优化算法复杂度分析简介

Jingguo Lan

The School of Management, USTC

2025 年 3 月 23 日

Table of Contents

研究内容

本讨论班聚焦于“随机优化方法与统计性质”，旨在探索随机优化算法和统计方法的理论基础及其内在联系，挖掘值得研究的问题。

研究内容：

- **经典随机优化方法**：包括优化算法收敛性分析、大规模随机优化算法（如 SGD 及其变体）、随机初始化、Sample Average Approximation (SAA) 等。帮助理解随机算法的基本原理及其复杂度分析技巧。
- **在线优化 (Online Optimization)**：涵盖 Online Gradient Descent、Online Mirror Descent、FTRL、Parameter-free Learning 和 Multi-Armed Bandit 等方法。解决序列数据中的动态决策问题，强调在不确定环境下的实时优化能力。
- **零阶优化 (Zeroth-order Optimization)**：针对梯度信息缺失的场景，通过随机采样和评估探索最优解。适用于目标函数复杂或不可微的情况。

讨论班安排

章节	理论核心	汇报人
算法复杂度理论分析	算法收敛性分析，凸/光滑/随机优化	兰敬国
大规模随机优化基础	经典 SGD 收敛理论; 方差缩减技术 二阶方法	张旋
SGD 及演进	动量加速原理; 自适应学习率机制 (AdaGrad/RMSProp/Adam)	金璋
随机初始化理论	随机初始化策略动机与原理	高哲
样本平均逼近方法	SAA 统计一致性; 蒙特卡洛采样; 收敛速率分析	金福隆
在线学习理论框架	OLO 框架； 一阶/二阶在线优化	金芝浩
FTRL 正则化	FTRL 收敛性证明; 在线镜像下降 Parameter-free	彭晖阳
多臂老虎机理论	随机 Bandit 分析;Upper Confidence Bound 统计学习	李嘉强
无梯度随机优化	随机扰动估计方法； 收敛速率分析 高维扩展性	罗皓天
无导数优化综述	确定性/随机算法； 无约束/有约束算法	章寒露

复杂度理论基础

问题背景

考虑通常形式下的优化问题，记做 $\mathcal{P}$ ：

$$f^* = \min_{x \in X} f(x)$$

根据约束集合 X 的类型和目标函数 $f(x)$ 的类型，可以对上述优化问题进行如下分类：

- **约束 / 无约束优化问题**：$X \subset \mathbb{R}^n$（约束）或 $X \equiv \mathbb{R}^n$（无约束）
- **光滑 / 非光滑优化问题**：$f(x)$ 在 X 上可导（光滑）或不可导（非光滑）
- **凸 / 强凸 / 非凸优化问题**：$f(x)$ 是凸函数（凸）、强凸函数（强凸）或非凸函数（非凸）
- **随机优化问题**：目标函数形式为 $f(x) = \mathbb{E}_\xi[F(x, \xi)]$

优化算法复杂度分析的理论基础

复杂度分析的理论可分为以下三个部分:

- **问题模型**
  对所要求解的优化问题进行分类和建模。
  例如: 线性规划、非线性规划、整数规划等。
- **算法模型**
  对求解优化问题的数值算法进行抽象和建模。
  例如: 梯度下降法、牛顿法、内点法等。
- **复杂度度量**
  定义算法模型在问题模型上的度量, 即复杂度。
  例如: 时间复杂度、空间复杂度、收敛速度等。

复杂度分析的问题模型

复杂度分析的问题模型分为三部分：全局信息、局部信息和解的精度。问题模型 $\mathcal{F}$ 可表示为：

$$\mathcal{F} \equiv (\Sigma, \mathcal{O}, \mathcal{T}_\epsilon)$$

- **全局信息** Σ
  描述问题集合 $\mathcal{F}$ 的共有特征，如目标函数的光滑性、可微性、约束类型等。算法 $\mathcal{S}$ 只能获取 Σ 中的信息。
- **局部信息** $\mathcal{O}$
  通过子程序 $\mathcal{O}$ 收集具体问题 $\mathcal{P}$ 的局部几何信息。例如，梯度法中的导数计算子程序。算法 $\mathcal{S}$ 可连续调用 $\mathcal{O}$ 获取局部信息。
- **解的精度** $\mathcal{T}_\epsilon$
  定义算法求解 $\mathcal{F}$ 时解的精度要求。不同问题集合 $\mathcal{F}$ 接受不同类型的精度度量方式。

算法 $\mathcal{S}$ 只能利用 Σ、$\mathcal{O}$ 和 $\mathcal{T}_\epsilon$ 三部分信息，逐步逼近最优解。

局部信息 $\mathcal{O}$

在复杂度分析中，算法通过子程序（Oracle）获取优化问题的局部信息。不同算法需要不同的局部信息：

- **梯度法**：返回函数值 $f(x_0)$ 和梯度 $\nabla f(x_0)$。
- **次梯度法**：返回次梯度信息 $\partial f(x_0)$。
- **牛顿法**：返回二阶导数信息 $\nabla^2 f(x_0)$。

子程序 $\mathcal{O}$ 具有以下两个关键性质：

1. **唯一性**：数值算法只能通过子程序获取局部信息。
2. **局部稳定性**：对测试点 x 的微小扰动，返回的局部信息变化不大。

局部稳定性是收敛性分析的关键假设。例如，分析梯度法时，假设目标函数或其导数具有 Lipschitz 连续性：

$$\|\mathcal{O}(x) - \mathcal{O}(y)\| \leq L\|x - y\|$$

如果目标函数不满足该性质，数值算法可能难以收敛或收敛性能较差。

解的精度 $\mathcal{T}_\epsilon$

解的精度 $\mathcal{T}_\epsilon$ 用于衡量算法求解优化问题的复杂度。根据问题类型不同，分为以下两类：

- **确定性优化问题**：
  - 当 $f(x)$ 是凸函数时，局部最优解即为全局最优解，算法第 k 步输出 x_k 满足以下条件之一时停止：

$$f(x_k) - f^* \le \epsilon, \quad \frac{f(x_k) - f^*}{f(x_k)} \le \delta, \quad \|\nabla f(x_k)\| \le \epsilon, \quad \|x_k - x^*\| \le \epsilon.$$

  - 当 $f(x)$ 是非凸函数时，采用梯度范数或解与最优解误差作为停止条件：

$$\|\nabla f(x_k)\| \le \epsilon, \quad \|x_k - x^*\| \le \epsilon.$$

- **随机优化问题**：随机优化算法的解是随机变量，需同时衡量解的精度和置信度：
  - ϵ 解：算法多次运行求得解的期望收敛效率：

$$\mathbb{E}[f(x_k) - f^*] \le \epsilon \quad \text{或} \quad \mathbb{E}[\|\nabla f(x_R)\|^2] \le \epsilon.$$

  - (ϵ, δ) 解：算法一次运行求得解的精度达到 ϵ 的置信率：

$$\mathrm{Prob}\{f(x_k) - f^* \ge \epsilon\} \le \delta \quad \text{或} \quad \mathrm{Prob}\{\|\nabla f(x_R)\|^2 \ge \epsilon\} \le \delta.$$

复杂度分析的算法模型

复杂度分析通过抽象迭代算法框架衡量优化算法的执行复杂度。以下是确定性和随机优化问题的算法框架：

- **确定性优化问题**：
  1. 在 x_k 调用子程序 $\mathcal{O}$，获取局部信息 $\mathcal{O}(x_k)$。
  2. 更新信息集合：$I_k = I_{k-1} \cup (x_k, \mathcal{O}(x_k))$。
  3. 应用规则处理 I_k，生成新迭代点 x_{k+1}。
  4. 验证停止条件 $\mathcal{T}_\epsilon$，满足则输出 $\bar{x}$，否则继续迭代。
- **随机优化问题**：在第 1 步中引入随机性：
  1. 根据分布随机抽样 ξ_k，调用随机子程序 $\mathcal{SFO}(x_k, \xi_k)$ 获取局部信息。
  2. 更新信息集合：$I_k = I_{k-1} \cup (x_k, \xi_k, \mathcal{SFO}(x_k, \xi_k))$。
  3. 后续步骤与确定性优化问题相同。
- **输出结果**：算法最终输出近似解 $\bar{x} = \mathcal{S}(x_0)$。

抽象迭代算法框架为复杂度分析提供了统一的理论基础。

复杂度分析的度量

有了问题模型和算法模型，我们定义算法 $\mathcal{S}$ 在问题 $\mathcal{P}$ 上的效率。定义两种复杂度：

- **分析复杂度**：求解问题 $\mathcal{P}$ 到精度 ϵ 所需调用子程序 $\mathcal{O}$ 的总次数。
- **算术复杂度**：求解问题 $\mathcal{P}$ 到精度 ϵ 所需执行的所有算术操作（包括子程序内部和算法本身）。

复杂度上界与下界：

- **上界**：给定算法 $\mathcal{S}$，$\mathcal{F}$ 的复杂度上界为：

$$\mathrm{Compl}_{\mathcal{S}}(\epsilon) = \sup_{\mathcal{P}\in\mathcal{F}} N_{\mathcal{S}}(\mathcal{P}, \epsilon)$$

- **下界**：对算法集 $\mathcal{M}$，$\mathcal{F}$ 的复杂度下界为：

$$\mathrm{Compl}(\epsilon) = \inf_{\mathcal{S}\in\mathcal{M}} \sup_{\mathcal{P}\in\mathcal{F}} N_{\mathcal{S}}(\mathcal{P}, \epsilon)$$

  上界由高效算法决定，下界由病态问题决定。

收敛率与复杂度的关系：

- **次线性收敛率**：$f(x_k) - f^* \leq \frac{c}{\sqrt{k}}$，对应复杂度为 $\mathcal{O}\left(\frac{1}{\epsilon^2}\right)$。
- **线性收敛率**：$\|x_k - x^*\| \leq c(1-q)^k$，对应复杂度为 $\mathcal{O}\left(\ln\frac{1}{\epsilon}\right)$。

盒约束全局优化问题的复杂度分析

问题模型 2.1

考虑如下全局优化问题：

$$\min_{x \in B_n} f(x),$$

其中约束集合 $B_n = \{x \in \mathbb{R}^n \mid 0 \le x^{(i)} \le 1, i = 1, \ldots, n\}$，目标函数 $f(x)$ 在 B_n 上相对于 ℓ_∞ 范数是 Lipschitz 连续的：

$$|f(x) - f(y)| \le L\|x - y\|_\infty, \quad \forall x, y \in B_n.$$

问题模型 $\mathcal{F} = (\Sigma, \mathcal{O}, \mathcal{T}_\epsilon)$：

- **全局信息** Σ： $f(x)$ 在 B_n 上 ℓ_∞-Lipschitz 连续。
- **局部信息** $\mathcal{O}$： 零阶子程序（$\mathcal{ZO}$），返回函数值 $f(x_0)$。
- **解的精度** $\mathcal{T}_\epsilon$： 求近似解 $\bar{x} \in B_n$，满足 $f(\bar{x}) - f^* \le \epsilon$。

算法模型

算法 1.2 无导数全局优化算法 $\mathcal{S}(p)$

输入： 盒约束集合每一维划分数 p，问题维数 n.

1. 构造包含 $(p+1)^n$ 个点的**测试点列**：

$$x_{(i_1,\ldots,i_n)} = \left(\frac{i_1}{p}, \frac{i_2}{p}, \cdots, \frac{i_n}{p}\right). \tag{1.27}$$

   其中 $(i_1, \ldots, i_n) \in \{0, \ldots, p\}^n$.

2. 遍历点列 $x_{(i_1,\ldots,i_n)}$，寻找使目标函数值 $f(x_{(i_1,\ldots,i_n)})$ 最小的点，记为 $\bar{x}$.

输出： $(\bar{x}, f(\bar{x}))$

无导数全局优化算法 $\mathcal{S}(p)$ 将 B_n 均匀分成 $(p+1)^n$ 个网格点，然后计算每个网格点上的函数值，最后返回函数值最小的网格点作为近似解。容易看到，$\mathcal{S}(p)$ 也属于我们的抽象迭代算法框架，它需要迭代 $(p+1)^n$ 次，全部遍历测试点列才能找到函数值最小的点。

复杂度分析

定理 1

记 f^* 为问题模型 *1* 的全局最优解，利用无导数全局优化算法求解．有，

$$f(\bar{x}) - f^* \leq \frac{L}{2p}$$

对于问题模型 *2.1* 中的每一个具体问题 $\mathcal{P} \in \mathcal{F}$ ，算法 $\mathcal{S}(p)$ 的分析复杂度满足：

$$N_{\mathcal{S}(p)}(\mathcal{P}, \epsilon) \leq \left(\left\lfloor \frac{L}{2\epsilon} \right\rfloor + 2 \right)^n$$

其中 $\lfloor a \rfloor$ 表示 a 的整数部分。

两边关于问题集合 $\mathcal{F}$ 中取极大，我们可以得到问题集合 $\mathcal{F}$ 的复杂度上界的一个估值：

$$\mathrm{Compl}_{\mathcal{S}(p)}(\epsilon) = \max_{\mathcal{P} \in \mathcal{F}} N_{\mathcal{S}(p)}(\mathcal{P}, \epsilon) \leq \left(\left\lfloor \frac{L}{2\epsilon} \right\rfloor + 2 \right)^n$$

也就是说，存在一个算法（比如 $\mathcal{S}(p)$ ）在解决 $\mathcal{F}$ 中每一个具体问题 $\mathcal{P}$ 时（近似解满足 $f(\bar{x}) - f^* \leq \epsilon, \bar{x} \in B_n$ ），所需要调用的 $\mathcal{ZO}$ 子程序次数之多不超过 $\left(\left\lfloor \frac{L}{2\epsilon} \right\rfloor + 2 \right)^n$

复杂度下界

我们会提出一些问题:

- 第一，我们在估计算法 $\mathcal{S}(p)$ 时太过粗略，是否存在更好的界？
- 第二，是否存在其他算法，其效率比 $\mathcal{S}(p)$ 要好？

要回答这两个问题，我们就需要推导问题集合 $\mathcal{F}$ 的复杂度下界。

定理 2

令 $\epsilon < \frac{1}{2}L$，对于对应的问题集合 $\mathcal{F}$ 来说，对于其中每一个具体问题 $\mathcal{P} \in \mathcal{F}$，对于其解算法集合 $\mathcal{M} := \mathcal{M}(\mathcal{F}, \mathcal{ZO})$ 中的任意一个算法 $\mathcal{S} \in \mathcal{M}$，其分析复杂度满足:

$$N_{\mathcal{S}}(\mathcal{P}, \epsilon) \geq \left(\left\lfloor \frac{L}{2\epsilon} \right\rfloor \right)^n$$

proof

只需要证明存在某一目标函数，使得 $\mathcal{S}_0$ 在求解该目标函数时的精度大于等于 ϵ 。

- 存在某一非测试点 $\hat{x}$ 以及集合 $B=\left\{x \mid \hat{x} \leq x \leq \hat{x}+\frac{1}{p} e\right\} \subseteq B_n$ 使得 B 中不包含任何测试点。
- 令 $x_*=\hat{x}+\frac{1}{2p} e$ ，在集合 B 上构造函数

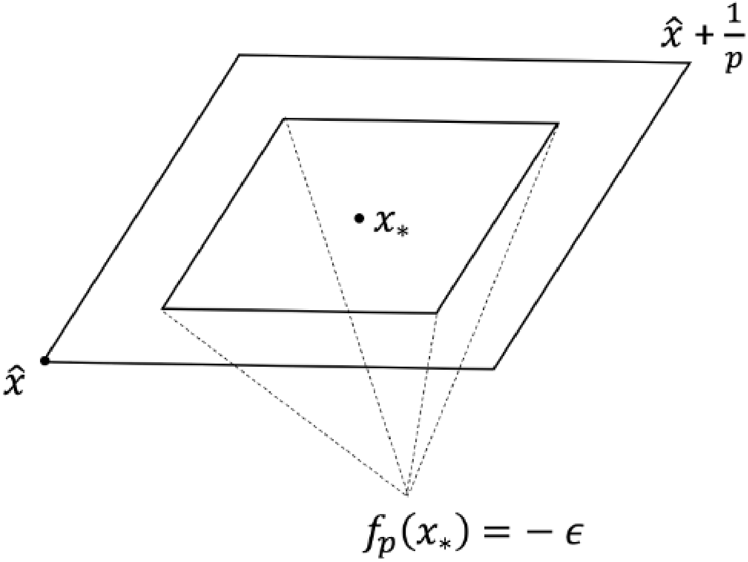
$$f_p(x)=\min \left\{0, L\left\|x-x_*\right\|_{\infty}-\epsilon\right\}, \quad \text{当} x \in B.$$

- 令 $f_p(x)=0$ 当 $x \in B_n \backslash B$ 。注意到 $\left\|x_*-\hat{x}\right\|_{\infty} \geq \frac{\epsilon}{L}$ ，因此 $f_p(\hat{x})=0$ 。
- 注意到 $f_p(x)$ 是 ℓ_{∞} - Lipschitz 连续（Lipschitz 常数为 L ），全局最优解为 $f_p\left(x_*\right)=-\epsilon$。利用 $\mathcal{S}_0$ 求解 $f_p(x)$ 时，由于在所要测试点列处调用 $\mathcal{ZO}$ 子程序都返回 $f_p\left(x_k\right)=0$ ，因此求得近似最优解为 $f_p(\bar{x})=0$ ，于是有

$$f_p(\bar{x})-f_p\left(x_*\right) \geq \epsilon$$

- 因此我们得到结论：若测试点列的个数 $N<\left(\left\lfloor\frac{L}{2 \epsilon}\right\rfloor\right)^n$（子程序 $\mathcal{ZO}$ 的调用次数）则对应的解算法求得的精度不可能比 ϵ 更好。即问题模型 2.1 的复杂度下界为 $\left(\left\lfloor\frac{L}{2 \epsilon}\right\rfloor\right)^n$ 。

图示



光滑优化问题的复杂度分析

问题

考虑优化问题

$$f^* = \min_{x\in X} f(x)$$

目标函数 $f(x)$ 是光滑的情况。首先我们定义光滑函数的一些子集合:

定义 3

设 X 为 $\mathbb{R}^n$ 的一个子集，记 $C_L^{k,p}(X)$ 为具有如下性质的函数集合:

1. 任何函数 $f \in C_L^{k,p}(X)$ 在 X 上 k 次连续可微.
2. 任何函数 $f \in C_L^{k,p}(X)$ 的 p 阶导数在 X 上 Lipschitz 连续（对于某常数 L ）,

$$\left\| f^{(p)}(x) - f^{(p)}(y) \right\| \le L\|x-y\|, \qquad \text{对任意} x, y \in X.$$

记 $\mathcal{F}_L^{k,p}(X)$ 表示函数集合 $C_L^{k,p}(X)$ 与凸函数集合的交集。

无约束非凸优化的复杂度分析

考虑无约束非凸优化问题集合 $\mathcal{F} = (\Sigma, \mathcal{O}, \mathcal{T}_\epsilon)$:

问题模型描述:

- **全局信息** Σ: 目标函数 $f \in C_L^{1,1}(\mathbb{R}^n)$ (Lipschitz 梯度连续), 且 $f(x)$ 不一定是凸函数; $f(x)$ 下有界: $\exists M, f(x) \geq M, \forall x \in \mathbb{R}^n$; 约束集合 $X \equiv \mathbb{R}^n$。
- **局部信息** $\mathcal{O}$: 一阶子程序 ($\mathcal{FO}$), 返回函数值 $f(x_0)$ 和梯度 $\nabla f(x_0)$。
- **解的精度** $\mathcal{T}_\epsilon$: 求局部极小值的近似解 $\bar{x} \in \mathbb{R}^n$, 满足 $\|\nabla f(\bar{x})\| \leq \epsilon$。

算法 2.1 梯度法

输入: $x_0 \in \mathbb{R}^n$, 步长序列 $\{h_k\}$, 对 $k = 0, 1, \cdots$ 执行迭代,

$$x_{k+1} = x_k - h_k \nabla f(x_k). \tag{2.13}$$

定理

定理 4

取 $\bar{x}$ 满足 $\|\nabla f(\bar{x})\| = \min_{0\leq k\leq N} \|\nabla f(x_k)\|$，则算法的收敛速度为

$$\|\nabla f(\bar{x})\| \leq \frac{1}{\sqrt{N+1}} \left[\frac{L}{\omega}\left(f\left(x_0\right) - f^*\right)\right]^{1/2}$$

问题模型 *2.1* 的分析复杂度上界为

$$N(\epsilon) \leq \frac{L\left(f\left(x_0\right) - f^*\right)}{\omega\epsilon^2}$$

其中 ω 为常数。

梯度法求解该问题模型时候，表现为全局次线性收敛，分析复杂度上界为 $\mathcal{O}\left(\frac{L}{\epsilon^2}\right)$。

问题模型 2.2：无约束非凸优化

考虑无约束非凸优化问题集合 $\mathcal{F} = (\Sigma, \mathcal{O}, \mathcal{T}_\epsilon)$：

问题模型描述：

- **全局信息** Σ：目标函数 $f \in C_L^{2,2}(\mathbb{R}^n)$（二阶导数 Lipschitz 连续），且 $f(x)$ 不一定是凸函数；存在局部极小点 x^*，且 $\nabla^2 f(x^*)$ 正定；存在常数 $0 < m \leq M < \infty$，使得 $mI_n \preceq \nabla^2 f(x^*) \preceq MI_n$；初始点 x_0 距离 x^* 足够近；约束集合 $X \equiv \mathbb{R}^n$。
- **局部信息** $\mathcal{O}$：一阶子程序（$\mathcal{FO}$），返回函数值 $f(x_0)$ 和梯度 $\nabla f(x_0)$。
- **解的精度** $\mathcal{T}_\epsilon$：求局部极小值的近似解 $\bar{x} \in \mathbb{R}^n$，满足 $\|\bar{x} - x^*\| \leq \epsilon$。

非凸问题

定理 5

假设梯度法的初始点 x_0 距离局部极小点 x^* 足够近，满足 $r_0 = \|x_0 - x^*\| < \bar{r} = \frac{2m}{L}$ ，且步长取 $h_k \equiv \frac{2}{m+M}$ ，则梯度法求解问题模型 *2.2* 的收敛速率为，

$$\|x_k - x^*\| \leq \frac{\bar{r} r_0}{\bar{r} - r_0}\left(1 - \frac{2m}{M + 3m}\right)^k$$

复杂度上界为

$$\frac{M + 3m}{2m}\left[\ln\left(\frac{\bar{r} r_0}{\bar{r} - r_0}\right) + \ln\frac{1}{\epsilon}\right].$$

梯度法求解该问题模型时候，表现为局部线性收敛，且复杂度上界为 $\mathcal{O}\left(\ln\frac{1}{\epsilon}\right)$ 。

凸问题

定理 6

若 $f \in \mathcal{F}_L^{1,1}(\mathbb{R}^n)$，令步长取 $h_k \equiv h = \frac{2}{L}$，则梯度法求解凸优化问题的收敛速率为

$$f(x_k) - f^* \leq \frac{2L\|x_0 - x^*\|^2}{k+4}$$

记 $D_0 = \|x_0 - x^*\|$，则复杂度上界为

$$\mathcal{O}\left(\frac{LD_0}{\epsilon}\right)$$

梯度法求解该问题模型时候，表现为全局次线性收敛，且复杂度上界为 $\mathcal{O}\left(\frac{1}{\epsilon}\right)$。

强凸问题

定理 7

令 $Q = L/\mu$. 若 $f \in \mathcal{F}_{L,\mu}^{1,1}(\mathbb{R}^n)$ ，令步长取 $h_k \equiv h = \frac{1}{L}$ ，则梯度法求解强凸问题模型的收敛速率为

$$\|x_k - x^*\|^2 \le \left(\frac{Q-1}{Q+1}\right)^{2k} \|x_0 - x^*\|^2$$

$$f(x_k) - f^* \le \frac{L}{2}\left(\frac{Q-1}{Q+1}\right)^{2k} \|x_0 - x^*\|^2$$

目标函数值对应的复杂度上界为

$$\log\left(\frac{LD_0^2}{\epsilon}\right) / \log\left(\frac{Q-1}{Q+1}\right)^2$$

梯度法求解该问题模型时候，表现为全局线性收敛，且复杂度上界为 $\mathcal{O}\left(\ln\frac{1}{\epsilon}\right)$ 。

复杂度下界

定理 8

（光滑凸优化问题的复杂度下界）对任意 $x_0 \in \mathbb{R}^n$ 和 $1 \leq k \leq \frac{1}{2}(n-1)$，存在 $f \in C_L^{\infty,1}(\mathbb{R}^n)$，使得对任意 $x_k \in x_0 + \operatorname{span}\{\nabla f(x_0), \ldots, \nabla f(x_{k-1})\}$ 都有

$$f(x_k) - f^* \geq \frac{3L\|x_0 - x^*\|^2}{32(k+1)^2}$$

令 $D_0 = \|x_0 - x^*\|$，将不等式右端等于 ϵ，可以得到一阶算法（只利用梯度信息的算法）求解凸优化问题 $C_L^{\infty,1}(\mathbb{R}^n)$ 的复杂度下界为:

$$\mathcal{O}\left(\sqrt{\frac{L}{\epsilon}} D_0\right)$$

与梯度法的复杂度上界 $\mathcal{O}(LD_0/\epsilon)$ 比较，发现梯度法并不是最优算法。

Nesterov 加速梯度算法框架

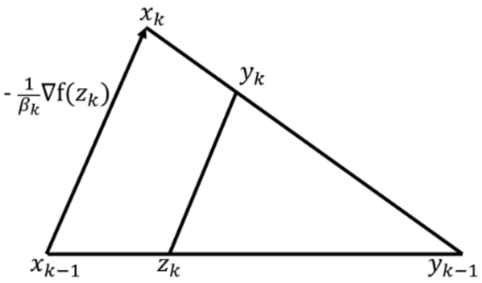
算法 2.3 Nesterov 加速梯度算法框架

输入： 令 $x_0 = y_0$, 选取序列 $\{\gamma_k\}$ 和 $\{\beta_k\}$ 使其满足 $L\gamma_k \le \beta_k$, $\gamma_1 = 1$。

对于$k = 1, ..., N$，执行迭代

1. $z_k = (1 - \gamma_k)y_{k-1} + \gamma_k x_{k-1}$，
2. $x_k = \mathrm{argmin}_{x\in X} \left\{\langle \nabla f(z_k), x\rangle + \frac{\beta_k}{2}\|x - x_{k-1}\|_2^2\right\}$，
3. $y_k = (1 - \gamma_k)y_{k-1} + \gamma_k x_k$。

输出: y_N



非光滑优化问题的复杂度分析

投影次梯度法

算法 3.1 投影次梯度法

输入：$x_0 \in \mathbb{R}^n$，对 $k = 0, 1, \cdots$ 执行迭代，

$$x_{k+1} = \underset{x \in X}{\operatorname{argmin}} \|x - (x_k - \gamma_k g(x_k))\|_2. \tag{3.3}$$

其中 $\gamma_k > 0$ 且 $g(x_k) \in \partial f(x_k)$.

我们可以用对目标函数做近似的角度来理解投影次梯度法的迭代

$$\begin{aligned} x_{k+1} &= \underset{x \in X}{\operatorname{argmin}} \frac{1}{2} \left\| x - (x_k - \gamma_k g(x_k)) \right\|_2^2 \\ &= \underset{x \in X}{\operatorname{argmin}} \gamma_k \langle g(x_k), x - x_k \rangle + \frac{1}{2} \|x - x_k\|_2^2 \\ &= \underset{x \in X}{\operatorname{argmin}} \gamma_k \langle g(x_k), x \rangle + \frac{1}{2} \|x - x_k\|_2^2 \\ &= \underset{x \in X}{\operatorname{argmin}} f(x_k) + \langle g(x_k), x - x_k \rangle + \frac{1}{2\gamma_k} \|x - x_k\|_2^2 . \end{aligned}$$

收敛定理

定理 9

我们考虑 $X \subset \mathbb{R}^n$ 闭且凸，$f: X \to \mathbb{R}^n$ 连续且凸，其次梯度满足 $\|g(x)\| \leq M$ 对任意 $x \in X$ 成立。令 $x_k, k = 1, \cdots, N$，由（3. 3）产生，且定义

$$\bar{x}_s^N = \frac{\gamma_s x_s + \cdots + \gamma_N x_N}{\gamma_s + \cdots + \gamma_N} = \left(\sum_{k=s}^{N} \gamma_k\right)^{-1} \sum_{k=s}^{k} \gamma_k x_k$$

则有

$$f\left(\bar{x}_s^N\right) - f^* \leq \left(2\sum_{k=s}^{N} \gamma_k\right)^{-1} \left[\|x_s - x^*\|_2^2 + M^2 \sum_{k=s}^{N} \gamma_k^2\right].$$

- **固定步长策略**：令 $D_X = \max_{x_1, x_2 \in X} \|x_1 - x_2\|$，取 $\gamma_k = \sqrt{\frac{D_X^2}{NM^2}}, \quad k = 1, \cdots, N$ 则，$f\left(\bar{x}_1^N\right) - f^* \leq \frac{MD_X}{2\sqrt{N}}$.
- **变步长策略**：取 $\gamma_k = \sqrt{\frac{D_X^2}{kM^2}}, \quad k = 1, \cdots$ 则，$f\left(\bar{x}_{\lceil k/2 \rceil}^N\right) - f^* \leq \mathcal{O}(1)\frac{MD_X}{\sqrt{k}}$.

镜像梯度法：从投影次梯度法的推广

投影次梯度法依赖于欧氏空间，引入非欧范数 $\|\cdot\|$ 和 Bregman 距离：

- **范数与对偶范数**：设 $\|\cdot\|$ 是 $\mathbb{R}^n$ 上的范数，其对偶范数定义为：

$$\|\xi\|_* = \max\{\langle \xi, x\rangle : \|x\| \le 1\}.$$

- **距离生成函数**：函数 $\omega : X \to \mathbb{R}$ 连续可微且强凸（相对于 $\|\cdot\|$ 和参数 $\mu > 0$）：

$$\langle \nabla\omega(x) - \nabla\omega(y), x - y\rangle \ge \mu\|x - y\|^2, \quad \forall x, y \in X.$$

- **Bregman 距离**：基于 $\omega(x)$ 定义的 Bregman 距离为：

$$V(x, y) = \omega(y) - [\omega(x) + \langle \nabla\omega(x), y - x\rangle].$$

  当 $\omega(x) = \|x\|_2^2/2$ 时，$V(x, y) = \|y - x\|_2^2/2$。

算法 3.2：**镜像梯度法**：输入初始点 $x_0 \in \mathbb{R}^n$，对于 $k = 0, 1, \ldots$ 执行迭代：

$$x_{k+1} = \operatorname*{argmin}_{x\in X} \gamma_k\langle g(x_k), x\rangle + V(x_k, x),$$

其中 $\gamma_k > 0$，$g(x_k) \in \partial f(x_k)$ 是次梯度。

重心法

问题模型 3.1 考虑约束凸优化问题集合 $\mathcal{F}=(\Sigma,\mathcal{O},\mathcal{T}_\epsilon)$：

- 全局信息 Σ： 目标函数 $f(x)\in C(X)$ 且是凸函数，存在常数 $B>0$ 使得对任意 $x\in X$ 有 $-B\le f(x)\le B$； 约束集合 $X\subset\mathbb{R}^n$ 闭且凸，
- 局部信息 $\mathcal{O}:\mathcal{FO}$ 子程序，对于任意给定的 x_0 返回函数值 $f(x_0)$ 和次梯度 $\partial f(x_0)$ ，
- 解的精度 $\mathcal{T}_\epsilon$： 求全局极小点 $\bar{x}\in\mathbb{R}^n$ ，使得 $f(\bar{x})-f^*\le\epsilon$。

算法 3.3 重心法

令 $\mathcal{S}_1=X$，对 $k=1,...,N$ 执行迭代，

1. 计算集合 $\mathcal{S}_k$ 的重心

$$c_k=\frac{1}{\mathrm{vol}(\mathcal{S}_k)}\int_{x\in\mathcal{S}_k}xdx. \tag{3.34}$$

2. 在 c_k 处调用 $\mathcal{FO}$ 子程序得 $w_k\in\partial f(c_k)$，更新集合 $\mathcal{S}_{k+1}$

$$\mathcal{S}_{k+1}=\mathcal{S}_k\cap\{x\in\mathbb{R}^n:\ (x-c_k)^Tw_k\le 0\}. \tag{3.35}$$

输出：$x_N\in\mathrm{argmin}_{1\le r\le N}\ f(c_r)$

定理

定理 10

重心法 *3.3* 在求解问题模型（*3.* *1*）时的收敛速率为，

$$f\left(x_N\right)-f^*\leq 2B\left(1-\frac{1}{e}\right)^{N/n}$$

复杂度上界为

$$\mathcal{O}\left(n\log\frac{2B}{\epsilon}\right).$$

从计算角度来说，由于每一步都需要计算集合的体积，即进行积分计算，因此重心法计算量过大，往往很难执行。重心法只是提供一个理论上的可行策略。

椭球法

算法 3.4 椭球法（The ellipsoid method）

输入：令 $\mathcal{E}_0$ 为包含约束集合 X 且半径为 R 中心为 c_0 的球，令 $H_0 = R^2 I_n$，

对 $k = 1, ..., N$ 执行迭代

1. 若 $c_k \notin X$，则调用子程序返回向量 $w_k \in \mathbb{R}^n$，过 c_k 且垂直 w_k 的平面将椭球 $\mathcal{E}_k$ 分割，使得约束集 $X \subset \{x : (x - c_k)^T w_k \le 0\}$;

   若 $c_k \in X$，则调用子程序返回向量 $w_k \in \partial f(c_k)$.

2. 构造新的椭球 $\mathcal{E}_{k+1} = \{x : (x - c_{k+1})^T H_{k+1}^{-1} (x - c_{k+1}) \le 1\}$ 使得

$$\{x \in \mathcal{E}_k : (x - c_k)^T w_k \le 0\} \subset \mathcal{E}_{k+1}. \tag{3.44}$$

   其中 c_{k+1}，H_{k+1} 满足，

$$c_{k+1} = c_k - \frac{1}{n+1} \frac{H_k w}{\sqrt{w^T H_k w}}, \tag{3.45}$$

$$H_{k+1} = \frac{n^2}{n^2 - 1} \left(H_k - \frac{2}{n+1} \frac{H_k w w^T H_k)}{w^T H_k w} \right). \tag{3.46}$$

输出：若 $\{c_1, ..., c_N\} \cap X \neq \varnothing$，则输出

$$x_N \in \mathrm{argmin}_{c \in \{c_1, ..., c_N\} \cap X} \, f(c). \tag{3.47}$$

定理 11

椭球法 *3.4* 在求解问题模型 *3.2* 时的收敛速率为

$$f(x_N) - f^* \le \frac{2BR}{r} \exp\left(-\frac{N}{2n^2}\right)$$

复杂度上界为

$$\mathcal{O}\left(n^2 \log \frac{BR}{r\epsilon}\right).$$

- 从子程序调用的分析复杂度来看，椭球法要比重心法差，前者需要调用 $\mathcal{FO}$ 子程序 $\mathcal{O}\left(n^2 \log\left(\frac{2BR}{r\epsilon}\right)\right)$ 次，而后者仅需要 $\mathcal{O}\left(n \log\left(\frac{2B}{\epsilon}\right)\right)$ 。
- 从计算角度来看，椭球法要比重心法更容易执行。

随机优化算法的复杂度分析

背景

考虑随机优化问题，

$$\min_{x\in X}\left\{f(x):=\mathbb{E}_{\xi}[F(x,\xi)]\right\} \tag{4.1}$$

其中 $X\subset\mathbb{R}^n$ 是一个非空有界闭凸集合。ξ 是一个随机向量，其分布 P 的支撑集满足 $\Xi\subset\mathbb{R}^d$ 。定义函数 $F:X\times\Xi\to\mathbb{R}$ ，对任意 $\xi\in\Xi, F(x,\xi)$ 都是凸函数，且关于 ξ 的期望

$$\mathbb{E}[F(x,\xi)]=\int_{\Xi}F(x,\xi)dP(\xi)$$

是有意义的，且对任意 $x\in X$ 都为有限值。

- 即使 ξ 的分布 P 已知，随着维数的增加，积分的计算也会变得很困难
- 对任意 $x\in X$ ，我们需要计算 $f(x)$ 的一个近似值 $\hat{f}(x)$ ，使得 $|\hat{f}(x)-f(x)|\leq\epsilon$ ，而这需要在不同的 $\xi\in\Xi$ 处计算 $\mathcal{O}\left(\frac{1}{\epsilon^m}\right)$ 次 $f(x,\xi)$
- 在高维情况下，需要采用随机算法，比如蒙特卡洛抽样技巧来降低求解的复杂度。
- 需要假设：对于随机向量 ξ ，存在已知的蒙特卡洛策略，能够产生一系列独立同分布的样本：$\{\xi_i\}_{i=1}^N$ 。

机器学习模型与优化问题

对于分类或回归问题，给定 n 个训练样本 $\{(x_1, y_1), \ldots, (x_n, y_n)\}$，其中：

- 样本特征 $x_i \in \mathcal{X}$，标签 $y_i \in \mathcal{Y}$；
- 样本独立从分布 D 中抽样。

目标是求解预测模型 $h(x; \omega) : \mathcal{X} \to \mathcal{Y}$，其参数 ω 通过优化问题确定：

$$\min_{\omega} R(\omega) + r(\omega),$$

其中：

- 期望风险 $R(\omega) = \mathbb{E}_{(x,y)\sim D}[\ell(h(x; \omega), y)]$；
- 正则项 $r(\omega)$（如 $r(\omega) = \|\omega\|_2^2$ 或 $r(\omega) = \|\omega\|_1$）控制模型复杂度。

由于分布 D 未知，我们用经验风险 $R_n(\omega)$ 近似 $R(\omega)$：

$$R_n(\omega) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i; \omega), y_i).$$

因此，优化问题变为：

$$\min_{\omega} R_n(\omega) + r(\omega).$$

关键问题：如何衡量经验风险优化问题的解对期望风险优化问题解的关系？

随机优化问题的解概念

在随机优化问题中，目标是最小化期望形式的目标函数：

$$f(x) = \mathbb{E}_{\xi}[F(x, \xi)].$$

由于分布未知，精确计算期望困难。我们通过采样数据 $\{\xi_i\}$ 和对应的目标值 $\{F(x, \xi_i)\}$ 来近似解。

定义 5.1：ϵ **近似解**：随机变量 $\tilde{x} \in X$ 满足：

$$\mathbb{E}_{\tilde{x}}[f(\tilde{x}) - f^*] \leq \epsilon,$$

则称 $\tilde{x}$ 为随机优化问题的 ϵ 近似解。

定义 5.2：(ϵ, δ) **近似解**：随机变量 $\tilde{x} \in X$ 满足：

$$\text{Prob}(f(\tilde{x}) - f^* \geq \epsilon) \leq \delta,$$

则称 $\tilde{x}$ 为随机优化问题的 (ϵ, δ) 近似解。

解释：

- ϵ 表示精度，δ 表示置信水平；
- 当 $\delta = 0$，(ϵ, δ) 近似解退化为确定性解。

采样平均逼近算法

采样平均逼近目标函数

$$\hat{f}_N(x) = \frac{1}{N}\sum_{i=1}^{N} F(x, \xi_i) \qquad (5.2)$$

当采样的样本总量 N 变大，对任意的 $x \in X$ 以及 $\epsilon > 0$ ，我们有结论，

$$\text{Prob}\left(\left|\frac{1}{N}\sum_{i=1}^{N} F(x, \xi_i) - f(x)\right| \geq \epsilon\right) \to 0$$

定理 12

假定函数 $F(x,\xi)$ 任意变差有界，即存在 $V < \infty$ ，其中

$$V = \max\{F(x_1, \xi_1) - F(x_2, \xi_2) : x_1, x_2 \in X, \xi_1, \xi_1 \in \Xi\}$$

则对任意 $\epsilon > 0$ 和 $\delta \in (0, 1)$ ，当抽样样本数量取 $N = \left\lceil \frac{V^2}{2\epsilon^2} \log\left(\frac{2}{\delta}\right)\right\rceil$ 时我们有，

$$\text{Prob}\left\{\left|\hat{f}_N(x) - f(x)\right| > \epsilon\right\} \leq \delta$$

Remark

从上述定理我们知道，对任意 $x \in X$ ，为了对问题 4.1 中 $f(x)$ 做 ϵ 精度的逼近，且逼近的置信水平为 δ ，我们需要至少对随机变量抽样 $\left\lceil \frac{V^2}{2\epsilon^2} \log\left(\frac{2}{\delta}\right) \right\rceil$ 次。定理 12 告诉我们，

$$\mathrm{Prob}\left\{\left|\hat{f}_N\left(x^*\right) - f\left(x^*\right)\right| > \epsilon\right\} \leq \delta$$

而我们更关注是否有，

$$\mathrm{Prob}\left\{\left|\hat{f}_N\left(\hat{x}^*\right) - f\left(x^*\right)\right| > \epsilon\right\} \leq \delta$$

成立。因为采样平均逼近算法实际中求解的是优化问题 5.2。另外，考虑到数值优化算法只能对 $\hat{x}^*$ 求解到某一 ϵ 精度，我们需要知道 $\hat{x}^*$ 的 ϵ 近似解是否也是 x^* 的 (ϵ, δ) 近似解？

逼近定理

定理 13

若 $X \subset \mathbb{R}^n$，令 $D := \sup_{x,y\in X} \|x-y\|$，假设存在 $L > 0$，使得对任意 $x, y \in X$ 和 $\xi \in \Xi$ 有 $|F(x,\xi) - F(y,\xi)| \le L\|x-y\|$。如果 *5.2* 中 N 满足

$$N \ge \mathcal{O}(1)\left(\frac{DL}{\epsilon}\right)^2\left[n\ln\left(\frac{DL}{\epsilon}\right) + \ln\left(\frac{1}{\delta}\right)\right]$$

则优化问题 *5.2* 每个精度为 $\epsilon/2$ 的近似解都是随机优化问题 *5.1* 的 (ϵ,δ) 近似解。

若 $\hat{f}_N(x) \in C_L^{0,1}(X)$ 且是凸函数，利用投影次梯度法求 5.2 精度为 $\epsilon/2$ 的近似解需要对 $\hat{f}_N(x)$ 求 $\mathcal{O}\left(L^2D^2/\epsilon^2\right)$ 次导数，若使该解为 5.1 的 (ϵ,δ) 近似解，则总共需要对 $F(x,\xi)$ 求导的次数不少于：

$$\mathcal{O}\left(\frac{D^4L^4}{\epsilon^4}\left[n\ln\left(\frac{DL}{\epsilon}\right) + \ln\left(\frac{1}{\delta}\right)\right]\right)$$

忽略常数项后，抽样平均逼近算法求解随机优化问题 5.1 的复杂度为

$$\mathcal{O}\left(\frac{n}{\epsilon^4}\ln\frac{1}{\epsilon} + \frac{1}{\epsilon^4}\ln\frac{1}{\delta}\right)$$

随机逼近算法

随机逼近算法是随机梯度算法的原型，其核心特点在于利用随机次梯度近似目标函数 $f(x)$ 的梯度，避免了多次计算。

关键假设：存在子程序 $\mathcal{SFO}$，对任意 $x \in X$ 和随机样本 $\xi \in \Xi$，返回函数值 $F(x,\xi)$ 和随机次梯度 $G(x,\xi)$，满足：

$$\mathbb{E}_{\xi}[G(x,\xi)] = g(x) \in \partial f(x).$$

算法 5.2 随机逼近算法框架 $\mathcal{S}$

输入：$x_0 \in X$，初始信息集合 $I_{-1} = \emptyset$，对 $k = 0, 1, \cdots, N$ 执行迭代

1. 根据 ξ 的分布生成样本 ξ_k，
2. 在 ξ_k 处调用子程序 $\mathcal{SFO}$，更新信息集合 $I_{k+1} = I_k \cup (x_k, \xi_k, \mathcal{SFO}(x_k, \xi_k))$，
3. 根据信息集合 I_{k+1} 更新迭代点 x_{k+1}，

输出：$\bar{x} = \mathcal{S}(x_0)$.

集成随机逼近算法

利用机器学习中集成学习的概念，我们可以得到集成随机逼近算法

算法 5.3 集成随机逼近算法 $\mathcal{M}$

输入：$x_0 \in \mathbb{R}^n$，随机逼近基算法 $\mathcal{S}$，每个基算法迭代次数 N, 集成次数 K。

对 $j = 1, ..., K$ 执行迭代，

1. 调用随机逼近基算法 $\mathcal{S}$，得到近似解 $\bar{x}_j = \mathcal{S}(x_0)$.

输出：$\hat{x} = \mathcal{M}(x_0) = \frac{1}{K} \sum_{j=1}^{K} \bar{x}_j$.

定理

定理 14

令 N 和 K 按如下方式选取，

$$N = \left\lceil C_{\mathcal{S}}^{-1}\left(\frac{\epsilon V_f}{2}\right)\right\rceil, \quad K = \left\lceil \frac{2}{\epsilon^2}\ln\frac{1}{\delta}\right\rceil$$

则算法 *5.3* 的解 $\hat{x}$ 是随机优化问题 *5.1* 的 $(\epsilon V_f, \delta)$ 近似解，即，

$$\mathrm{Prob}\left\{f(\hat{x}) - f^* \geq \epsilon V_f\right\} \leq \delta$$

且算法 *5.3* 总共需要调用子程序 $\mathcal{SFO}$ 的次数，即计算复杂度 T 满足，

$$T = K \cdot N \leq \left(1 + C_{\mathcal{S}}^{-1}\left(\frac{\epsilon V_f}{2}\right)\right) \cdot \left(1 + \frac{2}{\epsilon^2}\ln\frac{1}{\delta}\right)$$

References I

王奇超, 文再文, 蓝光辉, 等. 优化算法的复杂度分析 [J]. 中国科学: 数学, 2020,50(09):1271- 1336.