优化算法复杂度分析

Complexity Analysis for Optimization Algorithms

Hailiang Zhao

http://hliangzhao.me

2022 年 10 月 4 日

第一部分
理论基础

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**：$f(x) = \mathbb{E}_\xi \left[F(x, \xi) \right]$，$\xi$ 是随机变量

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** $\mathcal{T}_\epsilon$：用于衡量 $\mathcal{S}$ 取得的解和最优解之间的差距。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**。

我们首先来讨论目标函数信息。全局约定如下：**设 $X \subset \mathbb{R}^n$ 是闭凸集合，X^* 是问题(1)最优解的集合，x_* 是 x 在 X^* 上的投影。**

我们将具体问题按照目标函数的特征进行划分：

- $C(X)$（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 $C_L(X)$ 和强凸函数集合的交集。即，若 $f(x) \in \mathcal{F}_{L,\mu}^{0,1}(X)$，则还有：

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

- $\mathcal{F}_L^{1,1}(X)$：是 $C_L^{1,1}(X)$ 和凸函数集合的交集。
- $\mathcal{S}_{L,\mu}^{1,1}(X)$：是 $C_L^{1,1}(X)$ 和具有二阶增长性的凸函数集合的交集。即，若 $f(x) \in \mathcal{S}_{L,\mu}^{1,1}(X)$，则还有：

$$f(x) - f^* \geq \frac{\mu}{2}\|x - x_*\|^2. \tag{7}$$

全局信息 Σ

- $\mathcal{W}_{L,\mu}^{1,1}(X)$：是 $C_L^{1,1}(X)$ 和弱强凸函数集合的交集。即，若 $f(x) \in \mathcal{W}_{L,\mu}^{1,1}(X)$，则还有：

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

- $\mathcal{F}_{L,\mu}^{1,1}(X)$：是 $C_L^{1,1}(X)$ 和强凸函数集合的交集。即，若 $f(x) \in \mathcal{F}_{L,\mu}^{1,1}(X)$，则还有：

$$f(y) \geq f(x) + \langle \partial f(x), y - x \rangle + \frac{\mu}{2}\|y - x\|^2, \quad \forall x, y \in X. \tag{9}$$

显然有：

$$\mathcal{F}_{L,\mu}^{1,1}(X) \subset \mathcal{W}_{L,\mu}^{1,1}(X) \subset \mathcal{S}_{L,\mu}^{1,1}(X) \subset \mathcal{F}_L^{1,1}(X). \tag{10}$$

全局信息 Σ

对于约束集合信息，若 X 是闭凸集合，则问题(1)可以用**投影梯度法**求解。

更特殊地，若 X 是单纯形或凸多面体时，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**求解问题(1)。

我们会在后文解释为什么对于这类约束集合这些方法取得更好的收敛速度和更低的复杂度。

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（这是对 $\mathcal{S}$ 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 $\partial f(x_0)$
- $\mathrm{2nd}\mathcal{O}$（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：

$$y \in \operatorname*{argmin}_{x \in X} \|x_0 - x\|^2.$$

- $\mathcal{LO}$（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：$w^\top (x - c_0) \leq 0, \forall x \in X$。

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**

$$\text{（适用于凸函数）}\ f(x_k) - f^* \le \epsilon, \quad \frac{f(x_k) - f^*}{f(x_k)} \le \epsilon, \tag{11}$$

$$\text{（适用于非凸函数）}\ \|\nabla f(x_k)\| \le \epsilon, \quad \|x_k - x^*\| \le \epsilon. \tag{12}$$

- **随机性优化问题**：ϵ 解和 (ϵ, δ) 解

$$\mathbb{E}\Big[f(x_k) - f^*\Big] \le \epsilon, \quad \mathbb{E}\Big[\|\nabla f(x_R)\|^2\Big] \le \epsilon, \tag{13}$$

$$\mathbf{Pr}\Big\{f(x_k) - f^* \ge \epsilon\Big\} \le \delta, \quad \mathbf{Pr}\Big\{\|\nabla f(x_R)\|^2\Big\} \le \delta. \tag{14}$$

抽象迭代算法的运行框架

对于确定优化问题，我们有如下算法框架：

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

Input: $\epsilon > 0, x_0 \in X$，初始信息集合 $I_{-1} = \varnothing$
for $k = 0, 1, 2, \ldots$ **do**
  在 x_k 处调用子程序 $\mathcal{O}$，获得目标函数 $f(x)$ 和局部信息 $\mathcal{O}(x_k)$
  更新信息集合 $I_k = I_{k-1} \cup (x_k, \mathcal{O}(x_k))$
  应用 $\mathcal{S}$ 的规则处理 I_k 得到新的迭代点 x_{k+1}
  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。
end for
Output: 最终步的迭代点 $\bar{x} = \mathcal{S}(x_0)$

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

在上述抽象迭代算法框架中，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**

$$F := (F_1, F_2, ...). \tag{15}$$

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

$$x_{k+1} = x_0 + \text{span}\Big\{\nabla f(x_0), \nabla f(x_1), ..., \nabla f(x_k)\Big\}. \tag{16}$$

就对应着**一阶算法的集合**。

复杂度的度量

- **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：

$$\mathrm{Compl}_{\mathcal{S}}(\epsilon) := \sup_{\mathcal{P}\in\mathcal{F}} N_{\mathcal{S}}(\mathcal{P}, \epsilon). \tag{17}$$

- $\mathcal{F}$ 的复杂度下界：

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

为了得到问题集合 $\mathcal{F}$ 的复杂度上界，我们需要找到一个解算法 $\mathcal{S}$，使得 $\mathcal{F}$ 中的所有问题 $\mathcal{P}$ 都可以被其求解。对于 $\mathcal{F}$ 的复杂度下界，我们需要找到 $\mathcal{F}$ 中的一类病态问题，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 $\mathcal{O}(\ln \ln \frac{1}{\epsilon})$。

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，$\mathcal{B}(\cdot)$ 是范数球。

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，$\mathcal{B}(\cdot)$ 是范数球。

$\mathcal{F}_{L,\mu}^{0,1}(X)$	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，$\mathcal{B}(\cdot)$ 是范数球。

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

从下面开始，我们将依次分析各种算法适用的问题模型以及对应的复杂度。

第二部分

光滑优化问题的复杂度分析

纲要

1. 光滑优化问题与光滑函数子集合
2. 梯度法
   - 面向（无约束）光滑非凸函数
   - 面向（无约束）光滑凸/具有二阶增长性的凸/弱强凸/强凸函数
3. 牛顿法
4. （无约束）光滑凸/强凸优化问题的复杂度下界
5. Nesterov 加速梯度算法
   - 面向光滑凸函数（默认版本）
   - 面向光滑非凸函数（需改进）

光滑优化问题

我们将给出优化问题

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

的复杂度分析，其中 $f(x)$ 是光滑的。

首先，我们将定义一些光滑函数的子集合，并列出它们的一些性质。

光滑函数子集合 $C_L^{k,p}(X)$

$\forall X \in \mathbb{R}^n$:

- 定义 $C_L^{k,p}(X)$ 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):

$$\|f^{(p)}(x) - f^{(p)}(y)\| \leq L\|x - y\|, \forall x, y \in X. \tag{19}$$

- 定义 $\mathcal{F}_L^{k,p}(X)$ 为 $C_L^{k,p}(X)$ 和凸函数集合的交集。

光滑函数子集合的性质

性质 2.1: 若 $f_1 \in C_{L_1}^{k,p}(X), f_2 \in C_{L_2}^{k,p}(X)$，且 $\alpha, \beta \in \mathbb{R}$，则对 $L_3 = |\alpha|L_1 + |\beta|L_2$ 有 $\alpha f_1 + \beta f_2 \in C_{L_3}^{k,p}(X)$.

性质 2.2: $f \in C_L^{2,1}(\mathbb{R}^n) \subset C_L^{1,1}(\mathbb{R}^n)$ *iff* $\forall x \in \mathbb{R}^n \left[\|\nabla^2 f(x)\| \leq L \right]$。

性质 2.3: 若 $f \in C_L^{1,1}(\mathbb{R}^n)$，则 $\forall x, y \in \mathbb{R}^n$，有

$$\left| f(y) - f(x) - \langle \nabla f(x), y - x \rangle \right| \leq \frac{L}{2} \|y - x\|^2. \tag{20}$$

性质 2.4: 若 $f \in C_L^{2,2}(\mathbb{R}^n)$，则 $\forall x, y \in \mathbb{R}^n$，$\exists M > 0$满足

$$\left\| \nabla f(y) - \nabla f(x) - \langle \nabla^2 f(x), y - x \rangle \right\| \leq \frac{M}{2} \|y - x\|^2, \tag{21}$$

$$\left| f(y) - f(x) - \langle \nabla f(x), y - x \rangle - \frac{1}{2} \langle \nabla^2 f(x)(y - x), y - x \rangle \right| \leq \frac{M}{6} \|y - x\|^3. \tag{22}$$

光滑函数子集合的性质

性质 2.5：令 $f \in C_L^{2,2}(\mathbb{R}^n)$ 且 $\|y-x\| = r$ 则$\exists M > 0$满足

$$\nabla^2 f(x) - MrI_n \preceq \nabla^2 f(y) \preceq \nabla^2 f(x) + MrI_n. \quad (I_n\text{是单位阵}) \tag{23}$$

性质 2.6：若 $f \in \mathcal{F}_L^{1,1}(\mathbb{R}^n)$，则 $\forall x, y \in \mathbb{R}^n, \alpha \in [0,1]$ 有

$$0 \le f(y) - f(x) - \langle \nabla f(x), y-x \rangle \le \frac{L}{2}\|y-x\|^2, \tag{24}$$

$$f(x) + \langle \nabla f(x), y-x \rangle + \frac{1}{2L}\|\nabla f(x) - \nabla f(y)\|^2 \le f(y), \tag{25}$$

$$\frac{1}{L}\|\nabla f(y) - \nabla f(x)\|^2 \le \langle \nabla f(x) - \nabla f(y), x-y \rangle \le L\|x-y\|^2, \tag{26}$$

$$\begin{aligned} \alpha f(x) + (1-\alpha) f(y) &\le f(\alpha x + (1-\alpha)y) \\ &\quad + \frac{\alpha(1-\alpha)}{2L}\|\nabla f(x) - \nabla f(y)\|^2, \end{aligned} \tag{27}$$

$$\begin{aligned} \alpha f(x) + (1-\alpha) f(y) &\le f(\alpha x + (1-\alpha)y) \\ &\quad + \alpha(1-\alpha)\frac{L}{2}\|x-y\|^2. \end{aligned} \tag{28}$$

梯度法

接下来我们分析不同算法在不同光滑函数子集合中的收敛性和复杂度，并给出不同光滑函数子集合的复杂度上界。

我们首先考虑梯度法在求解

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

时的复杂度（即 $X=\mathbb{R}^n$）。

在梯度法中，新的迭代点通过如下方式得到：

$$x_{k+1} = x_k - h_k \nabla f(x_k), \tag{29}$$

其中 h_k 为步长。

梯度法

梯度法的步长 h_k 有三种选择方式：

- **固定步长和变步长策略**：$h_k = h > 0$ 或 $h_k = \frac{h}{\sqrt{k+1}}$；
- **精确先搜索步长法**：$h_k = \operatorname{argmin}_{h \geq 0} f\big(x_k - h\nabla f(x_k)\big)$；
- Goldenstein-Armijo **准则**：令 $x_{k+1} = x_k - h_k \nabla f(x_k)$，寻找满足以下不等式的 h_k：

$$\begin{aligned} \alpha\langle \nabla f(x_k), x_k - x_{k+1}\rangle &\leq f(x_k) - f(x_{k+1}) \\ \beta\langle \nabla f(x_k), x_k - x_{k+1}\rangle &\geq f(x_k) - f(x_{k+1}), \end{aligned}$$

其中 $0 < \alpha < \beta < 1$。

梯度法求解问题集合 $C_L^{1,1}(\mathbb{R}^n)$

问题模型 2.1

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

- 全局信息 Σ: 目标函数 $f \in C_L^{1,1}(\mathbb{R}^n)$, 约束集合 $X \equiv \mathbb{R}^n$, 且 $f(x)$ 不一定是凸函数; $f(x)$ 有下界, 即

$$\exists M \in \mathbb{R} \Big[\forall x \in X, f(x) \geq M \Big]. \tag{30}$$

- 局部信息 $\mathcal{O}$: $\mathcal{FO}$ 子程序, 对于任意给定的 x_0 返回 $f(x_0)$ 和 $\nabla f(x_0)$;
- 解的精度 $\mathcal{T}_\epsilon$: 求**局部**极小值的近似解 $\bar{x} \in \mathbb{R}^n$, 使得 $\|\nabla f(\bar{x})\| \leq \epsilon$。

梯度法求解问题集合 $C_L^{1,1}(\mathbb{R}^n)$

定理 2.1

使用梯度法求解问题模型 *2.1* 时，取 $\bar{x}$ 满足 $\|\nabla f(\bar{x})\| = \min_{0\le k\le N}\|\nabla f(x_k)\|$，则算法的收敛速度为：

$$\|\nabla f(\bar{x})\| \le \frac{1}{\sqrt{N+1}}\left[\frac{L}{\omega}\big(f(x_0)-f^*\big)\right]^{\frac{1}{2}}. \tag{31}$$

复杂度上界为

$$N(\epsilon) \le \frac{L(f(x_0)-f^*)}{\omega\epsilon^2}. \tag{32}$$

证明.

结合性质 *2.3*，可以发现三种步长设定方式均可得到 $f(x_k)-f(x_{k+1}) \ge \frac{\omega}{L}\|\nabla f(x_k)\|^2$。将其对所有的迭代步累加即可。 □

这意味着，梯度法求解 $C_L^{1,1}(\mathbb{R}^n)$ 时，表现为**全局次线性收敛**，复杂度上界为 $\mathcal{O}\big(\frac{1}{\epsilon^2}\big)$。

梯度法求解问题集合 $C_L^{2,2}(\mathbb{R}^n)$

问题模型 2.2

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

- 全局信息 Σ:
  1. 目标函数 $f \in C_L^{2,2}(\mathbb{R}^n)$，约束集合 $X \equiv \mathbb{R}^n$，且 $f(x)$ 不一定是凸函数;
  2. $f(x)$ 存在局部极小值点 x^*，且 $\nabla^2 f(x^*)$ 正定;
  3. $\exists 0 < m \leq M < \infty \left[mI_n \preceq \nabla^2 f(x^*) \preceq MI_n \right]$;
  4. 初始点 x_0 距离 x^* 足够近;
- 局部信息 $\mathcal{O}$: $\mathcal{FO}$ 子程序，对于任意给定的 x_0 返回 $f(x_0)$ 和 $\nabla f(x_0)$;
- 解的精度 $\mathcal{T}_\epsilon$: 求**局部**极小值的近似解 $\bar{x} \in \mathbb{R}^n$，使得 $\|\nabla f(\bar{x})\| \leq \epsilon$。

梯度法求解问题集合 $C_L^{2,2}(\mathbb{R}^n)$

定理 2.2

假设梯度法的初始点 x_0 距离极小值点 x^* 足够近，满足

$$r_0 = \|x_0 - x^*\| < \bar{r} = \frac{2m}{L}, \tag{33}$$

且步长取 $h_k = \frac{2}{m+M}$，则梯度法求解问题模型 *2.2* 的收敛速率为：

$$\|x_k - x^*\| \leq \frac{\bar{r} r_0}{\bar{r} - r_0}\left(1 - \frac{2m}{M+3m}\right)^k. \tag{34}$$

复杂度上界为

$$\frac{M+3m}{2m}\left[\ln\left(\frac{\bar{r} r_0}{\bar{r} - r_0}\right) + \ln\frac{1}{\epsilon}\right]. \tag{35}$$

这意味着，梯度法求解 $C_L^{2,2}(\mathbb{R}^n)$ 时，表现为**局部线性收敛**，复杂度上界为 $\mathcal{O}\left(\ln\frac{1}{\epsilon}\right)$。

梯度法求解问题集合 $\mathcal{F}_L^{1,1}(\mathbb{R}^n)$

问题模型 2.3

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

- 全局信息 Σ：$f \in \mathcal{F}_L^{1,1}(\mathbb{R}^n)$，或 $f \in \mathcal{F}_{L,\mu}^{1,1}(\mathbb{R}^n)$，或 $f \in \mathcal{W}_{L,\mu}^{1,1}(\mathbb{R}^n)$，或 $f \in \mathcal{S}_{L,\mu}^{1,1}(\mathbb{R}^n)$。分别是**凸**函数集合、**强凸**函数集合、**弱强凸**函数集合和**二阶增长类**函数集合；
- 局部信息 $\mathcal{O}$：$\mathcal{FO}$ 子程序（或 $\mathcal{PO}$ 子程序）；
- 解的精度 $\mathcal{T}_\epsilon$：求**全局**极小值的近似解 $\bar{x} \in \mathbb{R}^n$，使得 $f(\bar{x}) - f^* \leq \epsilon$ 或 $\|\bar{x} - x^*\| \leq \epsilon$。

梯度法求解问题集合 $\mathcal{F}_L^{1,1}(\mathbb{R}^n)$

定理 2.3

若 $f \in \mathcal{F}_L^{1,1}(\mathbb{R}^n)$，令步长 $h_k \equiv h = \frac{2}{L}$，则梯度法求解问题模型 *2.3* 的收敛速率为：

$$f(x_k) - f^* \leq \frac{2L\|x_0 - x^*\|^2}{k+4}. \tag{36}$$

记 $D_0 = \|x_0 - x^*\|$，则复杂度上界为

$$\mathcal{O}\Big(\frac{LD_0}{\epsilon}\Big). \tag{37}$$

这意味着，梯度法求解 $\mathcal{F}_L^{1,1}(\mathbb{R}^n)$ 时，表现为**全局次线性收敛**，复杂度上界为 $\mathcal{O}\big(\frac{1}{\epsilon}\big)$。

梯度法求解问题集合 $\mathcal{F}_{L,\mu}^{1,1}(\mathbb{R}^n)$

定理 2.4

若 $f \in \mathcal{F}_{L,\mu}^{1,1}(\mathbb{R}^n)$，令步长 $h_k \equiv h = \frac{1}{L}$，则梯度法求解问题模型 *2.3* 的收敛速率为：

$$f(x_k) - f^* \leq \frac{L}{2}\left(\frac{Q-1}{Q+1}\right)^{2k} \|x_0 - x^*\|^2, \tag{38}$$

$$\|x_k - x^*\|^2 \leq \left(\frac{Q-1}{Q+1}\right)^{2k} \|x_0 - x^*\|^2. \tag{39}$$

其中 $Q = \frac{L}{\mu}$。复杂度上界为

$$\frac{\log\left(\frac{LD_0^2}{\epsilon}\right)}{\log\left(\frac{Q-1}{Q+1}\right)^2}. \tag{40}$$

这意味着，梯度法求解 $\mathcal{W}_{L,\mu}^{1,1}(\mathbb{R}^n)$ 时，表现为**全局线性收敛**，复杂度上界为 $\mathcal{O}\left(\ln \frac{1}{\epsilon}\right)$。

梯度法求解问题集合 $\mathcal{W}_{L,\mu}^{1,1}(\mathbb{R}^n)$

定理 2.5

若 $f \in \mathcal{W}_{L,\mu}^{1,1}(\mathbb{R}^n)$，令步长 $h_k \equiv h = \frac{1}{L}$，则梯度法求解问题模型 *2.3* 的收敛速率为：

$$f(x_k) - f^* \leq \frac{L}{2}\left(\frac{Q-1}{Q+1}\right)^{k-1} \|x_0 - \bar{x}^k\|^2, \tag{41}$$

$$\|x_k - \bar{x}^k\|^2 \leq \left(\frac{Q-1}{Q+1}\right)^{k} \|x_0 - \bar{x}^k\|^2. \tag{42}$$

其中 $\bar{x}^k = \operatorname{argmin}_{k' \in \{1,\ldots,k\}} f(x_{k'})$。目标函数值对应的复杂度上界为

$$\frac{\log\left(\frac{LD_0^2}{\epsilon}\right)}{\log\left(\frac{Q-1}{Q+1}\right)}. \tag{43}$$

这意味着，梯度法求解 $\mathcal{W}_{L,\mu}^{1,1}(\mathbb{R}^n)$ 时，表现为**全局线性收敛**，复杂度上界为 $\mathcal{O}\left(\ln \frac{1}{\epsilon}\right)$。

梯度法求解问题集合 $\mathcal{S}_{L,\mu}^{1,1}(\mathbb{R}^n)$

定理 2.6

若 $f \in \mathcal{S}_{L,\mu}^{1,1}(\mathbb{R}^n)$，令步长 $h_k \equiv h = \frac{1}{L}$，则梯度法求解问题模型 *2.3* 的收敛速率为**全局线性收敛**：

$$f(x_k) - f^* \leq \frac{L}{2}\left(\frac{Q}{Q+1}\right)^{k-1} \|x_0 - \bar{x}^k\|^2, \tag{44}$$

$$\|x_k - \bar{x}^k\|^2 \leq \left(\frac{Q}{Q+1}\right)^{k} \|x_0 - \bar{x}^k\|^2. \tag{45}$$

目标函数值对应的复杂度上界为

$$\frac{\log\left(\frac{LD_0^2}{\epsilon}\right)}{\log\left(\frac{Q}{Q+1}\right)}. \tag{46}$$

这意味着，梯度法求解 $\mathcal{W}_{L,\mu}^{1,1}(\mathbb{R}^n)$ 时，表现为**全局线性收敛**，复杂度上界为 $\mathcal{O}\left(\ln \frac{1}{\epsilon}\right)$。

梯度法求解无约束凸优化问题的复杂度上界

对比定理 2.4、定理 2.5 和定理 2.6，可以发现，使用梯度法求解问题集合 $\mathcal{F}_{L,\mu}^{1,1,}(\mathbb{R}^n)$，$\mathcal{W}_{L,\mu}^{1,1,}(\mathbb{R}^n)$，$\mathcal{S}_{L,\mu}^{1,1,}(\mathbb{R}^n)$ 的复杂度上界依次递减。

随着目标函数 f 凸性的增强，梯度法的收敛速度虽然都是全局线性收敛，但是复杂度上界反而提升了。因此，**我们应当充分利用这种增强了的凸性以设计更好的方法来降低复杂度上界**。牛顿法就是一种典型的改进。

牛顿法

在牛顿法中，新的迭代点通过如下方式得到：

$$x_{k+1} \leftarrow x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k). \tag{47}$$

问题模型 2.4

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

- 全局信息 Σ：
  1. 目标函数 $f \in C_L^{2,2}(\mathbb{R}^n)$，约束集合 $X \equiv \mathbb{R}^n$，且 $f(x)$ 不一定是凸函数；
  2. $f(x)$ 存在局部极小值点 x^*，且 $\nabla^2 f(x^*)$ 正定；
  3. $\exists m > 0 \left[m I_n \preceq \nabla^2 f(x^*) \right]$；
  4. 初始点 x_0 距离 x^* 足够近；
- 局部信息 $\mathcal{O}$：$2nd\mathcal{O}$ 子程序，对于任意给定的 x_0 返回 $f(x_0)$、$\nabla f(x_0)$ 和 $\nabla^2 f(x_0)$；
- 解的精度 $\mathcal{T}_\epsilon$：求**局部**极小值的近似解 $\bar{x} \in \mathbb{R}^n$，使得 $\|\bar{x} - x^*\| \leq \epsilon$。

牛顿法求解问题集合 $C_L^{2,2}(\mathbb{R}^n)$

定理 2.7

假设牛顿法的初始点 x_0 距离极小值点 x^* 足够近，满足

$$\|x_0 - x^*\| < \bar{r} = \frac{3m}{L}, \tag{48}$$

则对任意的 k 满足

$$\|x_k - x^*\| \leq \bar{r}. \tag{49}$$

牛顿法求解问题模型 *2.4* 的收敛速率为：

$$\|x_{k+1} - x^*\| \leq \frac{L\|x_k - x^*\|^2}{2(m - L\|x_k - x^*\|)}. \tag{50}$$

复杂度上界为 $c \ln\ln \frac{\gamma}{\epsilon}$，其中 c 和 γ 为常数。

这意味着，牛顿法求解 $C_L^{2,2}(\mathbb{R}^n)$ 时，表现为**局部二阶收敛**，复杂度上界为 $\mathcal{O}\left(\ln\ln \frac{1}{\epsilon}\right)$。

总结

以上我们讨论了**梯度法**在求解

1. 无约束非凸优化问题集合 $C_L^{1,1}(\mathbb{R}^n)$ 和 $C_L^{2,2}(\mathbb{R}^n)$
2. 无约束凸优化问题集合 $\mathcal{F}_L^{1,1}(\mathbb{R}^n)$、强凸优化问题集合 $\mathcal{F}_{L,\mu}^{1,1}(\mathbb{R}^n)$、弱强凸优化问题集合 $\mathcal{W}_{L,\mu}^{1,1}(\mathbb{R}^n)$ 和具有二阶增长性质的凸优化问题集合 $\mathcal{S}_{L,\mu}^{1,1}(\mathbb{R}^n)$

时的收敛速度和复杂度上界。此外，我们还给出了**牛顿法**在求解无约束非凸优化问题集合 $C_L^{2,2}(\mathbb{R}^n)$ 的收敛速度和复杂度上界。

接下来，我们给出无约束光滑凸优化问题和无约束光滑强凸优化问题的复杂度下界。

无约束光滑凸优化问题的复杂度下界

定理 2.8

对任意 $x_0 \in \mathbb{R}^n$ 和 $1 \leq k \leq \frac{1}{2}(n-1)$，存在 $f \in C_L^{\infty,1}(\mathbb{R}^n)$，使得对任意的 $x_k \in x_0 + span\{\nabla f(x_0), ..., \nabla f(x_{k-1})\}$（即一阶算法，只利用梯度信息的算法）都有

$$f(x_k) - f^* \geq \frac{3L\|x_0 - x^*\|^2}{32(k+1)^2}. \tag{51}$$

令 $D_0 = \|x_0 - x^*\|$，令上式右端等于 ϵ，可以得到一阶算法求解光滑凸优化问题的复杂度下界为

$$\mathcal{O}\left(\sqrt{\frac{L}{\epsilon}}D_0\right). \tag{52}$$

定理 2.3 告诉我们，梯度法求解非光滑凸优化问题的复杂度上界为 $\mathcal{O}(\frac{LD_0}{\epsilon})$，相比之下可以发现梯度法并非最优算法。求解某一类问题集合的最优算法应当让其复杂度上界和下界尽可能接近。

无约束光滑强凸优化问题的复杂度下界

定理 2.9

对任意 $x_0 \in \mathbb{R}^\infty, \mu, L > 0$ 和 $Q = \frac{L}{\mu} > 1$，存在 $f \in C^{\infty,1}_{L,\mu}(\mathbb{R}^\infty)$，使得对任意 $x_k \in x_0 + span\{\nabla f(x_0), ..., \nabla f(x_{k-1})\}$ 都有

$$f(x_k) - f^* \geq \frac{\mu}{2}\left(\frac{\sqrt{Q}-1}{\sqrt{Q}+1}\right)^{2k} \|x_0 - x^*\|^2. \tag{53}$$

令上式右端等于 ϵ，可以得到一阶算法求解光滑强凸优化问题的复杂度下界为

$$\mathcal{O}\left(\sqrt{\frac{L}{\mu}} \max\left(\log \frac{\mu D_0}{\epsilon}, 1\right)\right). \tag{54}$$

Nesterov 加速梯度算法框架

定理 2.8 告诉我们，函数集合 $C_L^{\infty,1}(\mathbb{R}^n)$ 相对于解算法集合 $x_k \in x_0 + \text{span}\{\nabla f(x_0), ..., \nabla f(x_{k-1})\}$ 的复杂度下界为 $\mathcal{O}(1/\sqrt{\epsilon})$，复杂度上界为 $\mathcal{O}(1/\epsilon)$。这意味着梯度法相对于该解算法集合而言并不是最优的。接下来我们介绍 Nesterov 加速梯度法。

Algorithm 2: Nesterov 加速梯度算法框架

Input: 令 $x_0 = y_0$，选取序列 $\{\gamma_k\}$ 和 $\{\beta_k\}$ 满足 $L\gamma_k \leq \beta_k$，且 $\gamma_1 = 1$

for $k = 1, 2, ..., N$ **do**

$z_k \leftarrow (1 - \gamma_k)y_{k-1} + \gamma_k x_{k-1}$

$x_k \leftarrow \text{argmin}_{x \in X}\{\langle \nabla f(z_k), x\rangle + \frac{\beta_k}{2}\|x - x_{k-1}\|_2^2\}$

$y_k \leftarrow (1 - \gamma_k)y_{k-1} + \gamma_k x_k$

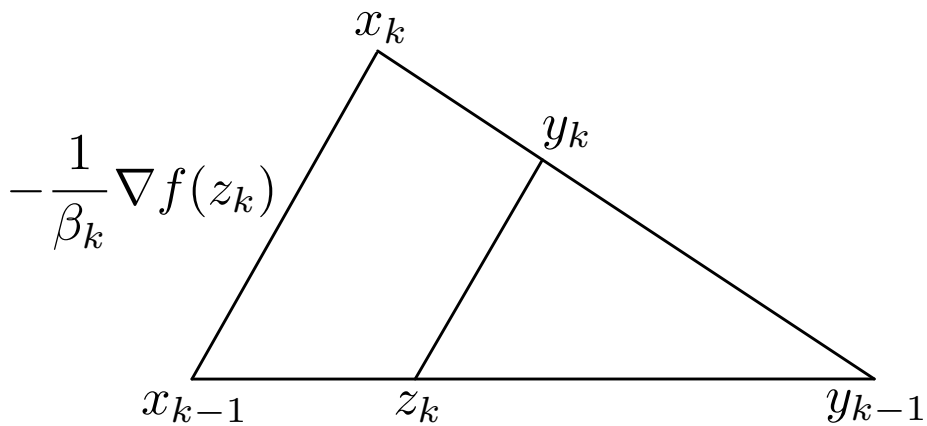
end for

Output: y_N

该算法中，收敛点列为 $\{y_k\}$。若 $X \equiv \mathbb{R}^n$，则第二步等价于 $x_k = x_{k-1} - \frac{1}{\beta_k}\nabla f(z_k)$（直接代入求出最优的 x）。

Nesterov 加速梯度算法框架

该算法中，收敛点列为 $\{y_k\}$。若 $X \equiv \mathbb{R}^n$，则第二步等价于 $x_k = x_{k-1} - \frac{1}{\beta_k}\nabla f(z_k)$（直接代入求出最优的 x）。此时，x_k、y_k 和 z_k 的迭代关系如下图所示：



注意到，这两个三角形是相似的，且相似比为 γ_k。

Nesterov 加速梯度算法的分析方法

选取不同的参数 γ_k 和 β_k，可以得到不同的收敛速度。接下来，我们通过构造序列 $\{\Gamma_k\}$ 来分析收敛性。

引理 2.1

令 $\gamma_t \in (0,1], t = 1, 2, \ldots,$ 构造序列

$$\Gamma_t = \begin{cases} 1 & t = 1 \\ (1-\gamma_t)\Gamma_{t-1} & t \geq 2. \end{cases} \tag{55}$$

若序列 $\{\Delta_t\}_{t\geq 0}$ 满足

$$\Delta_t \leq (1-\gamma_t)\Delta_{t-1} + B_t, \quad t = 1, 2, \ldots, \tag{56}$$

则 $\forall k$ 有

$$\Delta_k \leq \Gamma_k(1-\gamma_1)\Delta_0 + \Gamma_k \sum_{t=1}^{k} \frac{B_t}{\Gamma_t}. \tag{57}$$

Nesterov 加速梯度算法的分析方法

一般来说，我们会令 $\Delta_k = f(x_k) - f(x^*)$ 或 $\Delta_k = \|x_k - x^*\|_2^2$，由此，(56)变为

$$f(x_k) - f(x^*) \leq (1 - \gamma_k)\Big(f(x_{k-1}) - f(x^*)\Big) + B_k \tag{58}$$

或者

$$\|x_k - x^*\|_2^2 \leq (1 - \gamma_k)\|x_k - x^*\|_2^2 + B_k. \tag{59}$$

通常我们会构造序列 $\{\gamma_k\}$ 使其满足 $\gamma_1 = 1$，由此(57)变成

$$f(x_k) - f(x^*) \leq \Gamma_k \sum_{t=1}^{k} \frac{B_t}{\Gamma_t} \tag{60}$$

或

$$\|x_k - x^*\|_2^2 \leq \Gamma_k \sum_{t=1}^{k} \frac{B_t}{\Gamma_t}. \tag{61}$$

Nesterov 加速梯度算法的分析方法

可以发现，加速梯度法的收敛速度与 B_k 和 Γ_k 有关。**对于 B_k 我们通过放缩估计其上界，对于 γ_k 我们可以采用不同的构造方式**。例如，

$$\begin{array}{lll} \gamma_k = \frac{1}{k} & \Rightarrow & \Gamma_k = \frac{1}{k} \\ \gamma_k = \frac{1}{k+1} & \Rightarrow & \Gamma_k = \frac{2}{k(k+1)} \\ \gamma_k = \frac{3}{k+2} & \Rightarrow & \Gamma_k = \frac{6}{k(k+1)(k+2)} \end{array} \tag{62}$$

令

$$D_X = \sup_{x,y \in X} \|x - y\|,$$

下面给出加速梯度算法在取不同 β_k 和 γ_k 序列时的收敛速度。

Nesterov 加速梯度算法求解问题集合 $\mathcal{F}_L^{1,1}(\mathbb{R}^n)$

Nesterov 加速梯度算法在面对无约束凸优化问题集合时，具有如下收敛性结论：

定理 2.10 *

若 $f \in \mathcal{F}_L^{1,1}(\mathbb{R}^n)$，则 *Nesterov* 加速梯度算法求解相应的无约束凸优化问题的收敛速度为：

- 取 $\beta_k = L, \gamma_k = \frac{1}{k}$，则 $\Gamma_k = \frac{1}{k}, \frac{\beta_k \gamma_k}{\Gamma_k} = L$，我们有

$$f(y_k) - f(x^*) \leq \frac{L}{2k} D_X^2, \quad f(y_k) - f(x^*) \leq \frac{L}{2k} \|x_0 - x^*\|^2. \quad (63)$$

Nesterov 加速梯度算法求解问题集合 $\mathcal{F}_L^{1,1}(\mathbb{R}^n)$

定理 2.10 *（续）

若 $f \in \mathcal{F}_L^{1,1}(\mathbb{R}^n)$，则 *Nesterov* 加速梯度算法求解相应的无约束凸优化问题的收敛速度为：

- 取 $\beta_k = \frac{2L}{k}, \gamma_k = \frac{2}{k+1}$，则 $\Gamma_k = \frac{2}{k(k+1)}, \frac{\beta_k \gamma_k}{\Gamma_k} = 2L$，我们有

$$f(y_k) - f(x^*) \leq \frac{2L}{k(k+1)} D_X^2, \tag{64}$$

$$f(y_k) - f(x^*) \leq \frac{4L}{k(k+1)} \|x_0 - x^*\|^2. \tag{65}$$

- 取 $\beta_k = \frac{3L}{k+1}, \gamma_k = \frac{3}{k+2}$，则 $\Gamma_k = \frac{6}{k(k+1)(k+2)}, \frac{\beta_k \gamma_k}{\Gamma_k} = \frac{3LK}{2} \geq \frac{\beta_{k-1} \gamma_{k-1}}{\Gamma_{k-1}}$，我们有

$$f(y_k) - f(x^*) \leq \frac{9L}{2(k+1)(k+2)} D_X^2. \tag{66}$$

Nesterov 加速梯度算法求解问题集合 $\mathcal{F}_L^{1,1}(\mathbb{R}^n)$

证明.

充分利用 f 的凸性和 $Nesterov$ 的步骤可以得到：对任意序列 $\{\beta_k\},\{\gamma_k\},\{\Gamma_k\}$ 满足 $L\gamma_k \leq \beta_k$ 时有

$$f(y_k) - f(x) \leq \frac{\beta_k \gamma_k}{2} D_X^2. \tag{67}$$

若还有

$$\frac{\beta)k\gamma_k}{\Gamma_k} \geq \frac{\beta_{k-1}\gamma_{k-1}}{\Gamma_{k-1}}, \forall k \geq 2, \tag{68}$$

则

$$f(y_k) - f(x) \leq \Gamma_k \frac{\beta_1 \gamma_1}{2} \|x_0 - x\|^2. \tag{69}$$

依次带入不同的序列设定即可。 □

非凸函数的 Nesterov 加速梯度算法

对非凸函数问题集合直接利用 Nesterov 加速梯度算法框架不一定会有收敛性结果。我们需要对其做一些修改，从而使得收敛性存在。

我们首先给出非凸函数问题模型：

问题模型 2.5

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

- 全局信息 Σ：目标函数 $f \in C_L^{1,1}(\mathbb{R}^n)$，可能是非凸函数，约束集合 $X \equiv \mathbb{R}^n$；
- 局部信息 $\mathcal{O}$：$\mathcal{FO}$ 子程序；
- 解的精度 $\mathcal{T}_\epsilon$：求**局部**极小值点 $\bar{x} \in \mathbb{R}^n$ 使得 $\|\nabla f(\bar{x})\| \leq \epsilon$。

非凸函数的 Nesterov 加速梯度算法

非凸函数的加速梯度算法的步骤如下：

Algorithm 3: 非凸函数的 Nesterov 加速梯度算法

Input: 令 $x_0 = y_0, \gamma_k \in (0, 1]$，序列 $\{\beta_k\}$
for $k = 1, 2, ..., N$ **do**
  $z_k \leftarrow (1 - \gamma_k) y_{k-1} + \gamma_k x_{k-1}$
  $x_k \leftarrow x_{k-1} - \frac{1}{\beta_k} \nabla f(z_k)$
  $y_k \leftarrow z_k - \frac{1}{\gamma_k} \nabla f(z_k)$
end for
Output: z_N

该算法中，收敛点列为 $\{z_k\}$。

Nesterov 加速梯度算法求解问题集合 $C_L^{1,1}(\mathbb{R}^n)$

定理 2.11

取 $\gamma_k = \frac{2}{k+1}, \gamma_k = 2L$，则上述算法求解问题模型 *2.5* 时的收敛速率如下：

- 若取 $\beta_k \in \left[\frac{4k+4}{2k+3}L, 2L\right]$，则

$$\min_{0\le k\le N} \|\nabla f(z_k)\|^2 \le \frac{6L(f(x_0)-f^*)}{N}. \tag{70}$$

- 若问题模型 *2.5* 中的 f 是凸函数，取 $\beta_k = \frac{4L}{k}$，则

$$\min_{0\le k\le N} \|\nabla f(z_k)\|^2 \le \frac{96L^2\|x_0-x^*\|^2}{N(N+1)(N+2)}. \tag{71}$$

可以发现，在凸函数情况下，该算法的收敛速度为 $\mathcal{O}(\frac{1}{N^{3/2}})$，而梯度法的收敛速度为 $\mathcal{O}(\frac{1}{N^{1/2}})$。然而，当 f 是非凸函数时，该算法和梯度法的收敛速度相同（没有起到加速效果，但至少保证了收敛性的存在）。

第三部分

非光滑优化问题的复杂度分析

纲要

投影次梯度法

对于**非光滑有约束**问题

$$f^* = \min_{x \in X} f(x), \tag{72}$$

梯度法不可以直接使用。相应地，我们有投影次梯度法。其步骤如下：

Algorithm 4: 投影次梯度法

Input: $x_0 \in \mathbb{R}^n$

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

执行迭代：

$$x_{k+1} \leftarrow \operatorname*{argmin}_{x \in X} \left\| x - \left(x_k - \gamma_k g(x_k) \right) \right\|_2, \tag{73}$$

其中 $\gamma_k > 0$ 是步长，$g(x_k) \in \partial f(x_k)$。

end for

两个改变：（i）将梯度替换为次梯度；（ii）使用投影将解映射到约束集合。

投影次梯度法

经过变形，(73)可以改写为

$$x_{k+1} = \operatorname*{argmin}_{x\in X} \gamma_k \langle g(x_k), x\rangle + \frac{1}{2}\|x - x_k\|_2^2 \tag{74}$$

$$= \operatorname*{argmin}_{x\in X} \underbrace{f(x_k) + \langle g(x_k), x - x_k\rangle}_{f(x)\text{在}X\text{上的线性近似}} + \underbrace{\frac{1}{2\gamma_k}\|x - x_k\|_2^2}_{\text{和上一个迭代点不要太远}} . \tag{75}$$

引理 3.1

投影次梯度法确定的 x_{k+1} 对任意的 $x \in X$ 有

$$\gamma_k \langle g(x_k), x_{k+1} - x\rangle + \frac{1}{2}\|x_{k+1} - x_k\|^2 \le \frac{1}{2}\|x - x_k\|_2^2 + \frac{1}{2}\|x - x_{k+1}\|_2^2.$$

证明.

记 $\phi(x) = \gamma_k \langle g(x_k), x\rangle + \frac{1}{2}\|x_{k+1} - x_k\|_2^2$，该结论可由 $\phi(x)$ 的强凸性得到。 □

投影次梯度法的收敛速度

投影次梯度法适用的问题模型如下：

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：$X \subset \mathbb{R}^n$ **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；
- 局部信息 $\mathcal{O}$：$\mathcal{FO}$ 或 $\mathcal{PO}$ 等子程序；
- 解的精度 $\mathcal{T}_\epsilon$：求全局极小值点 $\bar{x} \in X$，使得 $f(\bar{x}) - f^* \leq \epsilon$。

投影次梯度法的收敛速度

定理 3.1

对于投影次梯度法确定的 $x_k, k = 1, ..., N$，定义

$$\bar{x}_s^N := \frac{\gamma_s x_s + ... + \gamma_n x_N}{\gamma_s + ...\gamma_N} = \left(\sum_{k=s}^{N} \gamma_k \right)^{-1} \sum_{k=s}^{N} (\gamma_k x_k), \tag{76}$$

则投影次梯度法求解问题模型 *3.1* 的收敛速度为

$$f(\bar{x}_s^N) - f^* \leq \left(2 \sum_{k=s}^{N} \gamma_k \right)^{-1} \left[\|x_s - x^*\|_2^2 + M^2 \sum_{k=s}^{N} \gamma_k^2 \right]. \tag{77}$$

证明.

根据引理 *3.1* 可以得到
$\gamma_k[f(x_k) - f(x)] \leq \frac{M^2}{2}\gamma_k^2 + \frac{1}{2}\|x - x_k\|_2^2 - \frac{1}{2}\|x - x_{k+1}\|_2^2$。将此式对 k 求和即可。 □

投影次梯度法的收敛速度

推论 3.1 *

对于投影次梯度法求解问题模型 3.1，选取不同的步长策略，则有如下收敛性结果：

- **固定步长策略**：令 $D_X = \max_{x_1,x_2\in X} \|x_1 - x_2\|$，取固定步长

$$\gamma_k = \sqrt{\frac{D_X^2}{NM^2}}, k = 1, ..., N, \tag{78}$$

则

$$f(\bar{x}_1^N) - f^* \leq \frac{MD_X}{2\sqrt{N}}, \quad \forall N \geq 1. \tag{79}$$

投影次梯度法的收敛速度

推论 3.1*（续）

对于投影次梯度法求解问题模型 3.1，选取不同的步长策略，则有如下收敛性结果：

- **变步长策略**：取变步长

$$\gamma_k = \sqrt{\frac{D_X^2}{kM^2}}, \quad k = 1, 2, ..., \tag{80}$$

则

$$f(\bar{x}^N_{\lceil k/2 \rceil}) - f^* \leq \mathcal{O}(1)\frac{MD_X}{\sqrt{k}}. \tag{81}$$

证明.

代入定理 *3.1* 即可。 □

镜像梯度法

投影次梯度法是在欧几里得结构下进行的。**我们将其扩展到非欧结构下，就得到了镜像梯度法**。

我们将范数 $\|x\|$ 的对偶范数 $\|z\|_*$ 定义为：

$$\|z\|_* := \sup_{\|z\|\leq 1} \langle z, x\rangle. \tag{82}$$

定义**相对于范数** $\|\cdot\|$ **和参数** μ **的距离生成函数** $\omega : X \to \mathbb{R}$，$\omega(\cdot)$ 满足连续可微且强凸，即

$$\langle \nabla\omega(x) - \nabla\omega(y), x - y\rangle \geq \mu\|x - y\|^2, \quad \forall x, y \in X. \tag{83}$$

我们进一步定义 Bregman **距离函数** $V(x, y)$：

$$V(x, y) := \omega(y) - \big(\omega(x) + \langle \nabla\omega(x), y - x\rangle\big). \tag{84}$$

注意，$V(x, \cdot)$ 是强凸函数。

镜像梯度法

我们将投影次梯度法中的迭代公式(74)中的范数距离改为 Bregman 距离，便得到了镜像（次）梯度法：

Algorithm 5: 镜像（次）梯度法

Input: $x_0 \in \mathbb{R}^n$

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

执行迭代：

$$x_{k+1} \leftarrow \operatorname*{argmin}_{x \in X} \gamma_k \langle g(x_k), x \rangle + V(x_k, x), \tag{85}$$

其中 $\gamma_k > 0$ 是步长，$g(x_k) \in \partial f(x_k)$。

end for

当取 $\omega(x) = \|x\|_2^2/2$ 时，$V(x, y) = \|y - x\|_2^2/2$，此式镜像梯度法退化为投影次梯度法。

镜像梯度法的收敛速度

引理 3.2

对于镜像梯度法确定的 x_{k+1}，对任意的 $x \in X$，我们有

$$\gamma_k \langle g(x_k), x_{k+1} - x \rangle + V(x_k, x_{k+1}) \leq V(x_k, x) - V(x_{k+1}, x). \tag{86}$$

定理 3.2

对于镜像梯度法确定的 $x_k, k = 1, ..., N$，定义

$$\bar{x}_s^N := \frac{\gamma_s x_s + ... + \gamma_n x_N}{\gamma_s + ...\gamma_N} = \left(\sum_{k=s}^{N} \gamma_k \right)^{-1} \sum_{k=s}^{N} (\gamma_k x_k), \tag{87}$$

则镜像梯度法求解问题模型 *3.1* 的收敛速度为

$$f(\bar{x}_s^N) - f^* \leq \left(\sum_{k=s}^{N} \gamma_k \right)^{-1} \left[V(x_s, x^*) + \frac{M^2}{2\mu} \sum_{k=s}^{N} \gamma_k^2 \right]. \tag{88}$$

镜像梯度法的收敛速度

同样地，镜像梯度法也有类似的收敛速度：

推论 3.2 *

对于镜像梯度法 *3.2*，令 $D_{\omega,X}^2 = \max_{x_1,x_2\in X} V(x_1,x_2)$，取变步长

$$\gamma_k = \sqrt{\frac{2\mu D_{\omega,X}^2}{kM^2}}, k = 1,2,... \tag{89}$$

则镜像梯度法求解问题模型 *3.1* 的收敛速度为

$$f(\bar{x}_s^N) - f^* \leq \frac{\sqrt{2}MD_{\omega,X}}{\sqrt{\mu k}}, \quad \forall k \geq 1. \tag{90}$$

证明.

代入定理 *3.2* 即可。 □

注意，镜像梯度法时投影次梯度法在非欧几里得结构下的推广。因此，二者具有相似的收敛速度。

重心法

重心法适用的问题模型如下：

问题模型 3.2

考虑**约束有界凸**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 $B > 0$ 使得对任意的 $x \in X$ 有 $-B \leq f(x) \leq B$（有界）；约束集合 $X \subset \mathbb{R}^n$ **闭且凸**；
- 局部信息 $\mathcal{O}$：$\mathcal{FO}$ 子程序；
- 解的精度 $\mathcal{T}_\epsilon$：求全局极小值点 $\bar{x} \in X$，使得 $f(\bar{x}) - f^* \leq \epsilon$。

重心法

对于问题模型 3.2，我们有重心法：

Algorithm 6: 重心法

令 $\mathcal{S}_1 = X$ **for** $k = 1, 2, ..., N$ **do**

计算集合 $\mathcal{S}_k$ 的重心：

$$c_k = \frac{1}{vol(\mathcal{S}_k)} \int_{x \in \mathcal{S}_k} x dx, \tag{91}$$

其中 $vol(\mathcal{S})$ 表示 $\mathcal{S}$ 的体积。

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

$$\mathcal{S}_{k+1} = \mathcal{S}_k \cap \{x \in \mathbb{R}^n : \langle x - c_k, w_k \rangle \leq 0\}. \tag{92}$$

end for

Output: $x_N \in \operatorname{argmin}_{1 \leq r \leq N} f(c_r)$

重心法的收敛速度

引理 3.3

设 $\mathcal{S}$ 是重心在原点的凸集合，即 $\int_{x\in\mathcal{S}} x dx = 0$，$vol(\mathcal{S})$ 表示 $\mathcal{S}$ 的体积，则对任意的 $w \in \mathbb{R}^n$，$w \neq 0$ 有

$$vol\Big(\mathcal{S} \cap \{x^\top w \geq 0\}\Big) \geq \frac{1}{\exp(1)} vol(\mathcal{S}). \tag{93}$$

定理 3.3

重心法在求解问题模型 *3.2* 时的收敛速率为

$$f(x_N) - f^* \leq 2B\Big(1 - \frac{1}{e}\Big)^{\frac{N}{n}}, \tag{94}$$

复杂度上界为

$$\mathcal{O}\Big(n \log \frac{2B}{\epsilon}\Big). \tag{95}$$

椭球与椭球法

重心法每一轮迭代都需要通过积分运算计算一次集合 $\mathcal{S}_k$ 的体积，复杂度过高。**椭球法借鉴了重心法的思想，改进了其计算的可行性**。后面我们可以看到，椭球法的分析复杂度虽然更高，但是更容易执行。

定义 5.1

(椭球) 椭球是具有如下形式的凸集合：

$$\mathcal{E} := \{x \in \mathbb{R}^n : (x-c)^\top H^{-1}(x-c) \leq 1\}, \tag{96}$$

其中 $c \in \mathbb{R}^n$，H 是一个对称正定矩阵。

几何上，c 是椭球的重心，H 的特征向量是椭球的半轴，半轴的长度是对应特征值的正平方根。

椭球法适用的问题模型

椭球法适用的问题模型如下：

问题模型 3.3

考虑**约束有界凸**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 $B > 0$ 使得对任意的 $x \in X$ 均有 $-B \leq f(x) \leq B$（有界）；约束集合 $X \subset \mathbb{R}^n$ **闭且凸**，且 $\mathcal{B}(r) \subset X \subset \mathcal{B}(R)$；
- 局部信息 $\mathcal{O}$：$\mathcal{FO}$ 子程序和 $\mathcal{SO}$ 子程序，后者返回分割平面垂直向量 w 使得

$$w^\top (x - c_0) \leq 0, \forall x \in X.$$

- 解的精度 $\mathcal{T}_\epsilon$：求全局极小值点 $\bar{x}$，使得 $f(\bar{x}) - f^* \leq \epsilon$。

椭球法

引理 3.4

令 $\mathcal{E}_0 = \{x \in \mathbb{R}^n : (x - c_0)^\top H_0^{-1}(x - c_0) \leq 1\}$，对任意的 $w \in \mathbb{R}^n$，$w \neq 0$，可以构造新椭球 $\mathcal{E}$ 使得

$$\{x \in \mathcal{E}_0 : w^\top (x - c_0) \leq 0\} \subset \mathcal{E}, \tag{97}$$

且原椭球 $\mathcal{E}_0$ 和新椭球 $\mathcal{E}$ 之间体积有如下关系：

$$vol(\mathcal{E}) \leq \exp\Big(-\frac{1}{2n}\Big) vol(\mathcal{E}_0). \tag{98}$$

当 $n \geq 2$ 时，新椭球 $\{x \in \mathbb{R}^n : (x - c)^\top H^{-1}(x - c) \leq 1\}$ 满足

$$c = c_0 - \frac{1}{n+1} \frac{H_0 w}{\sqrt{w^\top H_0 w}}, \tag{99}$$

$$H = \frac{n^2}{n^2 - 1}\left(H_0 - \frac{2}{n+1} \frac{H_0 w w^\top H_0}{w^\top H_0 w}\right). \tag{100}$$

椭球法

我们基于上述引理得到椭球法的步骤如下：

Algorithm 7: 椭球法

Input: 令 X_0 是包含 X 且半径为 R、重心为 c 的球，令 $H_0 = R^2 I_n$

for $k = 1, 2, ..., N$ **do**

  if $c_k \notin X$ **then**

    调用子程序返回向量 $w_k \in \mathbb{R}^n$：过 c_k 且垂直于 w_k 的平面将椭球 $\mathcal{E}_k$ 分割，使得 $X \subset \{x \in \mathbb{R}^n : (x - c_k)^\top w_k \leq 0\}$

  else

    调用子程序返回向量 $w_k \in \partial f(c_k)$

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

$$\{x \in \mathcal{E}_k : (x - c_k)^\top \leq 0\} \subset \mathcal{E}_{k+1}, \tag{101}$$

  其中 c_{k+1}, H_{k+1} 分别根据(99)和(100)迭代。

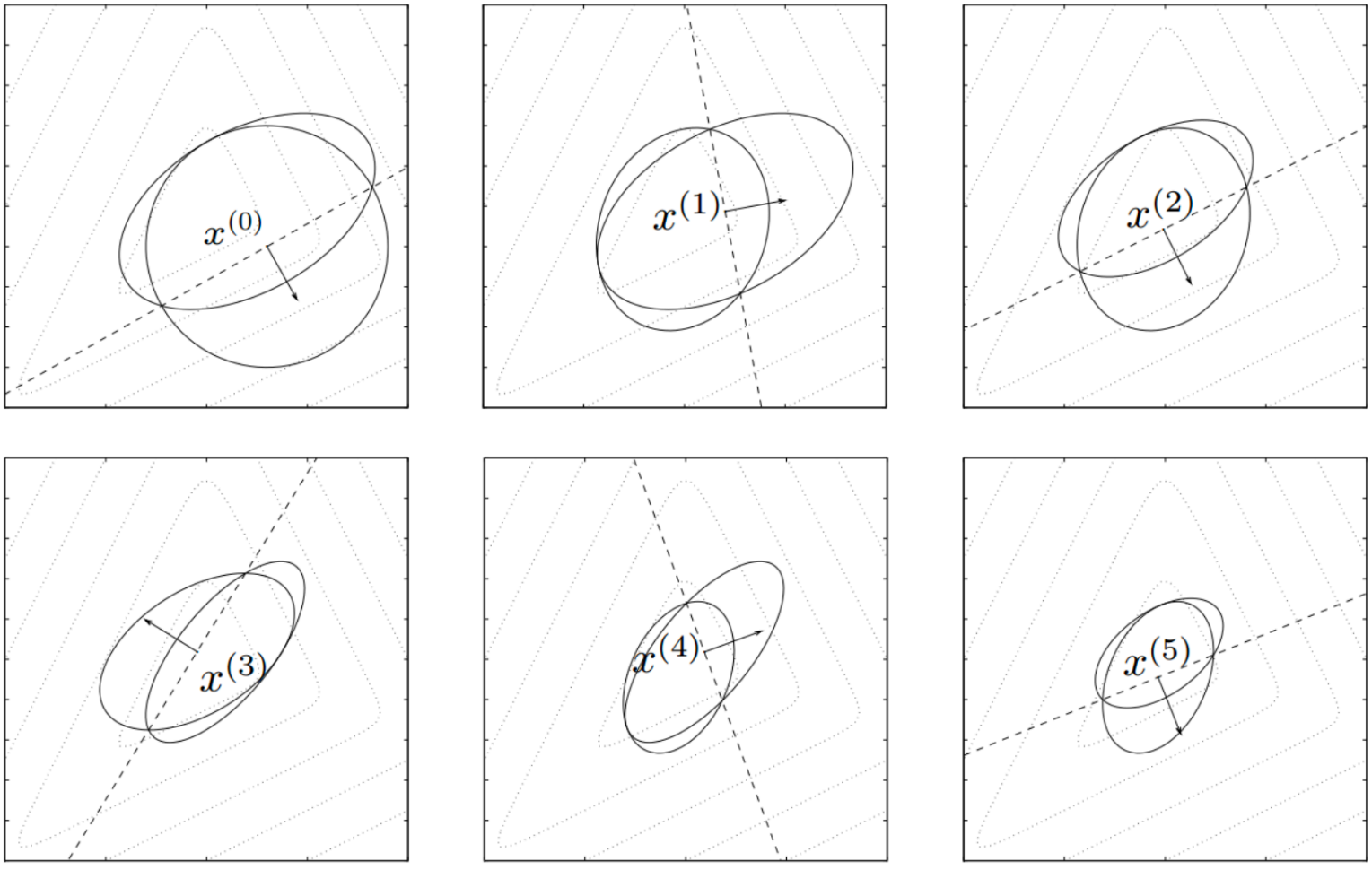
end for

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

$$x_N \in \operatorname*{argmin}_{c \in \{c_1, ..., c_N\} \cap X} f(c).$$

椭球法

椭球法的迭代过程如下图所示：



椭球法的收敛速度

定理 3.4

椭球法在求解问题模型 *3.3* 时的收敛速度为

$$f(x_N) - f^* \leq \frac{2BR}{r} \exp\Big(- \frac{N}{2n^2} \Big). \tag{102}$$

复杂度上界为

$$\mathcal{O}\Big(n^2 \log \frac{BR}{r\epsilon} \Big). \tag{103}$$

对比定理 3.3 和定理 3.4 可以发现：

- 椭球法的分析复杂度略差于重心法；
- 椭球法比重心法更容易计算和执行。

复合优化问题

考虑复合优化问题

$$\begin{aligned}\min_{x\in\mathbb{R}^n}\Phi(x) &:= \phi(x)+h(x)\\ &= \Big[\underbrace{f(x)}_{\in C^{1,1}_{L_f}(\mathbb{R}^n)\text{ 且非凸}} + \underbrace{g(x)}_{\in C^{1,1}_{L_g}(\mathbb{R}^n)\text{ 且凸}}\Big]\\ &+ \underbrace{h(x)}_{\text{定义域有界的简单凸函数，可能非光滑（正则化项）}}. \end{aligned}\tag{104}$$

显然有 $\phi(x)\in C^{1,1}_{L_\phi}$，且 $L_\phi = L_f + L_g$。接下来，我们将介绍面向复合优化问题的 Nesterov 加速算法。具体地，

1. 当 $f(x)\equiv 0$ 时，$\Phi(x)$ 是凸函数，我们有近似点梯度法和对应的两个加速算法（固定步长和变步长）；
2. 当 $f(x)\neq 0$ 时，$\Phi(x)$ 非凸，我们也给出了基于 *Nesterov* 加速算法框架的加速近似点梯度法。

凸复合优化问题与近似点梯度法

当 $f(x)=0$ 时，(104)是一个凸优化问题。我们将函数 $h(x)$ 的近似点算子 $\mathbf{prox}_h(\cdot)$ 定义如下：

$$\mathbf{prox}_h(x) := \operatorname*{argmin}_u \left(h(u) + \frac{1}{2}\|u-x\|_2^2 \right). \tag{105}$$

当 $h(x)$ 是闭凸函数时，$\mathbf{prox}_h(x)$ 存在且唯一。

对于凸复合优化问题，我们有近似点梯度法：

Algorithm 8: 近似点梯度法（面向凸复合优化问题）

Input: $x_0 \in \mathbb{R}^n$，步长序列 $\{t_k\}$
for $k = 1, 2, \ldots$ **do**
    执行迭代

$$x_k = \mathbf{prox}_{t_k h}\left(x_{k-1} - t_k \nabla g(x_{k-1}) \right). \tag{106}$$

end for

近似点梯度法的收敛速度

对于复合函数 $\Phi(\cdot)$，当选取不同的 $h(\cdot)$ 时：

- 当 $h(x)=0$ 时，$\mathbf{prox}_h(x)=x$，此时近似点梯度法退化为梯度法。
- 当 $h(x)=\mathcal{I}_C(x)$（凸紧集合 C 的示性函数）时，$\mathbf{prox}_h(x)=\operatorname{argmin}_{u\in C}\|u-x\|^2$，此时退化为投影梯度法。
- 当 $h(x)=\|x\|_1$ 时，$\mathbf{prox}_h(x)$ 被称为软阈算子（soft-threshold operator），此时近似点梯度法可以拿来求解机器学习中带 l_1 正则的目标优化函数。

定理 3.5

近似点梯度法求解凸复合优化问题时，若取固定步长 $t_k=\frac{1}{L_g}$，则

$$\Phi(x_k)-\Phi(x^*)\leq\frac{L_g}{2k}\|x_0-x_*\|^2. \tag{107}$$

算法复杂度为 $\mathcal{O}(\frac{1}{\epsilon})$。

加速近似点梯度法 FISTA（固定步长）

Algorithm 9: FISTA（近似点梯度法的一个加速版本）

Input: $x_0 \in \mathbb{R}^n$，步长序列 $\{t_k\}$

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

执行迭代

$$y = x_{k-1} + \frac{k-2}{k+1}(x_{k-1} - x_{k-2}), \tag{108}$$

$$x_k = \mathbf{prox}_{t_k h}\big(y - t_k \nabla g(y)\big). \tag{109}$$

end for

定理 3.6

FISTA 求解凸复合优化问题时，若取固定步长 $t_k = \frac{1}{L_g}$，则

$$\Phi(x_k) - \Phi(x^*) \leq \frac{2L_g}{(k+1)^2}\|x_0 - x_*\|^2. \tag{110}$$

算法复杂度为 $\mathcal{O}(\sqrt{\frac{1}{\epsilon}})$。

变步长加速近似点梯度法

近似点梯度法的另一种加速方式如下，它和 FISTA 的收敛速度一致。

Algorithm 10: 变步长加速近似点梯度法

Input: $x_0 \in \mathbb{R}^n$，步长序列 $\{\gamma_k\}$ 和 $\{\beta_k\}$ 满足 $\gamma_k \leq L_g\beta_k, \gamma_1 = 1$

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

执行迭代

$$z_k = (1-\gamma_k)y_{k-1} + \gamma_k x_{k-1}, \tag{111}$$

$$x_k = \mathbf{prox}_{(\beta_k/\gamma_k)h}\left(x_{k-1} - \frac{\beta_k}{\gamma_k}\nabla g(z_k)\right), \tag{112}$$

$$y_k = (1-\gamma_k)y_{k-1} + \gamma_k x_k. \tag{113}$$

end for

定理 3.7

变步长加速近似点梯度法求解凸复合优化问题时，若取 $\gamma_k = \frac{2}{k+1}$，$\beta_k = \frac{1}{L_g}$，则 (110)成立且法复杂度为 $\mathcal{O}(\sqrt{\frac{1}{\epsilon}})$。

非凸复合优化问题

当 $f(x) \neq 0$ 时，(104)是一个非凸优化问题。此时，我们选取梯度映射作为停止准则。其定义为

$$G_\alpha(x, y) := \frac{1}{\alpha}\Big[x - \mathbf{prox}_{\alpha h}(x - \alpha y)\Big]. \tag{114}$$

根据(106)，有

$$\begin{aligned} x_k - x_{k-1} &= \mathbf{prox}_{t_k h}\big(x_{k-1} - t_k \nabla g(x_{k-1})\big) - x_{k-1} && (115)\\ &= -t_k G_{t_k}\big(x_{k-1}, \nabla g(x_{k-1})\big). && (116) \end{aligned}$$

即，$G_{t_k}\big(x_{k-1}, \nabla g(x_{k-1})\big)$ 是近似点迭代步的负方向。此外，根据定义(105)可得

$$G_\alpha(x, \nabla\phi(x)) \in \nabla\phi(x) + \partial h\Big(x - \alpha G_\alpha(x, \nabla\phi(x))\Big). \tag{117}$$

这意味着$G_\alpha(x, \nabla\phi(x)) = 0$ *iff* x 是 $\phi(x) + h(x)$ 的局部极小值点。

非凸复合函数的加速梯度法框架

将在第二部分介绍的、面向非凸函数的 Nesterov 加速算法（算法3）稍作修改，就得到了面向非凸复合函数的加速近似点梯度法：

Algorithm 11: 非凸复合函数的加速近似点梯度法

Input: $x_0 = y_0 \in \mathbb{R}^n$，步长序列 $\{\gamma_k\}$ 满足 $\gamma_1 = 1, \gamma_k \in (0,1), \forall k \geq 2$

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

执行迭代

$$z_k = (1-\gamma_k) y_{k-1} + \gamma_k x_{k-1}, \tag{118}$$

$$x_k = \mathbf{prox}_{\gamma_k h}\big(x_{k-1} - \gamma_k \nabla\phi(z_k)\big), \tag{119}$$

$$y_k = \mathbf{prox}_{\beta_k h}\big(z_k - \beta_k \nabla\phi(z_k)\big). \tag{120}$$

end for

Output: z_N

我们利用 $\|G_{\beta_k}(z_k, \nabla\phi(z_k))\| = \frac{1}{\beta_k}(z_k - y_k)$ 作为该算法的停止准则，这实际上是解序列 $\{z_k\}$ 和 $\{y_k\}$ 之间的距离。

非凸复合函数的加速梯度法框架的收敛速率

定理 3.8

假设对任意的 $\alpha \in (0, +\infty)$ 以及任意的 $x, y \in \mathbb{R}^n$ 存在常数 $M > 0$ 使得 $\|\mathbf{prox}_{\alpha h}(x - \alpha y)\| \leq M$，则使用算法*11*求解复合优化问题时，若取 $\gamma_k = \frac{2}{k+1}, \beta_k = \frac{1}{2L_\phi}, \gamma_k = \frac{k\beta_k}{2}$，则对任意的 $N \geq 1$ 有

$$\min_{k=1,\ldots,N} \|G_{\beta_k}(z_k, \nabla\phi(z_k))\|^2 \leq 24L_\phi \left[\frac{4L_\phi \|x_0 - x_*\|^2}{N^2(N+1)} + \frac{L_f}{N}(\|x_*\|^2 + 2M^2) \right]. \tag{121}$$

当 $f(x) = 0$ 时，有

$$\Phi(y_N) - \Phi(x_*) \leq \frac{4L_g \|x_0 - x_*\|^2}{N(N+1)}. \tag{122}$$

当 $f(x) = 0$ 时，算法11的复杂度和 FISTA 相同，二者均为 $\mathcal{O}(L_g^2/\sqrt{\epsilon})$；否则算法11仍然收敛，但复杂度为 $\mathcal{O}(L_\phi^{2/3}/\epsilon^{1/3} + L_\phi L_f/\epsilon)$。

第四部分

条件梯度算法的复杂度分析

纲要

本部分将介绍面向**线性约束集合**的条件梯度法及其各种改进。条件梯度法的子程序为 $\mathcal{LO}$ 和 $\mathcal{FO}$。

1. 面向凸函数的条件梯度法（CndG）
2. 面向强凸函数的条件梯度法（Shrinking CndG，S-CndG）
3. 加速框架下的条件梯度法（PA-CndG）
4. 光滑优化问题的 $\mathcal{LO}$ 复杂度下界
5. 条件梯度法滑动法（CGS）：对 $\mathcal{LO}$ 和 $\mathcal{FO}$ 的调用次数同时达到最优

带线性约束的优化问题

考虑**带线性约束**的优化问题

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

其中 X 是一个线性约束集合。对此，我们可以采用投影梯度法来迭代：

$$x_{k+1} = \operatorname*{argmin}_{x\in X} \left\{ f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \|y - x_k\|^2 \right\}. \quad (123)$$

或者可以写为

$$x_{k+1} = \mathcal{P}_X(x_k - \alpha_k \nabla f(x_k)). \quad (124)$$

X 作为一个多面体，子迭代(123)是一个**二次规划问题**。求解二次规划问题的计算量是很大的，我们可以**去掉二次项**，**只做一次逼近**，这样(123)就会变成一个线性规划问题：

$$x_{k+1} = \operatorname*{argmin}_{x\in X} \left\{ f(x_k) + \langle \nabla f(x_k), x - x_k \rangle \right\}. \quad (125)$$

条件梯度法

(125)等价于

$$x_{k+1} = \operatorname*{argmin}_{x \in X} \langle \nabla f(x_k), x \rangle. \tag{126}$$

我们定义一个关于线性约束集合 X 的子程序 $\mathcal{LO}$，它对任意给定向量 $p \in \mathbb{R}^n$ 返回线性规划的解 y，满足

$$y \in \operatorname*{argmin}_{x \in X} \langle p, x \rangle. \tag{127}$$

$\mathcal{LO}$ 本身的计算复杂度是容易确定的（例如，单纯形法、内点法等，我们有现成的结论可以使用），因此我们可以专注于研究条件梯度法关于子程序 $\mathcal{LO}$ 的分析复杂度。

条件梯度法

条件梯度法（The Conditional Gradient (CndG) Method）又被称为 Frank-Wolfe 算法，近几年来，条件梯度法受到了机器学习和优化领域的广泛关注，主要由于如下原因:

- **子迭代复杂度低**：条件梯度法的子问题是求解一个线性逼近问题，在很多情况下会比投影次梯度法的非线性逼近子问题要简单。
- **简单性**：条件梯度法在实现时不需要精心挑选步长，且步长不依赖于导数的 *Lipschitz* 常数。而梯度法的实现却非常依赖步长的选择，需要反复的调整以达到收敛的目的。
- **解具有结构性**：条件梯度法的解是约束集合 X 极点的凸组合，因此具有稀疏性和低秩性。

接下来，我们将介绍条件梯度法在凸函数和强凸函数下的复杂度分析、具有 $\mathcal{LO}$ 子程序的算法在求解约束光滑优化问题时的复杂度下界，以及利用 Nesterov 加速梯度法框架导出加速条件梯度法，并分析其复杂度。

条件梯度法（面向凸函数）

我们首先给出条件梯度法求解凸函数优化问题的具体算法：

Algorithm 12: 条件梯度法（CndG，面向凸函数）

1 **Input:** $x_0 \in X$，令 $y_0 = x_0$，取步长 $\alpha_k \in [0,1]$
2 **for** $k = 1, 2, ..., N$ **do**
3 　在 y_{k-1} 处调用子程序 $\mathcal{FO}$，返回 $\nabla f(y_{k-1})$
4 　在 $\nabla f(y_{k-1})$ 处调用子程序 $\mathcal{LO}$，返回 $x_k \in \operatorname{argmin}_{x \in X} \langle \nabla f(x_{k-1}), x \rangle$
5 　令 $y_k = (1 - \alpha_k) y_{k-1} + \alpha_k x_k$
6 **end for**
Output: y_N

变换后可得 $y_k = \operatorname{conv}\{x_0, x_1, ..., x_k\}$，即收敛点 y_k 时点集 $\{x_0, x_1, ..., x_k\}$ 的凸组合。此外，步骤 5 可以变换为

$$y_k = y_{k-1} + \alpha_k (x_k - x_{k-1}), \tag{128}$$

即 y_k 时按照方向 $x_k - y_{k-1}$ 的方向按照步长 α_k 进行更新。

条件梯度法（面向凸函数）

CndG 中的步长有两种选择方式，其一是选取**严格递减到零**的变步长序列：

$$\alpha_k = \frac{1}{k+1}, \quad k = 1, 2, \ldots \tag{129}$$

另一种是采取**精确搜索**的步长：

$$\alpha_k = \underset{\alpha \in [0,1]}{\operatorname{argmin}} f\big((1-\alpha)y_{k-1} + \alpha x_k\big). \tag{130}$$

和投影次梯度法相比，CndG 在求解 l_p 范数约束问题和核范数约束问题时效率更高。

条件梯度法（面向凸函数）

CndG 中的步长有两种选择方式，其一是选取**严格递减到零**的变步长序列：

$$\alpha_k = \frac{1}{k+1}, \quad k = 1, 2, \ldots \tag{131}$$

另一种是采取精确搜索的步长：

$$\alpha_k = \underset{\alpha \in [0,1]}{\operatorname{argmin}} f\big((1-\alpha)y_{k-1} + \alpha x_k\big). \tag{132}$$

和投影梯度法相比，CndG 在求解 l_p 范数约束问题和核范数约束问题时效率更高。

条件梯度法的收敛速度（面向凸函数）

为了分析条件梯度法的收敛速度和复杂度，我们首先给出问题模型：

问题模型 4.1

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 $f \in C_L^{1,1}(X)$，且 $f(x)$ 凸（或强凸），约束集合 X 闭 t 凸有界；
- 局部信息 $\mathcal{O}$：$\mathcal{FO}$ 子程序和 $\mathcal{LO}$ 子程序（或“面向强凸函数的、改进的 $\mathcal{LO}$ 子程序”，后文会介绍）；
- 解的精度 $\mathcal{T}_\epsilon$：求全局极小点 $\bar{x} \in X$ 使得 $f(\bar{x}) - f^* \leq \epsilon$。

条件梯度法的收敛速度（面向凸函数）

定理 4.1 *

设 $f(x)$ 是凸函数，*CndG* 在求解问题模型 *4.1* 时取步长(131)或(132)，则产生的序列 $\{x_k\}$ 对任意的 $k = 1, 2, \ldots$ 满足

$$f(y_k) - f^* \leq \frac{2L}{k(k+1)} \sum_{i=1}^{k} \|x_i - y_{i-1}\|^2 \leq \frac{2L}{k+1} D_X^2, \tag{133}$$

且为了达到 ϵ 精度需要调用 $\mathcal{FO}$ 和 $\mathcal{LO}$ 子程序的次数最多为

$$\left\lceil \frac{2LD_X^2}{\epsilon} \right\rceil - 1. \tag{134}$$

条件梯度法的收敛速度（面向凸函数）

基于定理 4.1 的一些分析：

- 可以发现，CndG 寻找 ϵ 近似解所需要调用子程序 $\mathcal{LO}$ 和 $\mathcal{LO}$ 的次数相同，都不超过 $\mathcal{O}\left(\frac{LD_X^2}{\epsilon}\right)$。在下文中，我们会知道 CndG 求解 $C_L^{1,1}(X)$ 时，其子程序 $\mathcal{LO}$ 的调用次数，i.e.，$\mathcal{O}(1/\epsilon)$，达到了复杂度下界，因此是**最优的**。然而，$\mathcal{FO}$ 的调用次数没有达到复杂度下界 $\mathcal{O}(1/\sqrt{\epsilon})$。
- 其次，注意到 Lipschitz 常数 L 和 D_X 的都与设定的范数 $\|\cdot\|$ 有关，即 $L \equiv L_{\|\cdot\|}, D_X \equiv D_{X,\|\cdot\|}$。这意味着，我们可以选择合适的范数，从而使得分析复杂度**关于范数达到极小**：

$$\mathcal{O}(1) \inf_{\|\cdot\|} \left\{ \frac{L_{\|\cdot\|} D_{X,\|\cdot\|}}{\epsilon} \right\}. \tag{135}$$

- (133)依赖于 $\|x_i - y_{i-1}\|^2$，然而这个值不一定随着 k 的增加而趋近于零。

条件梯度法（面向强凸函数）

接下来我们考虑条件梯度法求解强凸函数。为了充分利用目标函数的强凸性从而提高收敛速度，我们对 $\mathcal{LO}$ 作出如下改进：输入某一给定的 $p \in \mathbb{R}^n$ 和 $x_0 \in X$，返回如下 x：

$$x = \operatorname*{argmin}_{x \in X, \|x - x_0\| \leq R} \langle p, x \rangle. \tag{136}$$

我们称之为面向强凸函数的、改进的 $\mathcal{LO}$ 子程序。同样地，改进的 $\mathcal{LO}$ 的复杂度与范数的选择有关。

条件梯度法（面向强凸函数）

基于改进的 $\mathcal{LO}$，我们给出收缩条件梯度法（The Shrinking CndG Method）：

Algorithm 13: 收缩条件梯度法（The Shrinking CndG Method）

Input: $p_0 \in X$，令 $R_0 = D_X$
for $t = 1, 2, \ldots$ **do**
  令 $y_0 = p_{t-1}$
  for $k = 1, \ldots, 8L/\mu$ **do**
    调用改进的 $\mathcal{LO}$ 子程序计算 $x_k \in \operatorname{argmin}_{x \in X_{t-1}} \langle \nabla f(y_{k-1}, x) \rangle$，
    其中 $X_{t-1} := \{x \in X : \|x - p_{t-1}\| \leq R_{t-1}\}$
    令 $y_k = (1 - \alpha_k) y_{k-1} + \alpha_k x_k$，其中 $\alpha_k \in [0, 1]$
  end for
  令 $p_t = y_k, R_t = R_{t-1}/\sqrt{2}$
end for

条件梯度法的收敛速度（面向强凸函数）

定理 4.2 *

设 $f(x) \in \mathcal{F}_{L,\mu}^{1,1}$，收缩条件梯度法在求解问题模型 *4.1* 时取步长(131)或(132)，则产生的序列 $\{p_k\}$ 对任意的 $t = 1, 2, \ldots$ 满足

$$f(p_t) - f^* \leq \frac{\mu R_0}{2^t}. \tag{137}$$

且为了达到 ϵ 精度需要调用 $\mathcal{FO}$ 和改进的 $\mathcal{LO}$ 子程序的次数最多为

$$\frac{8L}{\mu}\left\lceil \max\left(\log_2 \frac{\mu R_0}{\epsilon}, 1\right)\right\rceil. \tag{138}$$

加速框架下的条件梯度法

接下来，我们在加速梯度法框架中引入条件梯度法的 $\mathcal{LO}$ 子程序。主要的改动是：将非线性逼近子问题修改为线性逼近的子问题。

加速框架下的条件梯度法（The Primal Averaging CndG Method）的步骤如下：

Algorithm 14: 加速框架下的条件梯度法（The Primal Averaging CndG Method）

Input: $x_0 \in X$，步长序列 $\{\alpha_k\}$，$\alpha_k \in [0,1]$，令 $y_0 = x_0$
for $k = 1, 2, \ldots$ **do**
    令 $y_0 = p_{t-1}$
    for $k = 1, \ldots, 8L/\mu$ **do**
        $z_k = \frac{k-1}{k+1} y_{k-1} + \frac{2}{k+1} x_{k-1}$
        令 $p_k = \nabla f(z_k)$ 调用 $\mathcal{LO}$ 子程序计算 $x_k \in \operatorname{argmin}_{x \in X} \langle p_k, x \rangle$
        $y_k = (1 - \alpha_k) y_{k-1} + \alpha_k x_k$
    end for
end for

加速框架下的条件梯度法的收敛速度

定理 4.3 *

PA-CndG 在求解问题模型 *4.1* 时取步长(131)或(132)，则产生的序列 $\{x_k\}$ 和 $\{y_k\}$ 对任意的 $k = 1, 2, \ldots$ 满足

$$f(y_k) - f^* \leq \frac{2L}{k(k+1)} \sum_{i=1}^{k} \|x_i - x_{i-1}\|^2 \leq \frac{2L}{k+1} D_X^2. \tag{139}$$

对比(133)，可以发现，CndG 的收敛速度与 $\|x_k - y_{k-1}\|$ 的变化有关，而该值不一定随 k 趋于零；PA-CndG 的收敛速度与 $\|x_k - x_{k-1}\|$ 有关，该值是否随 k 趋于零取决于 X 的结构和 $\|p_{k+1} - p_k\|$。令 $\gamma_k = \frac{2}{k+1}, \alpha_k = \gamma_k$，则可推出 $\|z_{k+1} - z_k\| \leq 3\gamma_k D_X$，结合 $C_L^{1,1}(X)$ 的性质可得

$$\|p_{k+1} - p_k\|_* = \|\nabla f(z_{k+1}) - f(z_k)\|_* \leq 3\gamma_k D_X, \tag{140}$$

因此 $\|p_{k+1} - p_k\|$ 随 k 趋于零，这意味着，PA-CndG 的确相对于 CndG 增强了收敛性。

加速框架下的条件梯度法的收敛速度

引入更多条件进一步提高收敛性:

定理 4.4 *

PA-CndG 在求解问题模型 *4.1* 时取步长 $\alpha_k = \frac{2}{k+1}$，假设子程序 $\mathcal{LO}$ 满足

$$\|x_k - x_{k-1}\| \leq Q\|p_k - p_{k-1}\|_*^\rho, \quad k \geq 2, \tag{141}$$

则对于某些给定的 $\rho \in (0, 1]$ 和 $Q > 0$，对任意的 $k \geq 1$ 我们有如下收敛性结果

$$f(y_k) - f^* \leq \mathcal{O}(1) \begin{cases} \frac{Q^2 L^{2\rho+1} D_X^{2\rho}}{(1-2\rho)k^{2\rho+1}} & \rho \in (0, 0.5), \\ \frac{Q^2 L^2 D_X \log(k+1)}{k^2} & \rho = 0.5, \\ \frac{Q^2 L^{2\rho+1} D_X^{2\rho}}{(2\rho-1)k^2} & \rho \in (0.5, 1]. \end{cases} \tag{142}$$

条件梯度法的一般框架

首先，我们定义带 $\mathcal{LO}$ 子程序的解算法 $\mathcal{S}$ 的一般框架：

Algorithm 15: 条件梯度法的一般框架

Input: $x_0 \in X$
for $k = 1, 2, ..., N$ **do**
    确定线性逼近函数 $\langle p_k, \cdot \rangle$
    调用 $\mathcal{LO}$ 子程序计算 $x_k \in \operatorname{argmin}_{x \in X} \langle p_k, x \rangle$
    输出 $y_k \in conv\{x_0, ..., x_k\}$
end for

算法 12、13、14 都是该框架的特殊情形。当 f 光滑时，p_k 可以是某个可行点处的梯度，或历史梯度的线性组合；当 f 非光滑时，p_k 可以是 f 光滑逼近函数的梯度。p_k 也可以包含一些噪音或者二阶梯度信息。将上述算法框架与一阶算法进行比较。可以发现，

- 条件梯度法的一般框架解决线性逼近的子问题，而一阶算法需要求解非线性子问题（投影或近似点算子的计算）；
- 条件梯度法的一般框架对于算法的迭代方向 p_k 和输出的点 y_k 比一阶算法更为灵活。

光滑优化问题的 $\mathcal{LO}$ 复杂度下界

接下来，我们给出问题模型 4.1 关于 $\mathcal{LO}$ 子程序的分析复杂度下界。

定理 4.5 *

为了取得问题集合 $C^{1,1}_{L,\|\cdot\|}(X)$ 且目标函数为凸函数的 ϵ 近似解，包含条件梯度法的一般框架的解算法集合关于 $\mathcal{LO}$ 子程序的复杂度下界为

$$\left\lfloor \min\left\{\frac{n}{4}, \frac{LD_X^2}{8\epsilon}\right\}\right\rfloor - 1. \tag{143}$$

当 n 足够大时，光滑优化问题的 $\mathcal{LO}$ 复杂度下界为

$$\mathcal{O}\left(\frac{LD_X^2}{\epsilon}\right). \tag{144}$$

加速条件梯度法

由定理 4.5 可知，条件梯度法在 $\mathcal{LO}$ 子程序的调用次数上已经取得最优，但是对 $\mathcal{FO}$ 子程序并没有。下面，我们通过加速条件梯度法来减少对 $\mathcal{FO}$ 的调用次数。

Algorithm 16: 加速条件梯度法（条件梯度滑动算法，CGS）

---

1 **Input:** $x_0 \in X$，迭代次数 N，取 $\beta_k > 0, \eta_k > 0, \gamma_k \in [0,1]$，令 $y_0 = x_0$

2 **for** $k = 1, 2, ..., N$ **do**

3 $\quad z_k = (1-\gamma_k)y_{k-1} + \gamma_k x_{k-1}$

4 $\quad x_k = CndG(f'(z_k), x_{k-1}, \beta_k, \eta_k)$

5 $\quad y_k = (1-\gamma_k)y_{k-1} + \gamma_k x_k$

6 **end for**

Output: y_N

CGS 与 Nesterov 加速梯度法不同的地方在于：步骤 3 中 x_k 的更新方式。本质上，**加速条件梯度法是用条件梯度法来求解非线性优化子问题**。其中更新 x_k 的**子程序 CndG** 如下：

加速条件梯度法

Algorithm 17: 子程序 CndG：$u^+ = \mathrm{CndG}(g, u, \beta, \eta)$

1 $u_1 \leftarrow u, t \leftarrow 1$

2 调用 $\mathcal{LO}$ 求解如下子问题，令 v_t 为其最优解：

$$V_{g,u,\beta}(u_t) = \max_{x \in X} \langle g + \beta(u_t - u_1), u_t - x \rangle. \tag{145}$$

3 若 $V_{g,u,\beta}(u_t) \le \eta$，令 $u^+ = u$，停机

4 令 $u_{t+1} = (1 - \alpha_t)u_t + \alpha_t v_t$，其中

$$\alpha_t = \min\left\{1, \frac{\langle \beta(u - u_t) - g, v_t - u_t \rangle}{\beta \|v_t - u_t\|^2}\right\}. \tag{146}$$

5 取 $t \leftarrow t + 1$ 并转到步骤 2

可以发现，**子程序 CndG** 更新 x_k 的等价形式为求解如下非线性投影子问题：

$$\min_{x \in X}\left\{\phi(x) := \langle g, x \rangle + \frac{\beta_k}{2}\|x - u_1\|^2\right\}, \quad g = \nabla f(z_k). \tag{147}$$

加速条件梯度法的收敛速度和复杂度

此外，**子程序 CndG** 步骤 2 中的 $V_{g,u,\beta}$ 又被称为 Wolfe 间隔，是**约束优化问题最优性条件的满足程度**。步长 α_k 的选取实际上等价于

$$\alpha_t = \underset{\alpha\in[0,1]}{\operatorname{argmin}}\, \phi\big((1-\alpha)u_t, \alpha v_t\big). \tag{148}$$

为了分析加速条件梯度法的收敛速度和复杂度，我们可以采取分步的策略：

1. 确定**子程序 CndG** 的收敛性和复杂度；
2. 确定外迭代的收敛性和复杂度。

加速条件梯度法的收敛速度和复杂度

定理 4.6 *

若 $f \in C_L^{1,1}(\mathbb{R}^n)$ 且是凸函数，则加速条件梯度法求解问题模型 *2.4* 的收敛速率为：

- 取 $\beta_k = \frac{3L}{k+1}, \gamma_k = \frac{3}{k+2}$，以及 $\eta_k = \frac{LD_X^2}{k(k+1)}$ 时，外迭代收敛速度为：

$$f(y_k) - f(x^*) \leq \frac{15LD_X^2}{2(k+1)(k+2)}. \tag{149}$$

若外迭代总次数（$\mathcal{FO}$ 的分析复杂度）为 N，则内迭代总次数（$\mathcal{LO}$ 的分析复杂度）上界为 $\sum_{k=1}^{T} T_k \leq 9N^2 + 10N$。加速条件梯度法为了得到 ϵ 近似解所需的 $\mathcal{FO}$ 和 $\mathcal{LO}$ 的复杂度上界分别为

$$N(\mathcal{FO}) = \sqrt{\frac{15LD_X^2}{2\epsilon}}, N(\mathcal{LO}) = \frac{135LD_X^2}{2\epsilon} + 10\sqrt{\frac{15LD_X^2}{2\epsilon}}. \tag{150}$$

加速条件梯度法的收敛速度和复杂度

定理 4.6 * (续)

- 令外迭代总次数 N 固定，取 $\beta_k = \frac{2L}{k}, \gamma_k = \frac{2}{k+1}$，以及 $\eta_k = \frac{2LD_0^2}{Nk}$，则外迭代收敛速度为：

$$f(y_N) - f(x^*) \leq \frac{6LD_0^2}{N(N+1)}. \tag{151}$$

内迭代总次数（$\mathcal{LO}$ 的分析复杂度）为 $\sum_{k=1}^{T} T_k \leq \frac{6N^2 D_X^2}{D_0^2} + N$。加速条件梯度法为了得到 ϵ 近似解所需的 $\mathcal{FO}$ 和 $\mathcal{LO}$ 的复杂度上界分别为

$$N(\mathcal{FO}) = \mathcal{O}\left(D_0\sqrt{\frac{L}{\epsilon}}\right), N(\mathcal{LO}) = \mathcal{O}\left(\frac{LD_X^2}{\epsilon} + D_0\sqrt{\frac{L}{\epsilon}}\right). \tag{152}$$

加速条件梯度法的 $\mathcal{FO}$ 和 $\mathcal{LO}$ 均达到了 $C_L^{1,1}(X)$ 的复杂度下界。

第五部分
随机优化算法的复杂度分析

纲要

1. 随机优化算法的精度度量：ϵ 近似解和(ϵ, δ) 近似解
2. 求解随机优化问题的通用策略：
   - 采样平均逼近策略：先通过蒙特卡洛采样对目标函数进行近似，然后求解近似函数的最优值
   - 随机逼近策略：直接利用目标函数的随机梯度信息更新迭代。结合集成学习，还有集成随机逼近
3. 基于随机逼近算法框架，有
   - 随机梯度法
   - 随机镜像下降法
4. （随机梯度法）在光滑凸随机优化问题下的收敛速度
5. （改进的随机梯度法）在光滑非凸随机优化问题下的收敛速度

随机优化问题

考虑随机优化问题

$$\min_{x\in X} f(x) := \mathbb{E}_{\xi}\Big[F(x,\xi)\Big], \tag{153}$$

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 $\Xi := \{\forall \xi : p(\xi) \neq 0\} \subset \mathbb{R}^d$。定义函数 $F : X \times \Xi \to \mathbb{R}$，对任意的 $\xi \in \Xi$，$F(x,\xi)$ 都是凸函数，该函数对任意的 $x \in X$ 都是有限值且关于 ξ 的期望

$$\mathbb{E}[F(x,\xi)] = \int_{\xi\in\Xi} F(x,\xi) dp\xi \tag{154}$$

是有意义的。

显然，$f(\cdot)$ 在 X 上是凸的并且具有有限值。此外，因为凸函数在有界闭集上连续，所以 $f(\cdot)$ 在 X 上连续。

随机优化问题

因为需要在不同的 ξ 处计算 $f(\cdot)$，而这在高维情况下十分耗时，因此我们需要采用随机算法，例如，通过**蒙特卡洛抽样技巧**来降低复杂度。对此，我们需要如下假设：

假设 5.1

对于随机向量 ξ，存在已知的蒙特卡洛策略，能够产生一系列的独立同分布的样本：$\{\xi_i\}_{i=1}^N$。

接下来给出一些常见的随机优化问题模型。

随机优化问题

案例 5.1：随机效用模型

$$\min_{x\in X}\left\{f(x):=\mathbb{E}_{\xi}\left[\phi\Big(\sum_{i=1}^{n}\big(\frac{i}{n}+\xi_i\big)x_i\Big)\right]\right\},\tag{155}$$

其中 $X=\{x\in\mathbb{R}^n:x\geq 0,\sum_{i=1}^{n}x_i=1\}$，$\xi_i$ 相互独立且 $\xi_i\sim\mathcal{N}(0,1)$，$\phi(\cdot)$ 是逐段线性的凸函数：

$$\phi(t):=\max\Big\{a_1t+b_1,...,a_mt+b_m\Big\},\tag{156}$$

其中 $\{a_j\}_{j=1}^m,\{b_j\}_{j=1}^m$ 是已知常数。

随机优化问题

案例 5.2：机器学习模型

对于机器学习中的分类或回归问题，我们有 n 个训练样本 $\{(x_1, y_1), ..., (x_n, y_n)\}$，其中样本特征 $x_i \in \mathcal{X}$，样本标签 $y_i \in \mathcal{Y}$。假设其中的样本都是从某一未知分布 D 中**独立抽样**出来。我们需要求解分类或回归模型 $h(\cdot; w) : \mathcal{X} \to \mathcal{Y}$，其中 w 是预测函数的参数。为此我们构造误差函数 $l(h(\cdot, w), \cdot) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$，并定义其在分布 D 上的期望风险 $R(w)$：

$$R(w) := \mathbb{E}_{(x,y)\sim D}\Big[l(h(x; w), y)\Big]. \tag{157}$$

通过求解优化问题

$$\min_w R(w) + r(w) \tag{158}$$

来求得分类或回归模型 $h(\cdot; w)$ 的最优参数 w^*，其中 $r(\cdot)$ 是正则项，可以取 $r(w) = \|w\|_2^2$ 或 $r(w) = \|w\|_1$，用于控制模型的复杂度，防止过拟合。

随机优化问题

对于上述机器学习模型，因为 D 未知，所以期望风险无法计算。但是因为数据点独立同分布，所以我们可以用经验风险

$$R_n(w) := \frac{1}{n}\sum_{i=1}^{n} l(h(x_i; w), y_i) \tag{159}$$

来近似 $R(w)$。这意味着，我们可以通过求解

$$\min_w R_n(w) + r(w) \tag{160}$$

来得到 $h(\cdot; w^*)$ 的一个近似。

随机优化问题解的精度

由假设 5.1，对随机变量 ξ 采样得到样本数据 $\{\xi_i\}$，通过处理 $f(x,\xi_i)$ 我们可以得到(153)的一个近似解 $\tilde{x}$。注意到近似解 $\tilde{x}$ 是一个关于 ξ 的函数，因此它是一个随机变量。由此，我们引入如下解的精度的度量：

定义 5.1

由随机优化算法产生的随机变量 $\tilde{x} \in X$，如果其满足

$$\mathbb{E}_{\tilde{x}}[f(\tilde{x}) - f^*] \leq \epsilon, \tag{161}$$

则称其为随机优化问题(153)的ϵ 近似解。

因为无法精确计算期望，所以我们可以弱化用期望来衡量随机变量解 $\tilde{x}$ 的精确性，而采用**关于解不等式成立的概率置信水平**来衡量其解的好坏：

定义 5.2

由随机优化算法产生的随机变量 $\tilde{x} \in X$，如果其满足

$$\mathbf{Pr}\{f(\tilde{x}) - f^* \geq \epsilon\} \leq \delta, \tag{162}$$

则称其为随机优化问题(153)的(ϵ, δ) 近似解。δ 越小，精度越高。

随机优化问题

接下来将介绍：

- 两个策略：
  1. **采样平均逼近策略**：先通过采样（例如蒙特卡罗算法）对目标函数进行近似，然后求解近似函数的最优值
  2. **随机逼近策略**：典型的方法是随机梯度法。它直接利用目标函数的随机梯度信息来更新迭代点。结合集成学习，还有集成随机逼近算法
- 深入分析随机梯度法的收敛速度和复杂度
  1. 面向**非光滑**问题
  2. 面向**光滑凸**问题
  3. 面向**光滑非凸**问题

采样平均逼近算法

采样平均逼近算法具体如下：

Algorithm 18: 采样平均逼近算法

1 对随机变量 ξ 进行抽样，产生样本序列 $\xi_i, i = 1, ..., N$

2 构造 $f(x)$ 的逼近函数：$\hat{f}_N(x) := \frac{1}{N}\sum_{i=1}^{N} F(x, \xi_i)$。

3 求解确定优化问题

$$\min_{x \in X} \hat{f}_N(x). \tag{163}$$

当 N 很大时，$\forall x \in X, \epsilon > 0$，我们总有

$$\mathbf{Pr}\left(|\hat{f}_N^{(}x) - f(x)| \geq \epsilon\right) \to 0. \tag{164}$$

然而，这是一个很弱的结论。

采样平均逼近算法的收敛速度

引理 5.1

(Hoeffding 不等式) 设 $Z_1, ..., Z_N$ 是具有相同期望 $\mu > 0$ 且相互独立的随机变量，且 $\forall i = 1, ...N, \mathbf{Pr}\{0 \le Z_i \le V\} = 1$ 成立，则对于 $\bar{Z}_N = \frac{1}{N}\sum_{i=1}^{N} Z_i$，有

$$\mathbf{Pr}\{\bar{Z}_n - \mu \ge \epsilon\} \le \exp\Big(-\frac{2N\epsilon^2}{V^2}\Big). \tag{165}$$

当 $f(x, \xi)$ 关于 ξ 一致有界时，我们可以得到如下结论：

定理 5.1

若 $F(x, \xi)$ 任意变差有界，即存在 $V < \infty$ 满足

$$V = \max\big\{F(x_1, \xi_1) - F(x_2, \xi_2) : x_1, x_2 \in X, \xi_1, \xi_2 \in \Xi\big\}, \tag{166}$$

则对于任意 $\epsilon > 0$ 和 $\delta \in (0, 1)$，当抽样样本数量取 $N = \lceil \frac{V^2}{2\epsilon^2}\log(\frac{2}{\epsilon})\rceil$ 时，有

$$\mathbf{Pr}\{|\hat{f}_N(x) - f(x)| > \epsilon\} \le \delta. \tag{167}$$

采样平均逼近算法的收敛速度

设 x^* 表示问题(153)的最优解，$\hat{x}^*$ 表示问题(163)的最优解，定理 5.1 告诉我们：$\mathbf{Pr}\{|\hat{f}_N(x^*) - f(x^*)| > \epsilon\} \leq \delta$，然而我们更关注 $\mathbf{Pr}\{|\hat{f}_N(\hat{x}^*) - f(x^*)| > \epsilon\} \leq \delta$ 是否成立。换句话说，我们更想知道 $\hat{x}^*$ 的近似解是否也是 x^* 的近似解。

对此，我们有如下结论：

定理 5.2 *

假设 $X \subset \mathbb{R}^n$ 且存在 $L > 0$，使得 $F(x, \xi)$ 任意变差有界，若(163)中的 N 满足

$$N \geq \mathcal{O}(1)\left(\frac{DL}{\epsilon}\right)^2\left[n \ln\left(\frac{DL}{\epsilon}\right) + \ln\left(\frac{1}{\epsilon}\right)\right], \tag{168}$$

其中 $D := \sup_{x,y \in X} \|x - y\|$，则问题(163)每个精度为 $\epsilon/2$ 的近似解都是问题(153)的 (ϵ, δ) 近似解。

采样平均逼近算法的复杂度 *

基于定理 5.2，我们有如下复杂度结论：

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(163)精度为 $\epsilon/2$ 的近似解需要对 $\hat{f}_N(x)$ 求 $\mathcal{O}(\frac{L^2D^2}{\epsilon^2})$ 次梯度，若使该解为问题(153)的 (ϵ, δ) 近似解，则总共需要对 $F(x, \xi)$ 的求导次数不少于

$$\mathcal{O}\left(\frac{D^4L^4}{\epsilon^4}\left[n\ln\left(\frac{DL}{\epsilon}\right)+\ln\left(\frac{1}{\epsilon}\right)\right]\right), \tag{169}$$

也就是

$$\mathcal{O}\left(\frac{n}{\epsilon^4}\ln\frac{1}{\epsilon}+\frac{1}{\epsilon^4}\ln\frac{1}{\epsilon}\right). \tag{170}$$

这就是采样平均逼近算法的复杂度。

随机逼近算法

另一个策略叫随机逼近，它是随机梯度算法的原型。它直接利用随机梯度来近似 $f(x)$ 的梯度。

假设 5.2

存在子程序 $\mathcal{SFO}$ 对任意的 $x \in X$ 和随机样本 $\xi \in \Xi$，返回函数值 $F(x,\xi)$ 和一阶随机次梯度 $G(x,\xi)$，满足 $\mathbb{E}_\xi[G(x,\xi)] = g(x) \in \partial f(x)$。

Algorithm 19: 随机逼近算法框架 $\mathcal{S}$

Input: $x_0 \in X$，初始信息集合 $I_{-1} = \varnothing$
for $k = 0, 1, ..., N$ **do**
    根据 ξ 的分布生成样本 ξ_k
    在 ξ_k 处调用子程序 $\mathcal{SFO}$，更新信息集合
    $I_{k+1} = I_k \cup (x_k, \xi_k, \mathcal{SFO}(x_k, \xi_k))$
    根据信息集合 I_{k+1} 更新迭代点 x_{k+1}
end for
Output: $\bar{x} = \mathcal{S}(x_0)$

集成随机逼近算法

进一步地，结合集成学习，我们可以得到集成随机逼近算法：

Algorithm 20: 集成随机逼近算法框架 $\mathcal{M}$

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

for $j = 1, ..., K$ **do**

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

end for

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

分析随机逼近算法

为了分析随机逼近算法，我们对(153)的目标函数做如下假设：

- $f(x)$ 在 X 上的总变差有界，即 $V_f = \sup_{x \in X}\{f(x) - f^*\} < \infty$；
- $F(x, \xi)$ 在 $X \times \Xi$ 上的任意变差有界。

对随机逼近算法19的解 $\bar{x}$，若我们可以证明存在关于迭代次数 N 的单调递减函数 $C_{\mathcal{S}}(N)$ 使得

$$\mathbb{E}[f(\bar{x}) - f^*] \leq C_{\mathcal{S}}(N), \tag{171}$$

则当 N 满足 $C_{\mathcal{S}}(N) \leq \epsilon\delta$ 时，$\bar{x}$ 是(153)的 (ϵ, δ) 近似解。这是因为，根据 Markov 不等式有

$$\mathbf{Pr}\{f(\bar{x}) - f^* \geq \epsilon\} \leq \frac{\mathbb{E}[f(\bar{x}) - f^*]}{\epsilon} \leq \frac{C_{\mathcal{S}}(N)}{\epsilon} \leq \delta. \tag{172}$$

分析集成随机逼近算法

对于集成随机逼近算法20，我们则有如下引理和定理：

引理 5.2

设 $\hat{x}$ 是集成随机逼近算法*20*的解，则对任意的 $\beta > 0$ 有

$$\mathbf{Pr}\{f(\hat{x}) - f^* \geq C_{\mathcal{M}}(N) + \beta\} \leq \exp\left\{-\frac{2K\beta^2}{V_f^2}\right\}. \tag{173}$$

随机逼近算法的复杂度

定理 5.3

令 N 和 K 按如下方式选取

$$N = \left\lceil C_{\mathcal{S}}^{-1}\left(\frac{\epsilon V_f}{2}\right)\right\rceil, \quad K = \left\lceil \frac{2}{\epsilon^2}\ln\frac{1}{\delta}\right\rceil, \tag{174}$$

则集成随机逼近算法*20*的解 $\hat{x}$ 是问题(153)的 $(\epsilon V_f, \delta)$ 近似解，即

$$\mathbf{Pr}\Big\{f(\hat{x}) - f^* \geq \epsilon V_f\Big\} \leq \delta, \tag{175}$$

且集成随机逼近算法*20*总共需要调用子程序 $\mathcal{SFO}$ 的次数 T 满足

$$T = K \cdot N \leq \left(1 + C_{\mathcal{S}}^{-1}\left(\frac{\epsilon V_f}{2}\right)\right) \cdot \left(1 + \frac{2}{\epsilon^2}\ln\frac{1}{\delta}\right). \tag{176}$$

随机逼近算法的复杂度

由随机梯度法的收敛性结论

$$\mathbb{E}[f(\bar{x}^k) - f^*] \leq \frac{DM}{\sqrt{k}}, \tag{177}$$

我们有 $C_{\mathcal{S}}(N) = DM/\sqrt{N}$。结合(172)可知，为了得到问题(153)的 $(\epsilon V_f, \delta)$ 近似解，算法19中的 N 需要满足

$$N = \mathcal{O}\left(\frac{1}{\epsilon^2\delta^2}\left(\frac{DM}{V_f}\right)^2\right). \tag{178}$$

进一步地，若集成随机逼近算法20中的基算法是随机梯度法，则为了得到问题(153)的 $(\epsilon V_f, \delta)$ 近似解，算法20中的 N 和 T 需要满足

$$N = \mathcal{O}\left(\frac{1}{\epsilon^2}\left(\frac{MR}{V_f}\right)^2\right), \quad T = K \cdot N = \mathcal{O}\left(\frac{1}{\epsilon^4}\ln\frac{1}{\delta}\left(\frac{MR}{V_f}\right)^2\right). \tag{179}$$

随机逼近算法的复杂度

算法19与算法20的复杂度结论可以总结如下：

	算法 $\mathcal{S}$	算法 $\mathcal{M}$
基算法个数 K	1	$\frac{1}{\epsilon^2}\ln\frac{1}{\delta}$
基算法迭代次数 N	$\frac{1}{\epsilon^2\delta^2}\left(\frac{MR}{V_f}\right)^2$	$\frac{1}{\epsilon^2}\left(\frac{MR}{V_f}\right)^2$
总计算复杂度	$\frac{1}{\epsilon^2\delta^2}\left(\frac{MR}{V_f}\right)^2$	$\frac{1}{\epsilon^4}\ln\frac{1}{\delta}\left(\frac{MR}{V_f}\right)^2$

随机梯度法与随机镜像下降法：面向非光滑问题

基于到随机逼近算法框架，我们可以得到随机梯度法。

Algorithm 21: 随机梯度法（Stochastic Gradient Method）

Input: $x_0 \in \mathbb{R}^n$，步长序列 $\{\gamma_k\}$
for $k = 0, 1, \ldots$ **do**
    生成样本 ξ_k
    在 (x_k, ξ_k) 处调用子程序 $\mathcal{SFO}$ 得到随机梯度 $G(x_k, \xi_k)$，并迭代

$$x_{k+1} = \operatorname*{argmin}_{x \in X} \|x_k - \gamma_k G(x_k, \xi_k)\|_2. \tag{180}$$

end for

随机梯度法与随机镜像下降法：面向非光滑问题

就像基于次梯度法得到镜像下降法那样，我们可以推广随机梯度法得到随机镜像下降法。

Algorithm 22: 随机镜像下降法（Stochastic Mirror Descent Method）

Input: $x_0 \in \mathbb{R}^n$，步长序列 $\{\gamma_k\}$ 以及 *Bregman* 距离 $V(x,y)$

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

  生成样本 ξ_k

  在 (x_k, ξ_k) 处调用子程序 $\mathcal{SFO}$ 得到随机梯度 $G(x_k, \xi_k)$，并迭代

$$x_{k+1} = \operatorname*{argmin}_{x \in X} \gamma_k \langle G(x_k, x_k), x \rangle + V(x_k, x) \tag{181}$$

end for

随机镜像下降法的复杂度

当取 Bregman 距离为 $V(x,y)=\frac{1}{2}\|x-y\|^2$ 时，随机梯度法 21 与随机镜像下降法 22 等价。**因此我们只给出随机镜像下降法的复杂度分析**。

我们考虑随机优化问题(153)在下列随机梯度假设时的 ϵ 近似解和 (ϵ,δ) 近似解的复杂度。假设 $\{\xi_t\}$ 是关于 ξ 的独立同分布随机变量序列，考虑对任意的 $x\in X$，存在常数 $M_*>0$，且假设 5.2 产生的随机次梯度 $G(x,\xi_t)$ 满足下列条件之一：

假设 5.3 $\|G(x,\xi_t)\|_*\le M_*$

假设 5.4 $\mathbb{E}_{\xi_t}[\exp(\|G(x,\xi_t)\|_*^2/M_*^2)]\le\exp(1)$

假设 5.5 $\mathbb{E}_{\xi_t}[\|G(x,\xi_t)\|_*^2]\le M_*^2$

显然有

$$\text{假设 5.3}\Rightarrow\text{假设 5.4}\Rightarrow\text{假设 5.5} \tag{182}$$

即，假设 5.3 最强，假设 5.5 最弱。

随机镜像下降法的复杂度（基于假设 5.5）

定理 5.4 *

若假设 *5.1*、假设 *5.2* 和假设 *5.5* 成立，令

$$\bar{x}_s^k = \frac{\sum_{t=s}^k \gamma_t x_t}{\sum_{t=s}^k \gamma_t}, \tag{183}$$

其中 x_t 表示随机镜像下降法22的迭代点，则有如下收敛性结果：

$$\mathbb{E}[f(\bar{x}_s^k)] - f^* \leq \left(\sum_{t=s}^k \gamma_t\right)^{-1} \left[\mathbb{E}[V(x_s, x^*)] + (2\nu)^{-1} M_*^2 \sum_{t=s}^k \sum_{t=s}^k \gamma_t^2\right]. \tag{184}$$

随机镜像下降法的复杂度（基于假设 5.5）

推论 5.1 *

对随机镜像下降法22取固定步长 $\gamma_t = \sqrt{2\mu}D_{w,X}/(M\sqrt{k})$，则有如下收敛性结果：

$$\mathbb{E}[f(\bar{x}_1^k) - f^*] \leq D_{w,X} M \sqrt{\frac{2}{\mu k}}. \tag{185}$$

其 ϵ 近似解的复杂度为

$$\mathcal{O}\left(\frac{D_{w,X}^2 M^2}{\epsilon^2}\right). \tag{186}$$

定理 5.5 *

若假设 5.1、假设 5.2 和假设 5.5 成立，则随机镜像下降法22求得 (ϵ, δ) 近似解的复杂度为

$$\mathcal{O}\left(\frac{1}{\epsilon^2\delta^2}\right). \tag{187}$$

随机镜像下降法的复杂度（基于假设 5.4）

定理 5.6

若假设 5.1、假设 5.2 和假设 5.4 成立，则随机镜像下降法22求得 (ϵ, δ) 近似解的复杂度为

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\ln^2\frac{1}{\delta}\right). \tag{188}$$

随机镜像下降法的复杂度（基于假设 5.3）

引理 5.3

（Azuma-Hoeffding 不等式）设 $Z_1, Z_2, \ldots$ 是一个鞅差序列，且对 $i = 1, 2, \ldots$ 满足 $|Z_i| \leq V$，则下列不等式成立

$$\mathbf{Pr}\left\{\sum_{i=1}^{N} Z_i \geq c\right\} \leq \exp\left(-\frac{2c^2}{NV^2}\right), \tag{189}$$

$$\mathbf{Pr}\left\{\sum_{i=1}^{N} Z_i \leq c\right\} \leq \exp\left(-\frac{2c^2}{NV^2}\right). \tag{190}$$

基于上述引理我们可以得到如下结论：

定理 5.7

若假设 *5.1*、假设 *5.2* 和假设 *5.3* 成立，则随机镜像下降法*22*求得 (ϵ, δ) 近似解的复杂度为

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\ln\frac{1}{\delta}\right). \tag{191}$$

随机镜像下降法的复杂度

不同假设下的随机镜像下降法的复杂度总结如下：

假设条件	(ϵ, δ) 复杂度
$\|G(x, \xi_t)\|_* \leq M_*$	$\mathcal{O}\left(\frac{1}{\epsilon^2 \delta^2}\right)$
$\mathbb{E}_{\xi_t}\left[\exp\left(\|G(x, \xi_t)\|_*^2 / M_*^2\right)\right] \leq \exp(1)$	$\mathcal{O}\left(\frac{1}{\epsilon^2} \ln^2 \frac{1}{\delta}\right)$
$\mathbb{E}_{\xi_t}[\|G(x, \xi_t)\|_*^2] \leq M_*^2$	$\mathcal{O}\left(\frac{1}{\epsilon^2} \ln \frac{1}{\delta}\right)$

光滑凸随机优化问题

对于光滑凸随机优化问题，我们要求：

- $f(x)$ 光滑且是凸函数；
- $f(x)$ 的导数满足 $\forall x, y \in X, \|\nabla f(x) - \nabla f(y)\|_* \leq L\|x - y\|$ 成立。

基于假设 5.1 和假设 5.2，类似地，我们给出如下假设：

假设 5.6 存在 $\sigma > 0$，对任意的 $x \in X$ 有

$$(a) \quad \mathbb{E}[G(x, \xi)] \equiv \nabla f(x). \tag{192}$$

$$(b) \quad \mathbb{E}[\|G(x, \xi) - \nabla f(x)\|_*^2] \leq \sigma^2. \tag{193}$$

假设 5.7 对任意的 $x \in X$ 有

$$\mathbb{E}[\exp\{\|G(x, \xi_t) - \nabla f(x)\|_*^2\}] \leq \exp(1). \tag{194}$$

假设 5.6 是说随机梯度是无偏的，且方差小于 σ^2。由 Jensen 不等式，我们有如下包含关系：假设 5.7 $\Rightarrow$ 假设 5.6 (b)。

随机镜像下降法的复杂度

引理 5.4

设 $\{\xi_t\}$ 是独立同分布的随机变量序列，记 $\xi_{[t]} = (\xi_1, ..., \xi_t)$，设 $Z_t = Z_t(\xi_{[t]})$ 是关于 $\xi_{[t]}$ 的 *Borel* 函数，且 $\mathbb{E}[Z_t|\xi_{[t]}] = 0$ 和 $\mathbb{E}[\exp\{Z_t^2/\sigma_t^2\}|\xi_{[t]}] \leq \exp(1)$ 几乎处处成立，其中 σ_t 为常数，则对任意的 $\lambda > 0$ 有

$$\mathbf{Pr}\left\{\frac{\sum_{t=1}^N Z_t}{\sum_{t=1}^N \sigma^2} \geq \lambda\right\} \leq \exp\left(-\frac{\lambda^2}{3}\right). \tag{195}$$

我们将基于这个引理得到相关的复杂度结论。

随机镜像下降法的复杂度（基于假设 5.6）

定理 5.8 *

取步长 γ_t 满足 $0 < \gamma_t < \mu/(2L)$，令

$$\tilde{x}_{k+1} = \frac{\sum_{t=1}^{k} \gamma_t x_{t+1}}{\sum_{t=1}^{k} \gamma_t}, \tag{196}$$

其中 x_t 表示随机镜像下降法22的迭代点，则有如下收敛性结果：

- 在假设 5.6 下，

$$f(\tilde{x}_{k+1}) - f^* \leq K_0(k), \tag{197}$$

其中

$$K_0(k) := \frac{D_{w,X}^2 + 2\mu^{-1}\sigma^2 \sum_{t=1}^{k} \gamma_t^2}{\sum_{t=1}^{k} \gamma_t}. \tag{198}$$

随机镜像下降法的复杂度（基于假设 5.6、假设 5.7）

定理 5.8 *（续）

取步长 γ_t 满足 $0 < \gamma_t < \mu/(2L)$，令

$$\tilde{x}_{k+1} = \frac{\sum_{t=1}^{k} \gamma_t x_{t+1}}{\sum_{t=1}^{k} \gamma_t}, \tag{199}$$

其中 x_t 表示随机镜像下降法*22*的迭代点，则有如下收敛性结果：

- 在假设 *5.6*、假设 *5.7* 下：

$$\mathbf{Pr}\{f(\tilde{x}_{k+1}) - f^* \geq K_0(k) + \lambda K_1(k)\} \leq e^{-\lambda^2/3} + e^{-\lambda}, \tag{200}$$

其中

$$K_1(k) := \frac{2D_{w,X}\sqrt{\frac{2\sigma^2}{\mu}\sum_{t=1}^{k}\gamma_t^2} + \frac{2\sigma^2}{\mu}\sum_{t=1}^{k}\gamma_t^2}{\sum_{t=1}^{k}\gamma_t}. \tag{201}$$

随机镜像下降法的复杂度

当我们取固定步长

$$\gamma^* = \min\left\{\frac{\mu}{2L}, \sqrt{\mu D_{w,X}^2 2k\sigma^2}\right\} \tag{202}$$

时，可以得到光滑随机凸优化问题的 (ϵ, δ) 复杂度上界为

$$\mathcal{O}\left(\frac{L\Omega^2}{\epsilon} + \frac{\Omega^2\sigma^2}{\epsilon^2}\ln^2\frac{1}{\delta}\right). \tag{203}$$

光滑非凸随机优化问题

对于非凸随机优化问题，我们需要改进随机梯度法如下：

Algorithm 23: 随机迭代的随机梯度法（Randomized Stochastic Gradient (RSG) method）

Input: $x_1 \in \mathbb{R}^n$，步长序列 $\{\gamma_k\}$，最大迭代次数 N，随机迭代次数 R 的概率分布 $P_R(\cdot)$

设 R 是概率分布为 $P_R(\cdot)$ 的随机变量

for $k = 1, 2, ..., R$ **do**

在 (x_k, ξ_k) 处调用子程序 $\mathcal{SFO}$ 得到随机梯度 $G(x_k, \xi_k)$，并迭代

$$x_{k+1} = x_k - \gamma_k G(x_k, \xi_k). \tag{204}$$

end for

随机迭代的随机梯度法与随机梯度法的区别在于**其总迭代步数事先确定，且服从某一概率分布**。

随机迭代的随机梯度法的收敛速度

定理 5.9 *

选择步长 $\{\gamma_k\}$ 使得 $\gamma_k \leq 2/L$，且概率密度函数 $P_R(\cdot)$ 取

$$P_R(k) := \mathbf{Pr}\{R = k\} = \frac{2\gamma_k - L\gamma_k^2}{\sum_{k=1}^{N}(2\gamma_k - L\gamma_k^2)}, \tag{205}$$

则在假设 *5.6* 下，

- 对任意的 $N \geq 1$ 我们有

$$\frac{1}{L}\mathbb{E}_{R,\xi_{[N]}}[\|\nabla f(x_R)\|^2] \leq \frac{D_f^2 + \sigma^2 \sum_{k=1}^{N} \gamma_k^2}{\sum_{k=1}^{N}(2\gamma_k - L\gamma_k^2)}. \tag{206}$$

- 若 f 是凸函数，则对任意的 $N \geq 1$ 我们有

$$\mathbb{E}_{R,\xi_{[N]}}[f(x_R) - f^*] \leq \frac{D_X^2 + \sigma^2 \sum_{k=1}^{N} \gamma_k^2}{\sum_{k=1}^{N}(2\gamma_k - L\gamma_k^2)}. \tag{207}$$

随机镜像下降法的复杂度

基于定理 5.9，我们可以得到随机迭代的随机梯度法在求解非凸光滑的随机优化问题时的复杂度。

推论 5.2 *

对任意的常数 γ，取步长 $\{\gamma_k\}$：

$$\gamma_k = \min\left\{\frac{1}{L}, \frac{\gamma}{\sigma\sqrt{N}}\right\}, \tag{208}$$

则在假设 *5.6* 下，

- 对任意的 $N \geq 1$ 我们有

$$\frac{1}{L}\mathbb{E}_{R,\xi_{[N]}}[\|\nabla f(x_R)\|^2] \leq C(N) := \frac{LD_f^2}{N} + \left(\gamma + \frac{D_f^2}{\gamma}\right)\frac{\sigma}{\sqrt{N}}. \tag{209}$$

- 若 f 是凸函数，则对任意的 $N \geq 1$ 我们有

$$\mathbb{E}_{R,\xi_{[N]}}[f(x_R) - f^*] \leq \frac{LD_X^2}{N} + \left(\gamma + \frac{D_X^2}{\gamma}\right)\frac{\sigma}{\sqrt{N}}. \tag{210}$$

随机镜像下降法的复杂度 *

利用 Markov 不等式 $\mathbb{E}_{R,\xi_{[N]}}[\|\nabla f(x_R)\|^2] \leq C(N)$，随机迭代的随机梯度法跑一次的 (ϵ, δ) 复杂度为

$$\mathbf{Pr}\left\{\|\nabla f(x_R)\|^2 \geq \frac{L^2 D_f^2}{N}\lambda + \frac{2LD_f\sigma}{\sqrt{N}}\lambda\right\} \leq \frac{1}{\lambda}. \tag{211}$$

令 $\delta = 1/\lambda$ 并取 $\frac{L^2 D_f^2}{N}\lambda + \frac{2LD_f\sigma}{\sqrt{N}}\lambda \leq \epsilon$，可得随机迭代的随机梯度法 (ϵ, δ) 复杂度至多为

$$\mathcal{O}\left(\frac{1}{\delta\epsilon} + \frac{\sigma^2}{\delta^2\epsilon^2}\right). \tag{212}$$

集成随机迭代的随机梯度法

受到集成随机梯度法的启发，我们也可以对随机迭代的随机梯度法进行集成。

Algorithm 24: 集成随机迭代的随机梯度法 (Two-phase RGS (2-RSG) method)

Input: $x_1 \in \mathbb{R}^n$，最大迭代次数 N，集成次数 S，采样次数 T

for $d = 1, 2, ..., S$ **do**

  调用 RSG 算法，输入为 x_1，最大迭代次数为 N，步长 $\{\gamma_k\}$。RSG 迭代次数 R 的概率分布为 $P_R(\cdot)$。令 $\bar{x}_s$ 表示 RSG 的输出。

end for

从 $\bar{x}_1, ..., \bar{x}_S$ 中选择 $\bar{x}^*$ 输出，满足：

$$\|g(\bar{x}^*)\| = \min_{s=1,...,S} \|g(\bar{x}_s)\|, \quad g(\bar{x}_s) := \frac{1}{T} \sum_{k=1}^{T} G(\bar{x}_s, \xi_k). \tag{213}$$

Output: $\bar{x}^*$

集成随机迭代的随机梯度法的复杂度

定理 5.10 *

在假设 *5.6* 下，集成随机迭代的随机梯度法满足：

- $C(N)$ 服从(209)中的定义，对任意的 $\lambda > 0$ 我们有：

$$\mathbf{Pr}\{\|\nabla f(\bar{x}^*)\| \geq 2(4LC(N) + \frac{3\lambda\sigma^2}{T})\} \leq \frac{S+1}{\lambda} + 2^{-S}. \quad (214)$$

- 令 $\epsilon > 0$，$\delta \in (0,1)$ 给定，若参数 S, N, T 取

$$S = S(\delta) := \lceil \log(\frac{2}{\delta}) \rceil, \quad (215)$$

$$N = N(\epsilon) := \lceil \max\{\frac{32L^2D_f^2}{\epsilon}, 32L(\gamma + \frac{D_f^2}{\gamma})\frac{\sigma}{\epsilon}\} \rceil, \quad (216)$$

$$T = T(\epsilon, \delta) := \lceil \frac{24(S+1)\sigma^2}{\delta\epsilon} \rceil \quad (217)$$

则集成随机迭代的随机梯度法为得到 (ϵ, δ) 近似解至多需要调用 $S(\delta)(N(\epsilon) + T(\epsilon, \delta))$ 次 $\mathcal{SFO}$ 子程序。

基于随机迭代的算法的复杂度汇总

综上，随机梯度法在求解光滑函数时有如下 (ϵ, δ) 复杂度结论:

1. **随机迭代的随机梯度法**在假设 5.6 下 (ϵ, δ) 复杂度为

$$\mathcal{O}\left(\frac{1}{\delta\epsilon} + \frac{\sigma^2}{\delta^2\epsilon^2}\right). \tag{218}$$

2. **集成随机迭代的随机梯度法**在假设 5.6 下 (ϵ, δ) 复杂度为

$$\mathcal{O}\left(\frac{\log(1/\delta)}{\epsilon} + \frac{\sigma^2}{\epsilon^2}\log\frac{1}{\delta} + \frac{\log^2(1/\delta)\sigma^2}{\delta\epsilon}\right). \tag{219}$$

3. **集成随机迭代的随机梯度法**在假设 5.6 和假设 5.7 下 (ϵ, δ) 复杂度为

$$\mathcal{O}\left(\frac{\log(1/\delta)}{\epsilon} + \frac{\sigma^2}{\epsilon^2}\log\frac{1}{\delta} + \frac{\log^2(1/\delta)\sigma^2}{\epsilon}\right). \tag{220}$$