目　录

第一章　绪论

随着现实世界中数据的爆炸式增长，机器学习、金融、信息等领域涌现出一系列大规模优化问题，这些问题具有约束条件多、变量维度高、目标函数结构复杂等特点，对传统的优化方法提出了挑战。在求解大规模优化问题时，梯度下降法和牛顿法等传统方法往往面临计算速度慢、冗余计算量大、内存占用高、容易收敛到局部极值等困难。对此，研究者们对传统的算法进行改良，给出了一系列适用于大规模数据的随机优化算法。

在本章第 1 节，我们将简单介绍机器学习中的大规模优化问题以及求解大规模优化问题的随机梯度下降法（SGD），指出其缺陷和改进方案，由此引入自适应随机梯度方法（AdaGrad）；在本章第 2 , 3 节，我们则给出论文的结构和主要结论，并列举相关研究；在本章第 4 节，我们对论文的问题设定和所采用的记号做出说明。

1.1　引言：大规模优化问题与随机梯度方法

1.1.1　大规模优化问题

近年来人工智能的崛起给计算数学领域带来了大量亟待解决的大规模优化问题。在论文中，我们所提到的大规模优化问题均以机器学习为背景，相应的模型参见第四章。下面我们对机器学习中一类常见的优化问题作简单介绍。

机器学习中一项重要任务是根据输入的样本 $\mathbf{x}$ 预测其标签 $\mathbf{y}$. 解决该问题的普遍思路是，构造一个含参数 $\mathbf{w}$ 的预测函数 $h(\cdot, \mathbf{w})$，并根据已有数据，不断调整 $\mathbf{w}$ 的取值，使得预测函数返回值与真实值之间的差距尽可能小。具体而言，预测值 $h(\mathbf{x}, \mathbf{w})$ 与真实标签 $\mathbf{y}$ 之间的差距可以用**损失函数**（loss function）$l(h(\mathbf{x}; \mathbf{w}), \mathbf{y})$ 来刻画，损失越小表明预测越准确。我们希望预测函数在已有样例 $\{(\mathbf{a}_i, \mathbf{y}_i)\}_{i=1}^m$ 上的表现尽量好，这归结于极小化平均损失

$$\min_{\mathbf{w}\in D} R_{emp} := \frac{1}{m}\sum_{i=1}^{m} l(h(\mathbf{a}_i;\mathbf{w}),\mathbf{y}_i), \tag{1-1}$$

上面问题的目标函数 R_{emp} 又被称为**经验风险**（empirical risk）。

另一方面，为防止过拟合，我们还希望所得的模型不能过于复杂。模型复杂度通常由正则项 $r(\mathbf{w})$（常取 L^1 **正则项** $\|\mathbf{w}\|_1$ 或者 L^2 **正则项** $\|\mathbf{w}\|_2^2$）的大小来刻画。我们希望经验风险尽量小，同时正则项不能太大，因而可以考虑求解

$$\min_{\mathbf{w}\in D} R_{srm} := \frac{1}{m}\sum_{i=1}^{m} l(h(\mathbf{a}_i;\mathbf{w}),\mathbf{y}_i) + \lambda r(\mathbf{w}), \tag{1-2}$$

其中 $\lambda \geqslant 0$ 为**正则参数**，用以权衡经验风险和模型复杂度；目标函数 R_{srm} 称为**结构风险**（structural risk）。结构风险小的模型在已有的训练数据集和未知的测试数据集上都有较好的表现 [1]。

机器学习中许多优化问题都可以归结于极小化结构风险。在论文的数值实验部分，我们将考虑 Logistic 回归、支持向量机、多层感知机三种模型，它们所对应的优化问题均可以写成问题 (1-2) 的形式 [2]。

模型	**损失函数类型**	**性质**	**表达式**
Logistic 回归	交叉熵损失函数	光滑、凸	$\ln(1+e^{-y(\mathbf{w}\cdot\mathbf{x})})$
支持向量机	Hinge 损失函数	非光滑、凸	$\max\{0, y(\mathbf{w}\cdot\mathbf{x}+b)\}$
多层感知机	平方损失函数	光滑、非凸	$\sum_{j=1}^{l}(\hat{y}_j^k - y_j^k)^2$

1.1.2 随机梯度下降法

在问题 (1-2) 中，我们记 $f_i(\mathbf{w}) = l(h(\mathbf{a}_i;\mathbf{w}),\mathbf{y}_i) + \lambda r(\mathbf{w})$，则该问题可以写为

$$\min_{x\in D} f(x) := \frac{1}{m}\sum_{i=1}^{m} f_i(x), \tag{1-3}$$

其中 D 为问题的可行域。假设 $f_1,\cdots,f_m$ 均可微，则求解问题 (1-3) 可以采用传统的梯度下降法（GD）。但此时我们面临以下困难：

(1) 计算速度慢。每次迭代时，为计算 ∇f，我们必须计算每个 ∇f_i 并将其相加。当 m 很大时，这将相当耗时。

(2) 冗余计算量大。由于每个 f_i 是由样例的性质所确定的，当数据集中存在大量相似样例时，$\{f_i\}_{i=1}^m$ 中的很多函数也彼此相似。GD 在每次迭代时都重

复计算这些相似的函数的梯度，产生大量冗余计算。

(3) 内存占用大。为计算 f 的梯度，每次迭代时都需加载全部样本的信息。

(4) 难以对模型进行动态更新。添加新的样例时，我们需要重新计算每一步的梯度，难以满足在线学习（online learning）的需求。

(5) 问题病态时收敛速度慢。在“峡谷”区域（目标函数等高线为扁平的椭圆），GD 所产生的序列不断地在斜坡上振荡，逼近局部极值点的速度很慢。

最早出现的用于求解大规模优化问题(1-3)的优化方法是**随机梯度下降法**（SGD, stochastic gradient descent method）。该方法由 Herbert E. Robbins 在 1951 年首次提出 [3] ，并于 21 世纪在机器学习中普及。前面提到，若数据集中存在大量相似样例，则 $\{f_i\}_{i=1}^m$ 中很多函数也彼此相似。SGD 的基本想法是，在计算梯度时，随机地用这些函数中的一个来代替原函数。具体而言，在第 k 次迭代时，我们随机选取 $a_k \in \{1,2,\cdots,m\}$，用 ∇f_{a_k} 来代替 ∇f：

$$x^{k+1} = P_D(x^k - \eta_k \nabla f_{a_k}(x^k)),\ a_k \sim U(\{1,2,\cdots,m\}),\ k = 1,2,\cdots, \qquad (1\text{-}4)$$

其中 $P_D(x) = \text{argmin}_{u\in D} \|u - x\|_2$ 为 x 在可行域 D 上投影；$\eta_k > 0$ 表示步长；随机变量 a_k 相互独立，且服从均匀分布。

SGD 随机选择一个函数的操作不具有代表性，因而实际计算时我们往往随机选择一小批函数 $\{f_i\}_{i\in I_k}$ $(|I_k| \ll m)$，用 $\frac{1}{|I_k|}\sum_{i\in I_k} \nabla f_i$ 来代替 ∇f，这种方法又被称为**小批量梯度下降法**（MBGD, mini-batch gradient descent method）。有的文献也用术语 SGD 来表示 MBGD，在后文我们也不做区分 [1]。

相比于 GD，SGD 计算效率高、内存占用小、可用于在线学习。但 SGD 依然具有下面几个重要的不足：

(1) SGD 较难逃脱鞍点，在非凸问题上未必表现出好的收敛性质；

(2) SGD 未能克服 GD 在病态问题中收敛慢的缺陷；

(3) SGD 对步长的选取敏感：固定步长下 SGD 并不一定收敛；较大的步长会使迭代过程产生波动，甚至导致发散；较小的步长则导致收敛缓慢；

(4) SGD 在每个维度上都采用相同的步长 η_k，而不同的维度上目标函数的性质可能完全不同。

---

1 一般而言，SGD 只要求迭代时代入的梯度 $g(x^k, a_k)$ 为 $\nabla f(x^k)$ 的无偏估计，即 $\mathbf{E}(g(x^k, a_k)|x^k) = \nabla f(x^k)$. 在后文中，为了表述的方便，我们还是以 $g(x^k, a_k) = \nabla f_{a_k}(x^k)$, $a_k \sim U(\{1,\cdots,m\})$ 为例。

下面我们将要介绍的一系列基于 SGD 的随机优化算法可以改进上述不足之处。例如，通过自适应步长改善 (3) 和 (4)；利用动量加速技术改善 (1) 和 (2).

1.2 论文的结构与主要结论

在随后的章节中，我们将：

1. 介绍自适应随机梯度方法 AdaGrad 及其标量形式 AdaGrad-Norm，给出其迭代格式与动机（2.1 节）；
2. 说明 AdaGrad 在凸优化问题中具有 $O\left(1/\sqrt{T}\right)$ 的收敛率（定理 2.1 ）以及光滑非凸优化问题具有 $O\left(\ln T/\sqrt{T}\right)$ 的收敛速度（定理 2.4 ）；
3. 回顾 AdaGrad 和 AdaGrad-Norm 在任意初始步长下所具有的 $O\left(\sqrt{T}\right)$ 的遗憾界（定理 2.9，定理 2.11 ）；与 SGD 依赖于步长选取的相同遗憾界（引理 2.7 ）作对比；
4. 指出 AdaGrad 和 AdaGrad-Norm 的遗憾界为后见之明意义下最优遗憾界的 $\sqrt{2}$ 倍（第 22 页）；
5. 介绍改善 AdaGrad 步长过度衰减等缺陷的一系列算法（Ada-系列算法），如 RMSProp（算法 2，P26 ）, AdaDelta（算法 3，P27 ）, Adam（算法 4，P30 ）等，给出它们的迭代格式与动机；
6. 利用逐分量自适应步长的 Ada-系列算法对 Logistic 回归（4.1 节）、支持向量机（4.2 节）、多层感知机（4.3 节）三种模型进行训练，与各分量步长统一的 SGD 和 AdaGrad-Norm 算法进行对比，验证前者在稀疏数据集上的优越表现，以及在光滑凸优化、非光滑凸优化、光滑非凸优化三类问题上相对于后者的一致的优秀性能。

1.3 相关研究

AdaGrad 的提出可以追溯到 Peter Auer et al. (2002) [4] ，该文章首次将自适应步长的随机梯度方法用于解决在线学习中的优化问题。经典的 AdaGrad 算法则是分别由 H. Brendan McMahan, Matthew Streeter (2010) [5] 和 John Duchi et al. (2011) [6] 独立提出的。AdaGrad 这一名称正是来自后者的论文 *Adaptive Subgradient Methods for Online Learning and Stochastic Optimization*，该文章系统地介绍

了一系列自适应随机梯度方法，并将其用于随机优化。目前机器学习中广泛运用的 AdaGrad 及其标量形式 AdaGrad-Norm 是此论文中的一个特例，即该方法的对角形式（diagonal adaptation）。

作为 SGD 最有效的改进方法之一，AdaGrad 被广泛应用于现实中各种在线优化和随机优化问题。实验表明，AdaGrad 能够对出现频率很少的特征作出及时的响应，并在稀疏的数据集上表现出良好的性能 [7] 。具体而言，在文本分类、图像识别、薪资预测等现实问题上，AdaGrad 都具有优越表现 [6] ；在深度学习中，AdaGrad（及其改进算法 RMSprop, AdaDelta, Adam 等）也被广泛应用。然而，也有研究指出，AdaGrad 等自适应步长的算法在神经网络的训练中可能表现出较差的泛化性能（即使能找到更好的局部最小值） [8] 。

相较于 AdaGrad 在各领域的广泛应用，AdaGrad 的理论性质的研究则相对落后。AdaGrad 的标量形式 AdaGrad-Norm 的研究相对容易，结果也较多，比如 [6,9-16] ，但 AdaGrad-Norm 在实际优化问题中使用相对较少。而目前关于经典的 AdaGrad 的理论研究大多基于在线学习（而非随机优化）的情境。在此情境下，我们通过遗憾来描述其性能。Auer [4] 和 Duchi [6] 给出了 AdaGrad 的次线性遗憾界，并说明了 AdaGrad 的遗憾被理论最优遗憾所控制。对于 AdaGrad 遗憾界的研究还有 [17-18] 等。

在随机优化情境下，AdaGrad 的理论研究还相对较少。通过对遗憾界限的研究，我们可以得到算法在期望意义下 $O\left(1/\sqrt{T}\right)$ 的收敛率，但此时我们通常需要假定目标函数为凸函数、定义域为紧凸集，并且所得到的收敛率也并不一定是最优的。此外，对于目标函数非凸和定义域无界的情形，我们很难得到较强的收敛结论。非凸情形下已有的收敛结果，大多假设目标函数 L-光滑，并对随机梯度的噪声给出一定的限制，得到大约 $O\left(\ln T/\sqrt{T}\right)$ 的收敛速度。这里的噪声是指随机梯度与目标函数梯度的差距，多数文章要求

$$\left\|\nabla f_{a_i}(x)-\nabla f(x)\right\| \leqslant A(f(x)-f^*)+B\left\|\nabla f(x)\right\|^2+C, \tag{1-5}$$

其中 $\|\cdot\|$ 为范数，$A, B, C \geqslant 0$. (1-5) 式是较弱的噪声要求，本文的条件（随机梯度一致有界）蕴含着 (1-5) 式中 $B=1,\ A=0$ 的情况。此外，也有研究定性分析迭代点列是否趋于驻点，得到几乎必然收敛和渐进收敛的结论，比如 [12,19].

表 1-1　AdaGrad 与 AdaGrad-Norm 的收敛性结果汇总

优化方法	优化问题类型	反映收敛性的指标	参考文献
AdaGrad	凸，定义域有界	收敛率，期望意义下	[6,10]
AdaGrad	光滑非凸	目标函数梯度，期望意义下	[20-21]
AdaGrad	光滑非凸	目标函数梯度，大概率	[13,20,22]
AdaGrad-Norm	凸，定义域有界	收敛率，期望意义下	[6,9-10]
AdaGrad-Norm	光滑非凸	收敛率，期望意义下	[11-12]
AdaGrad-Norm	光滑非凸	收敛率，大概率	[12]
AdaGrad-Norm	光滑非凸	目标函数梯度，大概率	[13-16]

经典的 AdaGrad 算法最主要的缺点是：受历史梯度信息累积的影响，算法的步长单减趋于 0，因而在迭代后期收敛较慢，对新数据的学习能力较弱。基于 AdaGrad 的改进方法有 RMSProp, AdaDelta [23], Adam [24], AMSGrad [25] 等。这类算法也被称为 Ada-系列算法，在机器学习中应用广泛。对于这类算法的简单介绍，可参考 Sebastian Ruder (2016) [26], Léon Bottou (2016) [27] 等。

1.4　问题设定与记号说明

1.4.1　问题设定

论文中，我们考虑用随机梯度算法求解大规模优化问题[1]

$$\min_{x\in D} f(x) := \frac{1}{m}\sum_{i=1}^{m} f_i(x), \tag{1-6}$$

其中 D 为可行域（$\mathbb{R}^n$ 或者某紧凸集），$f_i(x)\ (i=1,2,\cdots,m)$ 是可微函数或凸函数，并且存在 $x^*\in D$，使得 $f^*=f(x^*)$ 是 f 的最小值。为了方便起见，下文我们主要就 f_i 为可微函数的情形展开讨论。如果 f_i 为不可微凸函数，我们用次梯度代替梯度，相应讨论是类似的。

问题 (1-6) 中，如果令 $f_i(\mathbf{w}) = l(h(\mathbf{a}_i,\mathbf{w});\mathbf{y}_i)+\lambda r(\mathbf{w})$，则该问题就化为问题 (1-2) 的特殊形式。此时 m 代表样本数量，往往很大；n 与特征数正相关；每个 f_i 都对应一个样例 $(\mathbf{a}_i,\mathbf{y}_i)$ 的损失（相差一个正则项），并且 f_i 由样例的性质

1　事实上，该问题是随机优化问题 $\min_{x\in D}\mathbf{E}_{\xi\sim\mathcal{D}}\left(f_\xi(x)\right)$ 的特殊形式，即 ξ 服从均匀分布的情况。论文中涉及的随机梯度方法都可以运用于一般的随机优化问题。

所唯一确定。我们所讨论的是一般问题(1-6)，但考虑特殊形式(1-2)有助于阐明方法的动机。

1.4.2 记号说明

- **梯度 次梯度**

  记号 $\nabla f(x) = (\nabla_1 f(x), \nabla_2 f(x), \cdots, \nabla_n f(x))$ 表示 f 在 x 处的梯度或者某一次梯度；

  $\partial f(x)$ 表示 f 在 x 处的次梯度集合。

- **向量运算**

  记号 $\|x\|_2$ 表示向量 x 的 l^2 范数，即 $\sqrt{\sum_{i=1}^n x_i^2}$；

  记号 $\|x\|_\infty$ 表示向量 x 的无穷范数，即 $\sup_{1\leqslant i\leqslant n}|x_i|$；

  记号 $\odot$ 表示向量的 Hadamard 积，即逐分量向量乘法

$$(a_1, \cdots, a_n) \odot (b_1, \cdots, b_n) := (a_1 b_1, \cdots, a_n b_n).$$

- **算法迭代格式**

  $x^k \in \mathbb{R}^n\ (k = 1, 2, \cdots)$ 表示迭代中产生的点列（行向量），上标表示指标，不表示幂次；x_j^k 中的下标 j 表示 x^k 的第 j 个分量；

  记号 P_D 表示向可行域 D 的投影；

  未经说明时，$\nabla f_{a_k}(x^k)$ 都代表第 k 次迭代时选择的随机梯度（或次梯度），这里 a_k 为服从均匀分布的、彼此独立的随机变量。

- **随机向量 期望 条件期望**

  我们将论文中的迭代点列 x^k、随机梯度 $\nabla f_{a_k}(x^k)$ 等都看作和诸 a_i 有关的随机向量。

  对于随机向量 $\mathbf{X}$，记号 $\mathbf{E}(\mathbf{X})$ 表示其数学期望（向量）；记号 $\mathbf{E}(\mathbf{X}|\mathbf{Y})$ 表示 $\mathbf{X}$ 关于随机向量 $\mathbf{Y}$ 的条件期望（随机向量）；$\mathbf{E}_k(\mathbf{X})$ 表示 $\mathbf{E}(\mathbf{X}|a_1, \cdots, a_k)$，$\mathbf{E}_0(\mathbf{X}) := \mathbf{E}(\mathbf{X})$；

  此外，随机变量 $X \geqslant Y$ 表示对于每个事件 $\omega \in \Omega$ 都有 $X(\omega) > Y(\omega)$.

- **样本 标签**

  记号 $(\mathbf{a}_i, y_i)$ 表示一个样例，其中 $\mathbf{a}_i$ 为样本，y_i 为其对应标签。

论文中的样本是指 $\mathbb{R}^d$ 中的向量，它的每个分量都代表一个特征；标签一般是实数，如果是向量，则用粗体 $\mathbf{y}_i$ 表示。

第二章　Adagrad 及其理论性质

本章我们将介绍求解大规模优化问题 (1-6) 的经典自适应随机梯度方法 AdaGrad 及其标量形式 AdaGrad-Norm，并给出其收敛性和遗憾界。

2.1　算法迭代格式及其动机

在用 SGD 求解大规模优化问题时，步长的选取往往是件难事。**自适应随机梯度下降法**（AdaGrad，adaptive stochastic gradient method）则通过动态调整步长解决这一困难。直观的想法是根据梯度信息来调整步长，在梯度较小（函数较为平缓）时采用大步长，梯度较大（函数较为陡峭）时采用小步长。基于此想法，我们考虑如下迭代格式：

$$\begin{aligned} G^k &= G^{k-1} + \left\|\nabla f_{a_k}(x^k)\right\|_2^2, \\ x^{k+1} &= P_D\left(x^k - \frac{\eta \nabla f_{a_k}(x^k)}{\sqrt{G^k + \varepsilon}}\right), \quad k = 1, \cdots, m, \end{aligned} \tag{2-1}$$

其中 $G^0 = 0$; $\eta > 0$ 为初始步长；$\varepsilon \approx 10^{-7}$ 为确保数值稳定性的小量，可避免分母产生 0. 由于迭代过程中记录了梯度的范数信息，上面的迭代算法常被称为 **AdaGrad-Norm**. 该算法的特点是迭代过程中步长递减，梯度较小时步长减小慢，梯度较大时步长减小快。

AdaGrad-Norm 在第 k 次迭代时的步长选取可写为

$$\eta_k = \frac{\eta}{\sqrt{\sum_{i=1}^k \left\|\nabla f_{a_i}(x^i)\right\|_2^2 + \varepsilon}}.$$

在论文的 2.3 节，我们将给出这样选取步长的动机。简而言之，如此选取步长可以使该方法的遗憾界被后见之明意义下（in hindsight）最优遗憾界 $R_{\min}$ 控制，详见定理 2.9 .

AdaGrad-Norm 实现了步长的自动调整，但该方法在每个维度上采用相同步长，未能充分考虑不同的维度上 f 的不同性质。经典的 **AdaGrad** 算法正是在每个方向上根据相应梯度分量信息来确定步长：

$$\begin{aligned} &G_j^k = G_j^{k-1} + (\nabla_j f_{a_k}(x^k))^2,\ j = 1, \cdots, n; \\ &x^{k+1} = P_D\left(x^k - \left(\frac{\eta \nabla_1 f_{a_k}(x^k)}{\sqrt{G_1^k + \varepsilon}}, \cdots, \frac{\eta \nabla_n f_{a_k}(x^k)}{\sqrt{G_n^k + \varepsilon}}\right)\right), \quad k = 1, \cdots, m, \end{aligned} \tag{2-2}$$

其中 $\nabla_j f_{a_k}$ 表示次梯度 ∇f_{a_k} 的第 j 个分量；$G_j^0 = 0\,(j = 1, \cdots, n)$；$\eta > 0$ 为初始步长；$\varepsilon \approx 10^{-7}$ 为数值稳定性小量。

引入 Hadamard 积记号 $\odot$，(2-2) 式可简写为

$$\begin{aligned} G^k &= G^{k-1} + \nabla f_{a_k}(x^k) \odot \nabla f_{a_k}(x^k), \\ x^{k+1} &= P_D\left(x^k - \frac{\eta}{\sqrt{G^k + \varepsilon}} \odot \nabla f_{a_k}(x^k)\right), \quad k = 1, \cdots, m, \end{aligned} \tag{2-3}$$

其中 $G^0 = \mathbf{0}$；除法、根号、加法均视为逐分量运算。

实验表明，对于稀疏的梯度数据，AdaGrad 的表现显著较好，而 (2-3) 式中逐分量自适应步长的操作是提高收敛性能的关键。

对于定义域为 $D = \mathbb{R}^n$ 的大规模优化问题 (1-6)，我们将求解该问题 AdaGrad 和 AdaGrad-Norm 算法完整步骤总结如下：

Algorithm 1 AdaGrad & AdaGrad-Norm

输入： 组成 $f(x) = \frac{1}{m}\sum_{i=1}^m f_i(x)$ 的诸 $f_i(x)$ 及其梯度或次梯度 $g_i(x)$ 计算式；批量大小 N；初始步长 $\eta > 0$；初始点 x^1；最大迭代次数 maxit；$\varepsilon > 0$.

1: $G \leftarrow \mathbf{0}$（对 AdaGrad-Norm：$G \leftarrow 0$）；$x \leftarrow x^1$；

2: **for** $i = 1$ to maxit **do**

3: 随机等可能地选取 $\{\xi_1, \cdots, \xi_N\} \subseteq \{1, 2, \cdots, m\}$, $g \leftarrow \frac{1}{N}\sum_{j=1}^N g_{\xi_j}(x)$；

4: $G \leftarrow G + g \odot g$（对 AdaGrad-Norm：$G \leftarrow G + \|g\|_2^2$）；

5: $x \leftarrow x - \frac{\eta}{\sqrt{G+\varepsilon}} \odot g$（对 AdaGrad-Norm：$x \leftarrow x - \frac{\eta}{\sqrt{G+\varepsilon}} \cdot g$）；

6: **end for**

输出： 函数 f 极小值点的估计值 x.

2.2 收敛性

AdaGrad 算法在实际优化问题中展现出良好的收敛性质，但在理论上，关于 AdaGrad 算法收敛性的研究还比较少。在本节，我们分别讨论凸优化和非凸优化两种情境。在凸函数假设下，我们可以通过研究遗憾界（详见 2.3 节）来得到期望意义下 $O\left(1/\sqrt{T}\right)$ 的收敛率。这就是下面的定理：

定理 2.1 设问题 (1-6) 中 D 为紧凸集，$f_1, \cdots, f_m$ 为凸函数，且 $\forall\, x \in D$, $i \in \{1, 2, \cdots, m\}$, $\left\|\nabla f_i(x)\right\|_\infty \leqslant M$. 用 AdaGrad 算法 (2-2) 求解该问题，记 $\bar{x}_T = \frac{1}{T}\sum_{i=1}^{T} x^t$, $x^* = \operatorname{argmin}_{x\in D} f(x)$，则

$$\mathbf{E}\left(f(\bar{x}_T) - f(x^*)\right) \leqslant \frac{n}{T}\sqrt{TM^2+\varepsilon}\left(\frac{d^2}{2\eta}+\eta\right), \tag{2-4}$$

其中 $d = \sup_{x,y\in D,\, 1\leqslant j\leqslant n} \left|x_j - y_j\right|$.

证明 根据定理 2.11，我们得到 AdaGrad 的遗憾界

$$R(T) \leqslant \sum_{j=1}^{n} \sqrt{TM^2+\varepsilon}\left(\frac{d_j^2}{2\eta}+\eta\right), \tag{2-5}$$

其中 $d_j = \sup_{x,y\in D} \left|x_j - y_j\right|$. 由 $d_j \leqslant d$ 可得

$$R(T) \leqslant B(T) := n\sqrt{TM^2+\varepsilon}\left(\frac{d^2}{2\eta}+\eta\right), \tag{2-6}$$

利用命题 2.6 即得欲证结论。 □

注： 用类似的方法可以证明 SGD 在相同情境下也可以具有 $O\left(1/\sqrt{T}\right)$ 的收敛率，此时通常要求步长取 $\eta_k = \eta/\sqrt{k}$.

关于遗憾界的具体讨论，我们在下一节中展开。下面我们主要讨论目标函数非凸、定义域为 $\mathbb{R}^n$ 时 AdaGrad 的收敛性。具体而言，我们考虑如下的优化问题：

$$\min_{x\in\mathbb{R}^n} f(x) := \frac{1}{m}\sum_{i=1}^{m} f_i(x), \tag{2-7}$$

其中诸 f_i 均为可微函数。为得到 AdaGrad 的收敛性结论，我们还需如下假设：

假设 1 (目标函数下有界) $\forall\, x \in \mathbb{R}^n,\, f(x) \geqslant f_*$.

假设 2 (随机梯度一致有界) $\exists\, M,\ \text{s.t.}\, \forall\, 1 \leqslant i \leqslant m,\, \forall\, x \in \mathbb{R}^n,\, \|\nabla f_i(x)\|_\infty \leqslant M$.

假设 3 (目标函数 L-光滑) $\forall\, x, y \in \mathbb{R}^n,\, \|\nabla f(x) - \nabla f(y)\|_2 \leqslant L\,\|x - y\|_2$.

本节的主要结果是

$$\mathbf{E}_t\left(\left\|\nabla f\left(x^t\right)\right\|_2^2\right) := \mathbf{E}\left(\frac{1}{T}\sum_{k=1}^{T}\left\|\nabla f\left(x^k\right)\right\|_2^2\right) = O\left(\frac{\ln T}{T}\right). \tag{2-8}$$

该结果可直观理解如下：在 $T-1$ 次迭代后，从迭代点列 $\{x^k\}_{k=1}^T$ 中随机选取一个点作为输出，则该点处目标函数梯度值的期望至多与 $\ln T/T$ 同阶。上述收敛性的刻画相较于收敛率而言是较弱的，因为在 L-光滑假设下，由引理 A.2 知

$$\|\nabla f(x^t)\|_2^2 \leqslant 2L\left(f(x^t) - f(x^*)\right), \tag{2-9}$$

其中 $x^* = \text{argmin}_{x\in\mathbb{R}^n} f(x)$. 如果我们能得到期望意义下的收敛率

$$\mathbf{E}_t\left(f(x^t) - f(x^*)\right) = O\left(\frac{\ln T}{T}\right),$$

就不必对 $\|\nabla f(x^t)\|_2^2$ 进行估计。遗憾的是，收敛率的估计是相当困难的。

下面我们对 AdaGrad 的收敛性进行具体分析。为方便起见，我们将迭代格式 (2-2) 改写为

$$\begin{aligned} G_j^k &= \sum_{i=1}^{k}(g_j^i)^2,\ u_j^k = \frac{g_j^k}{\sqrt{G_j^k + \varepsilon}}, \quad j = 1, \cdots, n\,; \\ x^{k+1} &= x^k - \eta\left(u_1^k, \cdots, u_n^k\right), \qquad\qquad k = 1, \cdots, m, \end{aligned} \tag{2-10}$$

其中 $g_j^i = \nabla_j f_{a_i}(x^i)\ (i \in \mathbb{N}_+;\, j = 1, \cdots, n)$；$a_k \in U(\{1, 2, \cdots, m\})\ (k \in \mathbb{N}_+)$ 且诸 a_k 相互独立，$\eta > 0, \varepsilon \approx 10^{-7}$. 此外，我们记 $g^k = (g_1^k, \cdots, g_n^k)$, $G^k = (G_1^k, \cdots, G_n^k)$, $u^k = (u_1^k, \cdots, u_n^k)$，并用 $\mathbf{E}_i(\cdot)$ 表示条件期望 $\mathbf{E}(\cdot \,|\, a_1, \cdots, a_i)$. 根据迭代格式 (2-10)，不难看出 $x^k = x^k(a_1, \cdots, a_{k-1})$, $g^k = g^k(a_1, \cdots, a_k)$, $u^k = u^k(a_1, \cdots, a_k)$，并且

$$\mathbf{E}_{k-1}\left(g^k\right)=\nabla f\left(x^k\right),\ k=2,3,\cdots. \tag{2-11}$$

在收敛性的研究中，条件 (2-11) 是本质的，该条件表明随机梯度是真实梯度的无偏估计。只要满足条件 (2-11)，下文的收敛性结论就成立。因此，本文的结论可推广到批量大小大于 1 和 a_k 不服从均匀分布的情况。

AdaGrad 的更新方向不一定是下降方向，这导致收敛性分析较为困难。下面的引理给出了更新方向与负梯度方向的平均偏差的一个下界。

引理 2.2 沿用上文的记号和假设，并记 $\mathbf{E}_0\left(\cdot\right)=\mathbf{E}\left(\cdot\right)$，则对 $\forall\ k\in\mathbb{N}_+$，成立

$$\mathbf{E}_{k-1}\left(\nabla f(x^k)\cdot u^k\right)\geqslant\frac{\left(\nabla f(x^k)\right)^2}{2\sqrt{kM^2+\varepsilon}}-2M\mathbf{E}_{k-1}\left(\left\|u^k\right\|_2^2\right), \tag{2-12}$$

证明 只需要证明对 $1\leqslant j\leqslant n$ 有

$$\mathbf{E}_{k-1}\left(\nabla_j f(x^k)u_j^k\right)\geqslant\frac{\left(\nabla_j f(x^k)\right)^2}{2\sqrt{kM^2+\varepsilon}}-2M\mathbf{E}_{k-1}\left(\left(u_j^k\right)^2\right). \tag{2-13}$$

我们将 $\mathbf{E}_{k-1}\left(\nabla_j f(x^k)u_j^k\right)$ 写为

$$\mathrm{I}=\mathbf{E}_{k-1}\left(\nabla_j f(x^k)\frac{g_j^k}{\sqrt{\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon}}\right)$$

与剩余部分 II 的和。I 的估计是较容易的。注意到 $\nabla_j f(x^k)$ 和 $\mathbf{E}_{k-1}\left(G_j^k\right)$ 均为 $a_1,\cdots,a_{k-1}$ 的函数而与 a_k 无关，根据条件期望的性质可得

$$I=\frac{\nabla_j f(x^k)\mathbf{E}_{k-1}\left(g_j^k\right)}{\sqrt{\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon}}=\frac{\left|\nabla_j f(x^k)\right|^2}{\sqrt{\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon}}. \tag{2-14}$$

我们将 II 写为 $\mathrm{II}=\mathbf{E}_{k-1}\left(\nabla_j f(x^k)g_j^k\cdot\mathrm{III}\right)$，其中

$$\begin{aligned}\mathrm{III}&=\frac{1}{\sqrt{G_j^k+\varepsilon}}-\frac{1}{\sqrt{\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon}}\\&=\frac{\mathbf{E}_{k-1}\left(G_j^k\right)-G_j^k}{\left(\sqrt{G_j^k+\varepsilon}+\sqrt{\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon}\right)\sqrt{G_j^k+\varepsilon}\sqrt{\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon}},\end{aligned} \tag{2-15}$$

由于 $g_j^1, \cdots, g_j^{k-1}$ 只和 $a_1, \cdots, a_{k-1}$ 有关，故

$$\mathbf{E}_{k-1}\left(G_j^k\right) - G_j^k = \mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right) - \left(g_j^k\right)^2,$$

由此可见 $|\nabla_j f(x^k) g_j^k \cdot \mathrm{III}| \leqslant \mathrm{IV} + \mathrm{V}$，其中

$$\mathrm{IV} = \frac{\left|\nabla_j f(x^k) g_j^k\right| \cdot \mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right)}{\sqrt{G_j^k + \varepsilon}\left(\mathbf{E}_{k-1}\left(G_j^k\right) + \varepsilon\right)}, \quad \mathrm{V} = \frac{\left|\nabla_j f(x^k) g_j^k\right| \cdot \left(g_j^k\right)^2}{\left(G_j^k + \varepsilon\right)\sqrt{\mathbf{E}_{k-1}\left(G_j^k\right) + \varepsilon}}. \tag{2-16}$$

我们首先估计 IV. 由基本不等式，我们有

$$\begin{aligned}\mathrm{IV} &= 2 \cdot \frac{\left|\nabla_j f(x^k)\right|}{2\left(\mathbf{E}_{k-1}\left(G_j^k\right) + \varepsilon\right)^{\frac{1}{4}}} \cdot \frac{\left|g_j^k\right| \cdot \mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right)}{\left(\mathbf{E}_{k-1}\left(G_j^k\right) + \varepsilon\right)^{\frac{3}{4}}\left(G_j^k + \varepsilon\right)^{\frac{1}{2}}} \\ &\leqslant \frac{\left|\nabla_j f(x^k)\right|^2}{4\left(\mathbf{E}_{k-1}\left(G_j^k\right) + \varepsilon\right)^{\frac{1}{2}}} + \frac{\left|g_j^k\right|^2\left(\mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right)\right)^2}{\left(\mathbf{E}_{k-1}\left(G_j^k\right) + \varepsilon\right)^{\frac{3}{2}}\left(G_j^k + \varepsilon\right)},\end{aligned} \tag{2-17}$$

此外，根据 $G_j^k = \sum_{i=1}^k \left(g_j^i\right)^2$ 可见

$$G_j^k + \varepsilon > \left(g_j^k\right)^2, \ \mathbf{E}_{k-1}\left(G_j^k\right) + \varepsilon > \mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right), \tag{2-18}$$

从而由 (2-17) 式和 (2-18) 式我们得到

$$\mathrm{IV} \leqslant \frac{\left|\nabla_j f(x^k)\right|^2}{4\left(\mathbf{E}_{k-1}\left(G_j^k\right) + \varepsilon\right)^{\frac{1}{2}}} + \frac{\left|g_j^k\right|^2 \mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right)}{\left(\mathbf{E}_{k-1}\left(G_j^k\right) + \varepsilon\right)^{\frac{1}{2}}\left(G_j^k + \varepsilon\right)}. \tag{2-19}$$

由于 $\nabla_j f(x^k)$ 和 $\mathbf{E}_{k-1}(\cdot)$ 都只和 $a_1, \cdots, a_{k-1}$ 有关，在 (2-19) 式两边取条件期望可得

$$\mathbf{E}_{k-1}(\mathrm{IV}) \leqslant \frac{\left|\nabla_j f(x^k)\right|^2}{4\left(\mathbf{E}_{k-1}\left(G_j^k\right) + \varepsilon\right)^{\frac{1}{2}}} + \frac{\mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right)}{\left(\mathbf{E}_{k-1}\left(G_j^k\right) + \varepsilon\right)^{\frac{1}{2}}}\mathbf{E}_{k-1}\left(\frac{\left|g_j^k\right|^2}{G_j^k + \varepsilon}\right). \tag{2-20}$$

对 V 我们同样利用基本不等式进行估计：

$$\mathrm{V} \leqslant \frac{\left|\nabla_j f(x^k) \cdot g_j^k\right|^2}{4\mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right)\left(\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon\right)^{\frac{1}{2}}} + \frac{\left|g_j^k\right|^4 \mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right)}{\left(\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon\right)^{\frac{1}{2}}\left(G_j^k+\varepsilon\right)^2}. \quad (2\text{-}21)$$

对 (2-21) 式两边取条件期望，结合 (2-18) 式，可得与 (2-20) 式右端相同的上界

$$\mathbf{E}_{k-1}(\mathrm{V}) \leqslant \frac{\left|\nabla_j f(x^k)\right|^2}{4\left(\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon\right)^{\frac{1}{2}}} + \frac{\mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right)}{\left(\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon\right)^{\frac{1}{2}}}\mathbf{E}_{k-1}\left(\frac{\left|g_j^k\right|^2}{G_j^k+\varepsilon}\right). \quad (2\text{-}22)$$

此外，结合 $\left|g_j^k\right| \leqslant M$ 和 (2-18) 式，可见

$$\frac{\mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right)}{\left(\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon\right)^{\frac{1}{2}}} < \left(\mathbf{E}_{k-1}\left(\left(g_j^k\right)^2\right)\right)^{\frac{1}{2}} \leqslant M, \quad (2\text{-}23)$$

结合 (2-20) 式， (2-22) 式和 (2-23) 式，我们得到 II 的估计

$$\begin{aligned}\mathrm{II} &= \mathbf{E}_{k-1}\left(\nabla_j f(x^k) g_j^k \cdot \mathrm{III}\right) \geqslant \mathbf{E}_{k-1}(-\mathrm{IV}-\mathrm{V}) \\ &\geqslant -\frac{\left|\nabla_j f(x^k)\right|^2}{2\left(\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon\right)^{\frac{1}{2}}} - 2M\mathbf{E}_{k-1}\left(\frac{\left|g_j^k\right|^2}{G_j^k+\varepsilon}\right). \end{aligned} \quad (2\text{-}24)$$

结合 (2-14) 式和 (2-24) 式可见

$$\mathbf{E}_{k-1}\left(\nabla_j f(x^k) u_j^k\right) \geqslant \frac{\left|\nabla_j f(x^k)\right|^2}{2\left(\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon\right)^{\frac{1}{2}}} - 2M\mathbf{E}_{k-1}\left(\frac{\left|g_j^k\right|^2}{G_j^k+\varepsilon}\right). \quad (2\text{-}25)$$

注意到

$$2\left(\mathbf{E}_{k-1}\left(G_j^k\right)+\varepsilon\right)^{\frac{1}{2}} \leqslant 2\left(kM^2+\varepsilon\right)^{\frac{1}{2}}, \quad \left(u_j^k\right)^2 = \frac{\left|g_j^k\right|^2}{G_j^k+\varepsilon},$$

我们就得到了 (2-13) 式。 □

引理 2.3 沿用上文的记号和假设，AdaGrad 迭代 T 轮后，每步前进长度的平方和的期望值有如下估计：

$$\sum_{k=1}^{T}\mathbf{E}\left(\left\|\eta u^k\right\|_2^2\right)\leqslant n\eta^2\ln\left(1+\frac{TM^2}{\varepsilon}\right). \tag{2-26}$$

证明 根据 u^k 的定义有

$$\sum_{k=1}^{T}\mathbf{E}\left(\left\|u^k\right\|_2^2\right)=\sum_{j=1}^{n}\mathbf{E}\left(\sum_{k=1}^{T}\frac{\left(g_j^k\right)^2}{\sum_{s=1}^{k}\left(g_j^s\right)^2+\varepsilon}\right). \tag{2-27}$$

在引理 A.3 中令 $a_k=\left(g_j^k\right)^2$，得到

$$\sum_{k=1}^{T}\mathbf{E}\left(\left\|u^k\right\|_2^2\right)\leqslant\sum_{j=1}^{n}\mathbf{E}\left(\ln\left(1+\frac{\sum_{s=1}^{T}\left(g_j^s\right)^2}{\varepsilon}\right)\right)\leqslant n\ln\left(1+\frac{TM^2}{\varepsilon}\right). \tag{2-28}$$

最后一个不等号利用了假设 2. □

下面我们给出本节的主要结果。

定理 2.4 对于满足假设 1，2，3 的问题 (2-7)，我们利用 AdaGrad 算法 (2-10) 进行求解，$T-1$ 次迭代后，从迭代点列中随机选取一点，该点处梯度范数平方的期望至多与 $\ln T/\sqrt{T}$ 同阶。具体而言，我们有

$$\mathbf{E}\left(\frac{1}{T}\sum_{k=1}^{T}\left\|\nabla f\left(x^k\right)\right\|_2^2\right)\leqslant\frac{1}{\sqrt{T}}\left(A\ln\left(1+\frac{M^2}{\varepsilon}T\right)+B(f(x^1)-f_*)\right), \tag{2-29}$$

其中 $A=n(4M+L\eta)\sqrt{M^2+\varepsilon}$, $B=2\sqrt{M^2+\varepsilon}/\eta$.

证明 根据假设 3 和引理 A.1，$\forall\, k\in\mathbb{N}_+$，我们有

$$f(x^{k+1})\leqslant f(x^k)-\eta u^k\cdot\nabla f(x^k)+\frac{L}{2}\left\|\eta u^k\right\|_2^2, \tag{2-30}$$

(2-30) 式两端求条件期望，并利用引理 2.2 可见

$$\begin{aligned}\mathbf{E}_{k-1}\left(f(x^{k+1})\right)&\leqslant f(x^k)-\eta\mathbf{E}_{k-1}\left(u^k\cdot\nabla f(x^k)\right)+\frac{L\eta^2}{2}\mathbf{E}_{k-1}\left(\left\|u^k\right\|_2^2\right)\\&\leqslant f(x^k)-\frac{\eta\left\|\nabla f(x^k)\right\|_2^2}{2\sqrt{kM^2+\varepsilon}}+\left(2M\eta+\frac{L\eta^2}{2}\right)\mathbf{E}_{k-1}\left(\left\|u^k\right\|_2^2\right).\end{aligned} \tag{2-31}$$

记 $C=2M\eta+\frac{L\eta^2}{2}$, $M_1=\sqrt{TM^2+\varepsilon}\geqslant\sqrt{kM^2+\varepsilon}$，对 $k=1,2,\cdots,T$ 所对应的 (2-31) 式求和，可见

$$\sum_{k=1}^{T}\mathbf{E}_{k-1}\left(f(x^{k+1})\right)\leqslant\sum_{k=1}^{T}\left(f(x^k)-\frac{\eta\left\|\nabla f(x^k)\right\|_2^2}{2M_1}+C\mathbf{E}_{k-1}\left(\left\|u^k\right\|_2^2\right)\right). \quad (2\text{-}32)$$

由条件期望的性质，对 $k\in\mathbb{N}_+$ 有 $\mathbf{E}\left(\mathbf{E}_{k-1}\left(\cdot\right)\right)=\mathbf{E}\left(\cdot\right)$. 对 (2-32) 式两端求期望可见

$$\mathbf{E}\left(f(x^{T+1})\right)\leqslant\mathbf{E}\left(f(x^1)\right)+\sum_{k=1}^{T}\left(-\frac{\eta\mathbf{E}\left(\left\|\nabla f(x^k)\right\|_2^2\right)}{2M_1}+C\mathbf{E}\left(\left\|u^k\right\|_2^2\right)\right). \quad (2\text{-}33)$$

注意到 $\mathbf{E}\left(f(x^1)-f(x^{T+1})\right)\leqslant f(x^1)-f_*$，整理可得

$$\mathbf{E}\left(\frac{1}{T}\sum_{k=1}^{T}\left\|\nabla f(x^k)\right\|_2^2\right)\leqslant\frac{2M_1}{T\eta}\left(f(x^1)-f_*\right)+\frac{2CM_1}{T\eta}\sum_{k=1}^{T}\mathbf{E}\left(\left\|u^k\right\|_2^2\right). \quad (2\text{-}34)$$

结合 (2-34) 式和引理 2.3 中 (2-28) 式，我们有

$$\mathbf{E}\left(\frac{1}{T}\sum_{k=1}^{T}\left\|\nabla f(x^k)\right\|_2^2\right)\leqslant\frac{2M_1}{T\eta}\left(f(x^1)-f_*\right)+\frac{2CM_1n}{T\eta}\ln\left(1+\frac{TM^2}{\varepsilon}\right), \quad (2\text{-}35)$$

注意到

$$\frac{M_1}{T}=\frac{1}{\sqrt{T}}\sqrt{M^2+\frac{\varepsilon}{T}}\leqslant\frac{1}{\sqrt{T}}\sqrt{M^2+\varepsilon},$$

代入 (2-35) 式即得欲证结论。 □

2.3 遗憾界

遗憾（regret）是一种用于评价在线学习 [1] 算法性能的指标。虽然论文主要将随机梯度方法用于解决随机优化问题 (1-6) 而非在线优化问题，但通过对遗憾上界的估计，我们可以得到期望意义下的收敛、依概率收敛等重要结论（参见命题 2.6 ）。因此，对遗憾的讨论具有重要的理论意义。

1 机器学习中，提前加载所有数据并以此构建模型的方法被称为“批量学习”或者“离线学习”。在线学习（online learning）则考虑数据持续增长的情形，用当前时刻的数据更新模型。前文提到的 SGD 和 AdaGrad 等方法都可以平行地应用到在线学习中。

本节我们将证明：在凸函数和梯度一致有界的假设下，对于任何初始步长，AdaGrad-Norm 和 AdaGrad 都具有 $O(\sqrt{T})$ 的遗憾界（定理 2.9，定理 2.11），而 SGD 要达成相同的遗憾界则需要恰当地选取步长（引理 2.7）。上述结果印证了 SGD 对步长选取的敏感性，也从理论上说明了自适应随机梯度方法可以克服该缺陷。

定义 2.5 对于某一随机梯度方法，假设第 k 次迭代的起始点为 x^k，所选取的随机梯度（或次梯度）是 $\nabla f_{a_i}(x^k)$，称

$$R(T)=\sum_{i=1}^{T}\left(f_{a_i}(x^i)-f_{a_i}(x_T^*)\right) \tag{2-36}$$

为该方法迭代 T 轮所产生的遗憾，其中 $x_T^*=\operatorname{argmin}_{x\in D}\sum_{i=1}^{T}f_{a_i}(x)$；$D\subseteq\mathbb{R}^n$ 为问题的可行域，通常是紧凸集；$f_i\ (i=1,2,\cdots,m)$ 通常是凸函数。

在问题 (1-1) 的特殊形式下，遗憾就表示在线算法的总损失与离线算法的理论最小损失之间的差值。我们通常要求算法达到次线性的遗憾，即 $R(T)=o(T)$（有的文献也将该条件称为 Hannan 一致性）。此时，我们可以得到期望意义上的收敛和依概率收敛等结论。

下面的命题给出了遗憾在研究随机优化问题 (1-6) 时的作用。此时我们本质上是用在线学习的思路来求解随机优化问题，这被称为在线-批量转换（online-to-batch conversion）。

命题 2.6 设问题 (1-6) 中 f 为凸函数，求解该问题的某随机梯度算法的迭代点列为 $\{x^k\}_{k\in\mathbb{Z}_+}$，第 k 次迭代所选的随机梯度（或次梯度）是 $\nabla f_{a_k}(x^k)$（其中 $a_k\sim U(\{1,2,\cdots,m\})$ 相互独立），并且该方法具有与诸 a_k 无关的遗憾界 $B(T)\geqslant R(T)$，则

$$0\leqslant\mathbf{E}(f(\bar{x}_T))-f(x^*)\leqslant\frac{B(T)}{T}. \tag{2-37}$$

其中 $\bar{x}_T=\frac{1}{T}\sum_{i=1}^{T}x^t$, $x^*=\operatorname{argmin}_{x\in D}\ f(x)$. 特别地，若 $B(T)=o(T)$，我们有

$$\mathbf{E}\left(f\left(\bar{x}_T\right)\right)\to f(x^*),\quad f(\bar{x}_T)\xrightarrow{\mathbf{P}}f(x^*)\quad(T\to\infty).$$

证明 由遗憾的定义可见

$$B(T) \geqslant R(T) = \sum_{i=1}^{T} f_{a_i}(x^i) - \min_{x\in D} \sum_{i=1}^{T} f_{a_i}(x) \geqslant \sum_{i=1}^{T} f_{a_i}(x^i) - \sum_{i=1}^{T} f_{a_i}(x^*), \quad (2\text{-}38)$$

由于 $a_1, \cdots, a_T$ 相互独立，$x^i = x^i(a_1, \cdots, a_{i-1})$，故 x^i 与 a_i 独立，因而 $\mathbf{E}(f_{a_i}(x^i)) = \mathbf{E}(f(x^i))$. 对 (2-38) 式两边求期望，可得

$$B(T) \geqslant \mathbf{E}\left(\sum_{i=1}^{T} f(x^i)\right) - Tf(x^*) \geqslant T\left(\mathbf{E}(f(\bar{x}_T)) - f(x^*)\right),$$

此外，由 x^* 为极小值点知

$$\mathbf{E}(f(\bar{x}_T)) - f(x^*) = \mathbf{E}(f(\bar{x}_T) - f(x^*)) \geqslant 0,$$

故 (2-37) 式成立。

当 $B(T) = o(T)$ 时，由 (2-37) 式可见 $\mathbf{E}(f(\bar{x}_T)) - f(x^*) \to 0$ $(T \to \infty)$. 另一方面，对 $\forall\, \varepsilon > 0$，由 Markov 不等式知

$$\mathbf{P}\left(\{f(\bar{x}_T) - f(x^*) > \varepsilon\}\right) \leqslant \frac{1}{\varepsilon}\mathbf{E}\left(f(\bar{x}_T) - f(x^*)\right) \to 0 \quad (T \to \infty)$$

故 $f(\bar{x}_T) \xrightarrow{\mathbf{P}} f(x^*)$ $(T \to \infty)$. □

在下面的引理中，我们给出变步长的 SGD 的遗憾界。

引理 2.7 (Zinkevich) [28] 设问题 (1-6) 中 $f_1, \cdots, f_m$ 均为凸函数，利用迭代格式为 (1-4) 的步长递减的 SGD 求解该问题，则迭代 T 轮后所得的遗憾具有上界

$$R(T) \leqslant \frac{1}{2}\left(\frac{d^2}{\eta_T} + \sum_{i=1}^{T} \eta_i \left\|\nabla f_{a_i}(x^i)\right\|_2^2\right), \quad (2\text{-}39)$$

其中 $d = \operatorname{diam} D$，$\eta_1 \geqslant \eta_2 \geqslant \cdots \geqslant \eta_T$ 为步长。

证明 设 $x_T^* = \operatorname{argmin}_{x\in D} \sum_{i=1}^{T} f_{a_i}(x)$，由诸 f_i 的凸性知

$$R(T) = \sum_{i=1}^{T}\left(f_{a_i}(x^i) - f_{a_i}(x_T^*)\right) \leqslant \sum_{i=1}^{T}\left((x^i - x_T^*)\cdot \nabla f_{a_i}(x^i)\right). \quad (2\text{-}40)$$

记 $y^{i+1} = x^i - \eta_i \nabla f_{a_i}(x^i)$，则 $x^{i+1} = P_D(y^{i+1})$，并且

$$\left\|y^{i+1}-x_T^*\right\|_2^2=\left\|x^i-x_T^*\right\|_2^2+\eta_i^2\left\|\nabla f_{a_i}(x^i)\right\|_2^2-2\eta_i\nabla f_{a_i}(x^i)\cdot(x^i-x_T^*).$$

根据投影的定义，由 $x_T^*\in D$ 知 $\left\|y^{i+1}-x_T^*\right\|_2\geqslant\left\|x^{i+1}-x_T^*\right\|_2$，从而

$$2\eta_i\nabla f_{a_i}(x^i)\cdot(x^i-x_T^*)\leqslant\left\|x^i-x_T^*\right\|_2^2+\eta_i^2\left\|\nabla f_{a_i}(x^i)\right\|_2^2-\left\|x^{i+1}-x_T^*\right\|_2^2,\quad(2\text{-}41)$$

代入 (2-40) 式得

$$R(T)\leqslant\sum_{i=1}^{T}\frac{1}{2\eta_i}\left(\left\|x^i-x_T^*\right\|_2^2-\left\|x^{i+1}-x_T^*\right\|_2^2\right)+\sum_{i=1}^{T}\frac{\eta_i}{2}\left\|\nabla f_{a_i}(x^i)\right\|_2^2.\quad(2\text{-}42)$$

记 $I=\sum_{i=1}^{T}\frac{1}{2\eta_i}\left(\left\|x^i-x_T^*\right\|_2^2-\left\|x^{i+1}-x_T^*\right\|_2^2\right)$，重新组合求和得

$$\begin{aligned}I&\leqslant\frac{\left\|x^1-x_T^*\right\|_2^2}{2\eta_1}+\sum_{i=2}^{T}\left(\frac{1}{2\eta_i}-\frac{1}{2\eta_{i-1}}\right)\left\|x^i-x_T^*\right\|_2^2-\frac{\left\|x^{T+1}-x_T^*\right\|_2^2}{2\eta_T}\\&\leqslant\frac{d^2}{2\eta_1}+\sum_{i=2}^{T}\left(\frac{1}{2\eta_i}-\frac{1}{2\eta_{i-1}}\right)d^2=\frac{d^2}{2\eta_T}.\end{aligned}\quad(2\text{-}43)$$

结合 (2-42) 式和 (2-43) 式即得欲证结论。 □

上述引理的直接推论是，若随机梯度 $\nabla f_{a_i}(x^i)$ 均有界，在凸函数假设下，取步长 $\eta_k=\eta/\sqrt{k}$ $(\eta>0)$ 可以让 SGD 达到 $O(\sqrt{T})$ 的遗憾界。此外，对于强凸函数，我们有平行的结果：

引理 2.8 在引理 2.7 的条件下，若对 $i=1,\cdots,m$，f_i 还是 **λ-强凸函数**，即 $f_i(x)-\frac{\lambda}{2}\left\|x\right\|_2^2$ 为凸函数，并且

$$\frac{1}{\eta_1}\geqslant\lambda,\ \frac{1}{\eta_i}-\frac{1}{\eta_{i-1}}\geqslant\lambda,\ \ i=2,\cdots,T,$$

则遗憾界可以降低为

$$R(T)\leqslant\frac{1}{2}\left(d^2\left(\frac{1}{\eta_T}-\lambda T\right)+\sum_{i=1}^{T}\eta_i\left\|\nabla f_{a_i}(x^i)\right\|_2^2\right).\quad(2\text{-}44)$$

证明 对 $f_{a_i}(x)-\frac{\lambda}{2}\|x\|_2^2$ 应用凸函数的一阶条件易得

$$f_{a_i}(x^i)-f_{a_i}(x_T^*)\leqslant(x^i-x_T^*)\cdot\nabla f_{a_i}(x^i)-\frac{\lambda}{2}\left\|x^i-x_T^*\right\|_2^2, \tag{2-45}$$

从而

$$R(T)\leqslant\sum_{i=1}^{T}\left((x^i-x_T^*)\cdot\nabla f_{a_i}(x^i)-\frac{\lambda}{2}\left\|x^i-x_T^*\right\|_2^2\right). \tag{2-46}$$

其余证明与引理 2.7 是完全类似的。 □

由上述引理可见，若随机梯度 $\nabla f_{a_i}(x^i)$ 均有界，在 λ-强凸函数假设下，取步长 $\eta_k=1/(\lambda k)$ 可以让 SGD 达到 $O(\ln T)$ 的遗憾界。

AdaGrad-Norm 的遗憾界

在引理 2.7 中，取

$$\eta_k=\frac{d}{\sqrt{\sum_{i=1}^{T}\left\|\nabla f_{a_i}(x^i)\right\|_2^2}},\ \ k=1,\cdots,T,$$

则此时 $R(T)$ 达到最小值

$$R_{\min}=d\sqrt{\sum_{i=1}^{T}\left\|\nabla f_{a_i}(x^i)\right\|_2^2}. \tag{2-47}$$

上述最小值只能是理论上的，因为我们事先并不知道每个 $\nabla f_{a_i}(x^i)$ 的取值。不过，步长 η_i 的取法仍给我们带来的一些启发。AdaGrad-Norm 算法的动机正是基于此。如果我们只考虑已得的梯度信息，令

$$\eta_k=\frac{\eta}{\sqrt{\sum_{i=1}^{k}\left\|\nabla f_{a_i}(x^i)\right\|_2^2+\varepsilon}},\ \ k=1,2,\cdots,T, \tag{2-48}$$

根据引理 2.7，我们有遗憾界

$$R(T) \leqslant \frac{d^2}{2\eta}\sqrt{\sum_{i=1}^{T}\left\|\nabla f_{a_i}\left(x^i\right)\right\|_2^2+\varepsilon}+\frac{\eta}{2}\sum_{i=1}^{T}\frac{\left\|\nabla f_{a_i}\left(x^i\right)\right\|_2^2}{\sqrt{\sum_{j=1}^{i}\left\|\nabla f_{a_j}\left(x^j\right)\right\|_2^2+\varepsilon}}. \tag{2-49}$$

利用引理 A.4 可得

$$\sum_{i=1}^{T}\frac{\left\|\nabla f_{a_i}\left(x^i\right)\right\|_2^2}{\sqrt{\sum_{j=1}^{i}\left\|\nabla f_{a_j}\left(x^j\right)\right\|_2^2+\varepsilon}} \leqslant 2\sqrt{\sum_{i=1}^{T}\left\|\nabla f_{a_i}\left(x^j\right)\right\|_2^2+\varepsilon}. \tag{2-50}$$

结合 (2-49) 和 (2-50) 两式，我们可以得到 AdaGrad-Norm 算法的遗憾界估计，这就是下面的定理：

定理 2.9 设问题 (1-6) 中 $f_1,\cdots,f_m$ 均为凸函数，利用 AdaGrad-Norm 算法 (2-1) 求解该问题，则迭代 T 轮后所得的遗憾具有上界

$$R(T) \leqslant \left(\frac{d^2}{2\eta}+\eta\right)\sqrt{\sum_{i=1}^{T}\left\|\nabla f_{a_i}\left(x^i\right)\right\|_2^2+\varepsilon}, \tag{2-51}$$

其中 $d=\operatorname{diam} D$.

该定理的直接推论是，随机梯度 $\nabla f_{a_i}\left(x^i\right)$ 的 l^2 范数一致有界时，AdaGrad-Norm 也达到 $O(\sqrt{T})$ 的遗憾界。

注： 在定理 2.9 中取 $\eta=d/\sqrt{2},\varepsilon=0$，则 (2-51) 式给出的遗憾界达到最小值 $\sqrt{2}R_{\min}$，其中 $R_{\min}=d\sqrt{\sum_{i=1}^{T}\left\|\nabla f_{a_i}\left(x^i\right)\right\|_2^2}$ 是 (2-47) 式中所确定的 SGD 遗憾界的理论最小值。这表明即使我们不知道每步迭代时的梯度，AdaGrad-Norm 的遗憾界也不会劣于后见之明意义下最优遗憾界的 $\sqrt{2}$ 倍。

AdaGrad 的遗憾界

经典的 AdaGrad 可以看成是逐分量的 AdaGrad-Norm，其第 i 个分量在第 k 次迭代时的步长为

$$\eta_{k,i}=\frac{\eta}{\sqrt{\sum_{j=1}^{k}\left\|\nabla_i f_{a_j}\left(x^j\right)\right\|_2^2+\varepsilon}}.$$

一般而言，对于各分量步长不同的随机梯度算法，我们依然可以利用 (2-40) 式对遗憾进行估计：

$$R(T) \leqslant \sum_{i=1}^{T} \left((x^i - x_T^*) \cdot \nabla f_{a_i}(x^i) \right) = \sum_{j=1}^{n} B_j(T), \tag{2-52}$$

其中 $B_j(T) = \sum_{i=1}^{T} \left((x_j^i - (x_T^*)_j) \nabla_j f_{a_i}(x^i) \right)$ 是算法单独作用于第 j 个分量的遗憾界。这表明，把在每个分量上利用 (2-40) 式放缩所得的遗憾界相加，可以得到 $R(T)$ 的一个上界。据此，我们可以将引理 2.7 推广为：

引理 2.10 设问题 (1-6) 中 $f_1, \cdots, f_m$ 均为凸函数，利用迭代格式为

$$x_j^{k+1} = x_j^k - \eta_{k,j} \nabla_j f_{a_k}\left(x^k\right),\ j = 1, \cdots, n;\ k \in \mathbb{N}_+. \tag{2-53}$$

的逐分量步长递减随机梯度算法求解该问题，迭代 T 次后的遗憾具有估计

$$R(T) \leqslant \sum_{j=1}^{n} B_j(T) = \sum_{j=1}^{n} \frac{1}{2} \left(\frac{d_j^2}{\eta_{T,j}} + \sum_{i=1}^{T} \eta_{i,j} \left| \nabla_j f_{a_i}\left(x^i\right) \right|^2 \right), \tag{2-54}$$

其中 $B_j(T)$ 代表算法作用在第 j 个分量上的遗憾界，$d_j = \sup_{x,y \in D} |x_j - y_j|$，$\eta_{1,j} \geqslant \eta_{2,j} \geqslant \cdots \geqslant \eta_{T,j} > 0\ (j = 1, 2, \cdots, n)$.

根据引理 2.10 和定理 2.9，我们可以给出经典的 AdaGrad 的遗憾界：

定理 2.11 设问题 (1-6) 中 $f_1, \cdots, f_m$ 均为凸函数，利用 AdaGrad 算法 (2-3) 求解该问题，则迭代 T 轮后所得的遗憾具有上界

$$R(T) \leqslant \sum_{j=1}^{n} \left(\frac{d_j^2}{2\eta} + \eta \right) \sqrt{\sum_{i=1}^{T} \left| \nabla_j f_{a_i}\left(x^i\right) \right|^2 + \varepsilon}, \tag{2-55}$$

其中 $d_j = \sup_{x,y \in D} |x_j - y_j|$.

定理 2.11 的直接推论是，当随机梯度 $\nabla f_{a_i}\left(x^i\right)$ 的 l^∞ 范数一致有界时，AdaGrad 也可达到 $O(\sqrt{T})$ 的遗憾界。

注： 若 $\varepsilon = 0$，定义域 D 为矩体，且初始步长 η 在各分量上也取不同值，例如在第 j 个分量上取值 $d_j/\sqrt{2}$，那么 (2-55) 式所得的遗憾界可以达到最小值，并且

此最小值不大于定理 2.9 所给出的 $\sqrt{2}R_{\min}$. 事实上，由 Cauchy 不等式可见

$$\begin{aligned}\sum_{j=1}^{n}\left(\frac{d_j^2}{2\frac{d_j}{\sqrt{2}}}+\frac{d_j}{\sqrt{2}}\right)\sqrt{\sum_{i=1}^{T}\left|\nabla_j f_{a_i}\left(x^i\right)\right|^2} &\leqslant \sqrt{2}\sqrt{\sum_{j=1}^{n}d_j^2}\sqrt{\sum_{j=1}^{n}\sum_{i=1}^{T}\left|\nabla_j f_{a_i}\left(x^i\right)\right|^2} \\ &= \sqrt{2}d\sqrt{\sum_{i=1}^{T}\left\|f_{a_i}\left(x^i\right)\right\|_2^2}=\sqrt{2}R_{\min}.\end{aligned}$$

这表明逐分量 AdaGrad 相较于 AdaGrad-Norm，遗憾界有较好的控制。这也从理论上印证了 AdaGrad 较好的收敛性。

第三章　基于 AdaGrad 的改进算法

由迭代格式(2-2)可见，AdaGrad 算法步长的动态减小是通过在分母累积历史梯度的平方达成的。这样操作虽然能够充分利用函数的曲率信息，但也产生如下的弊端：

1. 迭代过程中步长单调递减趋于 0，导致迭代后期收敛放缓；
2. 历史梯度信息对当前迭代的影响过大，因而当模型训练到稳定阶段后，再添加新数据，算法将无法对这些新数据进行有效学习 [29] ；
3. 算法对初始步长和初始点的选取敏感；如果初始点附近的梯度较大，会造成步长过早下降，收敛放慢。

为了克服上述弊端，研究者提出了众多基于 AdaGrad 的改进算法。本章我们将介绍 RMSProp, AdaDelta, m-AdaGrad, Adam 等常用方法 [1] ，并分析其性能。

本章在介绍算法迭代格式时，主要考虑定义域 $D = \mathbb{R}^n$ 的大规模优化问题(1-6). 和之前一样，$\nabla f_{a_k}(x^k)$ 表示第 k 次迭代时选取的随机次梯度。

3.1　RMSProp 与 AdaDelta

AdaGrad 迭代过程中历史梯度信息权重过高，我们希望随着迭代的进行，距离当前时刻较远的梯度信息能被“遗忘”。**均方根传播算法** [2]（RMSProp, root mean square propagation）正是基于这样的思想。该方法与 AdaGrad 的不同之处在于，将迭代格式(2-3)中的 G^k 的递推式改为 **指数滑动平均**（exponential moving average）的形式：

$$\begin{aligned} G^k &= \rho G^{k-1} + (1-\rho)(\nabla f_{a_k}(x^k) \odot \nabla f_{a_k}(x^k)), \\ x^{k+1} &= x^k - \frac{\eta}{\sqrt{G^k + \varepsilon}} \odot \nabla f_{a_k}(x^k), \qquad\qquad k = 1, 2, \cdots, \end{aligned} \tag{3-1}$$

1　一般而言，此类逐分量自适应步长的随机梯度方法统称 **Ada-系列算法**。

2　RMSProp 由 Geoff Hinton 在其课堂 Neural Networks for Machine Learning 上提出，并未公开发表。论文所参考的内容主要来自 *http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf* 所提供的课件。

其中 $\rho \in (0,1)$ 为衰减参数，一般取 0.9.

由迭代格式(3-1)可见，RMSProp 的步长不必然趋于 0，并且受较远的梯度信息的影响较小。事实上，由归纳法可见，若对于 $\forall\ x \in \mathbb{R}^n, 1 \leqslant i \leqslant m$ 有 $\|\nabla f_i(x)\|_\infty \leqslant M$，则 G^k 的第 j 个分量 $G_j^k \leqslant M\ (1 \leqslant j \leqslant n, k \in \mathbb{N}_+)$. 这说明此时 RMSProp 在各个维度上的迭代步长存在正下界 $\eta/\sqrt{G_j^k + \varepsilon} \geqslant \eta/\sqrt{M + \varepsilon}$. 此外，如果极限 $\lim_{k\to\infty}(\nabla_j f_{a_k}(x^k))^2$ 存在，必然 $\lim_{k\to\infty} G_j^k = \lim_{k\to\infty}(\nabla_j f_{a_k}(x^k))^2$. 这说明 RMSProp 的步长主要受当前点附近的梯度的影响。

Algorithm 2 RMSProp

输入： 组成 $f(x) = \frac{1}{m}\sum_{i=1}^m f_i(x)$ 的诸 $f_i(x)$ 及其梯度或次梯度 $g_i(x)$ 计算式；批量大小 N；初始步长 $\eta > 0$；初始点 x^1；最大迭代次数 maxit；$\rho, \varepsilon > 0$.

1: $G \leftarrow \mathbf{0}$, $x \leftarrow x^1$;
2: **for** $i = 1$ to maxit **do**
3: 随机等可能地选取 $\{\xi_1, \cdots, \xi_N\} \subseteq \{1, 2, \cdots, m\}$, $g \leftarrow \frac{1}{N}\sum_{j=1}^N g_{\xi_j}(x)$;
4: $G \leftarrow \rho G + (1-\rho) g \odot g$;
5: $x \leftarrow x - \frac{\eta}{\sqrt{G+\varepsilon}} \odot g$;
6: **end for**

输出： 函数 f 极小值点的估计值 x.

RMSProp 的收敛性分析是更为困难的。Sashank J. Reddi et al. (2018) [25] 给出了在线凸优化中 RMSProp 存在非次线性遗憾界的案例。即使在凸优化问题中，RMSProp（以及后文所介绍的 Adam 等方法）也可能发散。关于 RMSProp 的收敛性大致有如下研究：Fangyu Zou et al. (2019) [30] 给出了 RMSProp 收敛的一个充分条件；Naichen Shi et al. (2021) [31] 说明了在适当的参数选取下 RMSProp 的收敛性；最近，南开大学 Huan Li et al. (2024) 给出了 RMSProp 在梯度噪声分量的方差有界时 $O(\sqrt{n}/T^{\frac{1}{4}})$ 的收敛速度。

AdaGrad 的另一常用改进方法是 **AdaDelta**. 该方法由 M. D. Zeiler (2012) 提出 [23]，其基本想法是在 RMSprop 的基础上，考虑对 $\Delta x^k = x^{k+1} - x^k$ 的分量平方进行累积并求均方根，用其代替(3-1)式中的 η. Zeiler 指出，AdaDelta 在梯度的噪声较大时表现较好，并且对不同的模型结构、数据模式和超参数的选择表现出较强的鲁棒性。AdaDelta 的迭代格式为

$$
\begin{aligned}
G^k &= \rho G^{k-1} + (1-\rho)\nabla f_{a_k}(x^k) \odot \nabla f_{a_k}(x^k), \\
x^{k+1} &= x^k - \sqrt{\frac{D^{k-1}+\varepsilon}{G^k+\varepsilon}} \odot \nabla f_{a_k}(x^k), \\
D^k &= \rho D^{k-1} + (1-\rho)\Delta x^k \odot \Delta x^k, \qquad\qquad k = 1, 2, \cdots,
\end{aligned}
\tag{3-2}
$$

其中 $G^0 = D^0 = \mathbf{0}$, $\rho \in (0,1)$. AdaDelta 无需手动确定初始步长。事实上我们有

$$
x^2 - x^1 = -\sqrt{\frac{\varepsilon}{(1-\rho)\nabla f_{a_1}(x^1) \odot \nabla f_{a_1}(x^1) + \varepsilon}} \odot \nabla f_{a_1}(x^1),
$$

这类似于在 (3-1) 中让 $\eta = \sqrt{\varepsilon}$.

Algorithm 3 AdaDelta

输入: 组成 $f(x) = \frac{1}{m}\sum_{i=1}^m f_i(x)$ 的诸 $f_i(x)$ 及其梯度或次梯度 $g_i(x)$ 计算式；批量大小 N；初始步长 $\eta > 0$；初始点 x^1；最大迭代次数 maxit；ρ, $\varepsilon > 0$.

1: $G \leftarrow \mathbf{0}$, $D \leftarrow \mathbf{0}$, $x \leftarrow x^1$;

2: **for** $i = 1$ to maxit **do**

3: 随机等可能地选取 $\{\xi_1, \cdots, \xi_N\} \subseteq \{1, 2, \cdots, m\}$, $g \leftarrow \frac{1}{N}\sum_{j=1}^N g_{\xi_j}(x)$;

4: $G \leftarrow \rho G + (1-\rho) g \odot g$, $\Delta x \leftarrow -\sqrt{\frac{D+\varepsilon}{G+\varepsilon}} \odot g$;

5: $x \leftarrow x + \Delta x$, $D \leftarrow \rho D + (1-\rho)\Delta x \odot \Delta x$;

6: **end for**

输出: 函数 f 极小值点的估计值 x.

AdaDelta 的另一重要特征是迭代过程中的量纲始终统一，这也是 (3-2) 式中引入 D^k 的原因之一。假设目标函数 $f(x)$ 的值是无单位的，那么在 SGD 的迭代格式 (1-4) 下，我们有

$$
U(\Delta x_j^k) = U(x_j^{k+1} - x_j^k) \propto U\left(\nabla_j f(x_j^k)\right) \propto \frac{1}{U(x_j^k)},
\tag{3-3}
$$

其中 $U(\cdot)$ 表示变量的单位。这说明 SGD 迭代过程中量纲不统一。AdaGrad 也有类似的问题（对于 AdaGrad 而言，Δx_j^k 是无单位的）。而 AdaDelta 在每一次迭代前后，量纲都是统一的。事实上，假设 $U(x^k) \propto u$ 对任意 $k \leqslant N$ 成立 $(N \geqslant 1)$，那么 $\forall\, k \leqslant N$, $D^k \propto u^2$. 根据迭代格式 (3-2) 我们有

$$U(\Delta x_j^{N+1}) \propto \sqrt{\frac{U(D^k)}{\left(U\left(\nabla_j f(x_j^2)\right)\right)^2}} \cdot U\left(\nabla_j f(x_j^2)\right) \propto u. \tag{3-4}$$

通过归纳可见，$U(x^k) \propto u$ 对 $k \in \mathbb{N}_+$ 均成立。

3.2 动量加速技术

在数据未作标准化处理、分布不均匀时，优化目标函数常呈现病态。此时目标函数等高线为扁平的椭圆，传统的 GD 和 SGD 所产生的迭代点列将沿着椭圆的短轴方向振荡，而沿着局部最小值方向进展缓慢。由 Boris Polyak 在 1964 年提出的**动量方法**（MSGD, momentum-based stochastic gradient descent method）[32] 则可以有效地抑制振荡、加速收敛。MSGD 在确定每步的更新方向时，综合考虑当前梯度方向和上次迭代前进方向。经典的 MSGD 的迭代格式为：

$$v^{k+1} = \gamma v^k - \eta_k \nabla f_{a_k}(x^k),\ x^{k+1} = x^k + v^{k+1},\ k = 1, 2, \cdots, \tag{3-5}$$

其中 $\gamma \in [0, 1)$, $\eta_k > 0$, $v^1 = \mathbf{0}$. v^{k+1} 可以看成第 k 次迭代前进的速度，由动量项 γv^k 和负随机梯度项 $-\eta_k \nabla f_{a_k}(x^k)$ 构成。动量项系数 γ 常取 0.9，表示迭代过程具有较大的惯性。

MSGD 的加速原理可直观理解如下：我们将 v^k, $-\eta_k \nabla f_{a_k}(x^k)$ 分别分解为朝向局部最优的方向 v_1^k, p_1^k 和垂直方向 v_2^k, p_2^k. 产生振荡时，有 $v_2^k \approx -p_2^k$. 此时 MSGD 产生的新的更新方向 $v^{k+1} \approx (\gamma v_1^k + p_1^k) + (1-\gamma) v_2^k$，其中向局部最优前进的速度增加到 $\gamma v_1^k + p_1^k$，而振荡则减小为 $(1-\gamma) v_2^k$.

另一种形式的 MSGD 又被称为“**重球方法**（HB, Heavy ball momentum）”，其迭代格式为：

$$v^{k+1} = \gamma v^k + (1-\gamma) \nabla f_{a_k}(x^k),\ x^{k+1} = x^k - \eta_k v^{k+1},\ k = 1, 2, \cdots, \tag{3-6}$$

其中 $v^1 = \mathbf{0}$, $\gamma \in [0, 1)$. 这里 v^k 代表了梯度的指数滑动平均，主要受本次和前几次迭代的梯度影响，而忽略较远的梯度信息。迭代点列前进的过程类似于 “小球滚下山” 的过程，朝向局部最优的方向，前进速度越来越快；垂直方向的振荡则

不断减小。

另一种常用的动量加速方法是 **Nesterov 加速算法** [33]，它可以视为 MSGD 的校正，主要不同之处在于，计算梯度时先预见性地走完惯性导致的一步，再根据当前位置更新梯度。这种预见性的更新方式能防止迭代过程走得太快，提高了响应能力 [34]。Nesterov 加速算法的迭代格式如下：

$$v^{k+1}=\gamma v^k-\eta_k\nabla f_{a_k}(x^k+\gamma v^k),\ x^{k+1}=x^k+v^{k+1},\ k=1,2,\cdots,$$

其中 $v^1=\mathbf{0}$, $\gamma\in[0,1)$.

动量方法可以和上文所提到的 AdaGrad 和 RMSProp 等方法结合，得到诸多动量加速的自适应方法。例如 **m-AdaGrad**：

$$\begin{aligned} v^{k+1}&=\beta_1 v^k+\nabla f_{a_k}\left(x^k\right),\\ G^k&=\beta_2 G^{k-1}+\nabla f_{a_k}(x^k)\odot\nabla f_{a_k}(x^k),\\ x^{k+1}&=x^k-\frac{\eta}{\sqrt{G^k+\varepsilon}}\odot v^{k+1}, \qquad k=1,2,\cdots, \end{aligned} \tag{3-7}$$

其中 $v^1=\mathbf{0}$, $\beta_1,\beta_2\in[0,1)$. 又如 **m-RMSProp**：

$$\begin{aligned} G^k&=\rho G^{k-1}+(1-\rho)\nabla f_{a_k}(x^k)\odot\nabla f_{a_k}(x^k),\\ v^{k+1}&=\alpha v^k-\frac{\eta}{\sqrt{G^k+\varepsilon}}\odot\nabla f_{a_k}\left(x^k\right),\\ x^{k+1}&=x^k+v^{k+1}, \qquad k=1,2,\cdots, \end{aligned} \tag{3-8}$$

其中 $v^1=\mathbf{0}$, $\rho,\alpha\in[0,1)$.

3.3 自适应矩估计

自适应矩估计（Adam, adaptive moment estimation）由 Diederik Kingma 和 Jimmy Ba 在 2014 年提出 [24]，可以说是近年来机器学习等领域最有效的优化算法。Adam 可视作 RMSProp 和 HB（重球动量）的结合，迭代时同时记录了梯度及其各分量平方的指数滑动平均：

$$S^k = \rho_1 S^{k-1} + (1-\rho_1)\nabla f_{a_k}(x^k),$$
$$G^k = \rho_2 G^{k-1} + (1-\rho_2)\nabla f_{a_k}(x^k) \odot \nabla f_{a_k}(x^k),\ k = 1, 2, \cdots,$$

其中 $\rho_1,\ \rho_2 \in [0, 1)$, $S^0 = G^0 = \mathbf{0}$.

然而，单纯采用指数滑动平均的形式会使迭代初期时 S^k 和 G^k 向 0 偏离。具体而言，我们假定 $\nabla f_{a_k}(x^k) \equiv \mathbf{c}$ 为常向量，此时由归纳法易见

$$S^k = (1-\rho_1^k)\cdot \mathbf{c},\ G^k = (1-\rho_2^k)\cdot \mathbf{c} \odot \mathbf{c},\ k = 1, 2, \cdots.$$

实际操作中衰减系数 ρ_1, ρ_2 常取 0.9 和 0.999，这样当 k 较小时 $1-\rho_1^k,\ 1-\rho_2^k$ 也将较小。为让 S^k 和 G^k 更准确地记录梯度信息，而尽量少地受参数 ρ_1, ρ_2 影响，我们可以考虑修正式

$$\hat{S}^k = \frac{S^k}{1-\rho_1^k},\ \hat{G}^k = \frac{G^k}{1-\rho_2^k},$$

并令

$$x^{k+1} = x^k - \frac{\eta}{\sqrt{\hat{G}^k + \varepsilon}} \odot \hat{S}^k.$$

综合上面的想法，求解定义域 $D = \mathbb{R}^n$ 的问题 (1-6) 的 Adam 算法完整步骤可总结如下：

Algorithm 4 Adam

输入： 组成 $f(x) = \frac{1}{m}\sum_{i=1}^m f_i(x)$ 的诸 $f_i(x)$ 及其梯度或次梯度 $g_i(x)$ 计算式；批量大小 N；初始步长 $\eta > 0$；衰减参数 $\rho_1,\ \rho_2 \in (0, 1)$；初始点 x^1；最大迭代次数 maxit；$\varepsilon > 0$.

1: $S \leftarrow \mathbf{0}$, $G \leftarrow \mathbf{0}$, $x \leftarrow x^1$；
2: **for** $i = 1$ to maxit **do**
3: 随机等可能地选取 $\{\xi_1, \cdots, \xi_N\} \subseteq \{1, 2, \cdots, m\}$, $g \leftarrow \frac{1}{N}\sum_{j=1}^N g_{\xi_j}(x)$；
4: $S \leftarrow \rho_1 S + (1-\rho_1)g$, $\hat{S} \leftarrow \frac{S}{1-\rho_1^i}$；
5: $G \leftarrow \rho_2 G + (1-\rho_2)g \odot g$, $\hat{G} \leftarrow \frac{G}{1-\rho_2^i}$；
6: $x \leftarrow x - \frac{\eta}{\sqrt{\hat{G}+\varepsilon}} \odot \hat{S}$；
7: **end for**

输出： 函数 f 极小值点的估计值 x.

在 Adam 方法中取 $\rho_1 = 0$, $\rho_2 \to 1^-$ 时，我们可以得到类似 AdaGrad 的迭代格式。具体而言，此时

$$\begin{aligned}\hat{S}^k &= \nabla f_{a_k}(x^k),\\ \hat{G}^k &= \frac{1}{k}\sum_{i=1}^{k} \nabla f_{a_i}(x^i) \odot \nabla f_{a_i}(x^i).\end{aligned}$$

事实上，由于

$$(1-\rho_2^k)\hat{G}^k = \rho_2(1-\rho_2^{k-1})\hat{G}^{k-1} + (1-\rho_2)\nabla f_{a_k}(x^k) \odot \nabla f_{a_k}(x^k),$$

那么对 $k \geqslant 2$，易于验证极限存在时总有

$$\begin{aligned}\lim_{\rho_2\to 1^-} \hat{G}^k &= \lim_{\rho_2\to 1^-}\left(\frac{\rho_2(1-\rho_2^{k-1})}{1-\rho_2^k}\hat{G}^{k-1} + \frac{1-\rho_2}{1-\rho_2^k}\nabla f_{a_k}(x^k)\odot\nabla f_{a_k}(x^k)\right)\\ &= \frac{k-1}{k}\lim_{\rho_2\to 1^-}\hat{G}^{k-1} + \frac{1}{k}\nabla f_{a_k}(x^k)\odot\nabla f_{a_k}(x^k).\end{aligned}$$

根据定义，对于 $k = 0$ 我们有 $\hat{G}^k = 0$，且

$$\lim_{\rho_2\to 1^-}\hat{G}^1 = \nabla f_{a_i}(x^i)\odot\nabla f_{a_i}(x^i).$$

由归纳法易见 $\rho_1 = 0$, $\rho_2 \to 1^-$ 时，我们有

$$\hat{G}^k = \frac{1}{k}\sum_{i=1}^{k}\nabla f_{a_i}(x^i)\odot\nabla f_{a_i}(x^i).$$

在提出 Adam 方法时，Kingma 等人验证了 Adam 在 Logistic 回归、卷积神经网络等模型上良好的性能 [24] 。Adam 方法的优点是：当数据集稀疏、梯度带噪声时，依然具有较好的收敛性；并且该方法易于调参，采用默认参数（$\rho_1 = 0.9$, $\rho_2 = 0.999$, $\eta = 0.001$）往往就能取得很好的效果。

Adam 方法的收敛性依然缺乏理论保证，Reddi et al. (2018) [25] 就给出了 Adam 在凸优化情形中不收敛的例子。和 AdaGrad 类似，当优化问题定义域为紧凸集、随机梯度一致有界时，Adam 方法具有次线性遗憾 $R(T) = O(\sqrt{T})$ [24] . 关于 Adam 方法在一般情形下的收敛性，可参考 [21,25,35] 等。

此外，Adam 方法还有一些变体，比如 Adamax 方法、AMSgrad 方法。

第四章　数值实验

在本章中，我们考虑三种常见的二分类模型：Logistic 回归、支持向量机和多层感知机（神经网络）。这三种模型虽然简单，却能够涵盖目标函数为光滑（强）凸函数、非光滑凸函数、光滑非凸函数的三大类优化问题。我们分别利用步长递减的 SGD, AdaGrad, AdaGrad-Norm, RMSProp, AdaDelta 和 Adam 方法对上述模型进行训练，给出最终的验证准确度（accuracy）、验证精确度（precision）、验证召回率（recall），以及目标函数取值与迭代时期[1]的关系。

论文实验采用 Spambase 和 Ads 两个数据集，其中 Spambase 为稠密数据，Ads 为稀疏数据。数据集的详细信息、各种优化方法的超参数设定（批量大小、初始步长、衰减系数等）、全局随机种子等内容，参见附录B.

4.1　Logistic 回归

Logistic 回归（Logistic Regression）是一种用于解决二分类问题的广义线性回归模型。该模型根据所提供的样本，返回 0 到 1 之间的连续数字，用以估计样本属于某类事物可能性。具体而言，给定样例 $\{(\mathbf{a}_1, y_1), (\mathbf{a}_2, y_2), \cdots, (\mathbf{a}_m, y_m)\}$（其中 $\mathbf{a}_i \in \mathbb{R}^d$, $y_i \in \{1, -1\}$），Logistic 回归模型假设样本 $\mathbf{x} \in \mathbb{R}^d$ 与标签 $y \in \{1, -1\}$ 对应的概率分别为

$$\mathbb{P}(y = 1 \,|\, \mathbf{x}) = s(\mathbf{w} \cdot \mathbf{x} + b),\ \mathbb{P}(y = -1 \,|\, \mathbf{x}) = 1 - s(\mathbf{w} \cdot \mathbf{x} + b),$$

这里 $s(t) = 1/(1 + e^{-t})$ 为 **sigmond 函数**，$\mathbf{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$ 为待定参数。如果模型构建恰当，每个样本都与其标签对应的概率应尽可能大，故我们需要求解问题

$$\max_{w \in \mathbb{R}^d, b, \in \mathbb{R}} F(\mathbf{w}, b) := \prod_{\substack{y_i = 1 \\ 1 \leqslant i \leqslant m}} s(\mathbf{a}_i \cdot \mathbf{w} + b) \prod_{\substack{y_i = -1 \\ 1 \leqslant i \leqslant m}} (1 - s(\mathbf{a}_i \cdot \mathbf{w} + b)).$$

上式两边取对数，并令 $\mathbf{v} = (\mathbf{w}, b)$, $\mathbf{c}_i = (\mathbf{a}_i, 1)$，问题可简化为

1　迭代时期数（epoch）的计算式为 epoch $= \lfloor tB/N \rfloor$，其中 t 为迭代次数，B 为批量大小，N 为数据量。在选用的数据量大小上，随机梯度方法的一个时期相当于非随机方法的一次迭代。

$$\min_{\mathbf{v}\in\mathbb{R}^{d+1}} \; c(\mathbf{v}) := \frac{1}{m}\sum_{i=1}^{m} \ln(1+e^{-y_i(\mathbf{v}\cdot\mathbf{c}_i)}). \tag{4-1}$$

问题(4-1)的目标函数是可微凸函数。事实上，记 $f_i(\mathbf{x}) = \ln(1+e^{-y_i(\mathbf{x}\cdot\mathbf{c}_i)})$，我们有

$$\nabla f_i(\mathbf{x}) = \frac{-y_i e^{-y_i(\mathbf{x}\cdot\mathbf{c}_i)}\mathbf{c}_i}{1+e^{-y_i(\mathbf{x}\cdot\mathbf{c}_i)}}, \quad \text{Hess}\, f_i(\mathbf{x}) = \frac{y_i^2 e^{-y_i(\mathbf{x}\cdot\mathbf{c}_i)}\mathbf{c}_i'\mathbf{c}_i}{(1+e^{-y_i(\mathbf{x}\cdot\mathbf{c}_i)})^2},$$

由于 $\mathbf{c}_i'\mathbf{c}_i$ 半正定，故 f_i 为凸函数。在实验中，我们对问题(4-1)进行 L^2 正则化处理，并置正则参数 $\lambda = 10^{-5}$. 这样，优化目标函数就成为强凸函数。

用 AdaGrad 求解 Logistic 回归问题的收敛性是有理论保证的。如果输入数据 $\mathbf{a}_i$ $(i=1,2,\cdots,m)$ 的 l^2 范数具有上界 M，可见 $\nabla f_i(\mathbf{x})$ 具有上界 $\max\{1,M\}$. 如果限制定义域有界，根据定理2.1可见 AdaGrad 求解此问题时具有 $O(1/\sqrt{T})$ 的收敛率。

我们用 Ada-系列算法对上文所述的 Logistic 回归模型进行训练，在迭代 100 个时期后，得到模型的验证准确度（accuracy）、验证精确度（precision）、验证召回率（recall）如图4-1（见下页）所示[1]。结果表明，总体而言 AdaGrad 和 RMSProp 的训练结果最优，Adam 和 AdaDelta 次之；AdaGrad-Norm 在非稀疏数据集 Spambase 上表现较好，在稀疏数据集 Ads 上则表现不佳，甚至劣于 SGD. 在非稀疏数据集 Spambase 上，各方法的差距不明显，而在稀疏数据集 Ads 上，逐分量自适应学习率的 Ada-系列算法在验证精确度上则显著优于其它方法。对于此现象的解释，可参见第五章。

在图4-2（见下页）中，我们给出了在首次实验中各方法对应的目标函数取值随迭代时期数的变化。结果表明，在非稀疏数据集 Spambase 上，各优化方法的函数值均呈现较稳定的下降趋势，而在稀疏数据集 Ads 上则呈现或多或少的振荡（SGD 和 AdaGrad-Norm 的振荡尤为强烈）。其原因可能是：对于稠密数据，选择一小批函数计算平均梯度，能够较好地代替目标函数的梯度，这样每次迭代时的更新方向都接近下降方向，因而更新过程中振荡较小。而对于稀疏数据集而言，选择较小的批量计算梯度将很难模拟真实的梯度方向。注意论文中我们采取的批量大小为 32，而 Ads 的非零元占比仅为 0.82%（见附录B）.

1 论文给出的结果均为重复实验 10 次后的平均数据。图4-1中，数据所显示的范围为 mean $\pm$ 1.96SE，其中 mean 为均值，SE 为标准误。

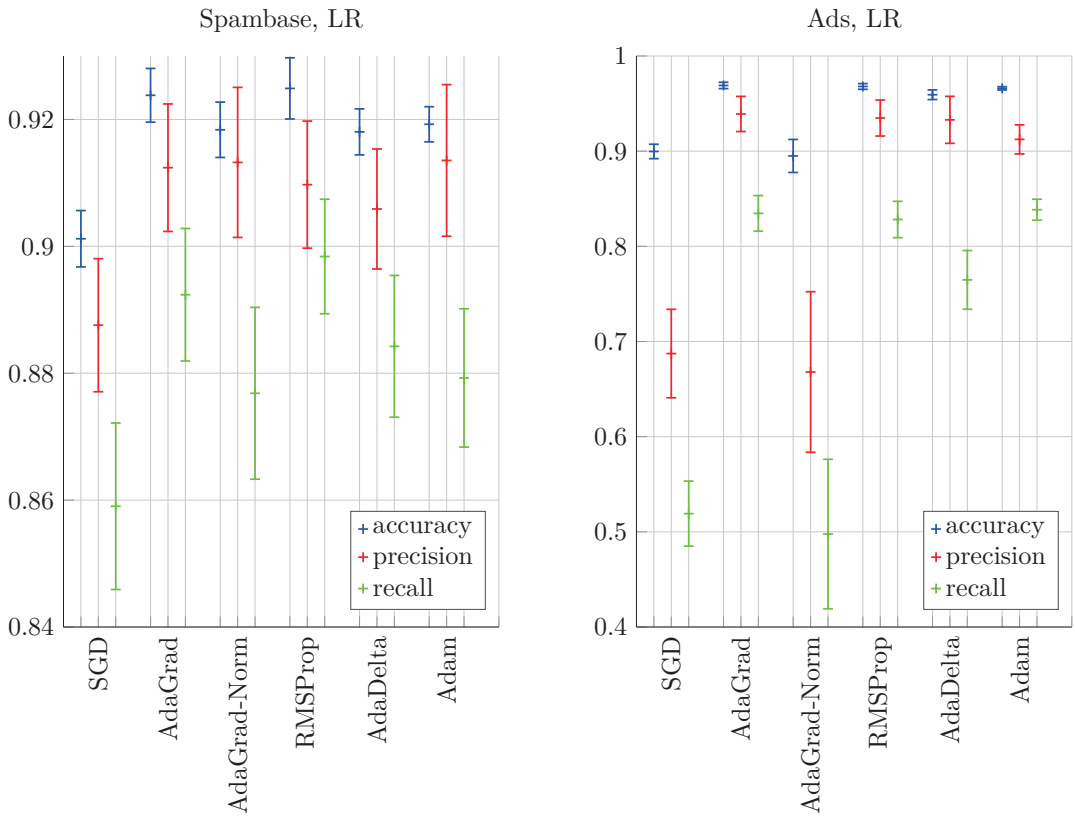

图 4-1　迭代 100 个时期后的验证准确度、验证精确度、验证召回率：Logistic 回归

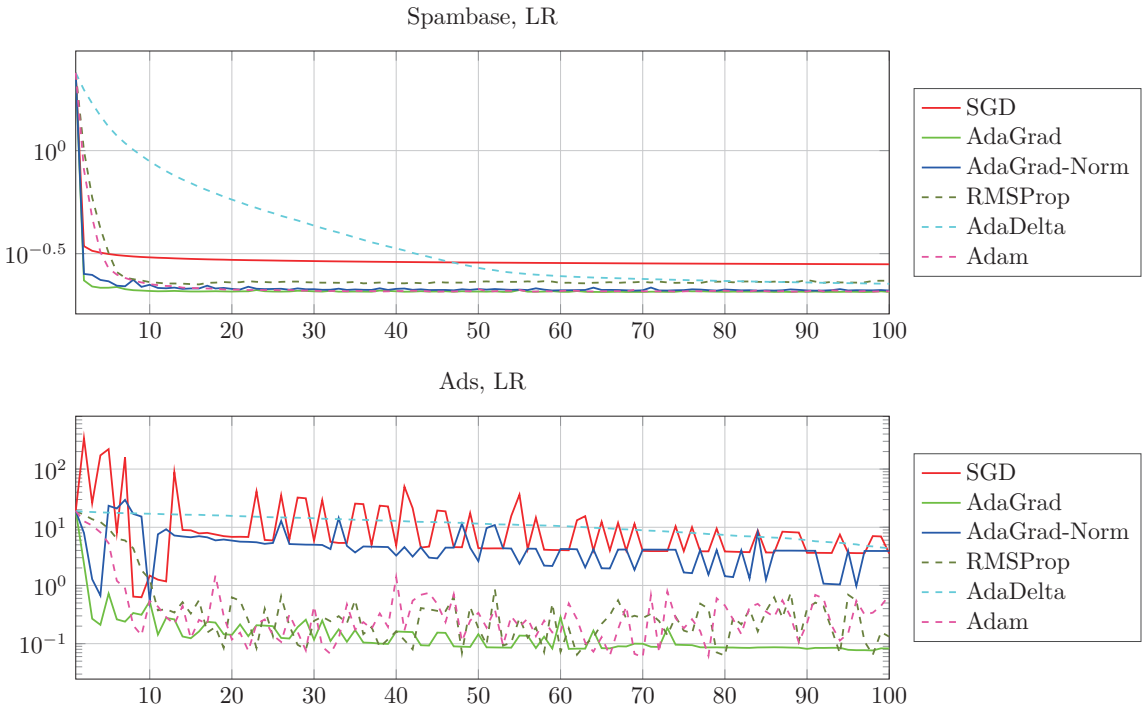

图 4-2　目标函数值随迭代时期数的变化：Logistic 回归

综合比较图4-1和图4-2，我们可以看出，优化目标函数值的下降程度可以大致反映训练效果。但 Ads 数据集下 AdaDelta 似乎是一个例外：该方法下目标函数值下降较慢，甚至不如 SGD，但其训练结果远好于 SGD. 值得注意的是，我们完全没有对 AdaDelta 进行任何参数调节（AdaDelta 无需手动确定初始步长），而其余方法是根据格点搜索来确定的最优初始步长。

4.2 支持向量机

支持向量机（SVM，Support Vector Machine）是一种用于解决分类和回归问题的机器学习模型。以二分类问题为例，给定样例 $\{(\mathbf{a}_1, y_1), \cdots, (\mathbf{a}_m, y_m)\}$（其中 $\mathbf{a}_i \in \mathbb{R}^d$, $y_i \in \{1, -1\}$），我们希望找到一个超平面 $\{\mathbf{x} \in \mathbb{R}^d : \mathbf{w} \cdot \mathbf{x} + b = 0\}$ 尽可能准确地分离标签为 1 和 -1 的两类样本。由于 $\mathbf{w}, b$ 乘上非零常数后所表示的超平面不变，故不妨设能够正确分离的样本中，标签为 1 样本 $\mathbf{x}$ 均满足 $\mathbf{w} \cdot \mathbf{x} + b \geqslant 1$，标签为 -1 的样本 $\mathbf{x}$ 均满足 $\mathbf{w} \cdot \mathbf{x} + b \leqslant -1$. 注意到函数 $\max\{0, y(\mathbf{w} \cdot \mathbf{x} + b)\}$ 在分类正确时值为 0，错误时值不小于 1，故可以将其作为损失函数。要使分类尽可能准确，我们需要极小化总损失 $\sum_{i=1}^{m} \max\{0, y_i(\mathbf{w} \cdot \mathbf{a}_i + b)\}$.

另一方面，在正确分离的样本中，异类样本之间的距离不小于 $d = 2/\|\mathbf{w}\|_2$. 我们还希望 d 能够尽可能大，这样超平面的分离效果受训练样本局部扰动的影响较小，并且对未知样例的泛化性能最强。综合以上两点，构造恰当的超平面可归结于求解非光滑凸优化问题

$$\min_{\mathbf{w} \in \mathbb{R}^d,\, b \in \mathbb{R}} \sum_{i=1}^{m} \max\{0, y_i(\mathbf{w} \cdot \mathbf{a}_i + b)\} + \lambda \|\mathbf{w}\|_2^2, \tag{4-2}$$

其中 $\lambda > 0$ 为正则参数。论文实验中，$\lambda = 1/(10^3 m)$，其中 m 为数据量。

记 $f_i(\mathbf{x}, b) = \max\{0, y_i(\mathbf{x} \cdot \mathbf{a}_i + b)\} + \lambda \|\mathbf{x}\|_2^2$，由于 f_i 为非光滑凸函数，在迭代时我们采取如下次梯度计算式：

$$\nabla f_i(\mathbf{x}, b) := \left(y_i \chi_{\{y_i(\mathbf{x} \cdot \mathbf{a}_i + b)\}}(\mathbf{x}, b) \mathbf{a}_i + 2\lambda \mathbf{x},\ y_i \chi_{\{y_i(\mathbf{x} \cdot \mathbf{a}_i + b)\}}(\mathbf{x}, b) \right).$$

如果限制定义域有界、输入数据一致有界，根据定理 2.1 可见 AdaGrad 求解此问题同样具有 $O(1/\sqrt{T})$ 的收敛率。

我们用 Ada-系列算法对上文所述的 SVM 模型进行训练，迭代 100 个时期

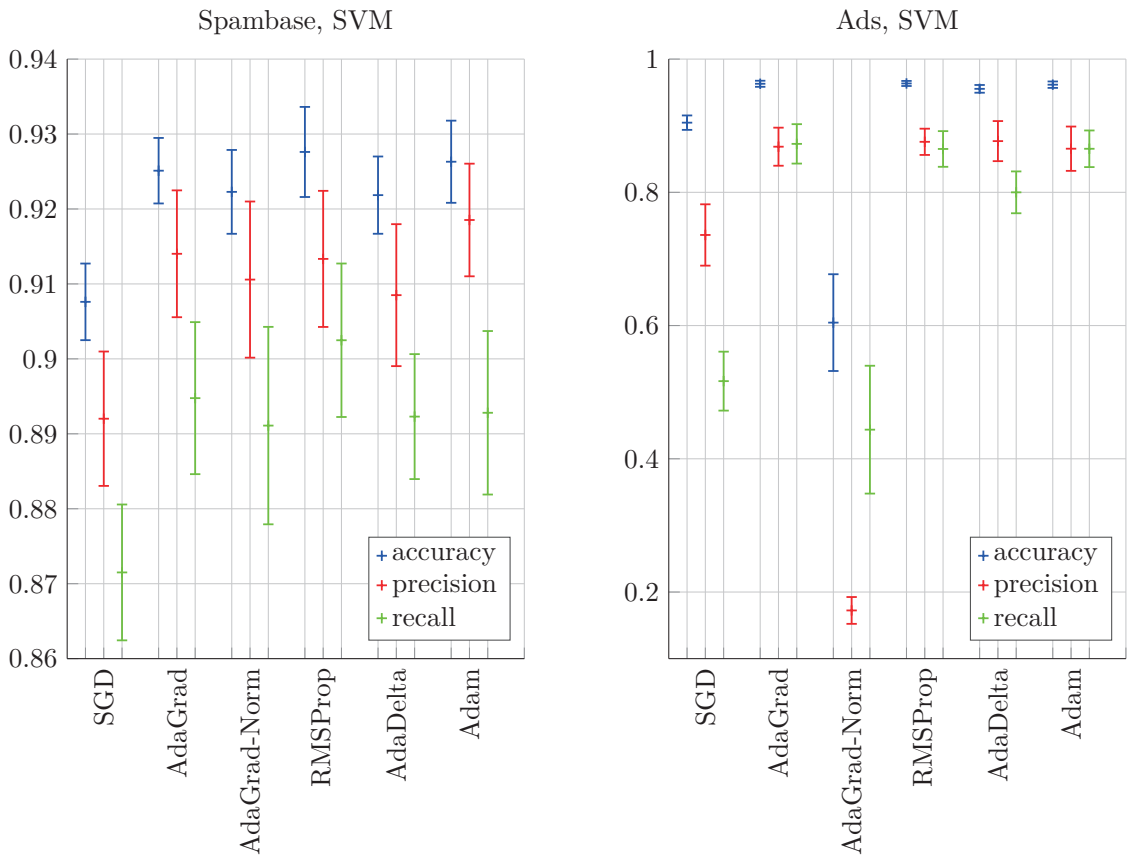

图 4-3　迭代 100 个时期后的验证准确度、验证精确度、验证召回率：SVM

后，模型的验证准确度、精确度、召回率如图 4-3 所示。结果表明，对于非稀疏数据集 Spambase，RMSProp 整体表现最好，接下来是 AdaGrad, AdaDelta, Adam 和 AdaGrad-Norm，不过这些方法之间差距不大；SGD 的表现也尚可，但略逊于以上方法；而对于稀疏数据集 Ads，Ada-系列算法的表现则显著较好。AdaGrad-Norm 算法精确度极低，可以认为不适合该数据集。

图 4-4（见下页）给出了各方法对应的目标函数取值随迭代时期数的变化。结果表明：在非稀疏数据集 Spambase 上，各方法平稳地趋于最小值，且函数值的大小大致反映了训练结果的好坏。在稀疏数据集 Ads 上，SGD, RMSProp 和 Adam 均有较强的振荡，AdaGrad 每间隔一段时间有一次明显振荡，最后趋于稳定的最小值；而 AdaGrad-Norm 似乎陷入了一段平台区域，学习过程难以进展。AdaDelta 的收敛是最慢的：在迭代 100 轮后，其函数值仍然较大；但图 4-3 的训练结果却表明 AdaDelta 具有较好的泛化性能。

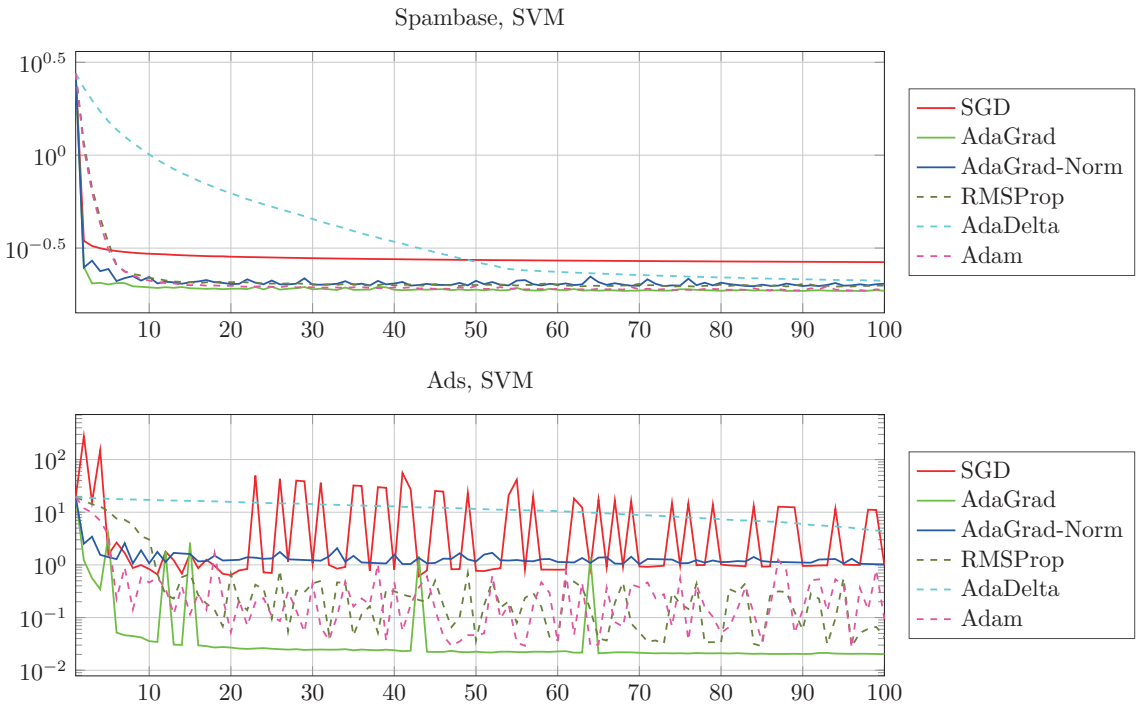

图 4-4　目标函数值随迭代时期数的变化：SVM

4.3　多层感知机

多层感知机（MLP，Multilayer Perceptron）是一种深度学习模型，可用于解决分类、回归和聚类等问题。MLP 由输入层、输出层和多个隐藏层构成，每层包含若干个神经元。图 4-5 的 (a) 和 (b) 分别给出了隐藏层数为 1 的 MLP 和其中隐藏层第 i 个神经元的结构示意图。

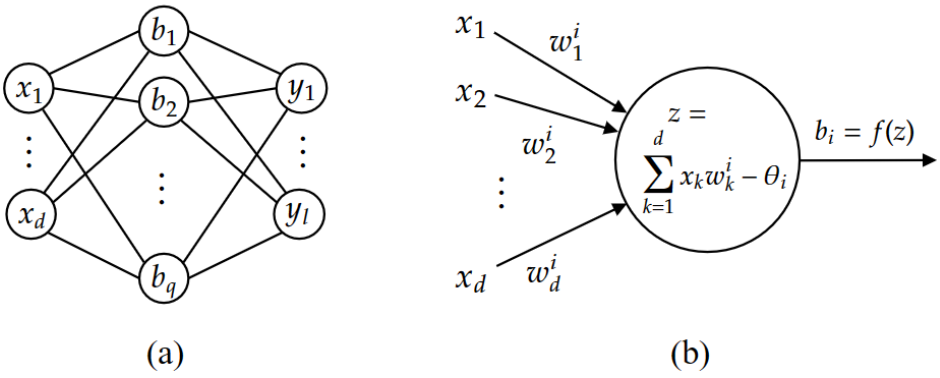

图 4-5　多层感知机

图4-5所示的 MLP 的工作原理可概括如下：输入层将记录外部数据的向量 $\mathbf{x}=(x_1,\cdots,x_d)$ 传递给隐藏层神经元；第 i 个隐藏层神经元按上图 (b) 所示规则产生输出 $b_i=f(\mathbf{w}^i\cdot\mathbf{x}-\theta_i)$，其中 $\mathbf{w}^i=(w_1^i,\cdots,w_d^i)$ 为权向量，$\theta_i\in\mathbb{R}$ 为阈值，$f(x)=1/(1+e^{-x})$ 为 Sigmond 激活函数。接下来，输出层中第 k 个神经元接收来自隐藏层的信号 $\mathbf{b}=(b_1,\cdots,b_q)$，按类似规则产生新的输出 $y_k=f(\mathbf{v}^k\cdot\mathbf{b}-\eta_k)$，其中 $\mathbf{v}^k$ 为权向量，η_k 为阈值。

给定样例 $\{(\mathbf{a}_1,\mathbf{y}_1),\cdots,(\mathbf{a}_m,\mathbf{y}_m)\}$ ($\mathbf{a}_i\in\mathbb{R}^d$, $\mathbf{y}_i=(y_1^i,\cdots,y_l^i)\in[0,1]^l$)，假设对样本 $\mathbf{a}_k$，MLP 的输出为 $\hat{\mathbf{y}}_k=(\hat{y}_1^k,\cdots,\hat{y}_l^k)$，则有均方误差 $E_k=\sum_{j=1}^{l}(\hat{y}_j^k-y_j^k)^2$. 为了构建恰当的模型，我们需要极小化总误差，这等价于求解光滑非凸优化问题

$$\min_{\substack{\mathbf{w}^1,\cdots,\mathbf{w}^q\in\mathbb{R}^d;\,\theta_1,\cdots,\theta_q\in\mathbb{R}\\ \mathbf{v}^1,\cdots,\mathbf{v}^l\in\mathbb{R}^q;\,\eta_1,\cdots,\eta_l\in\mathbb{R}}}\frac{1}{m}\sum_{k=1}^{m}E_k=\frac{1}{m}\sum_{k=1}^{m}\sum_{j=1}^{l}(\hat{y}_j^k-y_j^k)^2. \tag{4-3}$$

问题 (4-3) 的目标函数性质并不优秀。迭代时我们可能遇到大量鞍点和局部极值点，以及函数梯度奇异和梯度消失等现象 [29] 。在此条件下，我们所采用的各种随机优化方法都缺乏理论上的收敛保证。不过，实验表明，Ada-系列算法在此类问题中依然可以表现出较好的性能。

在实验中，我们只考虑一个隐藏层，隐藏神经元数量取 $q=5$. 实验所要解决的问题为二分类问题，此时输出神经元数量 $l=1$. 我们对此条件下相应的问题 (4-3) 进行 L^2 正则化处理，正则参数为 10^{-7}.

用 Ada-系列算法对上文所述的 MLP 模型进行训练，迭代 100 个时期后，模型的验证准确度、精确度、召回率如图4-6（见下页）所示。结果表明，在非稀疏数据集 Spambase 上，除 SGD 外的其余算法表现均较好，同时 SGD 仍具有超过 85% 的验证精确度。然而，在稀疏数据集 Ads 上，SGD 多次出现正例为 0 的训练结果，导致验证精确度标准差极大；同时，SGD 的召回率始终很低，可以认为 SGD 不适合该数据集。AdaGrad-Norm 的验证准确率最高，这是较为意外的结果。RMSProp, Adam 的训练结果也较好，AdaDelta 在验证召回率上稍低，AdaGrad 则在验证精确度上稍低。不过，这些方法的训练结果相差不大。

在图4-7（见下页）中，我们给出了在首次实验中各方法对应的目标函数取值随迭代时期数的变化。此时，各方法的振荡不如前两种模型激烈。事实上，这是因为目标函数较为平缓，产生了梯度消失现象。可以看出，SGD 很快陷入一

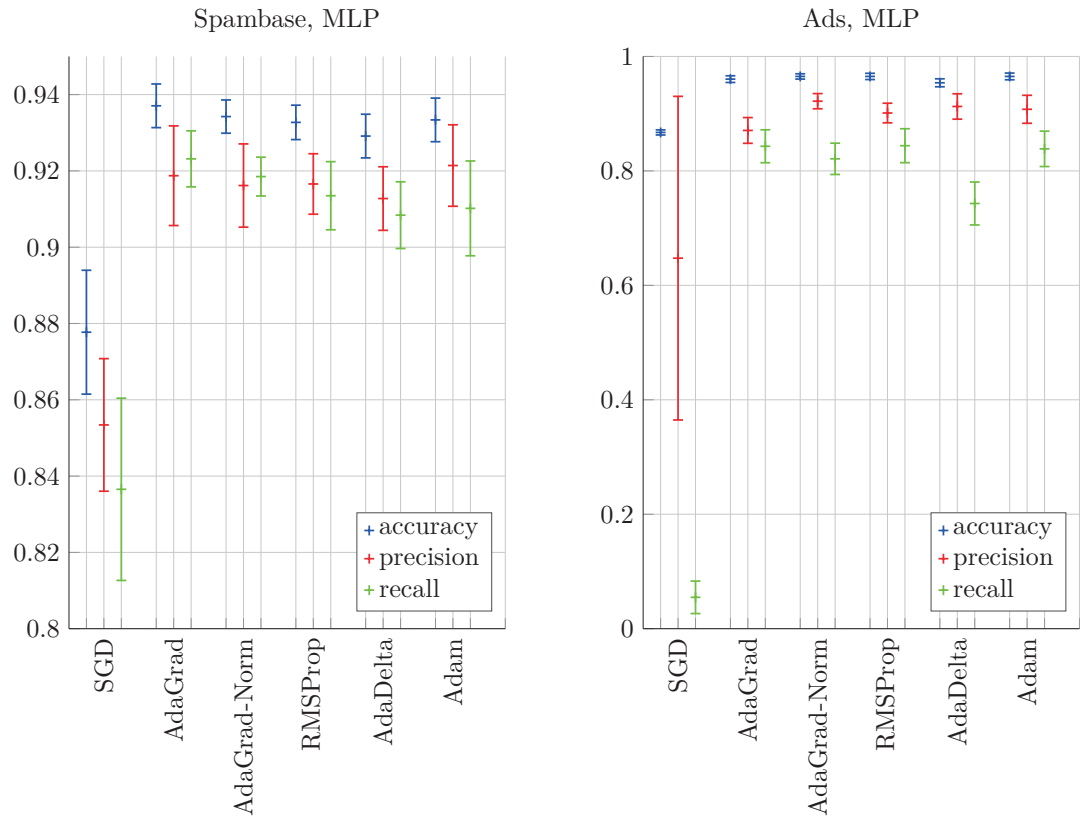

图 4-6　迭代 100 个时期后的验证准确度、验证精确度、验证召回率：MLP

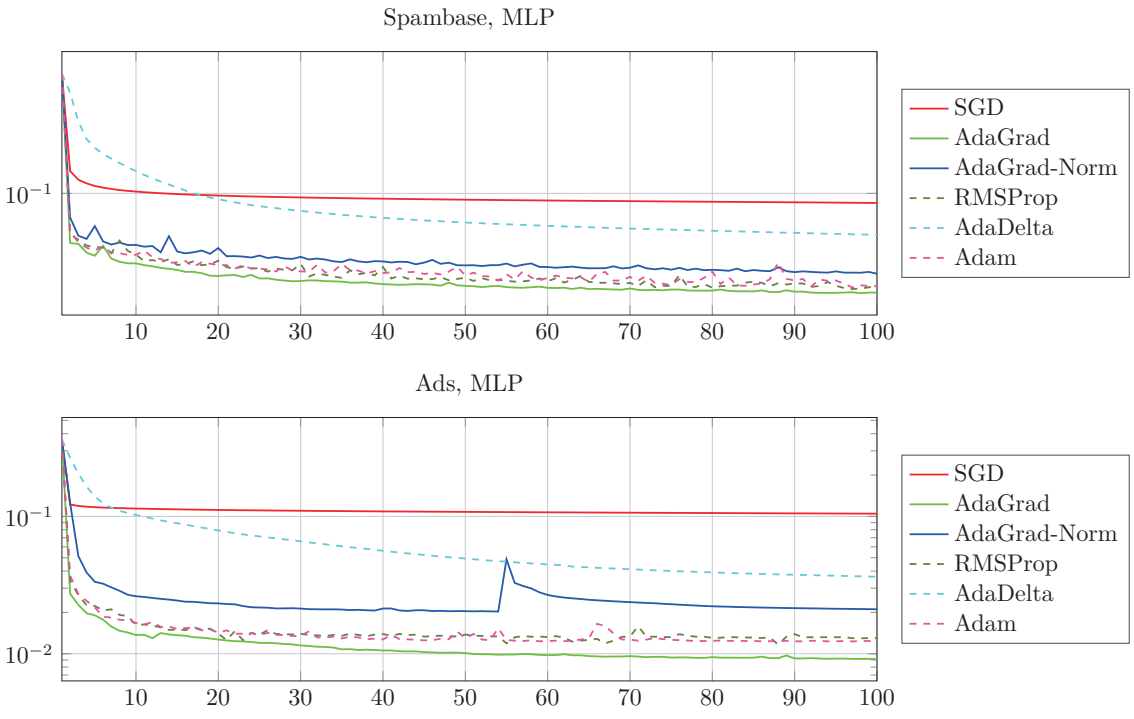

图 4-7　目标函数值随迭代时期数的变化：MLP

段平台区域，却始终未能走出该区域，这导致模型的训练结果较差。AdaGrad-Norm 在第 54 轮迭代时发生振荡，这并不是偶然的：在论文 10 次实验中，类似的振荡发生过 4 次。可以猜想此时迭代点列从一个鞍点或局部最小值中摆脱。AdaDelta 的收敛是较慢的，在迭代前期，其精确度高而召回率很低（< 0.4）。随着迭代进行，其召回率逐步提高，达到可以被接受的数值。AdaGrad 的收敛最快，到达的目标函数值也最小，但其训练结果并非最佳。这也表明单纯追求收敛速度并不一定带来较好的训练结果。

通过绘制极小值点附近的函数图像（随机选取两个维度），我们大致可以发现，MLP 的目标函数较为平坦。可以猜想此时出现了梯度消失现象。实验中，SGD 的步长始终单调递减（见附录 B ）；而由于目标函数梯度较小，AdaGrad-Norm 和 Ada-系列算法却能保持较大步长。这也解释了 SGD 表现较差的原因。

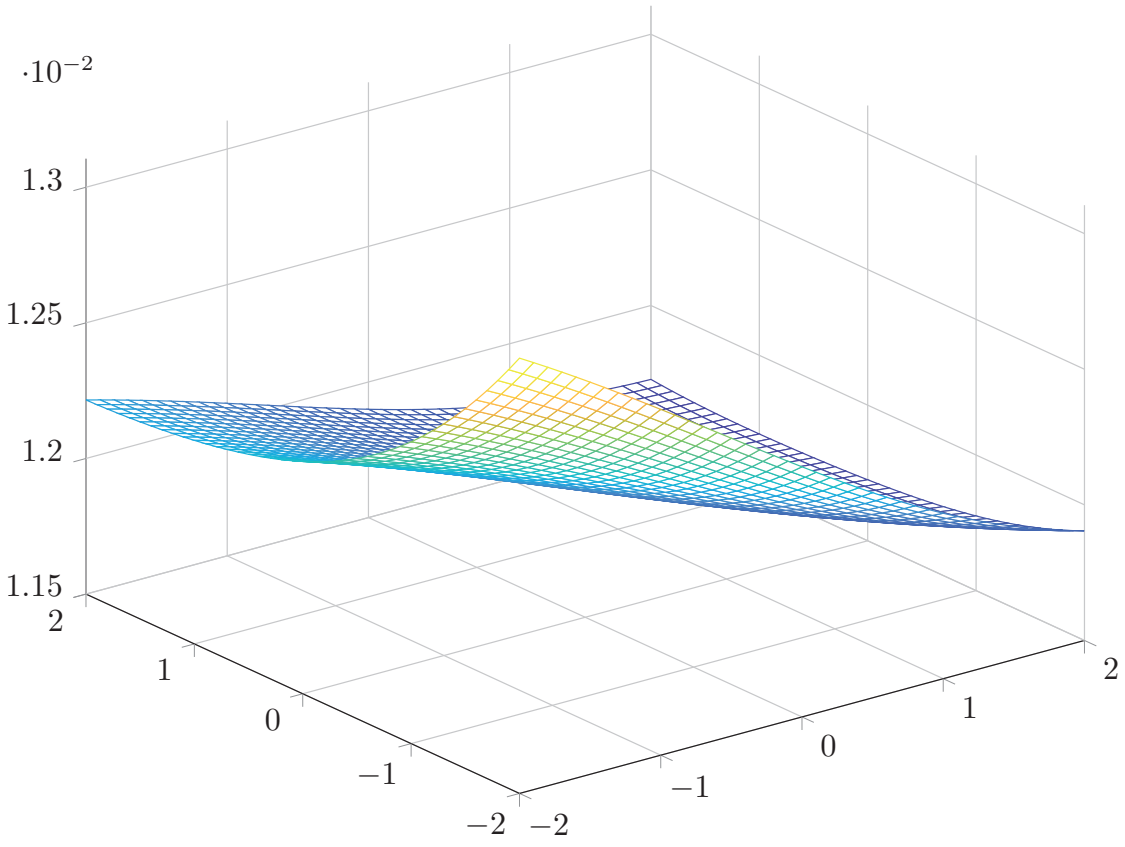

图 4-8　目标函数示意图：Ads, MLP

总结

在本章，我们分别考虑了对应三类优化问题的三种模型、稀疏和非稀疏两类数据集，以及六种随机优化方法：SGD, AdaGrad, AdaGrad-Norm, RMSProp, AdaDelta, Adam. 我们得出的主要结论是：Ada-系列算法（即 AdaGrad, RMSProp, AdaDelta, Adam）的整体表现较好，并且在稀疏数据集 Ads 上，差距更加明显。在总结与展望中，我们将给出造成此现象的原因。

第五章　总结与展望

关于 AdaGrad 的理论性质

在凸优化问题中，定理 2.1 给出了 AdaGrad 的 $O\left(1/\sqrt{T}\right)$ 的收敛率：

$$\mathbf{E}\left(f(\bar{x}_T)-f(x^*)\right)\leqslant\frac{n}{T}\sqrt{TM^2+\varepsilon}\left(\frac{d^2}{2\eta}+\eta\right).$$

该定理是由 AdaGrad 的遗憾界推导而来的，本质上是用在线优化的思路求解随机优化问题（在线-批量转换）。然而，一般而言，在线优化比随机优化困难，用在线-批量转换的思路未必能得到随机优化情形下最优的收敛速度。此外，定理 2.1 虽然给出了理论上的收敛性，但该收敛性是依赖于定义域直径的。在现实问题中，定义域的大小未必有明确的估计。

在光滑非凸优化问题中，定理 2.4 给出了 AdaGrad 的 $O\left(\ln T/\sqrt{T}\right)$ 的收敛速度。该定理能对 AdaGrad 的收敛作出理论性保证，但并不是理想的收敛速率的估计。具体而言，我们用目标函数在迭代点列处的平均梯度信息

$$\mathbf{E}_t\left(\left\|\nabla f(x^t)\right\|_2^2\right)=\mathbf{E}\left(\frac{1}{T}\sum_{k=1}^{T}\left\|\nabla f\left(x^k\right)\right\|_2^2\right)$$

来刻画收敛，这种刻画不如收敛率 $\mathbf{E}\left(f(\bar{x}_T)-f(x^*)\right)$ 明了且直接。此外，在定理 2.4 中，我们给出了 $\mathbf{E}_t\left(\left\|\nabla f(x^t)\right\|_2^2\right)$ 的一个上界

$$\frac{1}{\sqrt{T}}\left(A\ln\left(1+\frac{M^2}{\varepsilon}T\right)+B(f(x^1)-f_*)\right)=O\left(\frac{\ln T}{\sqrt{T}}\right).$$

但是，该估计是很宽松的。由于 A 和问题维数 n 成正比，B 和初始步长 η 成反比，且该估计中含有 $1/\varepsilon$ 项，那么当 n 较大而初始步长 η 和数值稳定性小量 ε 较小时，该上界并不是很好的估计。产生如上缺陷的原因是：在证明过程中，我们多次将随机梯度的分量放缩为其 l^∞ 范数的上界 M. 如果梯度较为稀疏，这将产

生很大的误差。为了避免过度放缩，我们可以考虑用梯度 l^2 范数的一致上界 M' 来取代 M 来进行收敛性分析，这样我们或许可以得到 $\mathbf{E}_t\left(\|\nabla f(x^t)\|_2^2\right)$ 与问题维数 n 无关的上界。

Ada-系列算法在稀疏数据下的优越性能

实验表明，Ada-系列算法在稀疏数据集上的训练结果和收敛结果都显著优于未进行逐分量自适应步长的方法。一个可能的原因是：如果某个特征很少出现，那么在该特征对应的分量上，梯度的累积将很少，因此 Ada-系列算法能在特征出现时作出及时的反应（在该方向给予一个较大步长）。而对于各分量步长统一的方法而言，当步长衰减到一定程度之后，将很难对新出现的特征给予反馈；如果数据集中某一特征出现很少却至关重要，这样的方法将忽视该特征的重要性，从而得出较差的结果。此外，由于稀疏数据储存结构的特殊性，我们很少对其进行标准化等预处理操作（否则可能破坏其稀疏性质，增加内存占用）。因此，数据在各个维度上的分布是极不均匀的，这种不均匀性导致了问题的病态。不过，Ada-系列算法在各个方向上的步长与其它方向的梯度无关，因而能够免受不均匀数据的影响。

我们对 Spambase 数据集的某些特征进行缩放，使其各维度不均匀，再重新用各方法进行训练（并重新用格点搜索确定最优初始步长），结果表明，Ada-系列算法仍具有较好性能，而 SGD 和 AdaGrad-Norm 的性能则产生较大下滑。例如，图 5-1 （见下页）就给出了数据进行缩放前和进行缩放后 Logistic 回归模型的训练结果。

关于 AdaGrad 对稀疏数据的适合性，Duchi 也给出了一个具体例子作为直观解释，参见 [6].

Ada-系列算法的性质总结

论文所介绍的 Ada-系列算法主要包括 AdaGrad, RMSProp, AdaDelta, Adam. 和 SGD, MSGD 等方法一样，它们都属于大规模优化问题中的常用算法。它们共同的优点是内存占用小、运算效率高、可用于在线学习等。下面我们对此类算法的各自独特的优缺点进行大致总结。

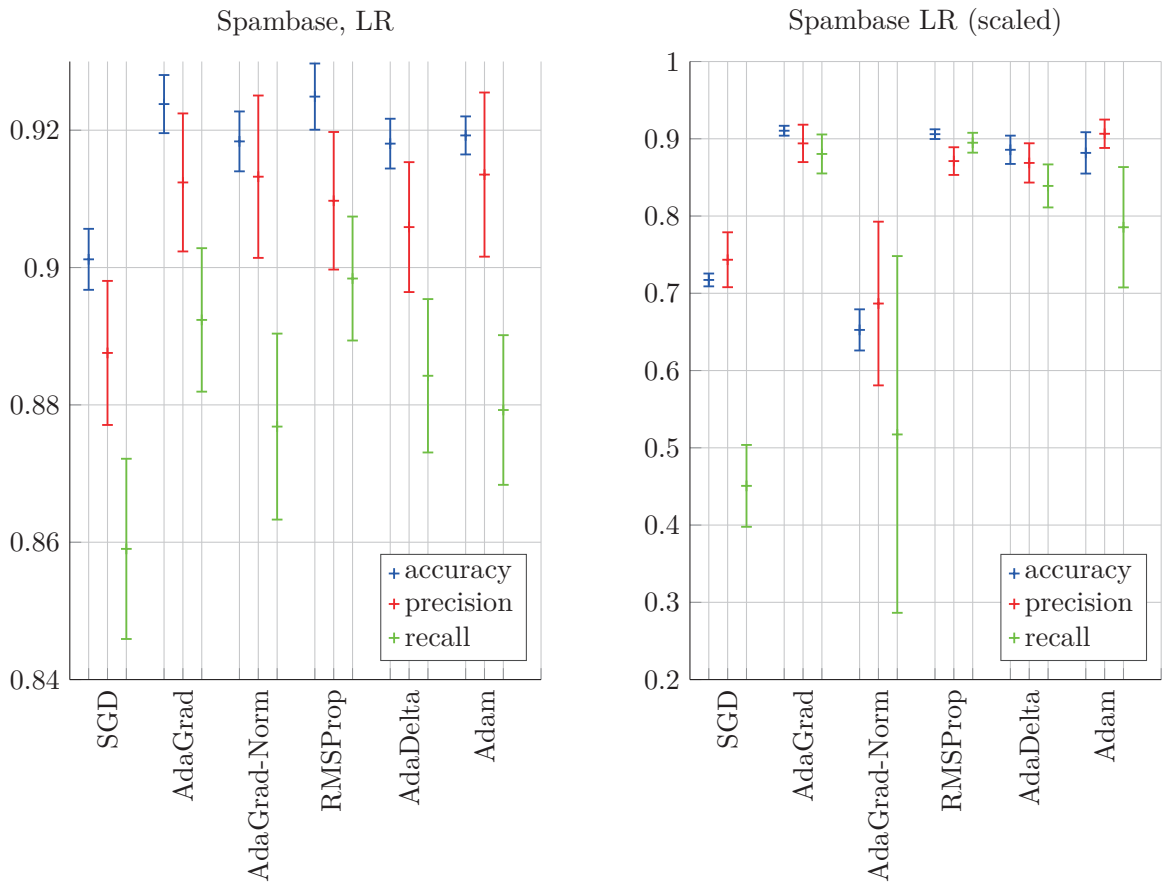

图 5-1　数据缩放前（左）和缩放后（右）的训练结果比较

- **SGD**：其优点是：超参数少，调整较方便；理论成果丰富，可解释性强；表现出较强的泛化能力 [36] 。其缺陷是：病态问题中收敛缓慢；对步长敏感，需要细致调节；难以逃脱鞍点；迭代过程不稳定，振荡较大。
- **AdaGrad**：其优点是：适合稀疏数据；在病态问题下收敛较快；无需手动调整步长。其缺陷是：步长单减趋于 0，迭代后期收敛缓慢；对新数据的学习能力较弱；在非凸优化中可能出现早停问题。
- **RMSProp**：其优点是：适合稀疏和带噪声的数据；在病态问题下收敛较快；无需手动调整步长，同时避免了步长过早衰减；适合在线学习，对新数据学习能力较强。其缺陷是：在神经网络等模型中泛化能力可能较弱；缺乏收敛性的理论保证，迭代可能发散。
- **AdaDelta**：其优点是：适合稀疏数据；需要调整的超参数较少；对不同模型和超参数的鲁棒性强。其缺陷是：在特殊问题中表现可能欠佳；迭代格式较复杂；缺乏收敛性的理论保证，可能发散。
- **Adam**：其优点是：适合稀疏和带噪声的数据；易于调参，默认参数就能取

得较好结果；在病态问题下收敛较快；适合在线学习，对新数据学习能力较强。其缺陷是: 在神经网络等模型中泛化能力可能较弱；迭代格式较复杂；缺乏收敛性的理论保证，可能发散。

虽然 Ada-系列算法在大规模优化问题中展现出种种优越性能，但我们仍要指出，在实际操作中 SGD 仍被广泛采用，并表现出不劣于 Ada-系列算法的性能。此外，在大规模优化问题中受内存和运算量限制未被广泛使用的二阶方法也有其改进形式，在实际计算时未必劣于一阶方法。

索引

参考文献

[1] 李航. 统计学习方法 [M]. 清华大学出版社, 2012.

[2] 刘浩洋, 户将, 李勇锋, 等. 最优化：建模、算法与理论 [M]. 高等教育出版社, 2021.

[3] ROBBINS H E. A Stochastic Approximation Method [J]. Annals of Mathematical Statistics, 1951, 22: 400-407.

[4] AUER P, CESA-BIANCHI N, GENTILE C. Adaptive and Self-Confident On-Line Learning Algorithms [J]. Journal of Computer and System Sciences, 2002, 64(1): 48-75.

[5] MCMAHAN H B, STREETER M J. Adaptive Bound Optimization for Online Convex Optimization [C]. The 23rd Conference on Learning Theory, 2010: 244-256.

[6] DUCHI J C, HAZAN E, SINGER Y. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization [J]. J. Mach. Learn. Res., 2011, 12: 2121-2159.

[7] DUCHI J C, JORDAN M I, MCMAHAN H B. Estimation, Optimization, and Parallelism when Data is Sparse [C]. Neural Information Processing Systems, 2013.

[8] WILSON A C, ROELOFS R, STERN M, et al. The Marginal Value of Adaptive Gradient Methods in Machine Learning [C]. Neural Information Processing Systems, 2017.

[9] LEVY K Y, YURTSEVER A, CEVHER V. Online Adaptive Methods, Universality and Acceleration [C]. Advances in Neural Information Processing Systems

31: Annual Conference on Neural Information Processing Systems, 2018: 6501-6510.

[10] ENE A, NGUYEN H L, VLADU A. Adaptive Gradient Methods for Constrained Convex Optimization [C]. AAAI Conference on Artificial Intelligence, 2020.

[11] LIU Z, NGUYEN T D, ENE A, et al. On the Convergence of AdaGrad on $\mathbb{R}^d$: Beyond Convexity, Non-Asymptotic Rate and Acceleration [C]. The Eleventh International Conference on Learning Representations, 2023.

[12] LI X, ORABONA F. On the Convergence of Stochastic Gradient Descent with Adaptive Stepsizes [C]. International Conference on Artificial Intelligence, 2018.

[13] LIU Z, NGUYEN T D, NGUYEN T H, et al. High Probability Convergence of Stochastic Gradient Methods [J]. ArXiv, 2023, abs/2302.14843.

[14] KAVIS A, LEVY K Y, CEVHER V. High Probability Bounds for a Class of Non-convex Algorithms with AdaGrad Stepsize [C]. The Tenth International Conference on Learning Representations, 2022.

[15] WANG B, ZHANG H, MA Z, et al. Convergence of AdaGrad for Non-convex Objectives: Simple Proofs and Relaxed Assumptions [C]. The Thirty Sixth Annual Conference on Learning Theory, 2023: 161-190.

[16] WARD R A, WU X, BOTTOU L. AdaGrad stepsizes: Sharp convergence over nonconvex landscapes, from any initialization [C]. International Conference on Machine Learning, 2018.

[17] MUKKAMALA M C, HEIN M. Variants of RMSProp and Adagrad with Logarithmic Regret Bounds [C]. Proceedings of the 34th International Conference on Machine Learning, 2017: 2545-2553.

[18] MCMAHAN H B, STREETER M J. Adaptive Bound Optimization for Online Convex Optimization [C]. The 23rd Conference on Learning Theory, 2010: 244-256.

[19] JIN R, XING Y, HE X. On the Convergence of mSGD and AdaGrad for Stochastic Optimization [C]. The Tenth International Conference on Learning Representations, 2022.

[20] ZHOU D, TANG Y, YANG Z, et al. On the Convergence of Adaptive Gradient Methods for Nonconvex Optimization [J]. ArXiv, 2018, abs/1808.05671.

[21] D'EFOSSEZ A, BOTTOU L, BACH F R, et al. A Simple Convergence Proof of Adam and Adagrad [J]. Trans. Mach. Learn. Res., 2020, 2022.

[22] HONG Y, LIN J. Revisiting Convergence of AdaGrad with Relaxed Assumptions [J]. ArXiv, 2024, abs/2402.13794.

[23] ZEILER M D. ADADELTA: An Adaptive Learning Rate Method [J]. CoRR, 2012, abs/1212.5701.

[24] KINGMA D P, BA J. Adam: A Method for Stochastic Optimization [C]. 3rd International Conference on Learning Representations, 2015.

[25] REDDI S J, KALE S, KUMAR S. On the Convergence of Adam and Beyond [C]. 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings, 2018.

[26] RUDER S. An overview of gradient descent optimization algorithms [J]. ArXiv, 2016, abs/1609.04747.

[27] BOTTOU L, CURTIS F E, NOCEDAL J. Optimization Methods for Large-Scale Machine Learning [J]. SIAM Rev., 2018, 60(2): 223-311.

[28] ZINKEVICH M A. Online Convex Programming and Generalized Infinitesimal Gradient Ascent [C]. International Conference on Machine Learning, 2003.

[29] 王东. 机器学习导论 [M]. 清华大学出版社, 2021.

[30] ZOU F, SHEN L, JIE Z, et al. A Sufficient Condition for Convergences of Adam and RMSProp [C]. IEEE Conference on Computer Vision, 2019: 1119+.

[31] SHI N, LI D, HONG M, et al. RMSprop converges with proper hyper-parameter [C]. International Conference on Learning Representations, 2021.

[32] POLYAK B. Some methods of speeding up the convergence of iteration methods [J]. Ussr Computational Mathematics and Mathematical Physics, 1964, 4: 1-17.

[33] NESTEROV Y. A method for solving the convex programming problem with convergence rate $O(1/k^2)$ [J]. Proceedings of the USSR Academy of Sciences, 1983, 269: 543-547.

[34] BENGIO Y, BOULANGER-LEWANDOWSKI N, PASCANU R. Advances in optimizing recurrent networks [J]. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2012: 8624-8628.

[35] CHEN C, SHEN L, ZOU F, et al. Towards Practical Adam: Non-Convexity, Convergence Theory, and Mini-Batch Acceleration [J]. J. Mach. Learn. Res., 2021, 23: 229:1-229:47.

[36] MA C, YING L. On Linear Stability of SGD and Input-Smoothness of Neural Networks [C]. Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems, 2021: 16805-16817.

[37] HOPKINS M, REEBER E, FORMAN G, et al. Spambase [DB/OL]. 1999. DOI: https://doi.org/10.24432/C53G6X.

[38] KUSHMERICK N. Internet Advertisements [DB/OL]. 1998. DOI: https://doi.org/10.24432/C5V011.

附录 A　技术性引理

引理 A.1 若 f 为 $\mathbb{R}^n$ 上的可微函数，并且 $\|\nabla f(x)-\nabla f(y)\|_2 \leqslant L\|x-y\|_2$，则 f 具有二次上界：

$$f(y) \leqslant f(x)+(y-x)\cdot\nabla f(x)+\frac{L}{2}\|y-x\|_2^2. \tag{A-1}$$

证明 $\forall\, x,y\in\mathbb{R}^n$，令 $F(t)=f(x+t(y-x))$，则 $F(0)=f(x)$, $F(1)=f(y)$，利用 N-L 公式以及条件 $\|f(x)-f(y)\|_2 \leqslant L\|x-y\|_2$ 可见

$$\begin{aligned}
f(y)-f(x)=F(1)-F(0)&=\int_0^1 F'(s)\,ds\\
&=\int_0^1 \nabla f(x+s(y-x))\cdot(y-x)\,ds\\
&=\int_0^1 (\nabla f(x+s(y-x))-\nabla f(x))\cdot(y-x)\,ds+\nabla f(x)\cdot(y-x)\\
&\leqslant\int_0^1 \|\nabla f(x+s(y-x))-\nabla f(x)\|_2\|y-x\|_2\,ds+\nabla f(x)\cdot(y-x)\\
&\leqslant\int_0^1 L\|y-x\|_2\,s\,ds+\nabla f(x)\cdot(y-x)\\
&=\frac{L\|y-x\|_2^2}{2}+\nabla f(x)\cdot(y-x).
\end{aligned}$$

故 $f(y)\leqslant f(x)+(y-x)\cdot\nabla f(x)+\frac{L}{2}\|y-x\|_2^2$. □

引理 A.2 若 f 为 $\mathbb{R}^n$ 上的可微函数，$\|\nabla f(x)-\nabla f(y)\|_2 \leqslant L\|x-y\|_2$，设 x^* 为 f 的最小值点，则 $\forall\, x\in\mathbb{R}^n$，我们有

$$\|\nabla f(x)\|_2^2 \leqslant 2L\left(f(x)-f(x^*)\right). \tag{A-2}$$

证明 由引理 A.1 可见

$$\inf_{y\in\mathbb{R}^n} f(y) \leqslant \inf_{y\in\mathbb{R}^n}\left(f(x)+(y-x)\cdot\nabla f(x)+\frac{L}{2}\|y-x\|_2^2\right), \tag{A-3}$$

注意到A-3式右边等于 $f(x)-\frac{\|\nabla f(x)\|_2^2}{2L}$，而左边为 $f(x^*)$，代入可得A-2式。 □

引理 A.3 对任意非负数列 $\{a_k\}_{k=1}^T$ 以及正数 ε，成立

$$\sum_{k=1}^{T}\frac{a_k}{\varepsilon+\sum_{j=1}^{k}a_j}\leqslant \ln\left(1+\frac{\sum_{s=1}^{T}a_s}{\varepsilon}\right). \tag{A-4}$$

证明 记 $S_k=\sum_{j=1}^{k}a_j$，在不等式 $x\geqslant \ln(1+x)$ $(-1<x<\infty)$ 中代入 $x=-\frac{a_k}{\varepsilon+S_k}$ 可见

$$\frac{a_k}{\varepsilon+S_k}\leqslant \ln(\varepsilon+S_k)-\ln(\varepsilon+S_k-a_k), \tag{A-5}$$

A-5式两边对 k 从1到 T 求和，注意到 $S_k-a_k=S_{k-1}$ $(k=2,3,\cdots,T)$，易见结论成立。 □

引理 A.4 对任意非负数列 $\{a_k\}_{k=1}^T$ 以及正数 ε，成立

$$\sum_{k=1}^{T}\frac{a_k}{\sqrt{\varepsilon+\sum_{j=1}^{k}a_j}}\leqslant 2\sqrt{\varepsilon+\sum_{k=1}^{T}a_k}\,. \tag{A-6}$$

证明 易于证明不等式

$$\frac{y}{\sqrt{x+y}}\leqslant 2(\sqrt{x+y}-\sqrt{x}), \tag{A-7}$$

其中 $x>0, x+y>0$. 在A-7式中取 $y=a_k$, $x=\varepsilon+\sum_{j=1}^{k-1}a_j$ 并对 $k=2,3,\cdots,T$ 求和，可见

$$\begin{aligned}\sum_{k=1}^{T}\frac{a_k}{\sqrt{\varepsilon+\sum_{j=1}^{k}a_j}}&\leqslant 2\sum_{k=2}^{T}\left(\sqrt{\varepsilon+\sum_{j=1}^{k}a_j}-\sqrt{\varepsilon+\sum_{j=1}^{k-1}a_j}\right)+\frac{a_1}{\sqrt{\varepsilon+a_1}}\\&\leqslant 2\sqrt{\varepsilon+\sum_{j=1}^{T}a_j}-2\sqrt{\varepsilon+a_1}+\frac{a_1}{\sqrt{\varepsilon+a_1}}.\end{aligned} \tag{A-8}$$

易见 $a_1/\sqrt{\varepsilon+a_1}\leqslant 2\sqrt{\varepsilon+a_1}$，故A-6式成立。 □

附录 B　实验数据来源与参数设定

论文实验中的数据集信息与预处理操作参见下表：

	垃圾邮件库 (Spambase) [37]	**网络广告信息 (Ads)** [38]
数据量	4601	3279
特征数	57	1558
非零元占比	22.59%	0.82%
正例/反例	垃圾邮件/非垃圾邮件 ≈ 0.806	广告/非广告 ≈ 0.163
数据集特点	连续值非稀疏矩阵	0 - 1 值稀疏矩阵
论文选用的特征	全部	第 4 至 1557 列
数据预处理操作	将第 55 至 57 列进行标准化	无
训练、验证集划分	$4:1$	$4:1$

论文中各优化方法初始步长由格点搜索确定，待选步长为

$$[10^{-4}, \cdots, 9\times 10^{-4}, 10^{-3}, \cdots, 9\times 10^{-3}, 0.01, \cdots, 0.09, 0.1, \cdots, 0.9, 1, 2, \cdots, 20].$$

格点搜索结果如下：

	SGD	**AdaGrad**	**AdaGrad-Norm**	**RMSProp**	**Adam**
Logistic 回归，Spambase	7	0.5	4	0.007	0.01
Logistic 回归，Ads	0.1	0.1	5	0.009	0.008
SVM，Spambase	4	0.6	5	0.007	0.008
SVM，Ads	0.07	0.6	0.3	0.008	0.009
MLP，Spambase	0.8	0.8	6	0.07	0.06
MLP，Ads	0.2	0.5	6	0.08	0.06

SGD 的步长由 $\eta_k = \eta/(1+0.1\eta k)$ 确定，其中 η 为初始步长。此外，Ada-系列算法中数值稳定性小量 ε 均取 10^{-7}；RMSProp 的衰减系数 $\rho = 0.9$，AdaDelta 的衰减系数 $\rho = 0.95$；Adam 的衰减系数 $\rho_1 = 0.9$, $\rho_2 = 0.999$. 各模型中各方法所选取的批量大小均取 32.

论文采用梅森旋转 MT19937 生成的随机数流，全局随机种子 seed $= 42$.