

无约束非凸优化高阶张量算法研究

林玉婷, 李永杰, 杨健茹

广东工业大学数学与统计学院, 广东 广州

收稿日期: 2025年9月16日; 录用日期: 2025年10月9日; 发布日期: 2025年10月17日

摘要

无约束优化问题在工程、经济及计算机视觉等领域有广泛应用, 其目标函数因系统复杂与环境不确定呈高度非凸特性, 使传统算法在效率与精度上双重受限。针对三次连续可微且三阶导数全局Lipschitz连续的非凸优化问题, 现有方法局限显著: 一阶算法(如SGD)收敛慢、易陷鞍点, 难以达全局优化; 二阶方法(如牛顿法)受Hessian矩阵存储计算开销限制, 精确求解要求也降低适用性; 三次正则化等三阶方法虽用高阶信息, 但计算成本高, 部分场景难构建全局光滑三阶逼近。文章基于信赖域范式与不动点迭代, 结合目标函数高阶导数信息, 通过减少迭代复杂度优化子问题求解, 衔接理论与实际并减弱参数敏感性, 设计了新型无约束非凸优化算法SSFPI-HOA。由数值实验可知, SSFPI-HOA算法能提升非凸问题最优解质量与计算效率, 为领域提供新理论技术支撑, 助力深度学习等领域实际问题求解。

关键词

无约束非凸优化, 高阶优化算法, 张量计算, 不动点迭代

Research on High-Order Tensor Algorithms for Unconstrained Non-Convex Optimization

Yuting Lin, Yongjie Li, Jianru Yang

School of Mathematics and Statistics, Guangdong University of Technology, Guangzhou Guangdong

Received: September 16, 2025; accepted: October 9, 2025; published: October 17, 2025

Abstract

Unconstrained optimization problems play an important role in fields such as engineering, economics, and computer vision. Their objective functions exhibit highly non-convex characteristics due to system complexity and environmental uncertainty, rendering traditional algorithms doubly constrained in terms of efficiency and accuracy. For non-convex optimization problems that are three times continuously differentiable with globally Lipschitz continuous third-order derivatives, exist-

ing methods have significant limitations: first-order algorithms (e.g., SGD) converge slowly, are prone to getting trapped in saddle points, and hardly achieve global optimization; second-order methods (e.g., Newton's method) are restricted by the storage and computational costs of Hessian matrices, and the requirement for accurate solution further reduces their applicability; third-order methods such as cubic regularization, although utilizing high-order information, incur high computational costs and struggle to construct globally smooth third-order approximations of objective functions in some scenarios. Based on the trust region paradigm and fixed-point iteration, this study combines the high-order derivative information of objective functions and designs a novel unconstrained non-convex optimization algorithm (SSFPI-HOA) by reducing iteration complexity, optimizing subproblem solving, bridging theory and practice, and reducing parameter sensitivity. Numerical experiments show that the SSFPI-HOA algorithm can improve the quality of optimal solutions and computational efficiency for non-convex problems, providing new theoretical and technical support for the field and facilitating the solution of practical problems in areas such as deep learning.

Keywords

Unconstrained Non-Convex Optimization, High-Order Optimization Algorithm, Tensor Computation, Fixed-Point Iteration

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

无约束非凸优化问题广泛存在于科学与工程优化控制、经济调度及计算机视觉图像恢复等领域。其目标函数多因系统复杂性与不确定性呈高度非凸特性，求解时面临效率与精度双重难题；尤其在深度学习中，多层神经网络的非线性非凸损失函数不仅使全局最优解难寻，局部最优解判断更属 NP 难问题，因此高阶优化方法至关重要。

当前大规模机器学习非凸问题的求解仍存瓶颈：一阶算法(如 SGD)虽计算廉价，但收敛缓慢(线性收敛)，易陷局部极小点且不满足二阶最优性；二阶算法(如牛顿法、信赖域法)受限于 Hessian 矩阵存储 ($O(n^2)$)、正定要求及求逆和分解的高开销，难以适配大规模问题；三次正则化方法虽兼顾性能，却因计算代价高昂，不适合大规模数据处理。

本文基于信赖域范式与不动点迭代，提出新型高阶优化算法。通过优化子问题求解与随机采样策略，解决大规模高维非凸问题，提升计算精度与速度，且证明算法全局收敛性，可为深度学习等领域的实际应用提供支撑。

符号：设 $\mathbb{R}$ 为实数空间，并且， $\mathbb{R}^d$ 表示 d 维实向量集合， $\mathbb{R}^{n \times m}$ 表示 $n \times m$ 维实矩阵集合， $\mathbb{R}^{n_1 \times n_2 \times \cdots \times n_p}$ 为 p 阶张量集合。在整个论文的写作中，使用小写粗体字母(如 $\mathbf{g} \in \mathbb{R}^n$)表示向量，大写粗体字母(如 $\mathbf{H} \in \mathbb{R}^{n \times n}$)表示矩阵，花体大写字母(如 $\mathcal{T} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_p}$)表示高阶张量，下标字母(如 $\mathbf{g}_i$, $\mathbf{H}_{i,j}$, $\mathcal{T}_{i_1, i_2, \dots, i_p}$)可表示向量、矩阵、张量中的具体分量。

2. 问题描述

考虑无约束非凸优化问题

$$\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}) \quad (2.1)$$

其中, $f(\mathbf{x}): \mathbb{R}^n \rightarrow \mathbb{R}$ 三次连续可微且有下界, 其三阶导数满足全局 Lipschitz 连续性: 存在常数 $L_p \geq 0$, 对于任意 $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, 有

$$\|\nabla^p f(\mathbf{x}) - \nabla^p f(\mathbf{y})\|_{[p]} \leq L_p \|\mathbf{x} - \mathbf{y}\| \quad (2.2)$$

成立, 其中 $\|\cdot\|_{[p]}$ 为 p 阶张量的谱范数, $\nabla^p f(\mathbf{x})$ 是 p 阶导数张量, 定义为:

$$\nabla^p f(\mathbf{x}) = \left[\frac{\partial^p f(\mathbf{x})}{\partial x_{j_1} \partial x_{j_2} \cdots \partial x_{j_p}} \right]_{j \in [n]^p},$$

这里 $[n] = \{1, \dots, n\}$, $j_1, \dots, j_p$ 为遍历 p 维索引。

多项式局部模型 $m(\mathbf{x}_k, \mathbf{d})$ 依赖 $f(\mathbf{x}_k + \mathbf{d})$ 在迭代点 $\mathbf{x}_k \in \mathbb{R}^n$ 处的三阶泰勒展开构建, 形式为:

$$m(\mathbf{x}_k, \mathbf{d}) := f(\mathbf{x}_k) + \nabla_x f(\mathbf{x}_k)^T \mathbf{d} + \frac{1}{2} \mathbf{d}^T \nabla_x^2 f(\mathbf{x}_k) \mathbf{d} + \frac{1}{6} \mathbf{d}^T \nabla_x^3 f(\mathbf{x}_k) [\mathbf{d}, \mathbf{d}]. \quad (2.3)$$

固定当前迭代点 $\mathbf{x}_k$, 第 i 次迭代中, 模型 $m(\mathbf{x}_k, \mathbf{d}^{(i)})$ 可表示为:

$$m(\mathbf{x}_k, \mathbf{d}^{(i)}) = f_i(\mathbf{x}_k) + \mathbf{g}_i(\mathbf{x}_k)^T \mathbf{d}^{(i)} + \frac{1}{2} \mathbf{H}_i(\mathbf{x}_k) [\mathbf{d}^{(i)}]^2 + \frac{1}{6} \mathcal{T}_i(\mathbf{x}_k) [\mathbf{d}^{(i)}]^3$$

其中, $f_i(\mathbf{x}_k) \in \mathbb{R}$ 表示当前迭代点处的函数值, $\mathbf{g}_i(\mathbf{x}_k)^T = \nabla_x f(\mathbf{x}_k) \in \mathbb{R}^n$ 表示梯度向量, $\mathbf{H}_i(\mathbf{x}_k) = \nabla_x^2 f(\mathbf{x}_k) \in \mathbb{R}^{n \times n}$ 表示 Hessian 矩阵, 且 $\mathcal{T}_i(\mathbf{x}_k) = \nabla_x^3 f(\mathbf{x}_k) \in \mathbb{R}^{n \times n \times n}$ 表示三阶导数张量。

3. 算法设计

Cartis 等人[1] [2]指出, 无约束非凸优化算法逼近局部极小值的核心步骤为:

- (1) 求解局部模型: 第 i 次迭代计算搜索方向 $\mathbf{d}_k \approx \arg \min_{\mathbf{s} \in \mathbb{R}^n} m(\mathbf{x}_k, \mathbf{d}^{(i)})$;
- (2) 更新迭代点: $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{d}_k$, 逐步降低目标函数值;
- (3) 终止条件: 函数值变化量小于阈值 ϵ , 或梯度范数 $\|\nabla f(\mathbf{x}_{k+1})\| < \epsilon$ 时, 输出近似局部极小值。

子问题求解效率直接影响算法整体性能。在最小化局部模型时, $\mathbf{d}_k$ 要满足一阶最优性条件(2.4)和二阶最优性条件(2.5):

$$\mathbf{g}_i(\mathbf{x}_k) + \mathbf{H}_i(\mathbf{x}_k) \mathbf{d}_k + \frac{1}{2} \mathcal{T}_i(\mathbf{x}_k) [\mathbf{d}_k]^2 = 0, \quad (2.4)$$

$$\mathbf{H}_i(\mathbf{x}_k) + \mathcal{T}_i(\mathbf{x}_k) [\mathbf{d}_k] \geq 0. \quad (2.5)$$

子问题算法设计的目标是通过求解关于 $\mathbf{d}_k$ 的二次方程(2.4)找到序列 $\{\mathbf{d}^{(i)}\}_{i \geq 0}$, 使其收敛至满足 $\|\mathbf{g}_i\| \leq \epsilon_1$ 与 $\lambda_{\min}[\mathbf{H}_i] \geq -\epsilon_2$, 其中 $\lambda_{\min}[\mathbf{H}_i]$ 表示对称矩阵 $\mathbf{H}_i$ 的最小特征值, (ϵ_1, ϵ_2) 分别是一阶最优性条件和二阶最优性条件所允许的误差。

为应对高维计算与实现高效收敛, 文章采用随机采样技术, 通过高斯分布随机选取采样指标集 C , 将高维向量、矩阵、张量降维采样, 显著压缩计算规模, 使子问题求解在内存与时间成本上更可行。接着, 采用不动点迭代算法思想将一阶最优性条件变形为不动点迭代函数

$$\mathbf{d}_{i+1} = -\left(\mathbf{H}_i(\mathbf{x}_k) + \frac{1}{2} \mathcal{T}_i(\mathbf{x}_k) \mathbf{d}_i\right)^{-1} \mathbf{g}_i(\mathbf{x}_k) \quad (2.6)$$

以逐步逼近的方式, 重复更新采样指标集对应的步长, 直至 $\|\mathbf{d}_{i+1}^C - \mathbf{d}^C\| < \epsilon$ 。

算法 1 随机采样不动点迭代法算法(SSFPI)

输入: 目标采样数 m , 下标总数 n , 当前外层迭代点 $\mathbf{x}_k$, 初始内层搜索方向 $\mathbf{d}_k^{(0)}$, 内层最大迭代次

数 T ，内层收敛阈值 ϵ_{inner}

输出：外层迭代 k 的最终搜索方向 $\mathbf{d}_{k+1}$

- 1) 采样指标集 $C \leftarrow$ 从 $[n]$ 中随机抽取 m 个不同下标;
- 2) $\mathbf{d}_k^{(0),C}, \mathbf{g}_k^C, \mathbf{H}_k^C, \mathcal{T}_k^C \leftarrow$ 根据 C 对 $\mathbf{d}_k^{(0)}, \mathbf{g}_k, \mathbf{H}_k, \mathcal{T}_k$ 采样;
- 3) 初始化内层迭代计数器 $t=0$ ，当前采样方向 $\mathbf{d}^C \leftarrow \mathbf{d}_k^{(0),C}$;
- 4) while $t < T$ do
- 5) $\mathbf{M}_t^C \leftarrow \mathbf{H}_k^C + \frac{1}{2} \mathcal{T}_k^C [\mathbf{d}^C]$;
- 6) if $\mathbf{M}_t^C$ 是不正定的 then
- 7) $\mathbf{M}_t^{C,+} \leftarrow$ 计算 $\mathbf{M}_t^C$ 的摩尔-彭罗斯伪逆;
- 8) else
- 9) $\mathbf{M}_t^{C,-1} \leftarrow$ 计算 $\mathbf{M}_t^C$ 的逆矩阵;
- 10) end if
- 11) 更新采样方向: $\mathbf{d}_k^{(t+1),C} = -\mathbf{M}_t^{C,+/-1} \mathbf{g}_k^C$;
- 12) if $\|\mathbf{d}_k^{(t+1),C} - \mathbf{d}^C\| < \epsilon_{inner}$ then
- 13) 终止内层迭代, 令 $\mathbf{d}_k^C = \mathbf{d}_k^{(t+1),C}$
- 14) else
- 15) 跳转至步骤 11
- 16) end if
- 17) 更新 $t = t + 1$
- 18) $\mathbf{d}^C \leftarrow \mathbf{d}_k^{(t+1),C}$
- 19) end while
- 20) 重构全量方向 $\mathbf{d}_k$ (非采样分量按 0 填充, 或通过插值补全)
- 21) return $\mathbf{d}_k$

子问题求解算法 SSFPI 中, 矩阵 $\mathbf{M}_t^C$ 非正定时会出现求逆困难且不稳定情况。传统的 Tikhonov 正则化会引入偏差、需调参。摩尔-彭罗斯伪逆借矩阵自身 SVD 计算, 无偏、稳定、无需调参, 能避免迭代中断、加速收敛, 还降低额外计算成本。结合三次正则化算法框架和信赖域算法, 可进一步得到高阶优化算法 SSFPI-HOA。

算法 2 高阶优化算法(SSFPI-HOA)

输入：目标函数 f 及其三阶连续导数 $\mathbf{g}(\mathbf{x}_k), \mathbf{H}(\mathbf{x}_k), \mathcal{T}(\mathbf{x}_k)$, 初始外层迭代点 $\mathbf{x}_0$, 外层收敛阈值 ϵ_1, ϵ_2 , 接受阈值 $0 < \eta < 1$

输出：局部最优解 $\mathbf{x}^*$, 外层迭代步数 k

初始化：设定初始 $\mathbf{x}_0 = \mathbf{1} \in \mathbb{R}^n$, 迭代计数器 $k=0$, 容差 $\epsilon_1, \epsilon_2 > 0$, $\eta = 0.1$

- 1) while True do
- 2) if $\|\mathbf{g}(\mathbf{x}_k)\| \leq \epsilon_1$ 且 $\lambda_{\min}[\mathbf{H}(\mathbf{x}_k)] \geq -\epsilon_2$ then
- 3) $\mathbf{x}^* = \mathbf{x}_k, K = k$;
- 4) return $\mathbf{x}^*, K$;
- 5) end if
- 6) 根据算法 1 (SSFPI) 更新步长 $\mathbf{d}_k$;
- 7) 根据(2.3)计算 $m(\mathbf{x}_k, \mathbf{d}_k)$

- 8) 计算 $\rho_k = \frac{f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{d}_k)}{f(\mathbf{x}_k) - m(\mathbf{x}_k, \mathbf{d}_k)}$
- 9) if $\rho_k > \eta$ 或 $f(\mathbf{x}_k + \mathbf{d}_k) < f(\mathbf{x}_k)$ then
- 10) $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{d}_k$
- 11) else
- 11) $\mathbf{x}_{k+1} := \mathbf{x}_k$
- 12) goto Step 6
- 13) end if
- 14) $k := k + 1$
- 15) end while

4. 数值实验

4.1. 具体的非凸问题求解

为探究非凸优化求解特性，选取两类典型非凸函数： $y = 2x^3 - 2x^2$ (函数 A)、 $y = x^4 - x^2$ (函数 B)。由图 1 可知，在一维情形下，这两个函数的鞍点均为 $x = 0$ 。

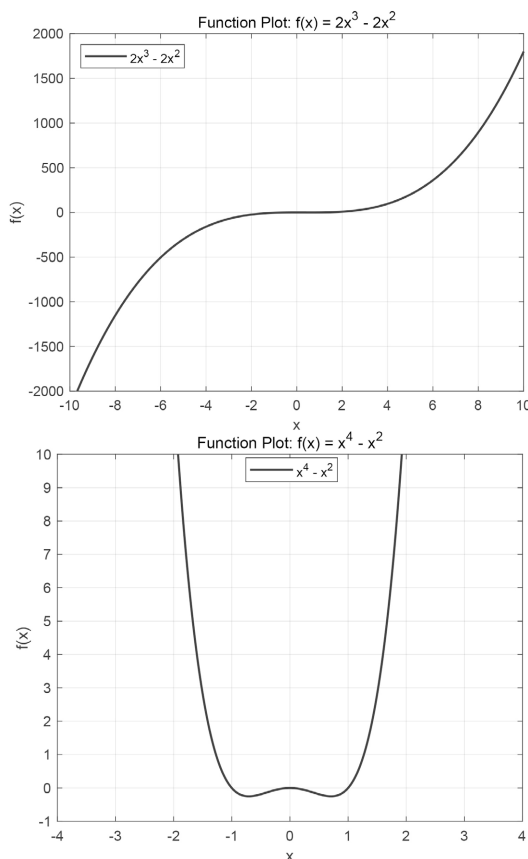


Figure 1. Function plots of Function A and Function B in the case of one dimension
图 1. 函数 A 和函数 B 在一维的情况下的函数图像

在后续的实验过程中，围绕问题 A 和问题 B，进行了如下细致的设置：维度 n (模拟不同规模问题，

探规模对算法性能的影响)、采样规模 m (调采样量, 探其对收敛的影响); x 初始为全 1 向量, 步长随机初始化以避免偏差, 最大迭代 35 (实际均提前收敛, 体现算法效率)。

Table 1. Analysis of results of the SSFPI algorithm under different dimensions and sampling sizes

表 1. SSFPI 算法不同维度 n 与采样规模 m 的结果分析

函数类型	维度 n	采样数 m	迭代次数	运行时间	收敛值
函数 A	10	2	13	3.25	-0.5926
	10	5	14	1.14	-1.4815
	20	2	13	8.35	-0.5926
	20	5	13	8.49	-1.4815
	20	10	14	8.20	-2.9630
函数 B	10	2	29	0.81	-0.4938
	10	5	30	0.89	-1.2346
	20	2	29	8.24	-0.4938
	20	5	30	8.60	-1.2346
	20	10	31	8.90	-2.4691

对表 1 中的数据进行深入分析, 可以得出如下结论: 对于所选取的两类非凸函数, 随着采样规模 m 的不断增大, 算法的收敛精度均呈现出明显的提升趋势, 计算结果也更为理想。然而, 与此同时, 计算时间也会相应地有所增长, 这体现出算法在收敛精度与计算效率之间存在一定的权衡关系, 也为后续进一步优化算法以更好地平衡这两者提供了方向。

4.2. 非凸逻辑回归问题的数值实验

为验证 SSFPI-HOA 算法在非凸逻辑回归问题中的性能表现, 选取如下非凸逻辑回归模型展开研究:

$$\min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^n \left(\frac{1}{1 + e^{-\mathbf{w}^T \mathbf{x}_i}} - y_i \right)^2 + \frac{\alpha}{2} \|\mathbf{w}\|^2 \quad (4-1)$$

其中, $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ 为样本集, $y_i \in \{0, 1\}$ 是样本标签, $\frac{1}{1 + e^{-\mathbf{w}^T \mathbf{x}_i}}$ 为非凸逻辑函数, 正则化参数 $\alpha = 10^{-5}$ 。问题(4-1)的损失函数 sigmoid 类别(文献[3] [4]同类型), 根据文献[5]可为其设计 sigmoidal programming 模型, 验证模型合理性。

实验数据采用来自 LIBSVM (可通过 <https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html> 获取)的 a1a 数据集, 该数据集样本数量为 1605, 数据维度为 119。将 SSFPI-HOA 与一阶增量梯度算法 SAGA、随机梯度下降 SGD 三种算法在 a1a 数据集上求解问题(4-1), 通过对比损失值(Loss)下降趋势、梯度范数(Norm_Grad)稳定性以及单次迭代耗时(Time), 来分析各算法的性能差异。

从图 2 中可以看出, SSFPI-HOA 算法的损失值下降最为显著且稳定, 在迭代过程中能快速且持续地降低损失; SAGA 算法损失值下降相对平缓; SGD 算法损失值下降幅度和速度均弱于前两者。这表明 SSFPI-HOA 在优化非凸逻辑回归损失函数时, 具有更强的目标函数优化能力。并且, SSFPI-HOA 算法的梯度范数整体波动较小, 保持着较好的稳定性; SAGA 算法梯度范数波动较大; SGD 算法梯度范数虽有下降趋势, 但波动情况也较为明显。梯度范数的稳定意味着算法在迭代过程中能更平稳地向最优解靠近, 减少震荡带来的无效迭代。但是, SSFPI-HOA 算法的单次迭代耗时明显低于 SAGA 和 SGD 算法, 且随着迭代轮次增加, 耗时稳定在较低水平。这体现出 SSFPI-HOA 在计算效率上的优势, 能以更少的时间成本完成迭代优化。

综合来看,在非凸逻辑回归问题中,SSFPI-HOA 算法在损失函数优化效果、梯度稳定性以及计算效率方面,均展现出优于 SAGA 和 SGD 算法的性能,尤其适合处理此类非凸优化任务。

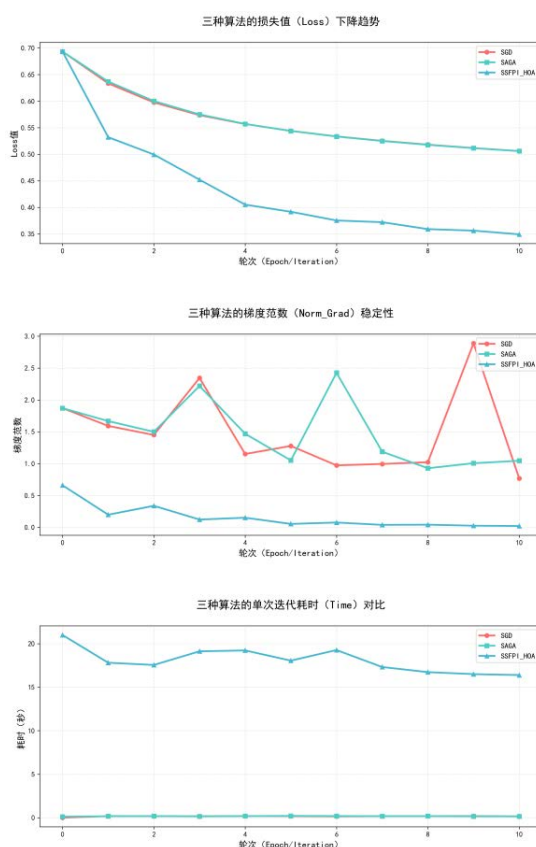


Figure 2. Comparison of results of different algorithms on the a1a dataset

图 2. 数据集 a1a 中不同算法的结果对比

基金项目

国家自然科学基金项目(12326302)。

参考文献

- [1] Cartis, C., Gould, N.I.M. and Toint, P.L. (2011) Adaptive Cubic Regularisation Methods for Unconstrained Optimization. Part I: Motivation, Convergence and Numerical Results. *Mathematical Programming*, **127**, 245-295. <https://doi.org/10.1007/s10107-009-0286-5>
- [2] Cartis, C., Gould, N.I.M. and Toint, P.L. (2011) Adaptive Cubic Regularisation Methods for Unconstrained Optimization. Part II: Worst-Case Function- and Derivative-Evaluation Complexity. *Mathematical Programming*, **130**, 295-319. <https://doi.org/10.1007/s10107-009-0337-y>
- [3] Ghadimi, S., Lan, G. and Zhang, H. (2019) Generalized Uniformly Optimal Methods for Nonlinear Programming. *Journal of Scientific Computing*, **79**, 1854-1881. <https://doi.org/10.1007/s10915-019-00915-4>
- [4] Mason, L., Baxter, J., Bartlett, P., et al. (1999) Boosting Algorithms as Gradient Descent. *Neural Information Processing Systems*. <https://api.semanticscholar.org/CorpusID:6101385>
- [5] Udell, M. and Boyd, S.P. (2013) Maximizing a Sum of Sigmoids. <https://api.semanticscholar.org/CorpusID:18061910>