优化算法复杂度分析（一）理论基础

Hailiang Zhao
http://hliangzhao.me

2022 年 9 月 24 日

问题分类

优化问题 $\mathcal{P}$:

$$f^* = \min_{x \in X} f(x) \tag{1}$$

可分类为:

1. **约束/无约束**: $X \subset \mathbb{R}^n / X \equiv \mathbb{R}^n$
2. **光滑/非光滑**: $f(x)$ 在 X 上可导/不可导
3. **凸/强凸/非凸**: $f(x)$ 是凸/强凸/非凸函数
4. **随机优化**: $f(x) = \mathbb{E}_\xi \big[F(x, \xi) \big]$

复杂度分析的问题模型

约定 $\mathcal{F} := \{\mathcal{P}\}$ 是具体问题 $\mathcal{P}$ 的集合，$\mathcal{S}$ 是待考察的数值解算法。

- **全局信息** Σ：$\mathcal{S}$ 所能获取的、$\mathcal{F}$ 中的共有特征信息（*e.g.*，目标函数是否光滑、可微、约束集合的类型、是否有界等）；
- **局部信息** $\mathcal{O}$：为了认识和求解 $\mathcal{P} \in \mathcal{F}$，$\mathcal{S}$ 需要逐步收集有关 $\mathcal{P}$ 的局部信息，然后根据这些信息给出寻找最优解的策略。这个过程被记为子程序 $\mathcal{O}$（*Oracle*）。例如，梯度法中求解 f 在给定点的导数的过程；
- **解的精度** $\mathcal{T}_\epsilon$：不同类型的 $\mathcal{F}$，其解的精度的度量方式不同。

由此，我们扩充问题集合 $\mathcal{F}$，得到复杂度分析理论中的问题模型 $\mathcal{F}$：

$$\mathcal{F} \equiv (\Sigma, \mathcal{O}, \mathcal{T}_\epsilon). \tag{2}$$

$\mathcal{S}$ 只能连续调用这三个部分来获得最优解的近似值。

全局信息 Σ

Σ 包含目标函数信息和约束集合信息。

目标函数若是凸函数，则可以做如下分类：设 $X \subset \mathbb{R}^n$ 是闭凸集合，$f: X \to \mathbb{R}^n$ 是凸函数，X^* 是(1)最优解的集合，x_* 是 x 在 X^* 上的投影。定义 f 的数个凸函数子集：

- $C(X)$（或记为 $C^0(X)$）：是所有连续函数的集合。
- $C_L(X)$（或记为 $C_L^{0,0}(X)$）：若 $f(x) \in C_L(X)$，则 $f(x)$ 具有 *Lipschitz* 连续性：

$$|f(x) - f(y)| \leq L\|x - y\|, \quad \forall x, y \in X. \tag{3}$$

- $C_L^{1,1}(X)$：若 $f(x) \in C_L^{1,1}(X)$，则 $f(x)$ 一阶可导且导数具有 *Lipschitz* 连续性：

$$|\nabla f(x) - \nabla f(y)| \leq L\|x - y\|, \quad \forall x, y \in X. \tag{4}$$

全局信息 Σ

- $C_L^{1,\alpha}(X)$：若 $f(x) \in C_L^{1,\alpha}(X)$，则 $f(x)$ 一阶可导且导数具有 *Hölder* 连续性，其中 $\alpha \in [0,1]$：

$$|\nabla f(x) - \nabla f(y)| \leq L\|x-y\|^{\alpha}, \quad \forall x, y \in X. \tag{5}$$

- $\mathcal{F}_{L,\mu}^{0,1}(X)$：是 $C_L(X)$ 中所有强凸函数组成的集合。即，若 $f(x) \in \mathcal{F}_{L,\mu}^{0,1}(X)$，则：

$$f(y) \geq f(x) + \langle \partial f(x), y-x \rangle + \frac{\mu}{2}\|y-x\|^2, \quad \forall x, y \in X. \tag{6}$$

- $\mathcal{F}_L^{1,1}(X)$：是 $C_L^{1,1}(X)$ 和凸函数集合的交集。

全局信息 Σ

- $\mathcal{F}_{L,\mu}^{1,1}(X)$：是 $C_L^{1,1}(X)$ 中所有强凸函数组成的集合。即，若 $f(x) \in \mathcal{F}_{L,\mu}^{1,1}(X)$，则：

$$f(y) \geq f(x) + \langle \partial f(x), y - x \rangle + \frac{\mu}{2}\|y - x\|^2, \quad \forall x, y \in X. \tag{7}$$

- $\mathcal{W}_{L,\mu}^{1,1}(X)$：是 $C_L^{1,1}(X)$ 中所有弱强凸函数组成的集合。即，若 $f(x) \in \mathcal{W}_{L,\mu}^{1,1}(X)$，则：

$$f^* \geq f(x) + \langle \nabla f(x), x_* - x \rangle + \frac{\mu}{2}\|x - x_*\|^2, \quad \forall x \in X. \tag{8}$$

- $\mathcal{S}_{L,\mu}^{1,1}(X)$：是 $C_L^{1,1}(X)$ 中所有具有二阶增长性的凸函数组成的集合。即，若 $f(x) \in \mathcal{S}_{L,\mu}^{1,1}(X)$，则：

$$f(x) - f^* \geq \frac{\mu}{2}\|x - x_*\|^2. \tag{9}$$

全局信息 Σ

显然有：$\mathcal{F}_{L,\mu}^{1,1}(X) \subset \mathcal{W}_{L,\mu}^{1,1}(X) \subset \mathcal{S}_{L,\mu}^{1,1}(X) \subset \mathcal{F}_{L}^{1,1}(X)$.

对于约束集合信息，若 X 是闭凸集合，可以在其上做投影，则(1)可以用投影梯度法求解。

更特殊地，若 X 是单纯形状或凸多面体时，即

$$X := \left\{ x \in \mathcal{R}^n : \sum_{i=1}^{n} x_n = 1, x_i \geq 0, i = 1, ..., n \right\},$$

则可以用条件梯度法求解(1)。

局部信息 $\mathcal{O}$

算法 $\mathcal{S}$ 通过子程序 $\mathcal{O}$ 来获取所求问题的局部信息。例如，$\forall x_0 \in X$，

1. **梯度法**的子程序返回函数值信息 $f(x_0)$ 和梯度信息 $\nabla f(x_0)$
2. **次梯度法**的子程序返回次梯度信息 $\partial f(x_0)$
3. **牛顿法**的子程序返回二阶导数信息 $\nabla^2 f(x_0)$

子程序 $\mathcal{O}$ 需要具备：

- **黑盒性**：$\mathcal{O}$ 是 $\mathcal{S}$ 获取局部信息的唯一来源
- **局部性**：对测试点 x 做微小扰动，$\mathcal{O}(x)$ 的变化不大（对 $\mathcal{S}$ 进行收敛性分析的关键假设）

常见的子程序 $\mathcal{O}$

- $\mathcal{ZO}$（无导数优化）：$\forall x_0 \in X$，返回 $f(x_0)$
- $\mathcal{FO}$（梯度法等）：$\forall x_0 \in X$，返回 $f(x_0)$、$\nabla f(x_0)$ 或 $\partial f(x_0)$
- 2nd$\mathcal{O}$（牛顿法）：$\forall x_0 \in X$，返回 $f(x_0)$、$\nabla f(x_0)$ 以及 $\nabla^2 f(x_0)$
- $\mathcal{SFO}$（随机梯度法等）：$\forall x_0 \in X$，返回函数值 $F(x_0, \xi_0)$ 和一阶随机梯度信息 $G(x_0, \xi_0)$
- $\mathcal{PO}$（投影梯度法）：$\forall x_0 \in \mathbb{R}^n$，返回 x_0 在 X 上的投影：
$$y \in \underset{x \in X}{\operatorname{argmin}} \|x_0 - x\|^2. \tag{10}$$
- $\mathcal{LO}$（条件梯度法）：当 X 是多面体时，给定 x_0 返回线性规划的解 $y \in \operatorname{argmin}_{x \in X} \langle x_0, x \rangle$
- $\mathcal{SO}$（椭球法）：若 X 是有界闭约束集合，则 $\forall c_0 \in \mathbb{R}^n$，若 $c_0 \in X$ 则返回真，否则返回一个向量 w 在 c_0 处形成的一个分割超平面：
$$w^\top (x - c_0) \le 0, \forall x \in X. \tag{11}$$

解的精度 $\mathcal{T}_\epsilon$

对于不同的问题，我们采用不同的解的精度来衡量算法的复杂度。

- **确定性优化问题**

$$f(x_k) - f^* \le \epsilon, \quad \frac{f(x_k) - f^*}{f(x_k)} \le \epsilon, \tag{12}$$

$$\|\nabla f(x_k)\| \le \epsilon, \quad \|x_k - x^*\| \le \epsilon. \tag{13}$$

- **随机性优化问题**

$$\mathbb{E}\Big[f(x_k) - f^*\Big] \le \epsilon, \quad \mathbb{E}\Big[\|\nabla f(x_R)\|^2\Big] \le \epsilon, \tag{14}$$

$$\mathbf{Pr}\Big\{f(x_k) - f^* \ge \epsilon\Big\} \le \delta, \quad \mathbf{Pr}\Big\{\|\nabla f(x_R)\|^2\Big\} \le \delta. \tag{15}$$

复杂度分析的算法模型

Algorithm 1: 抽象迭代算法 $\mathcal{S}$ 的运行框架（确定优化问题）

Input: $\epsilon > 0, x_0 \in X$，初始信息集合 $I_{-1} = \varnothing$

1 **for** $k = 0, 1, 2, \ldots$ **do**

2 　在 x_k 处调用子程序 $\mathcal{O}$，获得目标函数 $f(x)$ 和局部信息 $\mathcal{O}(x_k)$

3 　更新信息集合 $I_k = I_{k-1} \cup (x_k, \mathcal{O}(x_k))$

4 　应用 $\mathcal{S}$ 的规则处理 I_k 得到新的迭代点 x_{k+1}

5 　验证 x_k 是否满足停止条件 $\mathcal{T}_\epsilon$。若满足则输出 x_k；否则 $k \leftarrow k + 1$ 并转到步骤 2。

6 **end for**

Output: $\bar{x} = \mathcal{S}(x_0)$

对于随机优化问题，需要作出如下更改：

1. 步骤 2：调用子程序 $\mathcal{SFO}$ 得到局部信息 $\mathcal{SFO}(x_k, \xi_k)$
2. 步骤 3：更新信息集合 $I_k = I_{k-1} \cup (x_k, \xi_k, \mathcal{SFO}(x_k, \xi_k))$

复杂度分析的算法模型

在抽象迭代算法框架1中，新的迭代点 x_{k+1} 通过

$$x_{k+1} = F_k\Big(x_0, ..., x_k, \nabla f(x_0), ..., \nabla f(x_k), f(x_0), ..., f(x_k)\Big).$$

得到。即，每一个具体的 $\mathcal{S}$ 都对应着一组迭代规则函数

$$F := (F1, F2, ...). \tag{16}$$

我们将不同 F 的集合对应的解算法 $\mathcal{S}$ 的集合记做解算法集合 $\mathcal{M}$。例如

$$x_{k+1} = x_0 + \mathrm{span}\Big\{\nabla f(x_0), \nabla f(x_1), ..., \nabla f(x_k)\Big\}. \tag{17}$$

就对应一个解算法集合 $\mathcal{M}$（各种步长设定的梯度法的集合）。

复杂度分析的度量

- **分析复杂度**：$\mathcal{S}$ 将 $\mathcal{P}$ 求解到精度 ϵ 总共需要调用 $\mathcal{O}$ 的次数
- **算法复杂度**：$\mathcal{S}$ 将 $\mathcal{P}$ 求解到精度 ϵ 总共需要的算法操作（包含 $\mathcal{O}$ 内部的操作和 $\mathcal{S}$ 本身的操作）

我们主要关注分析复杂度。记 $\mathcal{S}$ 求解 $\mathcal{P}$ 的分析复杂度为 $N_{\mathcal{S}}(\mathcal{P},\epsilon)$，我们定义问题集合 $\mathcal{F}$ 的复杂度上界和下界：

- $\mathcal{F}$ 的复杂度上界：

$$\mathrm{Compl}_{\mathcal{S}}(\epsilon) := \sup_{\mathcal{P}\in\mathcal{F}} N_{\mathcal{S}}(\mathcal{P},\epsilon). \tag{18}$$

- $\mathcal{F}$ 的复杂度下界：

$$\mathrm{Compl}(\epsilon) := \inf_{\mathcal{S}\in\mathcal{M}} \mathrm{Compl}_{\mathcal{S}}(\epsilon) = \inf_{\mathcal{S}\in\mathcal{M}} \sup_{\mathcal{P}\in\mathcal{F}} N_{\mathcal{S}}(\mathcal{P},\epsilon). \tag{19}$$

为了 $\mathcal{F}$ 的复杂度下界，我们需要找到 $\mathcal{F}$ 中的一组病态问题，使得 $\mathcal{M}$ 中的算法的效率的都很低。

分析复杂度与收敛率的关系

我们可以从算法的收敛率中得到算法的分析复杂度：

- **次线性收敛率**：$f(x_k) - f^* \leq \frac{c}{\sqrt{k}}$，其中 c 为常数。令 $\frac{c}{\sqrt{k}} \leq \epsilon$，得到 $k \geq \frac{c^2}{\epsilon^2}$，因此分析复杂度为 $\mathcal{O}(\frac{1}{\epsilon^2})$。
- **线性收敛率**：$\|x_k - x^*\| \leq c(1-q)^k$，其中 c 为常数。同理可得到分析复杂度为 $\mathcal{O}(\ln\frac{1}{\epsilon})$。
- **二阶收敛率**：$\|x_{k+1} - x^*\| \leq c\|x_k - x^*\|^2$，其中 c 为常数。同理可得到分析复杂度为 $\mathcal{O}(\ln\ln\frac{1}{\epsilon})$。

建议读者自行进行推导。

算法复杂度表

我们将重心法记为 gravity，椭球法记为 ellipsoid，投影梯度法记为 PGD（Projected Gradient Method），加速梯度法记为 AGD（Accelerated Gradient Method），条件梯度法记为 CndG（Conditional Gradient Method），加速条件梯度法记为 CGS（Conditional Gradient Sliding Method）。

下表中的函数都是凸函数，$X \subseteq \mathbb{R}^n$ 是闭且凸的，且满足$\mathcal{B}(r) \subseteq X \subseteq \mathcal{B}(R)$。$Q = \frac{L}{\mu}$，其中 L 是梯度的 Lipschitz 常数，μ 是强凸函数对应的常数。

问题集合 $\mathcal{F}$	算法 $\mathcal{S}$	子程序 $\mathcal{O}$	收敛速率	分析复杂度
$C^0(X)$	gravity	$\mathcal{FO}+\mathcal{SO}$	$\exp(-\frac{k}{n})$	$n\log(\frac{B}{\epsilon})$
$C^0(X)$	ellipsoid	$\mathcal{FO}+\mathcal{SO}$	$\frac{R}{r}\exp(-\frac{k}{n^2})$	$n^2\log(\frac{BR}{r\epsilon})$
$C_L^{0,1}(X)$	PGD	$\mathcal{FO}+\mathcal{PO}$	$\frac{LR}{\sqrt{k}}$	$\frac{L^2R^2}{\epsilon^2}$

算法复杂度表

下表中的函数都是凸函数，$X \subseteq \mathbb{R}^n$ 是闭且凸的，且满足$\mathcal{B}(r) \subseteq X \subseteq \mathcal{B}(R)$。$Q = \frac{L}{\mu}$，其中 L 是梯度的 Lipschitz 常数，μ 是强凸函数对应的常数。

$\mathcal{F}_{L,\mu}^{0,1}(X)$	PGD	$\mathcal{FO}+\mathcal{PO}$	$\frac{L^2}{\mu k}$	$\frac{L^2}{\mu\epsilon}$
$C_L^{1,1}(X)$	PGD	$\mathcal{FO}+\mathcal{PO}$	$\frac{LR^2}{k}$	$\frac{LR^2}{\epsilon}$
$C_L^{1,1}(X)$	AGD	$\mathcal{FO}+\mathcal{PO}$	$\frac{LR^2}{k^2}$	$\frac{\sqrt{L}R}{\sqrt{\epsilon}}$
$C_L^{1,1}(X)$	CndG	$\mathcal{FO}+\mathcal{LO}$	$\frac{LR^2}{k}$	$\frac{LR^2}{\epsilon}$
$C_L^{1,1}(X)$	CGS	$\mathcal{FO}+\mathcal{LO}$	$\frac{LR^2}{k^2}$	$\mathcal{FO}: \sqrt{LR^2/\epsilon},\ \mathcal{LO}: \frac{LR^2}{\epsilon}$
$\mathcal{S}_{L,\mu}^{1,1}(X)$	PGD	$\mathcal{FO}+\mathcal{PO}$	$LR^2(\frac{Q}{Q+1})^k$	$\log(\frac{LR^2}{\epsilon})/\log(\frac{Q+1}{Q})$
$\mathcal{W}_{L,\mu}^{1,1}(X)$	PGD	$\mathcal{FO}+\mathcal{PO}$	$LR^2(\frac{Q-1}{Q+1})^k$	$\log(\frac{LR^2}{\epsilon})/\log(\frac{Q+1}{Q-1})$

算法复杂度表

下表中的函数都是凸函数，$X \subseteq \mathbb{R}^n$ 是闭且凸的，且满足$\mathcal{B}(r) \subseteq X \subseteq \mathcal{B}(R)$。$Q = \frac{L}{\mu}$，其中 L 是梯度的 Lipschitz 常数，μ 是强凸函数对应的常数。

$\mathcal{F}^{1,1}_{L,\mu}(X)$	PGD	$\mathcal{FO}+\mathcal{PO}$	$LR^2(\frac{Q-1}{Q+1})^{2k}$	$\log(\frac{LR^2}{\epsilon})/\log(\frac{Q+1}{Q-1})^2$
$\mathcal{F}^{1,1}_{L,\mu}(X)$	AGD	$\mathcal{FO}+\mathcal{PO}$	$LR^2(\frac{\sqrt{Q}-1}{\sqrt{Q}})^{k}$	$\log(\frac{LR^2}{\epsilon})/\log(\frac{\sqrt{Q}}{\sqrt{Q}-1})$
$\mathcal{F}^{1,1}_{L,\mu}(X)$	CndG	$\mathcal{FO}+\mathcal{LO}$	$\mu R/2^t$	$Q\log\left(\frac{\mu R}{\epsilon}\right)$
$\mathcal{F}^{1,1}_{L,\mu}(X)$	CGS	$\mathcal{FO}+\mathcal{LO}$	$\delta_0/2^t$	$\mathcal{FO}:\sqrt{Q}\log\frac{\delta_0}{\epsilon},\ \ \mathcal{LO}:\frac{LR^2}{\epsilon}$
$C^{1,\alpha}_L(\mathbb{R}^n)$	AGD	$\mathcal{FO}$	$\frac{2LR^{1+\alpha}}{(1+3\alpha)\log k}$	$\left(\frac{LR^{1+\alpha}}{\epsilon}\right)^{(2/1+3\alpha)}$