Advances in Applied Mathematics 应用数学进展, 2022, 11(10), 7173-7184
Published Online October 2022 in Hans. http://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2022.1110761

Hans汉斯

MCP正则优化问题的高效二阶算法

张原浩

上海理工大学理学院，上海

收稿日期：2022年9月18日；录用日期：2022年10月8日；发布日期：2022年10月18日

摘　要

Minimax Concave Penalty (MCP)正则优化问题在诸多科学领域有着广泛的应用，例如：机器学习、信号、图像恢复以及逻辑回归等问题。本文基于MCP正则项研究了一种二阶加速优化算法，该算法的主要思想是在于如何得到一个使得目标快速下降的方向，主要的方法是设计对偶半光滑牛顿法求解子问题。我们基于所提出的模型提出新的算法。通过数值试验部分的对比验证了我们所提出的算法的有效性和高效性。

关键词

非凸优化，临近算子，半光滑牛顿法

Efficient Second-Order Algorithm for MCP Regular Optimization Problems

Yuanhao Zhang

College of Science, University of Shanghai for Science and Technology, Shanghai

Received: Sep. 18th, 2022; accepted: Oct. 8th, 2022; published: Oct. 18th, 2022

Abstract

The Minimax Concave Penalty (MCP) regularization optimization problem is widely used in many scientific fields, such as machine learning, signal restoration, image restoration and logistic regression. This paper studies a second-order accelerated optimization algorithm based on MCP regularization term. The main idea of the algorithm is how to get a direction that makes the target drop quickly. The main method is to design dual semi-smooth Newton method to solve the subproblem. We propose a new algorithm based on the proposed model. The effectiveness and efficiency of the proposed algorithm are verified by numerical experiments.

文章引用: 张原浩. MCP 正则优化问题的高效二阶算法[J]. 应用数学进展, 2022, 11(10): 7173-7184.
DOI: 10.12677/aam.2022.1110761

Keywords

Non-Convex Optimization, Proximal Operator, Semi-Smooth Newton Method

1. 引言

本文考虑一类如下的优化问题：

$$\min_{x\in R^n} F(x) := f(x) + \eta \hat{g}(x) \tag{1-1}$$

其中 f 是光滑的并且梯度是 Lipschitz 连续的，$\hat{g}$ 是正常的下半连续函数(可能非凸非光滑)，$\eta > 0$ 是给定参数。对于加式类型的复合优化问题，近些年涌现出了许多优秀的优化方法如：临近梯度方法(PGA) [1]、交替方向乘子法(ADMM) [2]、DC 临近牛顿法[3]、半光滑牛顿法[4]等。并且该类型的优化问题在实际生活中有着广泛的应用，例如：非线性最小二乘[5]、鲁棒相位恢复[6]、低秩矩阵补全[7]、信号恢复[8]、逻辑回归[9]以及诸多机器学习问题。而随着科学技术的进一步发展，以及大数据时代的到来，优化算法在实际生活中的应用也是越来越广泛，从简单的凸的无约束优化算法到现在迅速发展的非凸带约束的优化算法，表明了人类永不止步的探索精神。从简单到难也意味着我们对于数据的处理以及要求越来越高，对应用于实际问题的算法要求越来越严格。随着时间的推移对于凸优化问题的研究已经发展的相当成熟，而对于非凸优化问题的研究还是寥寥无几，所以近几年来有大批学者着手于发展研究带有非凸非光滑正则项的优化问题，而大多数的非凸优化都是以求解稀疏解为目的的。就稀疏优化问题而言，最初的探究想法是将函数 $\hat{g}$ 设置为 L_0 范数，但是由于带 L_0 范数的优化模型是一个 NP 难的问题，而目前大部分的解决方法是利用 L_1 范数对其进行一个凸的替代，这样非凸的问题就变成了一个凸的优化问题，我们就可以使用现有的各种高效算法来求解，但事实上在实际生活中的许多工业或者自然科学等研究问题中目标大多是一个非凸的函数，这样我们原本的凸优化方法就显得不那么高效，由此非凸优化算法也就应运而生了。非凸优化问题在上述应用中也都有所发展。本文研究带非凸正则项的优化问题，目前研究发现一些性质比较好的非凸惩罚有 Minimax Concave Penalty (MCP) [2] [8] [10] [11] [12] [13] [14]、Smoothly Clipped Absolute Deviation (SCAD) [12] [15]等。最近的一些研究中发现了基于上述非凸正则项的一些变种同样有着良好的发展潜力有待我们研究，例如：GMCP [16] [17]，VMCP [13]等等。尽管对于现阶段的各类非凸惩罚已经有了大量的研究，但是大多数都是一阶方法，例如：邻近梯度法(PGA) [8] [18]、交替方向乘子法(ADMM) [2] [15]、坐标下降法(CD) [19]等。对于带非凸正则项的优化问题，二阶算法的研究就显得相对较少。

随着大数据时代的发展，显然凸优化方法一定程度上已经无法满足自然科学的更高要求，经过一段时间的研究和发现，非凸优化问题的求解虽然困难，但是通过变换也可以有效地解决，重要的是非凸优化问题能够弥补凸优化问题中的一些不足，非凸优化问题相比于凸优化问题而言优秀的算法虽然较少，但是目前仍有大批国内外学者在竭尽全力研究。其中较为突出的是非凸稀疏正则优化问题，其中性质良好的 MCP 正则项 2010 年由 Zhang [14]在理论和算法上作出了巨大的贡献，随后 2015 年由 Li 和 Lin [18]提出了解决非凸非光滑优化问题的加速近端梯度算法(APG)，其中单调 APG 具体框架如下：

$$
\begin{aligned}
y_k &= x_k + \frac{t_{k-1}}{t_k}\left(z_k - x_k\right) + \frac{t_{k-1}-1}{t_k}\left(x_k - x_{k-1}\right)\\
z_{k+1} &= prox_{\alpha_Y g}\left(y_k - \alpha_Y \nabla f\left(y_k\right)\right)\\
v_{k+1} &= prox_{\alpha_X g}\left(x_k - \alpha_X \nabla f\left(x_k\right)\right)\\
t_{k+1} &= \frac{\sqrt{4\left(t_k\right)^2 + 1} + 1}{2}\\
x_{k+1} &= \begin{cases} z_{k+1}, & \text{if } F\left(z_{k+1}\right) \le F\left(v_{k+1}\right)\\ v_{k+1}, & \text{otherwise} \end{cases}
\end{aligned}
$$

之后，2019 年 Shi 和 Huang [12]提出了一种二阶的，针对高维非凸稀疏优化问题的半光滑牛顿算法，并证明了他们所提出的算法局部超线性收敛于 KKT 点，同时数值试验也验证了该方法计算效率更高。但是，尽管该方法是二阶方法收敛速度快，而文中仅仅只考虑了残差下降并没有考虑目标的下降，其次该文章也只证明了局部的超线性收敛速度，并没有考虑全局收敛性。

在 2021 年 Wu 和 Li [8]等人推广了近端梯度算法，把传统近端项中的欧式范数改为了更一般的 Bregman 范数，进而设计出了新的一阶算法，所提出的新算法一般框架如下：

$$
\begin{aligned}
y^k &:= x^k + \alpha^k\left(x^k - x^{k-1}\right)\\
z^k &:= x^k + \beta^k\left(x^k - x^{k-1}\right)\\
x^{k+1} &\in \mathcal{P}^h_{\lambda_k g}\left(x^k; \nabla f\left(z^k\right) - \frac{1}{\lambda_k}\left(y^k - x^k\right)\right)
\end{aligned}
$$

然后在 KL 性质的假设条件下证明了全局收敛性和收敛速度，同时以 SCAD 和 MCP 两种非凸正则项为例通过数值实验，验证了该算法的有效性和高效性。但是由于他们所提出的算法仍属于一阶算法，收敛速度依然有待提高。

2. 相关基础知识

基础知识

为了方便后续文章模型的建立以及算法的构建，首先我们明确一些假设、符号、定义以及方法(对于基本的符号、定义和方法可以参考[11] [20] [21])。

定义 2.1. (梯度)给定函数 $f: R^n \to R$，且 f 在点 x 的一个邻域内有意义，若存在向量 $\dot{g} \in R^n$ 满足：

$$
\lim_{\delta \to 0} \frac{f\left(x+\delta\right) - f\left(x\right) - \dot{g}^{\mathrm{T}}\delta}{\|\delta\|} = 0
$$

就称 f 在点 x 处可微。此时 $\dot{g}$ 称为 f 在点 x 处的梯度。并记任意一阶可微函数 $f\left(x\right)$ 梯度为：$\nabla f\left(x\right)$。

假设 2.2. 假设目标函数 $F\left(x\right)$ 有下界，即 $F\left(x\right) \ge \bar{F}$，函数 $f: R^n \to R$ 是二次连续可微的，其梯度是 Lipschitz 连续的。对 $\forall k$ 满足 $\left[\lambda_{\min}\left(B_k\right) - \eta\delta\right] \succeq 0$。$\hat{g}$ 为 MCP 正则项。

其中函数 f 的梯度为 Lipschitz 常数为：$L_{\nabla f}$，$B = \nabla^2 f$，B_k 代表第 k 次迭代对 $\nabla^2 f$ 的近似。$\lambda_{\min}\left(A\right)$ 是矩阵 A 的特征值的最小值。η 是罚参数，δ 是与正则项相关的参数。

对于任意实值扩张函数 $f: R^n \to \left(-\infty, +\infty\right]$，它的定义域如下：

$$
\operatorname{dom} f := \left\{x \in R^n : f\left(x\right) < +\infty\right\} \tag{2-1}
$$

接着我们给出临近映射和 Moreau-envelope 的定义分别如下：

$$prox_{\mu f}(x) := \arg\min_u \left\{ f(u) + \frac{1}{2\mu}\|u-x\|^2 \right\} \tag{2-2}$$

$$M_f^{\mu}(x) := \inf_u \left\{ f(u) + \frac{1}{2\mu}\|u-x\|^2 \right\} \tag{2-3}$$

欧式空间上的内积和范数定义如下：

$$\langle x, y\rangle = x^{\mathrm{T}} y$$

$$\|x\|_2 = \sqrt{\langle x, x\rangle}$$

对于正则项 $\hat{g}(x)$，我们根据文献[14]的研究，采用性质较好和研究较多的非凸非光滑正则项 MCP，其具体定义如下：

$$\hat{g}(x) = \mathcal{P}_{MC}(x;\lambda,\gamma) = \sum_{i=1}^{n}\left\{P_{MC}(x_i;\lambda,\gamma)\right\} \tag{2-4}$$

其中 $x \in R^n$，$P_{MC}(x_i;\lambda,\gamma)$ 定义如下：

$$P_{MC}(x_i;\lambda,\gamma) = \lambda\int_0^{|x_i|}\left(1 - z/(\gamma\lambda)\right)_+ \mathrm{d}z = \begin{cases} \lambda|x_i| - \dfrac{1}{2\gamma}x_i^2, & \text{if } |x_i| \le \lambda\gamma \\ \dfrac{1}{2}\gamma\lambda^2, & \text{if } |x_i| > \lambda\gamma \end{cases} \tag{2-5}$$

根据定义我们容易得到函数 $\alpha\hat{g}(x)$ (其中 $\alpha \in (0,\gamma)$)的临近映射为：$\mathrm{firm}(x;\alpha\lambda,\lambda\gamma)$ [13]，具体表达式如下：

$$\begin{aligned} prox_{\alpha\hat{g}}(x) &= \arg\min_u \left\{ \hat{g}(u) + \frac{1}{2\alpha}\|x-u\|^2 \right\} \\ &= \mathrm{firm}(x;\alpha\lambda,\lambda\gamma) \\ &= \begin{cases} 0, & \text{if } |x| \le \alpha\lambda \\ \mathrm{sign}(x)\dfrac{\gamma(|x|-\alpha\lambda)}{\gamma-\alpha}, & \text{if } \alpha\lambda < |x| \le \lambda\gamma \\ x, & \text{if } |x| > \lambda\gamma \end{cases} \end{aligned}$$

进而根据文献[4]知道函数 $prox_{\alpha\hat{g}}(x)$ 是广义可微的，并且容易得到 $prox_{\alpha\hat{g}}(x)$ 的广义雅可比是一个对角阵，其对角元素为：

$$J_{mcp}^{ii}(x) = \begin{cases} 0, & \text{if } |x_i| < \alpha\lambda \\ \left[0, \dfrac{\gamma}{\gamma-\alpha}\right], & \text{if } |x_i| = \alpha\lambda \\ \dfrac{\gamma}{\gamma-\alpha}, & \text{if } \alpha\lambda < |x_i| < \lambda\gamma \\ \left[1, \dfrac{\gamma}{\gamma-\alpha}\right], & \text{if } |x_i| = \lambda\gamma \\ 1, & \text{if } |x_i| > \lambda\gamma \end{cases} \tag{2-6}$$

3. 模型、算法及实验结果

3.1. 模型提出

在有了基本的符号和知识后，本节我们将给出文章具体考虑的模型以及所提出的算法，最终通过数

值实验来验证我们算法的有效性以及高效性。本着重于求解一个使得目标快速下降的方向，然后对于如何求解该方向而设计快速算法。首先我们给出子问题在满足假设 2.2 的情况下，若当前迭代点为 x_k，我们假设下一步迭代点为 x_k+d，其中 d 是迭代搜索方向，将函数 f 在 x_k+d 这点进行二阶近似展开得到：

$$f\left(x_k+d\right)\approx f\left(x_k\right)+\nabla f\left(x_k\right)^{\mathrm{T}}d+\frac{1}{2}d^{\mathrm{T}}B_k d \tag{3-1}$$

我们令：$\varphi(d):=\nabla f\left(x_k\right)^{\mathrm{T}}d+0.5d^{\mathrm{T}}B_k d+\eta\hat{g}\left(x_k+d\right)$，进一步得到子问题形式如下：

$$\min_d \varphi(d):=\nabla f\left(x_k\right)^{\mathrm{T}}d+\frac{1}{2}d^{\mathrm{T}}B_k d+\eta\hat{g}\left(x_k+d\right) \tag{3-2}$$

问题(3-2)就是本文主要考虑的模型。根据文献[13]我们知道，当 $\delta\geq\frac{1}{\gamma}$ 时，$\hat{g}(x)+\frac{\delta}{2}\|y-x\|_2^2$ 是一个凸函数，基于这点我们将问题(3-2)等价变形为：

$$\begin{aligned}&\min_{d,y}\ \nabla f\left(x_k\right)^{\mathrm{T}}d+\frac{1}{2}d^{\mathrm{T}}\left(B_k-\eta\delta I\right)d+\eta\hat{g}\left(y\right)+\frac{\eta\delta}{2}\left\|y-x_k\right\|_2^2\\&\text{s.t.}\quad x_k+d=y\end{aligned} \tag{3-3}$$

进而我们容易得到问题(3-3)的对偶问题为：

$$\max_{\mu} c_\delta(\mu) \tag{3-4}$$

其中 μ 为对偶问题的自变量，$c_\delta(\mu)$ 为：

$$\begin{aligned}c_\delta(\mu)=&-\frac{1}{2}\left[\mu-\nabla f\left(x_k\right)\right]^{\mathrm{T}}\left(B_k-\eta\delta I\right)^{-1}\left[\mu-\nabla f\left(x_k\right)\right]-\frac{\|\mu\|_2^2}{2\eta\delta}\\&+\eta\hat{g}\left(prox_{\delta^{-1}\hat{g}}\left(x_k-\frac{\mu}{\eta\delta}\right)\right)+\frac{\eta\delta}{2}\left\|x_k-\frac{\mu}{\eta\delta}-prox_{\delta^{-1}\hat{g}}\left(x_k-\frac{\mu}{\eta\delta}\right)\right\|_2^2\end{aligned} \tag{3-5}$$

为了算法的构建我们引入一个记号：$\Delta_k(\alpha)(\alpha\in(0,1))$，具体定义如下：

$$\begin{aligned}\Delta_k(\alpha)&=\varphi(0)-\varphi(\alpha d_k)\\&=\eta\hat{g}\left(x_k\right)-\eta\hat{g}\left(x_k+\alpha d_k\right)-\alpha\nabla f\left(x_k\right)^{\mathrm{T}}d_k-\frac{\alpha^2}{2}d_k^{\mathrm{T}}B_k d_k\end{aligned} \tag{3-6}$$

3.2. 算法构建

算法 1. ALG1-求解问题(1-1)

初始化：$x_0,\eta,\varepsilon,\tau_1\in(0,1),\sigma_1\in(0,0.5),k_{out},K_{\max}$

While $k_{out}\leq K_{\max}$ do：

通过算法 2 得到下降方向：$d_k=\left(B_k-\eta\delta I\right)^{-1}\left(\mu_n-\nabla f\left(x_k\right)\right)$

If $\|\Delta_k(1)\|\leq\varepsilon$ then：

break；

end

令：$\alpha=1$，并计算：$\Delta_k(\alpha)$

While $F\left(x_k\right)-F\left(x_k+\alpha d_k\right)<\sigma_1\Delta_k(\alpha)$ do：

令：$\alpha=\tau_1\alpha$，计算：$\Delta_k(\alpha)$

Continued

end

令： $\alpha_k = \alpha$， $x_{k+1} = x_k + \alpha_k d_k$，用 L-BFGS 公式更新 $\left(B_k - \eta\delta I\right)^{-1}$

end

算法 2. 求解问题(3-4)

初始化： $\mu_0, \eta, \varepsilon, \tau_2 \in (0,1), \sigma_2 \in (0,0.5), n_{inn}, N_{\max}$

While $n_{inn} \le N_{\max}$ do：

计算梯度： $z_n = \nabla c_\delta(\mu_n)$

If $\|z_n\| \le \varepsilon$ then：

Break；

end

求解方程： $J_n d_{ssn}^n + \nabla c_\delta(\mu_n) = 0$ 得到搜索方向($J_n \in \partial\left(\nabla c_\delta(\mu_n)\right)$)

令： $\alpha = 1$

While $c_\delta(\mu_n) - c_\delta\left(\mu_n + \alpha d_{ssn}^n\right) \ge -\sigma_2 z_n^{\mathrm{T}} d_{ssn}^n$ do：

令： $\alpha = \tau_2 \alpha$

end

令： $\alpha_n = \alpha$， $\mu_{n+1} = \mu_n + \alpha_n d_{ssn}^n$

end

return μ_n

3.3. 数值实验及结果分析

本节我们将用随机合成的数据和实际数据来评估我们设计的算法 1，为了进行比较同时我们也编码了临近梯度法(PGA)具体内容由算法 3 提供。从算法框架可以看出我们的算法和 PGA 方法的主要区别是搜索方向 d_k 的计算。文章所有的代码都是用 MATLAB2020a 编写的，并在一台 8 核、芯片为 Apple M1、内存为8 GB 的微处理器上运行。对不同的算法将统一使用以下参数(随机数据的参数随后也会详细讨论。)：

$$\varepsilon = 10^{-6}, \sigma_1 = \sigma_2 = 10^{-4}, \tau_1 = \tau_2 = 0.2, \eta = 0.25, \gamma = 15, \lambda = 0.03 * \sqrt{2 * \log(n)} \tag{3-7}$$

不同的求解器停止条件我们统一使用：

$$\text{NormG} := \frac{\|d_k\|_2}{\|d_0\|_2 + 1} \le \varepsilon \tag{3-8}$$

算法 3. PGA [20] for MCP 正则优化问题

初始化： $x_0, \eta, \varepsilon, \tau_3 \in (0,1), \sigma_3 \in (0,0.5), j, J$

While $j < J$ do：

得到下降方向： $d_j = \frac{1}{\alpha_j}\left(x_j - prox_{\alpha_j \hat{g}}\left(x_j - \alpha_j \nabla f\left(x_j\right)\right)\right)$

If $\|\text{NormG}\| \le \varepsilon$ then

break；

Continued

end

令 $\alpha=1$，并计算：$\Delta_j(\alpha)$

While $F(x_j)-F(x_j+\alpha d_j)<\sigma_3\Delta_j(\alpha)$ do：

$\alpha=\tau_3\alpha$，计算：$\Delta_j(\alpha)$

end

令：$\alpha_j=\alpha$，$x_{j+1}=x_j+\alpha_j d_j$

end

首先我们考虑随机合成的数据。实验为稀疏信号恢复问题[8]，具体形式如下：

$$\min_{x\in R^n}\frac{1}{2}\|Ax-b\|_2^2+\eta\hat{g}(x) \tag{3-9}$$

其中 $\hat{g}(x)$ 我们选择(2-4)定义的非凸正则项，矩阵 $A\in R^{m\times n}$ 是测量矩阵，向量 $b\in R^m$ 是观测信号，η 是罚参数。我们使用 MATLAB 内置函数 randn，sprandn，normrnd，normalize 生成随机数据：

$$A=\text{normalize}(\text{randn}(m,n),\text{'range'})$$

$$x^*=\text{sprandn}(n,1,0.15)$$

$$eps=\text{normrnd}(0,0.01,[m,1])$$

$$b=Ax^*+eps$$

其中 $\text{normalize}(\text{randn}(m,n),\text{'range'})$ 是生成一个 $m\times n$ 的正态分布随机矩阵，并且按列归一化至[0, 1]。$\text{sprandn}(n,1,0.15)$，是生成一个随机稀疏向量，非零元素约有 $n\times1\times0.15$ 个。$\text{normrnd}(0,0.01,[m,1])$ 是生成一个均值为 0 标准差为 0.01 的正态分布随机向量。我们将对 m,n,σ_1,τ_1 四个参数用控制变量法进行实验以验证算法的有效性。评估指标为 CPU time (CPU 运行时间)、Iterations (迭代次数)。图例中的 eta 代表参数：η，同样的 gamma 代表参数：γ。

首先我们设置 $m=2400$，$\sigma_1=10^{-4}$，$\tau_1=0.2$，n 从 1000 变化至 1100 总共 11 组数据。数值结果从图 1 中可以看出随着 n 增大时，相较于 PGA 方法，我们的算法 1 迭代次数有着明显的优势，在 CPU 运行时间方面也都远小于 PGA 方法。

其次我们设置 $n=1000$，$\sigma_1=10^{-4}$，$\tau_1=0.2$，m 从 2410 变化至 2470 总共 7 组数据。数值结果如图 1 所示，从图中可以看出我们的算法相较于 PGA 方法，迭代次数和 CPU 运行时间是远小于 PGA 方法的，这证明我们所提出的方法有效并且是高效的。

再者，我们将数据的维数固定在 $m=2400$，$n=1000$，γ 从 10 变化至 20 一共 6 组数据，结果如图 1 所示，很容易看出随着 γ 的变化我们的算法是很稳健的，无论是从迭代次数还是 CPU 运行时间来看，都是远远比 PGA 更加的优越。最后我们将以同样的维数 $m=2400$，$n=1000$，但是 $\sigma_1=10^{-4}$，η 从 0.2 变化至 0.26 一共 7 组数据对算法进行测试。最终的结果由图 1 给出。

接下来由于 PGA 方法运行时间较长我们将对自己的算法进行评估，评估指标使用目标函数值(Value of (f + g))和恢复精度(nx_xstar)。以不同维数和不同的 η 值为例子说明我们算法的有效性和高效性。对于维度测试，我们将 m 固定在 5000 维，n 从 1000 变化至 3000，变化幅度为 500。$\eta=0.25$，$\gamma=15$ 其余参数不做改变。数值结果如图 2 所示，可以看到不同维度下我们的算法还是很稳健的。接着我们将维数保

持在 $m=3000$， $n=1500$， η 从 0.22 变化至 0.3，变化幅度为 0.02， $\gamma=15$ 其余参数不做改变。数值结果如图 2 所示，我们大致可以看到随 η 值变化的时候，当 η 保持在 0.24 到 0.26 这个区间的时候，算法效果是最优的，从而也证实了我们之前的实验设置 $\eta=0.25$ 是合适的选择。

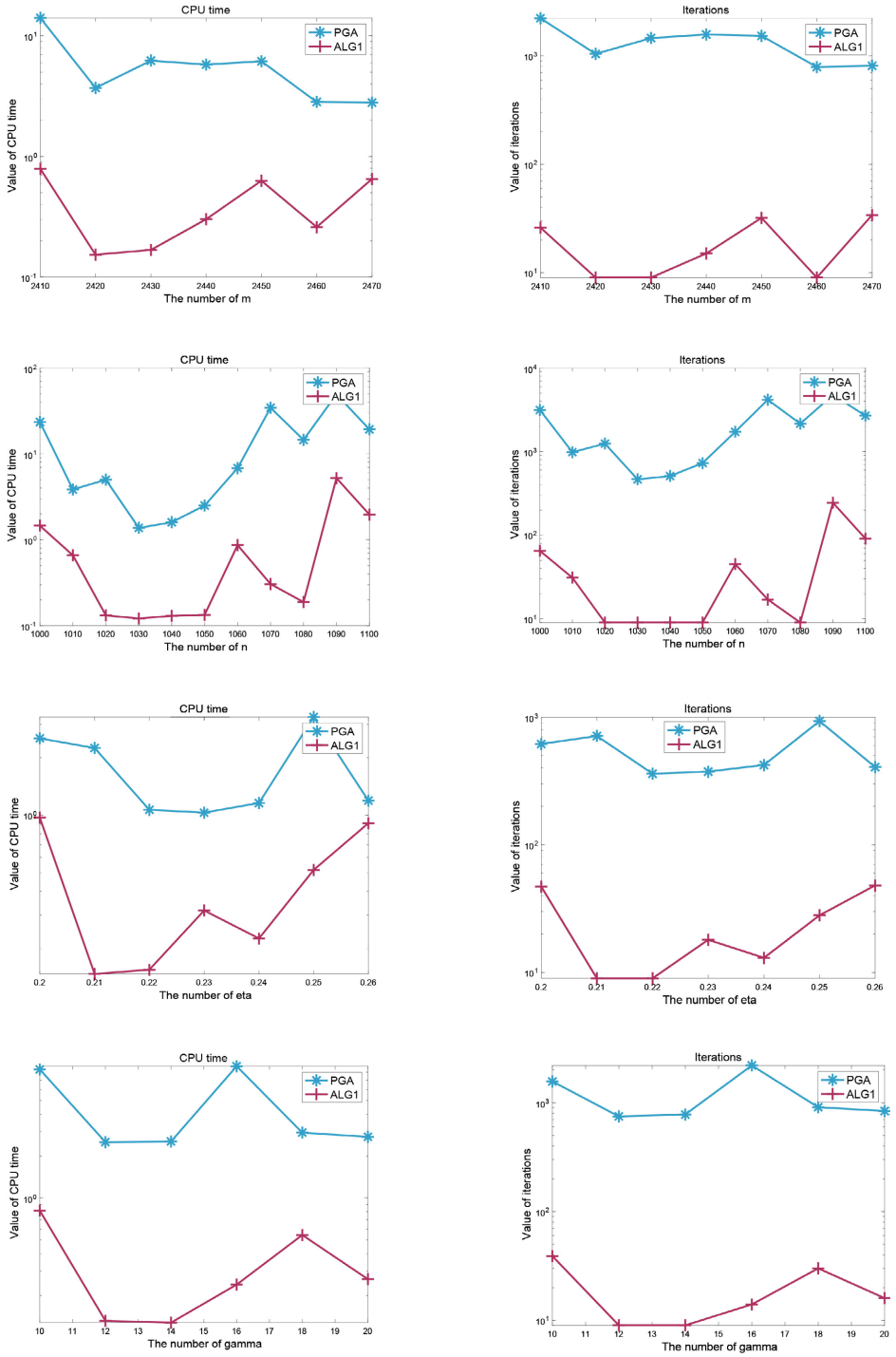

Figure 1. Comparison of CPU running time and iteration times between two algorithms with four parameters

图 1. 4 种参数下两算法间 CPU 运行时间和迭代次数的对比

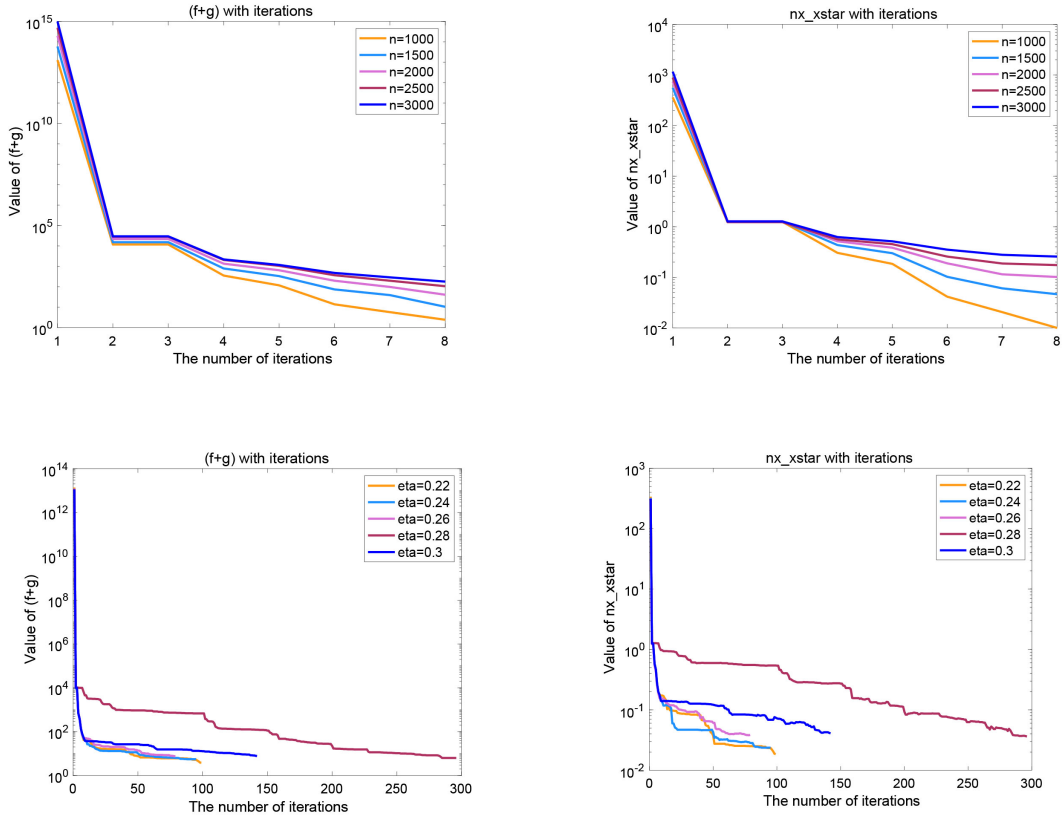

Figure 2. Performance of Algorithm 1 under different parameters
图 2. 算法 1 在不同参数下的表现

接下来我们考虑逻辑回归问题，给定数据集 $A\in R^{m\times n}$ 和标签向量 $b\in R^m$，逻辑回归问题表示如下：

$$\min \Phi_\eta(x) := \frac{1}{m}\sum_{i=1}^{m}\log\left(1+\exp\left(-b_i x^{\mathrm{T}} a_i\right)\right)+\eta\hat{g}(x) \tag{3-10}$$

其中 $A=(a_1,\cdots,a_m)$，$b_i\in\{-1,1\}$，$i=1,\cdots,m$。

这部分我们将从 LIBSVM [22]数据集中选取部分实例进行测试。每个实例给出了一个矩阵 A 和向量 b。具体的测试结果由表 1 和图 3 给出。我们将 CPU 运行时间(CPU(s))、算法迭代次数(Iter)和收敛精度(Res)作为本部分的评价指标，而 Optval 代表模型的最优值。其中收敛精度 Res 表示如下：

$$\mathrm{Res}=\frac{\|d_k\|_2}{\|d_0\|_2+1}\le\varepsilon$$

从表 1 可以清楚的看到，两种算法都可以有效地求解逻辑回归问题。而从图 3 中的两图我们可以清楚地看到，我们的算法 1 在 20 个不同的测试集上都有着优秀的表现，相比于传统的 PGA 方法而言迭代次数大大减少，同时也能收敛到更高的精度。虽然在表 1 中我们发现了测试集 5 我们的算法运行时间比 PGA 方法慢了一些，但是综合对比迭代次数和收敛的精度我我们的算法是远优于 PGA 方法的，所以对于时间上微小的差别基本可以忽略。从而这也验证了我们的算法在逻辑回归问题上有着卓越的表现，通过与 PGA 方法的对比也凸显出了我们的方法是更为有效和高效的。

综合上述实验以及结果的分析，验证了我们所设计的二阶算法在逻辑回归和稀疏信号恢复方面都是有效的，并且相较于传统的 PGA 方法在各个方面都更加高效，鲁棒性更好。

Table 1. Performance of two algorithms in 20 sets of logistic regression
表 1. 两算法在 20 组逻辑回归中的表现

测试集编号	行数	列数	算法 1				算法 3			
N	m	n	CPU(s)	Iter	Res	Optval	CPU(s)	Iter	Res	Optval
a1a	1605	119	3.16326e−01	9	3.73489e−09	0.693	3.69953e−01	75	8.83278e−07	0.693
a2a	2265	119	1.32050e−01	12	2.39626e−08	0.693	1.94620e−01	75	8.93744e−07	0.693
a3a	3185	122	1.00446e−01	10	1.46148e−08	0.693	2.55895e−01	75	8.93102e−07	0.693
a4a	4781	122	1.48468e−01	11	9.64087e−09	0.693	3.41970e−01	75	9.15061e−07	0.693
a5a	6414	122	8.31462e−01	13	1.24028e−08	0.693	4.17240e−01	75	9.02011e−07	0.693
a6a	11,220	122	1.26901e−01	8	2.18711e−09	0.693	6.77969e−01	75	8.85075e−07	0.693
a7a	16,100	122	1.79933e−01	10	1.16777e−08	0.693	9.49610e−01	75	8.65226e−07	0.693
a8a	22,696	123	2.44431e−01	10	1.89512e−08	0.693	1.32385e+00	75	8.64977e−07	0.693
a9a	32,561	123	2.69392e−01	8	2.85696e−09	0.693	1.88188e+00	75	9.16768e−07	0.693
ijcnn1	49,990	22	2.54796e−01	8	3.05836e−08	0.693	9.38021e−01	72	9.41579e−07	0.693
mushrooms	8124	112	1.32796e−01	8	9.91244e−09	0.693	5.52149e−01	75	9.27103e−07	0.693
phishing	11,055	68	2.15109e−01	12	1.46065e−08	0.693	5.97741e−01	74	9.39543e−07	0.693
w1a	2477	300	1.49633e−01	8	1.09425e−08	0.693	3.19617e−01	76	8.39927e−07	0.693
w2a	3470	300	1.43242e−01	8	7.10077e−09	0.693	4.09359e−01	76	8.54131e−07	0.693
w3a	4912	300	1.64765e−01	8	7.31374e−09	0.693	5.29249e−01	76	8.59312e−07	0.693
w4a	7366	300	1.70304e−01	8	5.10713e−09	0.693	7.32478e−01	76	8.42150e−07	0.693
w5a	9888	300	2.22083e−01	9	8.35186e−09	0.693	9.18242e−01	76	8.59644e−07	0.693
w6a	17,188	300	2.59537e−01	8	6.45706e−09	0.693	1.54480e+00	76	8.52992e−07	0.693
w7a	24,692	300	3.10005e−01	8	9.10777e−09	0.693	2.19111e+00	76	8.49034e−07	0.693
w8a	49,749	300	5.80007e−01	9	4.75253e−09	0.693	4.37231e+00	76	8.51638e−07	0.693

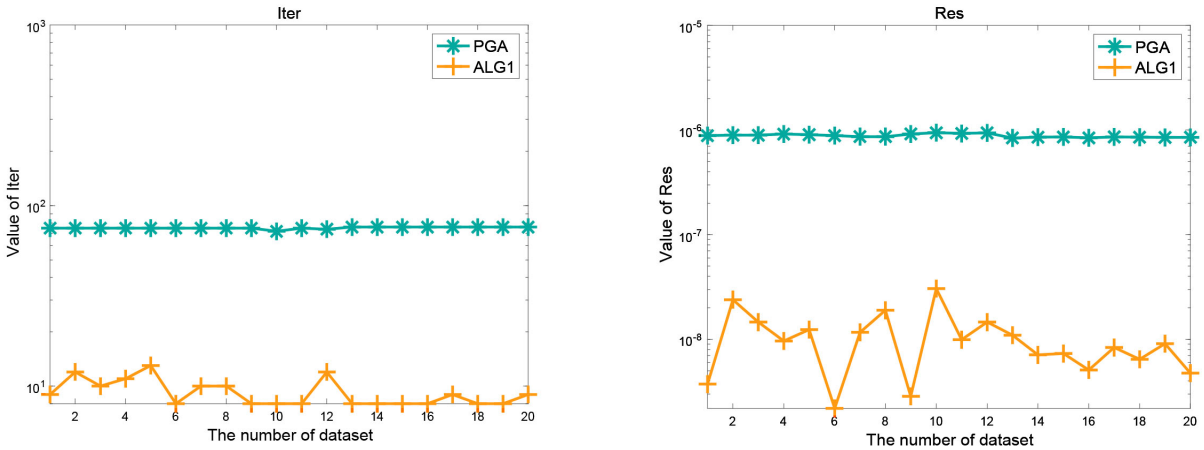

Figure 3. Performance of the two algorithms in the number of iterations and convergence accuracy in 20 datasets
图 3. 20 个数据集中两算法在迭代次数和收敛精度中的表现

4. 总结

本文基于带非凸正则项 MCP 的优化问题进行了二阶算法的研究，通过对目标可微项的二阶近似展开，得到了寻求下降方向的子问题，并利用对偶半光滑牛顿法对其进行求解。最终设计了一个新的二阶算法，通过大量的数值实验我们验证了在信号恢复和逻辑回归问题中，我们所提出的算法是有效并且高效的，而且鲁棒性更好。在随后的研究中我们希望将方法应用于流形优化，并且尽可能多地进行不同种类的数值实验来扩大我们算法的应用范围。

参考文献

[1] Chen, S.X., Ma, S.Q., Man-Cho Soand, A. and Zhang, T. (2020) Proximal Gradient Method for Nonsmooth Optimization over the Stiefel Manifold. *SIAM Journal on Optimization*, **30**, 210-239. https://doi.org/10.1137/18M122457X

[2] Jin, Z.F., Wan, Z.P., Jiao, Y.L. and Lu, X.L. (2016) An Alternating Direction Method with Continuation for Nonconvex Low Rank Minimization. *Journal of Scientific Computing*, **66**, 849-869. https://doi.org/10.1007/s10915-015-0045-0

[3] Rakotomamonjy, A., Flamary, R. and Gasso, G. (2015) Dc Proximal Newton for Nonconvex Optimization Problems. *IEEE Transactions on Neural Networks and Learning Systems*, **27**, 636-647. https://doi.org/10.1109/TNNLS.2015.2418224

[4] Li, X.D., Sun, D.F. and Toh, K.C. (2018) A highly Efficient Semismooth Newton Augmented Lagrangian Method for Solving Lasso Problems. *SIAM Journal on Optimization*, **28**, 433-458. https://doi.org/10.1137/16M1097572

[5] Marquardt, D.W. (1963) An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *Journal of the Society for Industrial and Applied Mathematics*, **11**, 431-441. https://doi.org/10.1137/0111030

[6] Duchi, J.C. and Ruan, F. (2019) Solving (Most) of a Set of Quadratic Equalities: Composite Optimization for Robust Phase Retrieval. *Information and Inference*: *A Journal of the IMA*, **8**, 471-529. https://doi.org/10.1093/imaiai/iay015

[7] Charisopoulos, V., Chen, Y.D., Davis, D., Díaz, M., Ding, L.J. and Drusvyatskiy, D. (2021) Low-Rank Matrix Recovery with Composite Optimization: Good Conditioning and Rapid Convergence. *Foundations of Computational Mathematics*, **21**, 1505-1593. https://doi.org/10.1007/s10208-020-09490-9

[8] Wu, Z.M., Li, C.S., Li, M. and Lim, A. (2021) Inertial Proximal Gradient Methods with Bregman Regularization for a Class of Nonconvex Optimization Problems. *Journal of Global Optimization*, **79**, 617-644. https://doi.org/10.1007/s10898-020-00943-7

[9] Wei, F.R. and Zhu, H.X. (2012) Group Coordinate Descent Algorithms for Nonconvex Penalized Regression. *Computational Statistics & Data Analysis*, **56**, 316-326. https://doi.org/10.1016/j.csda.2011.08.007

[10] Jiang, H., Zheng, W.H. and Dong, Y. (2021) Sparse and Robust Estimation with Ridge Minimax Concave Penalty. *Information Sciences*, **571**, 154-174. https://doi.org/10.1016/j.ins.2021.04.047

[11] Selesnick, I. (2017) Sparse Regularization via Convex Analysis. *IEEE Transactions on Signal Processing*, **65**, 4481-4494. https://doi.org/10.1109/TSP.2017.2711501

[12] Shi, Y.Y., Huang, J., Jiao, Y.L. and Yang, Q.L. (2019) A Semismooth Newton Algorithm for High-Dimensional Nonconvex Sparse Learning. *IEEE Transactions on Neural Networks and Learning Systems*, **31**, 2993-3006. https://doi.org/10.1109/TNNLS.2019.2935001

[13] Wang, S.B., Chen, X.F., Dai, W.W., Selesnick, I.W., Cai, G. and Cowen, B. (2018) Vector Minimax Concave Penalty for Sparse Representation. *Digital Signal Processing*, **83**, 165-179. https://doi.org/10.1016/j.dsp.2018.08.021

[14] Zhang, C.H. (2010) Nearly Unbiased Variable Selection under Minimax Concave Penalty. *The Annals of Statistics*, **38**, 894-942. https://doi.org/10.1214/09-AOS729

[15] Xu, J.W. and Chao, M.T. (2021) An Inertial Bregman Generalized Alternating Direction Method of Multipliers for Nonconvex Optimization. *Journal of Applied Mathematics and Computing*, **68**, 1-27. https://doi.org/10.1007/s12190-021-01590-1

[16] Cai, G.G., Wang, S.B, Chen, X.F., Ye, J.J. and Selesnick, I.W. (2020) Reweighted Generalized Minimax-Concave Sparse Regularization and Application in Machinery Fault Diagnosis. *ISA Transactions*, **105**, 320-334. https://doi.org/10.1016/j.isatra.2020.05.043

[17] Li, Z.P., Qiao, B.J., Wen, B., Li, Z.D. and Chen, X.F. (2021) Reweighted Generalized Minimax-Concave Sparse Regularization for Duct Acoustic Mode Detection with Adaptive Threshold. *Journal of Sound and Vibration*, **506**, Article ID: 116165. https://doi.org/10.1016/j.jsv.2021.116165

[18] Li, H. and Lin, Z.C. (2015) Accelerated Proximal Gradient Methods for Nonconvex Programming. *Proceedings of the*

28*th International Conference on Neural Information Processing Systems*, Vol. 1, Montreal, 7-12 December 2015, 379-387.

[19] Breheny, P. and Huang, J. (2011) Coordinate Descent Algorithms for Nonconvex Penalized Regression, with Applications to Biological Feature Selection. *The Annals of Applied Statistics*, **5**, 232-253. https://doi.org/10.1214/10-AOAS388

[20] Beck, A. (2017) First-order Methods in Optimization. Society for Industrial and Applied Mathematics, Philadelphia. https://doi.org/10.1137/1.9781611974997

[21] Rockafellar, R.T. and Wets, R.J. B. (1998) Variational Analysis. Vol. 317, Springer Science & Business Media, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-02431-3

[22] Chang, C. and Lin, C.J. (2011) Libsvm: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, **2**, Article No. 27. https://doi.org/10.1145/1961189.1961199