISTA 的迭代更新过程如下：

1. 计算动量项：

$$y^k = x^k + \frac{k-1}{k+2}(x^k - x^{k-1})$$

2. 更新变量：

$$x^{k+1} = \mathrm{Prox}_{t^k \mu \|\cdot\|_1}(y^k - t^k \nabla f(y^k))$$

1.2.3 与其他优化算法的对比

与近似点梯度法相比，FISTA 通过引入 Nesterov 加速技术，显著提高了收敛速度。在光滑部分 L 光滑的假设下，FISTA 可以在理论上获得 $O(1/k^2)$ 的收敛速度，而近似点梯度法只能达到 $O(1/k)$ 的收敛速度。相比于其他加速梯度方法，FISTA 在处理稀疏优化问题时表现尤为优越。

2 实验设计

本文考虑用近似点梯度法和 FISTA 算法 (Fast Iterative Shrinkage Thresholding Algorithm) 求解如下稀疏二分类逻辑回归问题：

$$\begin{aligned}\min_{x} \quad \ell(x) &\overset{def}{=} \frac{1}{m}\sum_{i=1}^{m} \ln\left(1+\exp\left(-b_i a_i^{\mathrm{T}} x\right)\right) + \lambda\|x\|_2^2 + \mu\|x\|_1 \\ &\overset{def}{=} f(x) + \mu\|x\|_1\end{aligned} \tag{1}$$

并比较它们的收敛速率以及参数选择对于收敛性和解的影响。

2.1 定义与记号

$\#$ 表示集合元素个数,$\|.\|$ 在没有角标时都表示二范数，$\|.\|_p$ 表示 p 范数。

近似点映射：

$$\begin{aligned}\mathrm{Prox}_{\mu\|x\|_1}(x) &= \operatorname*{argmin}_{t}\left(\frac{1}{2}\|t-x\|^2 + \mu\|t\|_1\right) \\ &= \mathrm{sign}(x)\max\{|x|-\mu, 0\}\end{aligned}$$

f 的微分具体表达式如下：

$$\begin{aligned}\nabla f(x) &= -\frac{1}{m}\sum_{i=1}^{m}(1-p_i(x))\, b_i a_i + 2\lambda x \\ p_i(x) &= \frac{1}{1+\exp\left(-b_i a_i^{\mathrm{T}} x\right)}\end{aligned}$$

2.2 算法

近似点梯度法是常用的解决带有非光滑部分优化问题的方法，其基本思想为对于光滑部分进行显式梯度下降，对于非光滑部分利用近似点映射进行隐式梯度下降。针对问题（1）, 下给出近似点梯度法的伪代码。

Algorithm 1 近似点梯度法

Require: 最大迭代次数 n，精确度 ε

Ensure: $k = 1$, 随机初始化 x_0

while $k \le n$ and $\|x^{k-1} - x^{k-2}\|^2 / t^{k-1} > 10^{-\varepsilon}$ **do**
    调用 3 选取合适的 t^k
    $x^k \leftarrow \mathrm{Prox}_{t^k\mu\|x\|_1}(x^{k-1} - t^k \nabla f(x^{k-1}))$
    $k \leftarrow k+1$
end while

FISTA 算法为近似点梯度法的一个加速算法，其基本思想为对于光滑部分在进行梯度下降时使用 Nesterov 加速算法。使得在光滑部分 L 光滑的假设下有更好的理论收敛

速度。针对问题（1）, 下给出近似点梯度法的伪代码。

Algorithm 2 FISTA 算法

Require: 最大迭代次数 n，精确度 ε

Ensure: $k = 1$, 随机初始化 x_0

while $k \le n$ and $\|x^{k-1} - x^{k-2}\|^2/t^{k-1} > 10^{-\varepsilon}$ **do**

  $y^k \leftarrow x^{k-1} + \frac{k-2}{k-1}(x^{k-1} - x^{k-2})$

  调用 3 选取合适的 t^k

  $x^k \leftarrow \mathrm{Prox}_{t^k \mu \|x\|_1}(y^k - t^k \nabla f(y^k))$

  $k \leftarrow k + 1$

end while

对于迭代系数 t_k 的选取，由于 $\frac{1}{t^k} \le L$ 时收敛速度有理论保证，其中 L 为 ∇f 的 Lipschitz 常数。我们采用线搜索，搜索的条件为：

$$f(x^k) \le f(y^k) + < \nabla f(y^k), x^k - y^k > + \frac{1}{2t_k}\|x^k - y^k\|_2^2 \tag{2}$$

Algorithm 3 线搜索

Require: $t^k = t^{k-1}$，参照点 y^k 以及其梯度 $\nabla f(y^k)$, 步长收缩的系数 γ

$x^k \leftarrow \mathrm{prox}_{t^k \mu \|\cdot\|_1}(y^k - t^k \nabla f(y^k))$

while x^k 与 y^k 不满足 2 式 **do**

  $t^k \leftarrow \gamma t^k$

  $x^k \leftarrow \mathrm{prox}_{t^k \mu \|\cdot\|_1}(y^k - t^k \nabla f(y^k))$

end while

3 实验结果

3.1 参数设置

本实验的数据集数据个数为 $m = 32561$ 个，每个数据为 $p = 123$ 维向量，均带有标签。

两个算法的最多迭代次数 $n = 10000$, 精确度 $\varepsilon = 6$, 其含义为当 $\|x_k - x_{k-1}/t_k\| \le 1e^{-\varepsilon}$ 时停止迭代, 随机化种子为 Seed = 3, 线搜索步长收缩系数为 $\gamma = 0.1$,(1) 中的 $\lambda = 1/(2 * m)$,$\mu = 0.001$。

3.2 两算法的收敛速率

收敛条件 $\log(\|x_k - x_{k-1}\|^2/t_k)$ 与迭代次数关系图为：

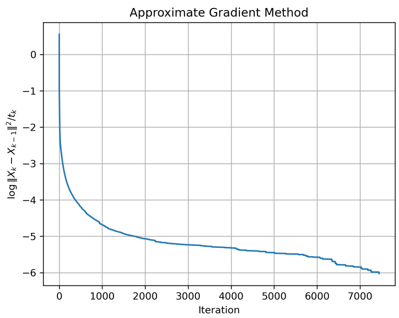


图 1: 近似点梯度法

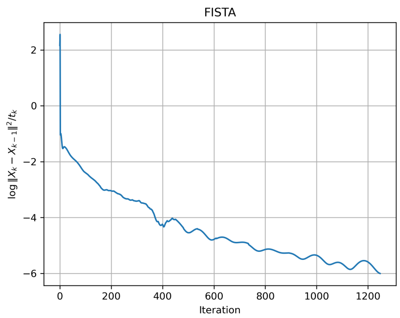


图 2: FISTA 算法

收敛条件 $\log(L(x^k) - L(x^*))$ 与迭代次数关系图为：

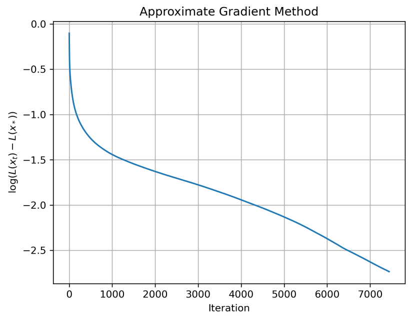


图 3: 近似点梯度法

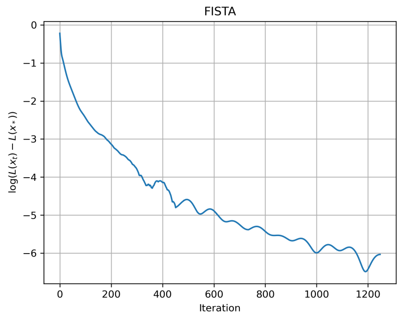
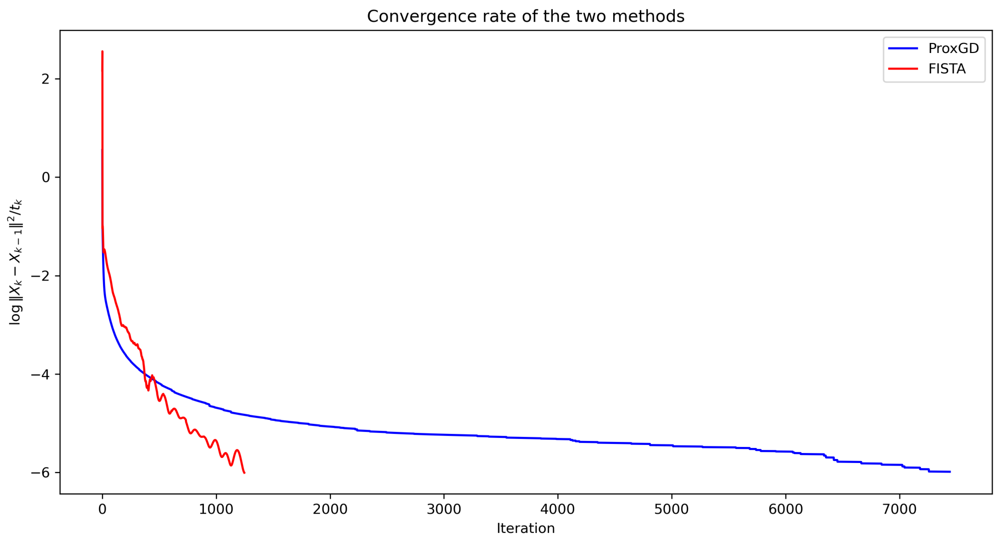
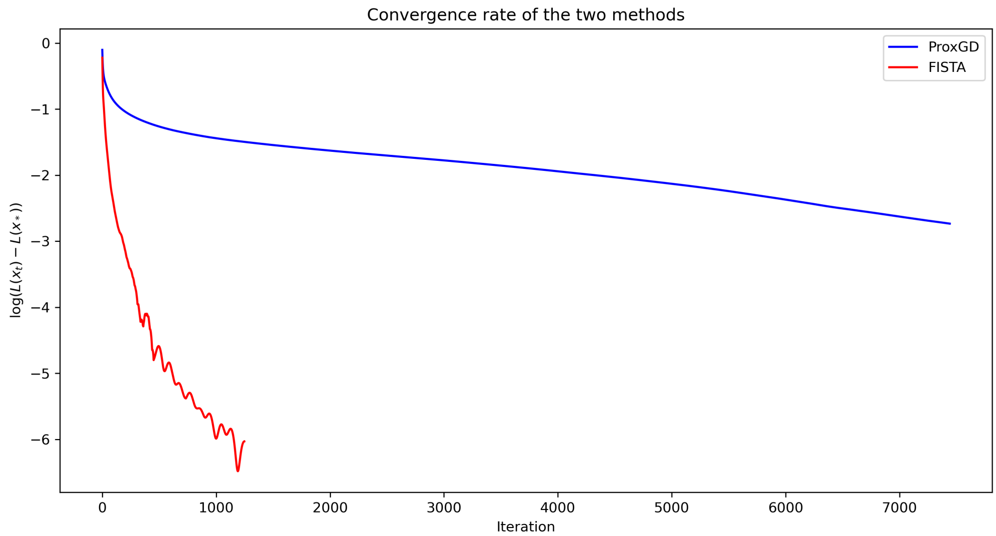


图 4: FISTA 算法

其中 x^* 用提前解出的一个高精度解来近似。





可以看到近似点梯度法用了 7434 次迭代停止，而 FISTA 算法只用了 1243 次迭代，且最后解的精度更高。

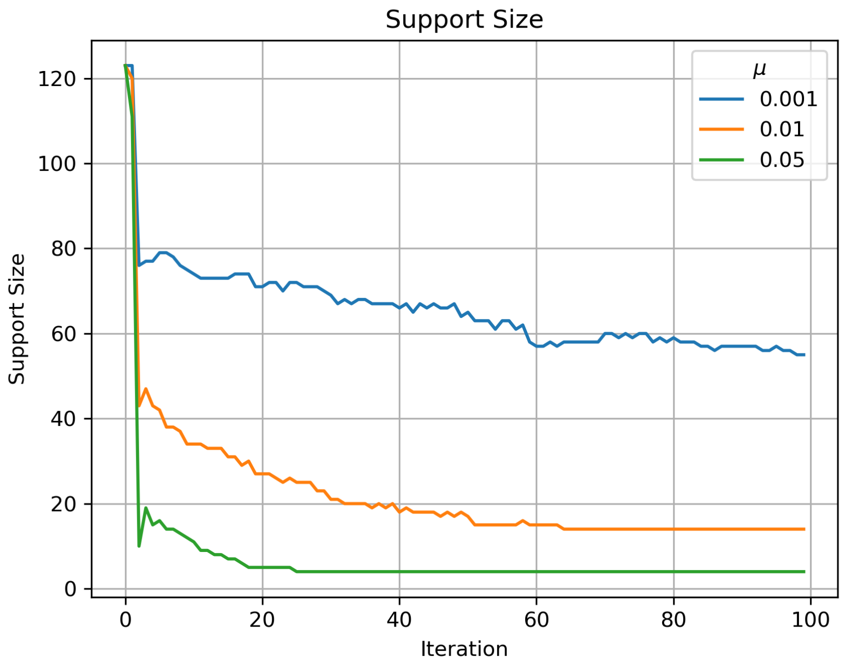
3.3 参数 μ 的选取对结果的影响

使用 FISTA 算法,其余参数设置不变,改变一范数正则项的系数 μ 为 $0, 1e^{-5}, 0.0001, 0.001, 0.01, 0.05$ 稀疏性计算公式为 sparsity $= \frac{\#\{i:x_i=0\}}{\dim(x)}$。得到如下实验结果：

μ	稀疏性	迭代次数
0.0	0.0	1635
1e-05	0.10569105691056911	1414
0.0001	0.37398373983739835	976
0.001	0.6829268292682927	492
0.01	0.8861788617886179	222
0.05	0.967479674796748	134

表 1: μ 的选取对解稀疏性的影响

可以看到随着 μ 的增大，对于非稀疏解的惩罚变大，因此最优解的稀疏度不断增大，达到最优解的迭代次数减少。



上图为 x_k 的支撑大小（即非零元素个数）与迭代次数的关系。随着迭代次数增加 x 的非零个数先快速下降，然后经过一段时间的缓慢下降，最后趋于稳定。这表明 1 范数的正则项实现了对于解的稀疏性的限制，越大的 μ 倾向于选择更稀疏的解。

4 总结

本文通过对稀疏逻辑回归问题分别使用近似点梯度法和 FISTA 算法进行了实验分析，得出了以下结论：

- FISTA 算法相比近似点梯度法在迭代次数和收敛速度方面表现更优，验证了其加速机制的有效性。
- 增大正则化参数 μ 可以显著提高解的稀疏性，同时减少迭代次数，这表明一范数正则项有效地控制了解的稀疏性。
- 实验结果与理论分析一致，FISTA 算法在处理大规模稀疏优化问题时具有明显优势。

4.1 未来工作

- 进一步扩展到更多类型的优化问题，例如非凸优化问题或其他形式的正则化问题。
- 探讨其他加速算法或自适应算法的性能，如 ADMM（交替方向乘子法）或基于机器学习的自适应学习率调整方法。

- 考虑在实际数据集上的应用，特别是在高维稀疏数据的背景下，进一步验证算法的性能和稳定性。

总之，本文的研究展示了近似点梯度法和 FISTA 算法在处理非光滑正则项优化问题中的有效性，特别是 FISTA 算法在收敛速度和解的稀疏性方面的优越性。