§ 讲 义

2019 年 6 月 5 日

目录

1 凸优化

1.1 引言

优化是机器学习的基础数学工具。在我们对学习问题进行建模以后，通常会将学习任务形式化为数学优化问题，然后借助优化技术求解。以监督学习为例，令$(\mathbf{x}, y)$代表“示例-标记”对，其分布记为$\mathcal{D}$，令$\mathcal{H}$为某函数空间。监督学习旨在从$\mathcal{H}$中寻找某函数h使得$h(\mathbf{x})$能够预测标记y。该问题可以形式化为：

$$\min_{h\in\mathcal{H}} \ \mathrm{E}_{(\mathbf{x},y)\sim\mathcal{D}}\big[\ell(h(\mathbf{x}), y)\big] \tag{1}$$

其中$\ell(\cdot,\cdot)$为衡量预测误差的损失函数。上述问题被称为风险最小化（Risk Minimization），本质上是一个随机优化问题，由于分布$\mathcal{D}$通常是不知道的，因此难以直接求解。但是，我们往往有一组从$\mathcal{D}$中独立采样得到的n个样本$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$，因此可以通过最小化在这些样本上的误差来选择预测函数。这就是著名的经验风险最小化：

$$\min_{h\in\mathcal{H}} \ \frac{1}{n}\sum_{i=1}^{n}\ell(h(\mathbf{x}_i), y_i)$$

然后，我们就需要设计合适的优化算法求解上述问题。本章重点介绍优化领域中的凸优化技术。

凸优化是一种特殊的优化问题，这类优化问题具有全局最优解，并且通常可以在多项式时间内求解。很多机器学习任务都可以建模为凸优化问题，因此在机器学习领域有广泛的应用。下面，我们介绍凸优化的基本定义，对偶问题，以及常用的梯度下降法。更详细的内容可以参考相关书籍[Boyd and Vandenberghe, 2004, Nesterov, 2004]。

1.2 凸优化问题

一般的优化问题可以表示为

$$\begin{aligned}\min_{\mathbf{x}} \quad & f_0(\mathbf{x}) \\ \text{s.t.} \quad & f_i(\mathbf{x}) \le 0, \quad i = 1, \ldots, n\end{aligned}$$

其中$\mathbf{x} = [x_1, \ldots, x_d]^\top \in \mathbb{R}^d$为$d$维的优化变量，$f_0 : \mathbb{R}^d \mapsto \mathbb{R}$代表目标函数，$f_i : \mathbb{R}^d \mapsto \mathbb{R}, i = 1, \ldots, n$表示约束函数。这个问题的含义是在满足约束$f_i(\mathbf{x}) \le 0$的情况下，寻找$\mathbf{x}$来最小化目标函数$f_0(\cdot)$。该问题的最优解可以定义为：

$$\{\mathbf{x}_* | f_0(\mathbf{x}_*) \le f_0(\mathbf{x}), \forall \mathbf{x} \in \Omega\}$$

其中集合$\Omega = \{\mathbf{x} | f_i(\mathbf{x}) \le 0, i = 1, \ldots, n\}$。

设计通用的优化算法是很困难的，但是对于某些特殊的优化问题，我们可以很容易的求解。比如针对回归任务的最小二乘（Least Squares）：

$$\min_{\mathbf{x}} \ f_0(\mathbf{x}) = \sum_{i=1}^{n}\left(\mathbf{a}_i^\top\mathbf{x} - b_i\right)^2 = \|A\mathbf{x} - \mathbf{b}\|_2^2 \tag{2}$$

其中$A=[\mathbf{a}_1,\ldots,\mathbf{a}_n]^\top\in\mathbb{R}^{n\times d}$代表数据矩阵，$\mathbf{b}=[b_1,\ldots,b_n]^\top\in\mathbb{R}^n$代表标记。对于最小二乘问题，是存在闭合最优解

$$\mathbf{x}_*=(A^\top A)^{-1}A^\top\mathbf{b}$$

并且存在大量的算法和工具来计算上述表达式，求解该问题的复杂度为$O(d^2n)$。一般而言，我们可以认为求解最小二乘是成熟的技术。当然，如果我们面临的问题规模特别大，比如n和d的数值非常大，求解最小二乘也会变得很困难，需要利用更加高级的技术。

另外一类成熟的问题是线性规划（Linear Programming）：

$$\begin{aligned}\min_{\mathbf{x}}\quad&\mathbf{c}^\top\mathbf{x}\\ \text{s.t.}\quad&\mathbf{a}_i^\top\mathbf{x}\le b_i,\quad i=1,\ldots,n\end{aligned}$$

其中$\mathbf{c},\mathbf{a}_1,\ldots,\mathbf{a}_n\in\mathbb{R}^d$，$b_1,\ldots,b_n\in\mathbb{R}$。线性规划的特点是其目标函数和约束函数都是线性的。虽然线性规划没有闭合解，但是这类问题求解容易，存在很多可靠的算法和工具。当$n\ge d$时，其复杂度也是$O(d^2n)$，求解线性规划同样可以认为是成熟的技术。

最后我们引入本节的重点—凸优化问题（Convex Optimization Problem）：

$$\begin{aligned}\min_{\mathbf{x}}\quad&f_0(\mathbf{x})\\ \text{s.t.}\quad&f_i(\mathbf{x})\le 0,\quad i=1,\ldots,n\end{aligned}$$

其中目标函数和约束函数都是凸的，也就是要满足下面的条件：

$$f_i(\theta\mathbf{x}+(1-\theta)\mathbf{y})\le\theta f_i(\mathbf{x})+(1-\theta)f_i(\mathbf{y}),\ \forall\theta\in[0,1].$$

对于一般的凸优化问题，同样没有闭合解，但是存在可靠的算法，因此可以认为凸优化几乎是成熟的技术。

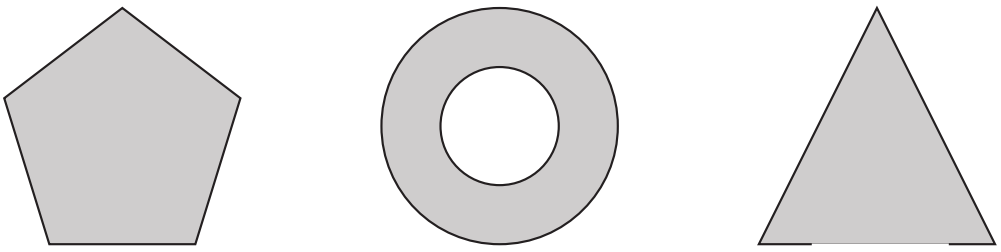
下面，我们介绍凸优化的基本概念。

定义1.1 (凸集合). *如果集合$\mathcal{C}$内任意两点的连线属于这个集合，即*

$$\mathbf{x}_1,\mathbf{x}_2\in\mathcal{C},0\le\theta\le 1\Rightarrow\theta\mathbf{x}_1+(1-\theta)\mathbf{x}_2\in\mathcal{C},$$

我们称集合$\mathcal{C}$为凸的。

我们可以根据定义判断一个集合是否是凸的。比如下面的3个集合中，很显然只有第一个集合是凸的，其他两个都不满足定义。



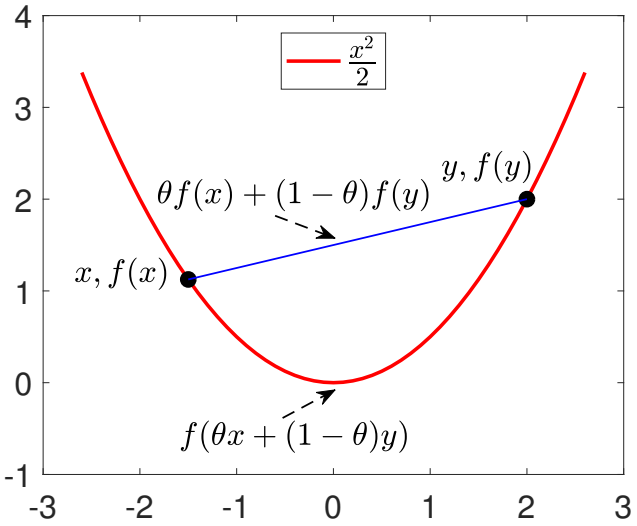
接下来我们引入凸函数的概念。

定义1.2 (凸函数). *当函数$f:\mathbb{R}^d\mapsto\mathbb{R}$满足下面两个条件时，我们称其为凸函数*

- *定义域*dom f*是凸集合；*
- *对于任意的*$\mathbf{x},\mathbf{y}\in\text{dom}\,f$, $\theta\in[0,1]$, *下面式子成立*

$$f(\theta\mathbf{x}+(1-\theta)\mathbf{y})\leq\theta f(\mathbf{x})+(1-\theta)f(\mathbf{y}).$$

其中第二个条件可以直观理解为：函数$f(\cdot)$上任意两点的连线，都在该函数上方。下面我们给出一个凸函数$f(x)=x^2/2$的例子：



从上面的例子可以看出该函数满足上述定义，因此是凸函数。与凸函数相对应的概念就是凹函数，其定义如下：

定义1.3 (凹函数). *当函数$f:\mathbb{R}^d\mapsto\mathbb{R}$满足下面两个条件时，我们称其为凹函数*

- *定义域*dom f*是凸集合；*
- *对于任意的*$\mathbf{x},\mathbf{y}\in\text{dom}\,f$, $\theta\in[0,1]$, *下面式子成立*

$$f(\theta\mathbf{x}+(1-\theta)\mathbf{y})\geq\theta f(\mathbf{x})+(1-\theta)f(\mathbf{y}).$$

通过上面的定义可以看出，如果函数$f(\cdot)$是凸函数，那么$-f(\cdot)$是凹函数，反之亦成立。

下面我们给出一些常见的凸函数。常见的1维凸函数有：

- 仿射函数: 定义在$\mathbb{R}$的函数$ax+b$, 其中$a,b\in\mathbb{R}$；
- 指数函数: 定义在$\mathbb{R}$的函数e^{ax}, 其中$a\in\mathbb{R}$；

- 幂函数: 定义在$\mathbb{R}_{++}$的函数x^a, 其中$a \geq 1$ 或$a \leq 0$;
- 绝对值幂函数: 定义在$\mathbb{R}$的函数$|x|^p$, 其中$p \geq 1$;
- 负熵: 定义在$\mathbb{R}_{++}$的函数$x \log x$。

常见的d维凸函数有：

- 仿射函数：定义在$\mathbb{R}^d$的函数$\mathbf{a}^\top \mathbf{x} + b$, 其中$\mathbf{a} \in \mathbb{R}^d$, $b \in \mathbb{R}$
- 范数: 定义在$\mathbb{R}^d$的函数$\|\mathbf{x}\|_p = (\sum_{i=1}^d |x_i|^p)^{1/p}$, 其中$p \geq 1$

除了根据定义判断一个函数是否是凸函数之外，还有其他的一些方式。下面我们介绍判断凸函数的一阶和二阶方法。首先引入函数梯度的概念，对于一个函数$f(\cdot) : \mathbb{R}^d \mapsto \mathbb{R}$，我们将其梯度记为

$$\nabla f(\mathbf{x}) = \left[\frac{\partial f(\mathbf{x})}{\partial x_1}, \ldots, \frac{\partial f(\mathbf{x})}{\partial x_d}\right]^\top \in \mathbb{R}^d$$

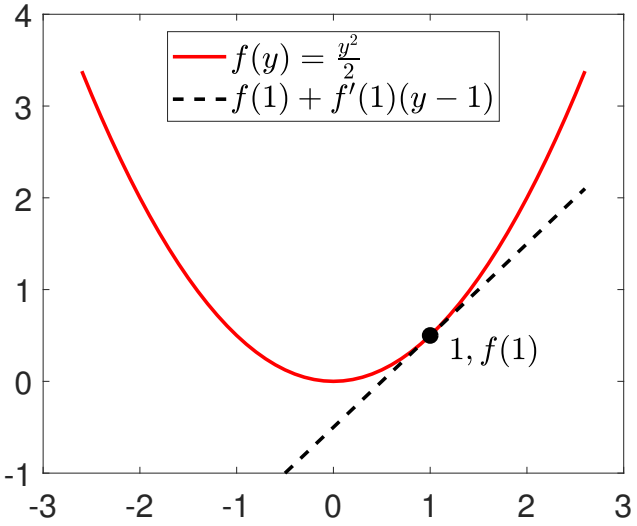
我们可以通过下面的定理判断函数是否为凸。

定理1.1 (凸函数的一阶条件). *假设函数f是可微分的。那么f是凸函数当且仅当*

- *定义域$\operatorname{dom} f$ 是凸集合;*
- *对于所有的$\mathbf{x}, \mathbf{y} \in \operatorname{dom} f$，下面的式子成立*

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) \tag{3}$$

上面第二个条件的直观含义是指：函数f在任意一个点$\mathbf{x} \in \operatorname{dom} f$的一阶泰勒展开是其下界。下面我们给出一个凸函数$f(y) = \frac{1}{2}y^2$的例子，以及该函数在$(1, f(1))$处的一阶展开。



显然，上述函数满足定理1.1中的判定条件。

接下来，我们介绍凸函数的二阶条件。首先引入Hessian矩阵，也就是二阶导数矩阵。函数$f(\cdot): \mathbb{R}^d \mapsto \mathbb{R}$在$\mathbf{x} \in \operatorname{dom} f$的Hessian矩阵记为$\nabla^2 f(\mathbf{x}) \in \mathbb{R}^{d \times d}$，其中

$$\nabla^2 f(\mathbf{x})_{ij} = \frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j}.$$

定理1.2 (凸函数的二阶条件). *假设函数f是二阶可微分的。那么f是凸函数当且仅当*

- *定义域*$\operatorname{dom} f$ *是凸集合；*
- *函数f在所有$\mathbf{x} \in \operatorname{dom} f$上的Hessian矩阵都是半正定的，即$\nabla^2 f(\mathbf{x}) \succeq 0$。*

定理1.2尤其适用于判断二次函数的凹凸性：二次函数$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top A\mathbf{x} + \mathbf{b}^\top \mathbf{x} + c$是凸函数当且仅当$A \succeq 0$。

除了根据定义1.3、定理1.1和定理1.2之外，我们还有一些快速判断凹凸性的方式。通过证明某些数学操作可以保持凸性质，我们可以对基本的凸函数进行变换，得到更复杂的凸函数。保持凸性质的数学操作有：

- 非负加权求和：如果$f_1, \ldots, f_n$是凸函数，并且$w_1, \ldots, w_n \geq 0$, 那么

  $$f = w_1 f_1 + w_2 f_2 + \cdots, w_n f_n$$

  为凸函数。

- 与仿射函数复合：如果f是凸函数，那么

  $$g(\mathbf{x}) = f(A\mathbf{x} + \mathbf{b})$$

  为凸函数。

- 最大值或上确界：如果$f_1, \ldots, f_n$是凸函数，那么

  $$f(x) = \max\{f_1(x), \ldots, f_n(x)\}$$

  为凸函数。如果对于所有的$y \in \mathcal{A}$, $f(x, y)$是关于$\mathbf{x}$的凸函数，那么

  $$g(x) = \sup_{y \in \mathcal{A}} f(x, y)$$

  为凸函数。

这里举一个例子，对于支持向量机中的Hinge损失$\ell(\mathbf{w}) = \max(0, 1 - y\mathbf{x}^\top \mathbf{w})$，我们可以通过最大值操作证明其为凸函数。

接下来了，我们介绍一个和凸函数非常相关的不等式。我们在定义1.3中所用到不等式

$$f(\theta\mathbf{x} + (1-\theta)\mathbf{y}) \leq \theta f(\mathbf{x}) + (1-\theta) f(\mathbf{y})$$

也被称为Jensen不等式。Jensen不等式可以拓展到多个函数求和的形式。假设f是凸函数，$\mathbf{x}_1,\ldots,\mathbf{x}_n \in \operatorname{dom} f$，$\theta_1,\ldots,\theta_n \geq 0$，并且$\theta_1+\cdots+\theta_n=1$，那么

$$f(\theta_1\mathbf{x}_1+\cdots+\theta_n\mathbf{x}_n) \leq \theta_1 f(\mathbf{x}_1)+\cdots\theta_n f(\mathbf{x}_n).$$

更一般地，假设f是凸函数，X为某随机变量，那么

$$f(\mathrm{E}[X]) \leq \mathrm{E}\left[f(X)\right].$$

到目前为止，我们已经了解了凸集合和凸函数的定义以及判定方法，下面给出凸优化问题的标准形式：

$$\begin{aligned} \min_{\mathbf{x}} \quad & f_0(\mathbf{x}) \\ \text{s.t.} \quad & f_i(\mathbf{x}) \leq 0, \quad i=1,\ldots,n \\ & \mathbf{a}_i^\top \mathbf{x} = b_i, \quad i=1,\ldots,p \end{aligned} \tag{4}$$

其中$f_0,f_1,\ldots,f_n$是凸函数。注意到在上面的优化问题中，存在不等号约束和等号约束，我们要求等号约束必须是仿射函数。凸优化问题有一个重要的特点：“凸优化问题的任意一个局部最优解就是全局最优解”。此外，对于凸优化问题，其最优解满足下面的性质。

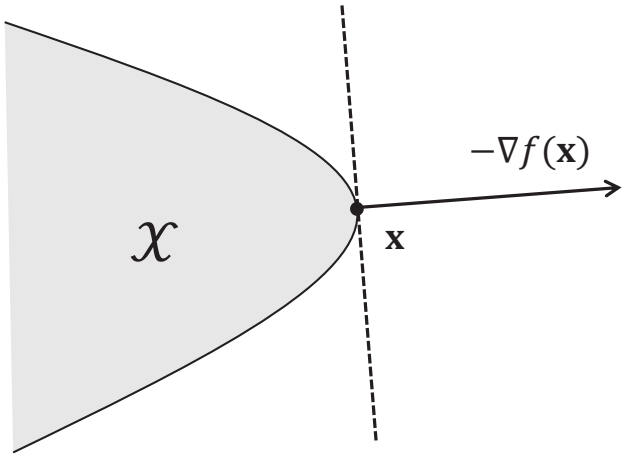
定理1.3 (凸优化最优解条件). *假设目标函数f_0是可微分的，定义可行域*

$$X=\left\{\mathbf{x}|f_i(\mathbf{x})\leq 0, \quad i=1,\ldots,n,\ \mathbf{a}_i^\top\mathbf{x}=b_i, \quad i=1,\ldots,p\right\}.$$

$\mathbf{x}$是问题(4)的最优解当且仅当$\mathbf{x}\in\mathcal{X}$，并且

$$\nabla f_0(\mathbf{x})^\top(\mathbf{y}-\mathbf{x})\geq 0,\ \forall \mathbf{y}\in\mathcal{X}.$$

上述定理可以直观理解为：$-\nabla f_0(\mathbf{x})$在$\mathbf{x}$处定义了一个可行域$\mathcal{X}$的支撑平面。



对于没有约束条件的凸优化问题，定理1.3中的判定条件可以退化为

$$\nabla f_0(\mathbf{x})=0$$

也就是说最优解处的梯度为0。

最后，我们引入共轭函数（Conjugate Function）的概念，这个概念将在下一节中的对偶问题中用到。

定义1.4. *函数$f:\mathbb{R}^d \mapsto \mathbb{R}$的共轭函数被定义为*

$$f^*(\mathbf{y}) = \sup_{\mathbf{x}\in \mathrm{dom}\, f} \left(\mathbf{y}^\top \mathbf{x} - f(\mathbf{x})\right)$$

从定义可以看出，$f^*(\mathbf{y})$反应的是线性函数$\mathbf{y}^\top\mathbf{x}$和$f(\mathbf{x})$之间最大的差值。共轭函数的定义域为

$$\mathrm{dom}\, f^* = \left\{\mathbf{y} | \sup_{\mathbf{x}\in \mathrm{dom}\, f} \left(\mathbf{y}^\top \mathbf{x} - f(\mathbf{x})\right) < \infty\right\}.$$

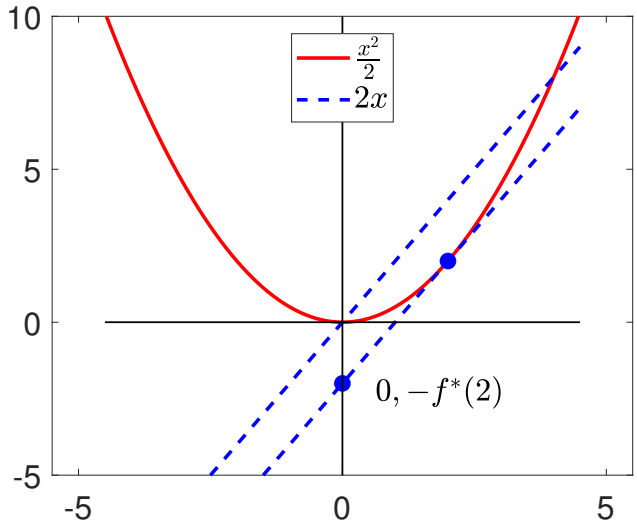
共轭函数有如下性质：

- 无论原函数f是否为凸函数，共轭函数f^*一定为凸。
- 如果函数f是可微分的，那么

$$f^*(\mathbf{y}) = \nabla f(\mathbf{x})^\top \mathbf{x} - f(\mathbf{x}) = -\left[f(\mathbf{x}) + \nabla f(\mathbf{x})^\top (0 - \mathbf{x})\right]$$

其中$\mathbf{y} = \nabla f(\mathbf{x})$。

下面，我们给出一个共轭函数的示意图



图中表示如何计算原函数$f(x) = \frac{1}{2}x^2$的共轭函数$f^*(y)$在$y = 2$处的数值。根据定义$f^*(2)$是线性函数$2x$和原函数$f(x)$之间的最大差值，也就是图中两条虚线之间的垂直距离。

1.3 对偶问题

在本节，我们介绍如何推导一个优化问题的对偶问题。研究对偶问题的原因在于，对于一个复杂的优化问题，其对偶问题可能非常容易求解。并且通过求解对偶问题，我们可以得到原始问题的最优解。

我们考虑一般的优化问题（未必是凸优化）：

$$\begin{aligned} \min_{\mathbf{x}} \quad & f_0(\mathbf{x}) \\ \text{s.t.} \quad & f_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, n \\ & h_i(\mathbf{x}) = 0, \quad i = 1, \ldots, p \end{aligned} \tag{5}$$

其中$\mathbf{x} \in \mathbb{R}^d$是优化变量。我们将上述问题的定义域记为$\mathcal{D}$，最优值记为$p_*$。

对于优化问题(5),其拉格朗日（Lagrangian）函数$L : \mathbb{R}^d \times \mathbb{R}^n \times \mathbb{R}^p \mapsto \mathbb{R}$定义为

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = f_0(\mathbf{x}) + \sum_{i=1}^{n} \lambda_i f_i(\mathbf{x}) + \sum_{i=1}^{p} \nu_i h_i(\mathbf{x})$$

其中λ_i是针对不等号约束$f_i(\mathbf{x}) \leq 0$引入的拉格朗日乘子，ν_i 为针对等号约束$h_i(\mathbf{x}) = 0$的拉格朗日乘子。根据拉格朗日函数，我们进一步可以定义拉格朗日对偶函数$g : \mathbb{R}^n \times \mathbb{R}^p \mapsto \mathbb{R}$

$$\begin{aligned} g(\boldsymbol{\lambda}, \boldsymbol{\nu}) &= \inf_{\mathbf{x} \in \mathcal{D}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \\ &= \inf_{\mathbf{x} \in \mathcal{D}} \left\{ f_0(\mathbf{x}) + \sum_{i=1}^{n} \lambda_i f_i(\mathbf{x}) + \sum_{i=1}^{p} \nu_i h_i(\mathbf{x}) \right\} \end{aligned}$$

可以证明，g是凹函数。函数g可以用来作为问题(5)最优值p_*的下界。我们有如下性质：

定理1.4 (下界性质). *如果$\boldsymbol{\lambda} \succeq 0$, 那么$g(\boldsymbol{\lambda}, \boldsymbol{\nu}) \leq p_*$。*

该定理可以通过下面的不等式来证明：

$$f_0(\tilde{\mathbf{x}}) \geq L(\tilde{\mathbf{x}}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \geq \inf_{\mathbf{x} \in \mathcal{D}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = g(\boldsymbol{\lambda}, \boldsymbol{\nu})$$

基于g，我们可以定义拉格朗日对偶问题（Lagrange Dual Problem）：

$$\begin{aligned} \max_{\boldsymbol{\lambda}, \boldsymbol{\nu}} \quad & g(\boldsymbol{\lambda}, \boldsymbol{\nu}) \\ \text{s.t.} \quad & \boldsymbol{\lambda} \succeq 0 \end{aligned} \tag{6}$$

该问题具备如下性质：

- 该问题最大化一个凹函数，因此是凸优化问题。
- 该问题的最优值d_*，是问题(5)最优值p_*的最优下界；

第一个性质告诉我们，无论原始问题是否为凸优化问题，对偶问题一定是凸优化问题，因此可以利用凸优化求解。第二个性质告诉我们

$$d_* \leq p_*$$

一定成立，也被称为弱对偶性（Weak Duality）。如果

$$d_* = p_*$$

我们称之为强对偶性（Strong Duality）。一般而言，我们只有弱对偶性成立，而强对偶性未必成立。判断强对偶性是否成立是一个比较困难的问题，具体的方式可以参考书籍[Boyd and Vandenberghe, 2004, 5.2.3节]。对于机器学习领域遇到的凸问题，强对偶性通常成立。

接下来，我们介绍著名的KKT （Karush-Kuhn-Tucker）条件，该条件可以用来刻画原始问题(5)和对偶问题(6)最优解之间的关系。令$\mathbf{x}_*$为原始问题(5)最优解，$(\boldsymbol{\lambda}_*, \boldsymbol{\nu}_*)$ 为对偶问题(6)最优解。当强对偶性成立时，

$$\begin{aligned} f_0(\mathbf{x}_*) = g(\boldsymbol{\lambda}_*, \boldsymbol{\nu}_*) = &\inf_{\mathbf{x}\in\mathcal{D}} \left\{ f_0(\mathbf{x}) + \sum_{i=1}^{n} \lambda_{*i} f_i(\mathbf{x}) + \sum_{i=1}^{p} \nu_{*i} h_i(\mathbf{x}) \right\} \\ \leq &f_0(\mathbf{x}_*) + \sum_{i=1}^{n} \lambda_{*i} f_i(\mathbf{x}_*) + \sum_{i=1}^{p} \nu_{*i} h_i(\mathbf{x}_*) \leq f_0(\mathbf{x}_*) \end{aligned}$$

也就意味着上式中的不等式应该是等式。因此我们得到下面两个条件一定成立：

- 互补松弛条件（Complementary slackness）：

$$\lambda_{*i} f_i(\mathbf{x}_*) = 0 \Rightarrow \begin{cases} \lambda_{*i} > 0 \Rightarrow f_i(\mathbf{x}_*) = 0 \\ f_i(\mathbf{x}_*) < 0 \Rightarrow \lambda_{*i} = 0 \end{cases}, \quad i = 1, \ldots, n$$

- $\mathbf{x}_*$是下面问题的最优解：

$$\mathbf{x}_* = \arg\min_{\mathbf{x}\in\mathcal{D}} \left\{ L(\mathbf{x}, \boldsymbol{\lambda}_*, \boldsymbol{\nu}_*) = f_0(\mathbf{x}) + \sum_{i=1}^{n} \lambda_{*i} f_i(\mathbf{x}) + \sum_{i=1}^{p} \nu_{*i} h_i(\mathbf{x}) \right\}.$$

通常而言，$\mathcal{D}$为全集或者$\mathbf{x}_*$位于$\mathcal{D}$的内部，因此拉格朗日函数$L(\mathbf{x}, \boldsymbol{\lambda}_*, \boldsymbol{\nu}_*)$在$\mathbf{x}_*$处梯度为0, 即

$$\nabla L(\mathbf{x}_*, \boldsymbol{\lambda}_*, \boldsymbol{\nu}_*) = \nabla f_0(\mathbf{x}_*) + \sum_{i=1}^{n} \lambda_{*i} \nabla f_i(\mathbf{x}_*) + \sum_{i=1}^{p} \nu_{*i} \nabla h_i(\mathbf{x}_*) = 0.$$

KKT 条件由下面几部分组成：

- 原始问题约束：

$$\begin{aligned} f_i(\mathbf{x}_*) \leq 0, \quad i = 1, \ldots, n \\ h_i(\mathbf{x}_*) = 0, \quad i = 1, \ldots, p \end{aligned}$$

- 对偶问题约束：

$$\boldsymbol{\lambda}_* \succeq 0$$

- 互补松弛条件：

$$\lambda_{*i} f_i(\mathbf{x}_*) = 0, \quad i = 1, \ldots, n$$

- $L(\mathbf{x}, \boldsymbol{\lambda}_*, \boldsymbol{\nu}_*)$在$\mathbf{x}_*$处梯度为0:

$$\nabla f_0(\mathbf{x}_*) + \sum_{i=1}^{n} \lambda_{*i} \nabla f_i(\mathbf{x}_*) + \sum_{i=1}^{p} \nu_{*i} \nabla h_i(\mathbf{x}_*) = 0$$

KKT条件具有如下重要性质：

- 对于任意优化问题，当强对偶性成立时，原始问题最优解和对偶问题最优解一定满足KKT条件。
- 对于凸优化问题，满足KKT条件的解一定是最优解，并且强对偶性成立。
- 对于凸优化问题，当强对偶性成立时，$\mathbf{x}_*$是原始问题最优解当且仅当存在$(\boldsymbol{\lambda}_*, \boldsymbol{\nu}_*)$满足KKT条件。

上述第一条性质告诉我们，当强对偶性成立时，KKT条件是最优解的必要条件。第二条性质告诉我们，对于凸优化问题，KKT条件同样是充分条件。第三条性质告诉我们，对于强对偶性成立的凸优化问题，KKT条件是充分必要条件。

1.4 优化方法

下面介绍具体的优化算法。为了便于表示，我们将优化问题写成如下形式：

$$\min_{\mathbf{w} \in \mathcal{W}} \quad f(\mathbf{w}) \tag{7}$$

其中$f(\cdot)$是凸函数，$\mathcal{W}$ 是凸集合。注意到，上一节所提到的原始问题(5)和对偶问题(6)都可以表示为上述形式。

在设计算法之前，首先需要确定如何衡量算法的性能。一般而言，优化算法是迭代算法，有两种等价的衡量准则：收敛速率（Convergence Rate）和迭代复杂度（Iteration Complexity）。假设算法迭代了T轮，$\mathbf{w}_T$为最终输出，$\mathbf{w}_*$为最优解。收敛速率旨在刻画优化误差$f(\mathbf{w}_T) - f(\mathbf{w}_*)$随迭代次数$T$的关系，常见的收敛关系有

$$f(\mathbf{w}_T) - f(\mathbf{w}_*) = O\left(\frac{1}{\sqrt{T}}\right), \ O\left(\frac{1}{T}\right), \ O\left(\frac{1}{T^2}\right), O\left(\frac{1}{\alpha^T}\right)$$

其中$\alpha > 1$。上面列举的收敛速率是越来越快，最后一种收敛速率$O(1/\alpha^T)$通常被称为线性收敛。迭代复杂度则是刻画为了达到ϵ最优解，需要的迭代次数。具体而言，迭代复杂度描述为了达到$f(\mathbf{w}_T) - f(\mathbf{w}_*) \leq \epsilon$，迭代次数$T$和$\epsilon$的关系。前面收敛速率所对应的迭代复杂度如下：

$$T = \Omega\left(\frac{1}{\epsilon^2}\right), \ \Omega\left(\frac{1}{\epsilon}\right), \ \Omega\left(\frac{1}{\sqrt{\epsilon}}\right), \Omega\left(\log \frac{1}{\epsilon}\right).$$

优化算法的收敛速率是和优化问题的性质密切相关。对于一般的凸优化问题，我们可以采用梯度下降（Gradient Descent）达到$O(1/\sqrt{T})$的收敛速率[Nesterov, 2004]。当函数强凸时，梯度下降的变种

可以达到$O(1/T)$的收敛速率[Hazan and Kale, 2011]。对于平滑函数，Nesterov的加速梯度下降可以达到$O(1/T^2)$的收敛速率[Nesterov, 2005, Tseng, 2008]。当函数强凸并且平滑，梯度下降可以达到线性收敛[Nesterov, 2007]。

下面我们介绍梯度下降，并分析其在凸函数情况下的收敛速率。梯度下降的基本流程如下：

1: 任意初始化$\mathbf{w}_1 \in \mathcal{W}$
2: **for** $t = 1, \ldots, T$ **do**
3:
$$\mathbf{w}'_{t+1} = \mathbf{w}_t - \eta_t \nabla f(\mathbf{w}_t)$$
4:
$$\mathbf{w}_{t+1} = \Pi_{\mathcal{W}}(\mathbf{w}'_{t+1})$$
5: **end for**
6: **return** $\bar{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}_t$

给第t次迭代，我们首先计算函数$f(\cdot)$在$\mathbf{w}_t$上的梯度，然后依据$\mathbf{w}'_{t+1} = \mathbf{w}_t - \eta_t \nabla f(\mathbf{w}_t)$更新当前解，其中$\eta_t > 0$为步长。这里需要注意的是，在原始问题(7)中存在$\mathbf{w} \in \mathcal{W}$的约束。但是我们通过梯度下降获得的中间解$\mathbf{w}'_{t+1}$未必属于集合$\mathcal{W}$。因此还需要通过投影操作$\mathbf{w}_{t+1} = \Pi_{\mathcal{W}}(\mathbf{w}'_{t+1})$保证下一轮的解$\mathbf{w}_{t+1} \in \mathcal{W}$。投影操作的定义如下：

$$\Pi_{\mathcal{W}}(\mathbf{y}) = \operatorname*{argmin}_{\mathbf{x} \in \mathcal{W}} \|\mathbf{x} - \mathbf{y}\|_2$$

其目的是在集合$\mathcal{W}$寻找距离输入最近的点。最后，我们将算法T次迭代的平均值最为最后的输出。

对于梯度下降，我们给出采用固定步长的理论保障。

定理1.5 (梯度下降收敛速率). 假设目标函数是G-*Lipschitz*连续，定义域$\mathcal{W}$直径为D，梯度下降采用固定步长η，即

$$\|\nabla f(\mathbf{w})\|_2 \leq G, \ \forall \mathbf{w} \in \mathcal{W}, \|\mathbf{x} - \mathbf{y}\|_2 \leq D, \ \forall \mathbf{x}, \mathbf{y} \in \mathcal{W}, \ \eta_t = \eta.$$

对于凸优化问题，我们有如下理论保障

$$f(\bar{\mathbf{w}}_T) - \min_{\mathbf{w} \in \mathcal{W}} f(\mathbf{w}) \leq \frac{D^2}{2\eta T} + \frac{\eta G^2}{2}.$$

令$\eta = \frac{D}{G\sqrt{T}}$，我们得到

$$f(\bar{\mathbf{w}}_T) - \min_{\mathbf{w} \in \mathcal{W}} f(\mathbf{w}) \leq \frac{GD}{\sqrt{T}} = O\left(\frac{1}{\sqrt{T}}\right).$$

证明. 对于任意的$\mathbf{w} \in \mathcal{W}$,

$$\begin{aligned}
&f(\mathbf{w}_t) - f(\mathbf{w}) \\
\overset{(3)}{\leq}&\langle \nabla f(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle = \frac{1}{\eta}\langle \mathbf{w}_t - \mathbf{w}'_{t+1}, \mathbf{w}_t - \mathbf{w} \rangle \\
=&\frac{1}{2\eta}\left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}'_{t+1} - \mathbf{w}\|_2^2 + \|\mathbf{w}_t - \mathbf{w}'_{t+1}\|_2^2\right) \\
=&\frac{1}{2\eta}\left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}'_{t+1} - \mathbf{w}\|_2^2\right) + \frac{\eta}{2}\|\nabla f(\mathbf{w}_t)\|_2^2 \\
\leq&\frac{1}{2\eta}\left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}_{t+1} - \mathbf{w}\|_2^2\right) + \frac{\eta}{2}\|\nabla f(\mathbf{w}_t)\|_2^2
\end{aligned}$$

其中最后一个不等号利用了投影操作的非扩展性质[Nemirovski et al., 2009, (1.5)]：

$$\|\Pi_{\mathcal{W}}(\mathbf{x}) - \Pi_{\mathcal{W}}(\mathbf{y})\|_2 \leq \|\mathbf{x} - \mathbf{y}\|_2, \forall \mathbf{x}, \mathbf{y}.$$

因为目标函数是G-Lipschitz连续，可以得到

$$f(\mathbf{w}_t) - f(\mathbf{w}) \leq \frac{1}{2\eta}\left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}_{t+1} - \mathbf{w}\|_2^2\right) + \frac{\eta}{2}G^2.$$

对上面的不等式从$t = 1$到T求和，得到

$$\begin{aligned}
\sum_{t=1}^{T} f(\mathbf{w}_t) - Tf(\mathbf{w}) \leq&\frac{1}{2\eta}\left(\|\mathbf{w}_1 - \mathbf{w}\|_2^2 - \|\mathbf{w}_{T+1} - \mathbf{w}\|_2^2\right) + \frac{\eta T}{2}G^2 \\
\leq&\frac{1}{2\eta}\|\mathbf{w}_1 - \mathbf{w}\|_2^2 + \frac{\eta T}{2}G^2 \leq \frac{1}{2\eta}D^2 + \frac{\eta T}{2}G^2.
\end{aligned}$$

最后，依据Jensen不等式，可以得到

$$\begin{aligned}
&f(\bar{\mathbf{w}}_T) - f(\mathbf{w}) = f\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{w}_t\right) - f(\mathbf{w}) \\
\leq&\frac{1}{T}\sum_{t=1}^{T} f(\mathbf{w}_t) - f(\mathbf{w}) \leq \frac{D^2}{2\eta T} + \frac{\eta G^2}{2}.
\end{aligned}$$

□

1.5 实例

本节以支持向量机为例，阐述如何推导对偶问题。支持向量机的优化问题如下

$$\min_{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}} \sum_{i=1}^{n} \max\left(0, 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2.$$

为了让推导过程更加一般化，我们将Hinge损失记为$\ell(\cdot)$，即

$$\ell(x) = \max(0, 1 - x)$$

其共轭函数为

$$\ell^*(y) = \sup_x(yx - \ell(x)) = \begin{cases} y, & -1 \le y \le 0 \\ \infty, & \text{其他} \end{cases}$$

这样，我们关注下面的一般性优化问题

$$\min_{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}} \sum_{i=1}^n \ell\left(y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2.$$

首先，引入辅助变量$u_i = y_i(\mathbf{w}^\top \mathbf{x}_i + b)$，$i = 1, \ldots, n$。将上述问题转换为(5)中的标准形式。

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}, \mathbf{u}\in\mathbb{R}^n} \quad & \sum_{i=1}^n \ell(u_i) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2 \\ \text{s.t.} \quad & u_i = y_i(\mathbf{w}^\top \mathbf{x}_i + b),\ i = 1 \ldots, n \end{aligned}$$

其中$\mathbf{u} = [u_1, \ldots, u_n]^\top$。上述问题的拉格朗日函数为

$$\mathcal{L}(\mathbf{w}, b, \mathbf{u}, \boldsymbol{\nu}) = \sum_{i=1}^n \ell(u_i) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2 + \sum_{i=1}^n \nu_i \left(u_i - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right)$$

其中$\boldsymbol{\nu} = [\nu_1, \ldots, \nu_n]^\top$, ν_i 为针对等号约束$u_i = y_i(\mathbf{w}^\top \mathbf{x}_i + b)$的拉格朗日乘子。

然后，对原始优化变量$\mathbf{w}, b, \mathbf{u}$求最小化，得到拉格朗日对偶函数

$$\begin{aligned} g(\boldsymbol{\nu}) &= \inf_{\mathbf{w},b,\mathbf{u}} \mathcal{L}(\mathbf{w}, b, \mathbf{u}, \boldsymbol{\nu}) \\ &= \inf_{\mathbf{w},b,\mathbf{u}} \sum_{i=1}^n \ell(u_i) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2 + \sum_{i=1}^n \nu_i \left(u_i - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right) \\ &= \inf_{\mathbf{w},b,\mathbf{u}} \sum_{i=1}^n (\ell(u_i) + \nu_i u_i)) + \left(\frac{\lambda}{2}\|\mathbf{w}\|_2^2 - \mathbf{w}^\top \sum_{i=1}^n \nu_i y_i \mathbf{x}_i\right) - b\sum_{i=1}^n \nu_i y_i. \end{aligned}$$

对$\mathbf{w}$求最小化，可以得到

$$\lambda\mathbf{w} - \sum_{i=1}^n \nu_i y_i \mathbf{x}_i = 0 \Rightarrow \mathbf{w} = \frac{1}{\lambda}\sum_{i=1}^n \nu_i y_i \mathbf{x}_i. \tag{8}$$

对b求最小化，可以得到

$$-\sum_{i=1}^n \nu_i y_i = 0. \tag{9}$$

对每一个u_i求最小化，可以得到

$$\begin{aligned} &\inf_{u_i} \left(\ell(u_i) + \nu_i u_i)\right) = -\sup_{u_i} \left(-\nu_i u_i - \ell(u_i)\right) \\ &= -\ell^*(-\nu_i) = \begin{cases} \nu_i, & 0 \le \nu_i \le 1 \\ -\infty, & \text{其他} \end{cases} \end{aligned} \tag{10}$$

利用(8)，(9)和(10)对$g(\boldsymbol{\nu})$进行化简，可以得到

$$g(\boldsymbol{\nu}) = \sum_{i=1}^n \nu_i - \frac{1}{2\lambda}\sum_{i=1}^n \sum_{j=1}^n \nu_i \nu_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j.$$

最后，我们就得到了下面的拉格朗日对偶问题：

$$\begin{aligned}\max_{\boldsymbol{\nu}\in\mathbb{R}^n} \quad & \sum_{i=1}^{n}\nu_i - \frac{1}{2\lambda}\sum_{i=1}^{n}\sum_{j=1}^{n}\nu_i\nu_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j \\ \text{s.t.} \quad & 0 \leq \nu_i \leq 1,\ i = 1\ldots, n \\ & \sum_{i=1}^{n}\nu_i y_i = 0\end{aligned}$$

此外，在对偶问题的推导过程中，也隐含了下面的KKT条件

$$\begin{aligned}& u_{*i} = y_i(\mathbf{w}_*^\top \mathbf{x}_i + b_*) \\ & 0 \leq \nu_{*i} \leq 1 \\ & \sum_{i=1}^{n}\nu_{*i} y_i = 0 \\ & \mathbf{w}_* = \frac{1}{\lambda}\sum_{i=1}^{n}\nu_{*i} y_i \mathbf{x}_i \\ & u_{*i} = \underset{u_i}{\operatorname{argmin}}\left(\ell(u_i) + \nu_{*i}u_i\right) = 1,\ 0 < \nu_{*i} < 1\end{aligned}$$

其中$(\mathbf{w}_*, b_*, \mathbf{u}_*)$是原始问题最优解，$\boldsymbol{\nu}_*$为对偶问题最优解。KKT条件可以帮助我们从对偶问题最优解得到原始问题最优解。

2 随机优化

2.1 引言

在本节，我们介绍如何求解随机优化（Stochastic Optimization）问题。随机优化可以定义为[Nemirovski et al., 2009]

$$\min_{\mathbf{w}\in\mathcal{W}} \ F(\mathbf{w}) = \mathrm{E}_\xi\left[f(\mathbf{w}, \xi)\right] = \int_\Xi f(\mathbf{w}, \xi) dP(\xi) \tag{11}$$

其中$\xi \in \Xi$为某随机变量。随机优化的困难之处在于上式中的随机变量ξ的分布通常是未知的，因此我们并不知道目标函数的具体形式。即使我们知道随机变量ξ的分布，但是由于高维积分非常困难，因此也难以准确计算目标函数。注意到，机器学习面临的风险最小化问题，即公式(1)，就是一种特殊的随机优化问题。因此，随机优化技术可以用来求解风险最小化，也反映了优化和机器学习之间的密切关系。

求解随机优化有两种方法：样本平均近似（Sample Average Approximation，SAA）和随机近似（Stochastic Approximation，SA）。接下来，我们分别对这两种方法进行介绍。

2.2 样本平均近似

样本平均近似就是机器学习领域的经验风险最小化（Empirical Risk Minimization）。其核心思想是对随机变量ξ进行独立采样，获得n个样本$\xi_1,\dots,\xi_n$，然后利用定义在n个样本上的平均函数$\widehat{F}(\mathbf{w})=\frac{1}{n}\sum_{i=1}^{n}f(\mathbf{w},\xi_i)$来近似未知的目标函数$F(\cdot)$。样本平均近似旨在求解下面的优化问题

$$\min_{\mathbf{w}\in\mathcal{W}}\ \widehat{F}(\mathbf{w})=\frac{1}{n}\sum_{i=1}^{n}f(\mathbf{w},\xi_i) \tag{12}$$

上述优化问题可以利用我们在第1章提到的优化算法，比如梯度下降求解。

令$\widehat{\mathbf{w}}\in\operatorname{argmin}_{\mathbf{w}\in\mathcal{W}}\widehat{F}(\mathbf{w})$为优化问题(12)的最优解。虽然$\widehat{\mathbf{w}}$可以最小化经验函数$\widehat{F}(\cdot)$，但我们的目标是最小化问题(11)中的期望函数$F(\cdot)$。因此，我们需要刻画$\widehat{\mathbf{w}}$对优化问题(11)的误差，也就是

$$F(\widehat{\mathbf{w}})-\min_{\mathbf{w}\in\mathcal{W}}F(\mathbf{w})$$

的大小。上述优化误差也被成为额外风险（Excess Risk）。分析$\widehat{\mathbf{w}}$的额外风险跟之前讲过的分析泛化误差有类似之处，下面我们阐述分析额外风险面临的困难。

令$\mathbf{w}_*\in\operatorname{argmin}_{\mathbf{w}\in\mathcal{W}}F(\mathbf{w})$为问题(11)的最优解。根据$\widehat{\mathbf{w}}$的定义，我们知道

$$\widehat{F}(\widehat{\mathbf{w}})=\frac{1}{n}\sum_{i=1}^{n}f(\widehat{\mathbf{w}},\xi_i)\leq\widehat{F}(\mathbf{w}_*)=\frac{1}{n}\sum_{i=1}^{n}f(\mathbf{w}_*,\xi_i). \tag{13}$$

这时候一个很自然的想法是对上面不等式左右两边同时求期望

$$\mathrm{E}\left[\widehat{F}(\widehat{\mathbf{w}})\right]=\mathrm{E}\left[\frac{1}{n}\sum_{i=1}^{n}f(\widehat{\mathbf{w}},\xi_i)\right]\leq\mathrm{E}\left[\widehat{F}(\mathbf{w}_*)\right]=\mathrm{E}\left[\frac{1}{n}\sum_{i=1}^{n}f(\mathbf{w}_*,\xi_i)\right]=F(\mathbf{w}_*)$$

由于$\mathbf{w}_*$和样本$\xi_1,\dots,\xi_n$无关，不等式右边求期望可以得到$F(\mathbf{w}_*)$。但是由于$\widehat{\mathbf{w}}$依赖于样本$\xi_1,\dots,\xi_n$，因此不等式左边的期望$\mathrm{E}[\widehat{F}(\widehat{\mathbf{w}})]$无法进一步化简，也就不能得到$F(\widehat{\mathbf{w}})$。

分析额外风险一般通过下面的思路求解：

$$\begin{aligned}
&F(\widehat{\mathbf{w}})-\min_{\mathbf{w}\in\mathcal{W}}F(\mathbf{w})=F(\widehat{\mathbf{w}})-F(\mathbf{w}_*)\\
\overset{(13)}{\leq}&F(\widehat{\mathbf{w}})-F(\mathbf{w}_*)+\widehat{F}(\mathbf{w}_*)-\widehat{F}(\widehat{\mathbf{w}})=F(\widehat{\mathbf{w}})-\widehat{F}(\widehat{\mathbf{w}})+\widehat{F}(\mathbf{w}_*)-F(\mathbf{w}_*)\\
\leq&\sup_{\mathbf{w}\in\mathcal{W}}\left[F(\mathbf{w})-\widehat{F}(\mathbf{w})\right]+\widehat{F}(\mathbf{w}_*)-F(\mathbf{w}_*)\leq 2\sup_{\mathbf{w}\in\mathcal{W}}\left|F(\mathbf{w})-\widehat{F}(\mathbf{w})\right|
\end{aligned}$$

这样，我们的问题就变成了如何分析随机过程$\sup_{\mathbf{w}\in\mathcal{W}}|F(\mathbf{w})-\widehat{F}(\mathbf{w})|$的上界。

对于风险最小化问题，在前几章的介绍中，大家已经见到了分析该上界的一般流程，涉及到集中不等式、VC维、Rademacher复杂度等技术。风险最小化问题的额外风险界存在丰富的研究结果[Vapnik, 1998, Koltchinskii, 2011]：当损失函数是Lipschitz连续时，我们可以得到$O(1/\sqrt{n})$的风险界[Bartlett and Mendelson, 2002] ；当损失函数是非负并且平滑时，可以得到$O(1/n+\sqrt{F_*/n})$的风险界，其中$F_*=F(\mathbf{w}_*)$为最小风险[Srebro et al., 2010]；当损失函数是强凸时，可以得到$O(1/[\lambda n])$的风险界，其中λ是强凸的参数[Sridharan et al., 2009]；当损失函数强凸并且平滑时，可以得到$O(1/[\lambda n^2]+LF_*/[\lambda n])$的风险界，其中$L$是平滑的参数[Zhang et al., 2017a]。

对于一般的随机优化问题而言，其额外风险界和风险最小化的额外风险界有所不同。当目标函数是Lipschitz连续时，我们可以得到$O(\sqrt{d/n})$的风险界，其中d是优化变量的维度[Shalev-Shwartz et al., 2009]；当目标函数是非负并且平滑时，可以得到$O(d/n + \sqrt{F_*/n})$的风险界[Zhang et al., 2017a]；当目标函数是强凸时，可以得到$O(1/[\lambda n])$的风险界[Shalev-Shwartz et al., 2009]；当目标函数强凸并且平滑时，可以得到$O(1/[\lambda n^2] + LF_*/[\lambda n])$的风险界，其中$L$是平滑的参数[Zhang et al., 2017a]。

2.3 随机近似

与样本平均近似不同，随机近似利用目标函数$F(\cdot)$的有噪声观测直接优化问题(11)。如果我们可以观测到目标函数值，就称为零阶随机优化[Nesterov, 2011]；如果可以观测到目标函数的梯度，就称为一阶随机优化[Kushner and Yin, 2003]。在本节，我们主要关注一阶随机优化。

首先，我们介绍随机近似的理论保证，以方便大家和样本平均近似相比较。我们把随机近似算法需要访问目标函数的次数记为T，然后刻画额外风险随T的变化。需要指出，对于随机近似，一般情况下并不需要区分一般的随机优化问题(11) 或是经验风险最小化问题(1)。换言之，随机近似算法对(11)和(1)是通用的，并且理论结果一致。当目标函数是Lipschitz连续时，我们可以得到$O(\sqrt{1/T})$的风险界[Zinkevich, 2003]；当目标函数是非负并且平滑时，可以得到$O(1/T + \sqrt{F_*/T})$的风险界[Srebro et al., 2010]；当目标函数是强凸时，可以得到$O(1/[\lambda T])$的风险界[Hazan and Kale, 2011]。当目标函数强凸并且平滑时，可以得到$O(1/[\lambda T^\alpha] + \kappa F_*/T)$的风险界，其中$\alpha > 1$为满足$T \geq \kappa^\alpha$的常数，$\kappa$是条件数[Zhang and Zhou, 2019]。可以看出，随机近似的理论保证和样本平均近似是类似的，但是一般情况下随机近似更加高效。

2.3.1 随机梯度下降（SGD）

接下来，我们介绍最著名的随机优化算法—随机梯度下降（Stochastic Gradient Descent，SGD）。顾名思义，SGD和前面介绍过的梯度下降非常类似，唯一的区别在于使用随机梯度代替真实梯度。SGD的一般流程如下：

1: 任意初始化$\mathbf{w}_1 \in \mathcal{W}$
2: **for** $t = 1, \ldots, T$ **do**
3:
$$\mathbf{w}'_{t+1} = \mathbf{w}_t - \eta_t \mathbf{g}_t$$
4:
$$\mathbf{w}_{t+1} = \Pi_{\mathcal{W}}(\mathbf{w}'_{t+1})$$
5: **end for**
6: **return** $\bar{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}_t$

其中我们要求$\mathbf{w}_t$的随机梯度$\mathbf{g}_t$是真实梯度$\nabla F(\mathbf{w}_t)$的无偏估计，即

$$\mathrm{E}_t[\mathbf{g}_t] = \nabla F(\mathbf{w}_t).$$

对于随机优化问题(11)而言，可以通过下面的方式实现SGD：

1: 任意初始化$\mathbf{w}_1 \in \mathcal{W}$
2: **for** $t = 1, \ldots, T$ **do**
3: 从分布中采样ξ_t
4:

$$\mathbf{w}'_{t+1} = \mathbf{w}_t - \eta_t \nabla f(\mathbf{w}_t, \xi_t)$$

5:

$$\mathbf{w}_{t+1} = \Pi_{\mathcal{W}}(\mathbf{w}'_{t+1})$$

6: **end for**
7: **return** $\bar{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}_t$

注意到，我们同样可以把SGD用于求解样本平均近似面临的优化问题(12)。这时候，我们只需要从已有的n个样本中采样计算随机梯度。从上面的描述可以看出，SGD每一次迭代只需要利用1个样本，因此SGD每一次迭代的计算复杂度非常低，特别适用于大规模机器学习。

对于一般的Lipschitz凸函数，SGD可以达到$O(1/\sqrt{T})$的额外风险界。该收敛速率从期望意义上成立，并且也以大概率成立。我们有如下定理：

定理2.1 (随机梯度下降收敛速率). 假设随机梯度上界为G，定义域$\mathcal{W}$直径为D，梯度下降采用固定步长η，即

$$\|\mathbf{g}_t\|_2 \leq G, \ \forall t \in [t], \quad \|\mathbf{x} - \mathbf{y}\|_2 \leq D, \ \forall \mathbf{x}, \mathbf{y} \in \mathcal{W}, \quad \eta_t = \eta = \frac{D}{G\sqrt{T}}.$$

对于随机凸优化，我们有

$$\mathrm{E}\left[F(\bar{\mathbf{w}}_T)\right] - F(\mathbf{w}_*) \leq \frac{D^2}{2\eta T} + \frac{\eta G^2}{2} = \frac{GD}{\sqrt{T}} = O\left(\frac{1}{\sqrt{T}}\right).$$

其中$\mathbf{w}_* \in \mathrm{argmin}_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w})$为最优解。此外，以至少$1-\delta$的概率，我们有

$$F(\bar{\mathbf{w}}_T) - F(\mathbf{w}_*) \leq \frac{GD}{\sqrt{T}}\left(1 + 2\sqrt{2\log\frac{1}{\delta}}\right) = O\left(\frac{1}{\sqrt{T}}\right).$$

证明. 对于任意的$\mathbf{w} \in \mathcal{W}$,

$$
\begin{aligned}
& F(\mathbf{w}_t) - F(\mathbf{w}) \\
\overset{(3)}{\leq} & \langle \nabla F(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle = \langle \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle + \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle \\
= & \frac{1}{\eta} \langle \mathbf{w}_t - \mathbf{w}'_{t+1}, \mathbf{w}_t - \mathbf{w} \rangle + \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle \\
= & \frac{1}{2\eta} \left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}'_{t+1} - \mathbf{w}\|_2^2 + \|\mathbf{w}_t - \mathbf{w}'_{t+1}\|_2^2 \right) + \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle \\
= & \frac{1}{2\eta} \left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}'_{t+1} - \mathbf{w}\|_2^2 \right) + \frac{\eta}{2} \|\mathbf{g}_t\|_2^2 + \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle \\
\leq & \frac{1}{2\eta} \left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}_{t+1} - \mathbf{w}\|_2^2 \right) + \frac{\eta}{2} \|\mathbf{g}_t\|_2^2 + \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle \\
\leq & \frac{1}{2\eta} \left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}_{t+1} - \mathbf{w}\|_2^2 \right) + \frac{\eta}{2} G^2 + \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle.
\end{aligned}
$$

对上面的不等式从$t = 1$到T求和，得到

$$
\begin{aligned}
& \sum_{t=1}^{T} F(\mathbf{w}_t) - T F(\mathbf{w}) \\
\leq & \frac{1}{2\eta} \left(\|\mathbf{w}_1 - \mathbf{w}\|_2^2 - \|\mathbf{w}_{T+1} - \mathbf{w}\|_2^2 \right) + \frac{\eta T}{2} G^2 + \sum_{t=1}^{T} \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle \\
\leq & \frac{1}{2\eta} \|\mathbf{w}_1 - \mathbf{w}\|_2^2 + \frac{\eta T}{2} G^2 + \sum_{t=1}^{T} \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle \\
\leq & \frac{1}{2\eta} D^2 + \frac{\eta T}{2} G^2 + \sum_{t=1}^{T} \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle.
\end{aligned}
$$

最后，依据Jensen不等式，可以得到

$$
\begin{aligned}
& F(\bar{\mathbf{w}}_T) - F(\mathbf{w}) = F\left(\frac{1}{T} \sum_{t=1}^{T} \mathbf{w}_t \right) - F(\mathbf{w}) \\
\leq & \frac{1}{T} \sum_{t=1}^{T} F(\mathbf{w}_t) - F(\mathbf{w}) \\
\leq & \frac{D^2}{2\eta T} + \frac{\eta G^2}{2} + \frac{1}{T} \sum_{t=1}^{T} \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle.
\end{aligned} \tag{14}
$$

可以看出，与梯度下降分析的唯一区别在于多了一项$\frac{1}{T} \sum_{t=1}^{T} \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle$。

下面，我们首先证明SGD期望意义上的收敛速率。注意到我们要求$\mathrm{E}[\mathbf{g}_t] = \nabla F(\mathbf{w}_t)$，因此

$$
\mathrm{E}\left[\sum_{t=1}^{T} \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w} \rangle \right] = 0.
$$

对公式(14)求期望可以得到

$$
\mathrm{E}\left[F(\bar{\mathbf{w}}_T) \right] - F(\mathbf{w}) \leq \frac{D^2}{2\eta T} + \frac{\eta G^2}{2} = \frac{GD}{\sqrt{T}}
$$

其中我们令$\eta = D/[G\sqrt{T}]$。

在前面的分析中，我们证明了从期望意义上，$\bar{\mathbf{w}}_T$的额外风险在$O(1/\sqrt{T})$量级。由于在实际应用中，一般只能运行SGD算法1次，因此我们想要知道单次运行SGD所能达到的效果，这也就需要提供大概率的理论保障。

为了分析SGD的理论保障，我们需要引入针对鞅的集中不等式[Cesa-Bianchi and Lugosi, 2006]。

定理2.2 (针对鞅的Azuma不等式). 假设$X_1, X_2, \ldots$是鞅差序列（*Martingale Difference Sequence*），并且

$$|X_i| \leq c_i.$$

那么，我们有

$$\Pr\left(\sum_{i=1}^{n} X_i \geq t\right) \leq \exp\left(-\frac{t^2}{2\sum_{i=1}^{n} c_i^2}\right),$$
$$\Pr\left(\sum_{i=1}^{n} X_i \leq -t\right) \leq \exp\left(-\frac{t^2}{2\sum_{i=1}^{n} c_i^2}\right).$$

由于存在$\mathrm{E}[\mathbf{g}_t] = \nabla F(\mathbf{w}_t)$的要求，$\langle \nabla F(\mathbf{w}_1) - \mathbf{g}_1, \mathbf{w}_1 - \mathbf{w}\rangle, \ldots$就组成了一个鞅差序列，因此可以利用定理2.2求鞅差之和的上界。根据假设，可以得到

$$|\langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w}\rangle| \leq \|\nabla F(\mathbf{w}_t) - \mathbf{g}_t\|_2 \|\mathbf{w}_t - \mathbf{w}\|_2 \leq D(\|\nabla F(\mathbf{w}_t)\|_2 + \|\mathbf{g}_t\|_2) \leq 2GD$$

其中利用了Jensen不等式来获得$\|\nabla F(\mathbf{w}_t)\|_2$的上界

$$\|\nabla F(\mathbf{w}_t)\|_2 = \|\mathrm{E}[\mathbf{g}_t]\|_2 \leq \mathrm{E}\|\mathbf{g}_t\|_2 \leq G.$$

根据定理2.2，可以得到以至少$1-\delta$的概率

$$\sum_{t=1}^{T} \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w}\rangle \leq 2GD\sqrt{2T\log\frac{1}{\delta}}.$$

将上式代入(14)，得到以至少$1-\delta$的概率

$$F(\bar{\mathbf{w}}_T) - F(\mathbf{w}) \leq \frac{D^2}{2\eta T} + \frac{\eta G^2}{2} + 2GD\sqrt{\frac{2}{T}\log\frac{1}{\delta}} = \frac{GD}{\sqrt{T}}\left(1 + 2\sqrt{2\log\frac{1}{\delta}}\right).$$

□

2.3.2 阶段随机梯度下降（Epoch-GD）

在本节，我们考虑目标函数$F(\cdot)$是强凸的，也就是

$$F(\mathbf{w}) + \langle \nabla F(\mathbf{w}), \mathbf{w}' - \mathbf{w}\rangle + \frac{\lambda}{2}\|\mathbf{w}' - \mathbf{w}\|_2^2 \leq F(\mathbf{w}'),\ \forall \mathbf{w}, \mathbf{w}' \in \mathcal{W}. \tag{15}$$

在这种情况，可以将SGD的步长设置为$\eta_t = 1/[\lambda t]$。通过理论分析，可以证明这种情况下SGD的额外风险界为$O(\log T/T)$ [Hazan et al., 2007, Kakade and Tewari, 2009]。与之前的$O(1/\sqrt{T})$速率相

比，$O(\log T/T)$ 已经是巨大的提升，但这并不是最优收敛速率。最近的研究表明，SGD的一些变种，比如阶段梯度下降（Epoch Gradient Descent，Epoch-GD）[Hazan and Kale, 2011]、后平均的SGD [Rakhlin et al., 2012]，可以达到最优的$O(1/T)$收敛速率。

下面，我们介绍阶段随机梯度下降（Epoch-GD）。Epoch-GD的基本流程如下：

1: 任意初始化$\mathbf{w}_1^1 \in \mathcal{W}$，设定$k=1$
2: **while** $\sum_{i=1}^{k} T_i \leq T$ **do**
3: **for** $t=1$ to T_k **do**
4: 得到$\mathbf{w}_t^k$的随机梯度$\mathbf{g}_t^k$
5:

$$\mathbf{w}_{t+1}^k = \Pi_{\mathcal{W}}\left(\mathbf{w}_t^k - \eta_k \mathbf{g}_t^k\right)$$

6: **end for**
7: $\mathbf{w}_1^{k+1} = \frac{1}{T_k}\sum_{t=1}^{T_k} \mathbf{w}_t^k$
8: $T_{k+1} = 2T_k$，并且$\eta_{k+1} = \eta_k/2$
9: $k = k+1$
10: **end while**
11: 返回$\mathbf{w}_1^k$

从上面的算法可以看出，Epoch-GD的是一个两层循环算法，其中内层就是采用固定步长的SGD。在第7步，算法把当前轮SGD的最终解传递给下一轮，作为下一轮的初始值。在第8步，算法将下一轮SGD的迭次数加倍，并将步长减少一半。整体而言，算法通过While循环中的$\sum_{i=1}^{k} T_i \leq T$判断语句控制所需要的随机梯度个数，保证算法最多计算T次随机梯度。

可以证明，在期望意义上Epoch-GD的额外风险界为$O(1/\lambda T)$。这一部分的证明比较简单，因此我们直接引用Hazan and Kale [2014, 定理5]中的结论。

定理2.3 (Epoch-GD的收敛速率). *假设随机梯度上界为G，目标函数$F(\cdot)$为λ强凸。将Epoch-GD的初始参数设置为$T_1=4$并且$\eta_1 = 1/\lambda$，Epoch-GD的最终输出满足*

$$\mathrm{E}\left[F(\mathbf{w}_1^k)\right] - F(\mathbf{w}_*) \leq \frac{16G^2}{\lambda T} = O\left(\frac{1}{\lambda T}\right)$$

其中$\mathbf{w}_* \in \mathrm{argmin}_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w})$为最优解。

证明Epoch-GD以大概率成立的额外风险界相对困难。Hazan and Kale [2011]提出对Epoch-GD算法进行修改，通过添加额外的约束来证明大概率的风险界。但是，这种做法会导致优化算法变得更加复杂。下面，我们通过利用更加高级的分析技术，来说明不需要修改Epoch-GD就可以得到大概率的风险界。

定理2.4. 假设随机梯度上界为G，目标函数$F(\cdot)$为λ强凸。定义

$$\tilde{\delta} = \frac{\delta}{k^\dagger} \tag{16}$$

$$k^\dagger = \left\lfloor \log_2 \left(\frac{2T}{\alpha} + 1 \right) \right\rfloor \tag{17}$$

$$\alpha = 24 + \frac{128}{3} \log \frac{\lceil 2 \log_2 T \rceil}{\tilde{\delta}} \tag{18}$$

将*Epoch-GD*的初始参数设置为$T_1 = \alpha/2$ 并且$\eta_1 = 1/\lambda$. 以至少$1-\delta$的概率，*Epoch-GD*的最终输出满足

$$F(\mathbf{w}_1^k) - F(\mathbf{w}_*) \leq \frac{2\alpha G^2}{\lambda T} = O\left(\frac{\log \log T}{\lambda T} \right)$$

其中$\mathbf{w}_* \in \operatorname{argmin}_{\mathbf{w} \in \mathcal{W}} F(\mathbf{w})$为最优解。

证明. 根据函数强凸的性质，可以得到下面的不等式[Hazan and Kale, 2011, (2)]

$$\frac{\lambda}{2} \|\mathbf{w} - \mathbf{w}_*\|^2 \leq F(\mathbf{w}) - F(\mathbf{w}_*), \ \forall \mathbf{w} \in \mathcal{W}. \tag{19}$$

此外，由于随机梯度的上界为G，可以证明[Hazan and Kale, 2011]

$$\|\mathbf{w} - \mathbf{w}_*\| \leq \frac{2G}{\lambda}, \ \forall \mathbf{w} \in \mathcal{W} \tag{20}$$

$$F(\mathbf{w}) - F(\mathbf{w}_*) \leq \frac{2G^2}{\lambda}, \ \forall \mathbf{w} \in \mathcal{W}. \tag{21}$$

首先，我们分析SGD在强凸函数下的收敛性质，并得到如下引理。

引理2.1. 假设随机梯度上界为G，目标函数$F(\cdot)$为λ强凸。运行T次*SGD*更新

$$\mathbf{w}_{t+1} = \Pi_{\mathcal{W}} \left(\mathbf{w}_t - \eta \mathbf{g}_t \right)$$

其中$\mathbf{g}_t$是函数$F(\cdot)$在$\mathbf{w}_t$处的随机梯度。那么以至少$1-\delta$的概率，

$$\sum_{t=1}^{T} F\left(\mathbf{w}_t\right) - T F(\mathbf{w}_*) \leq \frac{\eta T G^2}{2} + \frac{\|\mathbf{w}_1 - \mathbf{w}_*\|^2}{2\eta} + \frac{4G^2}{\lambda} \left(1 + \frac{8}{3} \log \frac{m}{\delta} \right). \tag{22}$$

其中$m = \lceil 2 \log_2 T \rceil$。

在引理2.1的基础上，我们可以分析Epoch-GD外层循环的性质，并得到如下引理。

引理2.2. 将*Epoch-GD*的参数设置为$T_1 = \alpha/2$和$\eta_1 = 1/\lambda$，其中α定义在*(18)*. 对于任意的k，以至少$(1-\tilde{\delta})^{k-1}$的概率

$$\Delta_k = F(\mathbf{w}_1^k) - F(\mathbf{w}_*) \leq V_k = \frac{G^2}{\lambda 2^{k-2}}.$$

Epoch-GD外层循环的次数，是由满足$\sum_{i=1}^{k} T_i \leq T$的最大$k$决定的。由于

$$\sum_{i=1}^{k} T_i = \sum_{i=1}^{k} \alpha 2^{i-2} = \frac{\alpha}{2}(2^k - 1).$$

因此，最后一次迭代的次数$k^\dagger$与公式(17)吻合，而算法的最后输出是$\mathbf{w}_1^{k^\dagger+1}$. 根据引理2.2，以至少$(1-\tilde{\delta})^{k^\dagger}$的概率

$$F(\mathbf{w}_1^{k^\dagger+1}) - F(\mathbf{w}_*) = \Delta_{k^\dagger+1} \leq V_{k^\dagger+1} = \frac{G^2}{2^{k^\dagger-1}\lambda} = \frac{2G^2}{2^{k^\dagger}\lambda} \leq \frac{2\alpha G^2}{\lambda T}$$

其中我们利用了下面的不等式

$$2^{k^\dagger} \geq \frac{1}{2}\left(\frac{2T}{\alpha}+1\right) \geq \frac{T}{\alpha}.$$

最后，我们证明概率$(1-\tilde{\delta})^{k^\dagger}$ 大于$1-\delta$。由于函数$(1-\frac{1}{x})^x$ 在$x>1$时是增函数，因此

$$(1-\tilde{\delta})^{k^\dagger} = \left(1-\frac{\delta}{k^\dagger}\right)^{k^\dagger} = \left(\left(1-\frac{1}{k^\dagger/\delta}\right)^{k^\dagger/\delta}\right)^{\delta} \geq \left(\left(1-\frac{1}{1/\delta}\right)^{1/\delta}\right)^{\delta} = 1-\delta.$$

□

2.3.3 引理2.1的证明

由于$F(\cdot)$是强凸的，因此

$$\begin{aligned}
&F(\mathbf{w}_t) - F(\mathbf{w}_*) \\
\overset{(15)}{\leq}& \langle \nabla F(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w}_* \rangle - \frac{\lambda}{2}\|\mathbf{w}_t - \mathbf{w}_*\|^2 \\
=& \langle \mathbf{g}_t, \mathbf{w}_t - \mathbf{w}_* \rangle + \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w}_* \rangle - \frac{\lambda}{2}\|\mathbf{w}_t - \mathbf{w}_*\|^2.
\end{aligned}$$

仿照公式(14)的推导过程，可以得到

$$\sum_{t=1}^{T} F(\mathbf{w}_t) - TF(\mathbf{w}_*) \leq \frac{\eta T G^2}{2} + \frac{\|\mathbf{w}_1 - \mathbf{w}_*\|^2}{2\eta} + \sum_{t=1}^{T}\langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w}_* \rangle - \frac{\lambda}{2}\sum_{t=1}^{T}\|\mathbf{w}_t - \mathbf{w}_*\|^2. \quad (23)$$

定义鞅差序列

$$\delta_t = \langle \nabla F(\mathbf{w}_t) - \mathbf{g}_t, \mathbf{w}_t - \mathbf{w}_* \rangle$$

为了得到$\sum \delta_t$的上界，我们将利用剥离技术（Peeling）[Bartlett et al., 2005]和针对鞅的Bernstein不等式[Cesa-Bianchi and Lugosi, 2006]。

定理2.5 (针对鞅的Bernstein不等式)**.** 假设$X_1, \ldots, X_n$是定义在滤波$\mathcal{F} = (\mathcal{F}_i)_{1\leq i\leq n}$上的有界鞅差序列并且$|X_i| \leq K$。令

$$S_i = \sum_{j=1}^{i} X_j$$

为对应的鞅。将条件方差（*Conditional Variances*）记为

$$\Sigma_n^2 = \sum_{t=1}^{n} \mathrm{E}\left[\delta_t^2 | \mathcal{F}_{t-1}\right].$$

那么对于所有的正数t和ν，

$$\Pr\left(\max_{i=1,\ldots,n} S_i > t \ and \ \Sigma_n^2 \leq \nu\right) \leq \exp\left(-\frac{t^2}{2(\nu + Kt/3)}\right).$$

因此，

$$\Pr\left(\max_i S_i > \sqrt{2\nu\tau} + \frac{2}{3}K\tau \ and \ \Sigma_n^2 \leq \nu\right) \leq e^{-\tau}.$$

首先，注意到我们定义的鞅差序列是有界的

$$|\delta_t| \leq \|\nabla F(\mathbf{w}_t) - \mathbf{g}_t\|\|\mathbf{w}_t - \mathbf{w}_*\| \overset{(20)}{\leq} 2G\frac{2G}{\lambda} = \frac{4G^2}{\lambda}.$$

为了方便起见，我们定义

$$A_T = \sum_{t=1}^{T} \|\mathbf{w}_t - \mathbf{w}_*\|^2 \overset{(20)}{\leq} \frac{4G^2T}{\lambda^2}.$$

对于条件方差，下面的不等式成立：

$$\Sigma_T^2 = \sum_{t=1}^{T} \mathrm{E}_{t-1}\left[\delta_t^2\right] \leq 4G^2 \sum_{t=1}^{T} \|\mathbf{w}_t - \mathbf{w}_*\|^2 = 4G^2 A_T.$$

根据剥离技术，我们首先考虑

$$A_T \leq \frac{4G^2}{\lambda^2 T}$$

的情况。在这种情况下：

$$\sum_{t=1}^{T} \delta_t \leq 2G \sum_{t=1}^{T} \|\mathbf{w}_t - \mathbf{w}_*\|_2 \leq 2G\sqrt{T}\sqrt{\sum_{t=1}^{T} \|\mathbf{w}_t - \mathbf{w}_*\|_2^2} \leq \frac{4G^2}{\lambda}.$$

接下来，我们将另外一种情况

$$A_T \in \left(\frac{4G^2}{\lambda^2 T}, \frac{4G^2 T}{\lambda^2}\right]$$

分解成$m = \lceil 2\log_2 T \rceil$种子情况，即

$$A_t \in \left(2^{i-1}\frac{4G^2}{\lambda^2 T}, 2^i \frac{4G^2}{\lambda^2 T}\right], \ i = 1, \ldots, \lceil 2\log_2 T \rceil.$$

最后，通过下面的一系列变换，可以证明

$$\begin{aligned}
&\Pr\left[\sum_{t=1}^{T}\delta_t \geq 2\sqrt{4G^2A_T\tau}+\frac{2}{3}\frac{4G^2}{\lambda}\tau+\frac{4G^2}{\lambda}\right]\\
=&\Pr\left[\sum_{t=1}^{T}\delta_t \geq 2\sqrt{4G^2A_T\tau}+\frac{2}{3}\frac{4G^2}{\lambda}\tau+\frac{4G^2}{\lambda}, A_T \leq \frac{4G^2}{\lambda^2 T}\right]\\
&+\Pr\left[\sum_{t=1}^{T}\delta_t \geq 2\sqrt{4G^2A_T\tau}+\frac{2}{3}\frac{4G^2}{\lambda}\tau+\frac{4G^2}{\lambda}, \frac{4G^2}{\lambda^2 T} < A_T \leq \frac{4G^2T}{\lambda^2}\right]\\
=&\Pr\left[\sum_{t=1}^{T}\delta_t \geq 2\sqrt{4G^2A_T\tau}+\frac{2}{3}\frac{4G^2}{\lambda}\tau+\frac{4G^2}{\lambda}, \Sigma_T^2 \leq 4G^2A_T, \frac{4G^2}{\lambda^2 T} < A_T \leq \frac{4G^2T}{\lambda^2}\right]\\
\leq&\sum_{i=1}^{m}\Pr\left[\sum_{t=1}^{T}\delta_t \geq 2\sqrt{4G^2A_T\tau}+\frac{2}{3}\frac{4G^2}{\lambda}\tau+\frac{4G^2}{\lambda}, \Sigma_T^2 \leq 4G^2A_T, \frac{4G^2}{\lambda^2 T}2^{i-1} < A_T \leq \frac{4G^2}{\lambda^2 T}2^{i}\right]\\
\leq&\sum_{i=1}^{m}\Pr\left[\sum_{t=1}^{T}\delta_t \geq \sqrt{2\frac{16G^4 2^i}{\lambda^2 T}\tau}+\frac{2}{3}\frac{4G^2}{\lambda}\tau, \Sigma_T^2 \leq \frac{16G^4 2^i}{\lambda^2 T}\right] \leq me^{-\tau}
\end{aligned}$$

其中最后一个不等式利用了定理2.5。然后令

$$\tau = \log\frac{m}{\delta} = \log\frac{\lceil 2\log_2 T\rceil}{\delta}.$$

可以得到以至少$1-\delta$的概率

$$\sum_{t=1}^{T}\delta_t \leq 2\sqrt{4G^2A_T\log\frac{m}{\delta}}+\frac{8G^2}{3\lambda}\log\frac{m}{\delta}+\frac{4G^2}{\lambda}. \tag{24}$$

将不等式(24)代入(23)，可以得到以至少$1-\delta$的概率

$$\begin{aligned}
\sum_{t=1}^{T}F(\mathbf{w}_t)-TF(\mathbf{w}_*) \leq& \frac{\eta TG^2}{2}+\frac{\|\mathbf{w}_1-\mathbf{w}_*\|^2}{2\eta}+2\sqrt{4G^2A_T\log\frac{m}{\delta}}+\frac{8G^2}{3\lambda}\log\frac{m}{\delta}+\frac{4G^2}{\lambda}-\frac{\lambda}{2}A_T\\
\leq& \frac{\eta TG^2}{2}+\frac{\|\mathbf{w}_1-\mathbf{w}_*\|^2}{2\eta}+\frac{32G^2}{3\lambda}\log\frac{m}{\delta}+\frac{4G^2}{\lambda}.
\end{aligned}$$

2.3.4 引理2.2的证明

首先，很容易证明下面式子成立：

$$T_k = \frac{\alpha G^2}{\lambda V_k} = \alpha 2^{k-2} \tag{25}$$

$$\eta_k = \frac{V_k}{2G^2} = \frac{1}{\lambda 2^{k-1}} \tag{26}$$

接下来，我们利用数学归纳法进行证明。当$k=1$时，

$$\Delta_1 = F(\mathbf{w}_1^1)-F(\mathbf{w}_*) \overset{(21)}{\leq} \frac{2G^2}{\lambda} = \frac{G^2}{\lambda 2^{1-2}} = V_1.$$

假设对某$k \geq 1$，$\Delta_k \leq V_k$ 以至少$(1-\tilde{\delta})^{k-1}$的概率成立。结合引理2.1, 以至少$(1-\tilde{\delta}) \cdot (1-\tilde{\delta})^{k-1} = (1-\tilde{\delta})^k$的概率,

$$\begin{aligned}
&\Delta_{k+1} = F(\mathbf{w}_1^{k+1}) - F(\mathbf{w}_*) \\
\leq &\frac{1}{T_k}\sum_{t=1}^{T_k} F(\mathbf{w}_t^k) - F(\mathbf{w}_*) \\
\leq &\frac{\eta_k G^2}{2} + \frac{\|\mathbf{w}_1^k - \mathbf{w}_*\|^2}{2\eta_k T_k} + \frac{1}{T_k}\left(1 + \frac{8}{3}\log\frac{m_k}{\tilde{\delta}}\right)\frac{4G^2}{\lambda} \\
\overset{(19)}{\leq} &\frac{\eta_k G^2}{2} + \frac{\Delta_k}{\eta_k T_k \lambda} + \frac{1}{T_k}\left(1 + \frac{8}{3}\log\frac{m_k}{\tilde{\delta}}\right)\frac{4G^2}{\lambda} \\
\overset{(25),(26)}{\leq} &\frac{V_k}{4} + \frac{2V_k}{\alpha} + \frac{\lambda V_k}{\alpha G^2}\left(1 + \frac{8}{3}\log\frac{m_k}{\tilde{\delta}}\right)\frac{4G^2}{\lambda} \\
= &\frac{V_k}{4} + \frac{V_k}{\alpha}\left(6 + \frac{32}{3}\log\frac{m_k}{\tilde{\delta}}\right).
\end{aligned}$$

其中$m_k = \lceil 2\log_2 T_k \rceil$.

根据公式(18)中α的定义，可以得到以至少$(1-\tilde{\delta})^k$概率

$$\Delta_{k+1} \leq \frac{V_k}{2} = V_{k+1}.$$

2.4 实例

本节介绍如何利用随机优化求解逻辑回归问题，目标是最小化

$$\begin{aligned}
\min_{\mathbf{w}} \quad & F(\mathbf{w}) = \mathrm{E}_{(\mathbf{x},y)\sim\mathcal{D}}\Big(\log\big(1+\exp(-y\mathbf{w}^\top\mathbf{x})\big)\Big) \\
\text{s.t.} \quad & \|\mathbf{w}\| \leq R
\end{aligned}$$

因为目标函数是凸函数，因此可以采用随机梯度下降求解。因为步长依赖于随机梯度的上界，所以需要计算该上界。我们假设对于所有的$(\mathbf{x}, y) \sim \mathcal{D}$, $\|\mathbf{x}\| \leq G$。这样

$$\left\|\frac{\exp(-y\mathbf{w}^\top\mathbf{x})}{1+\exp(-y\mathbf{w}^\top\mathbf{x})}y\mathbf{x}\right\| \leq \|\mathbf{x}\| \leq G.$$

因为定义域的直径为$2R$，依据定理2.1将步长设置为$\eta = \frac{2R}{G\sqrt{T}}$，这样的随机梯度下降流程如下：

1: 任意初始化$\mathbf{w}_1 \in \mathcal{W}$

2: **for** $t = 1, \ldots, T$ **do**

3: 从分布$\mathcal{D}$中随机采样$(\mathbf{x}_t, y_t)$

4:
$$\mathbf{w}'_{t+1} = \mathbf{w}_t + \frac{2R}{G\sqrt{T}}\frac{y_t\exp(-y_t\mathbf{w}^\top\mathbf{x}_t)}{1+\exp(-y_t\mathbf{w}^\top\mathbf{x}_t)}\mathbf{x}_t$$

5:
$$\mathbf{w}_{t+1} = \frac{R}{\max(\|\mathbf{w}'_{t+1}\|, R)}\mathbf{w}'_{t+1}$$

6: **end for**
7: **return** $\bar{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^{T}\mathbf{w}_t$

根据定理2.1，以至少$1-\delta$的概率，

$$F(\bar{\mathbf{w}}_T) - \min_{\|\mathbf{w}\|\leq R} F(\mathbf{w}) \leq \frac{2GR}{\sqrt{T}}\left(1+2\sqrt{2\log\frac{1}{\delta}}\right).$$

3 完全信息在线学习

3.1 引言

经验风险最小化是批量式的，假设所有的训练数据预先给定，通过最小化所有训练数据上的经验风险得到分类器。当数据规模非常大时，这种学习模式计算复杂度高、响应慢，无法用于实时性要求高的场景。与批量学习不同，在线学习（Online Learning）假设训练数据持续到来，通常利用一个训练样本更新当前模型，显著降低了学习算法的空间复杂度和时间复杂度，实时性强。在大数据时代，在线学习成为解决大数据规模大、增长快的重要技术手段，引起了学术界和工业界的广泛关注。

在线学习是回答一系列问题的过程，同时，算法可以得到历史问题的答案并进行学习[Shalev-Shwartz, 2011]。根据算法对历史答案了解程度的不同，在线学习又可以进一步划分为完全信息在线学习（Full-information Online Learning）和赌博机在线学习（Bandit Online Learning）。在完全信息在线学习问题中，学习算法可以观测到历史问题的完整答案；与之不同，赌博机在线学习只能观测到部分答案。在本章，我们主要关注完全信息在线学习。

在线学习可以被形式化为学习器和对手之间的博弈过程：

- 在每一个时刻t，学习器从解空间$\mathcal{W}$选择决策$\mathbf{w}_t$；同时，对手选择一个损失函数$f_t(\cdot):\mathcal{W}\mapsto\mathbb{R}$;
- 学习器遭受损失$f_t(\mathbf{w}_t)$，并更新模型获得下一时刻的解$\mathbf{w}_{t+1}$;

对于完全信息在线学习，学习器可以观测到完整的损失函数$f_t(\cdot)$，因此可以利用函数的信息（比如梯度）更新模型。对于赌博机在线学习，学习器通常只能观测到损失函数在所选决策$\mathbf{w}_t$上的值$f_t(\mathbf{w}_t)$，不能观测到损失函数在其他决策上的数值。这里，我们通过赛马和赌博机的例子来阐述两者的区别。在赛马比赛中，我们在每一轮选择一匹马，然后所有的马进行比赛；在当前轮次比赛结束之后，我们不仅可以看到所选马的比赛结果，还可以看到其他马的情况。这种场景，学习器观测到了完整的损失函数，就属于完全信息在线学习。在使用赌博机时，面对多个按钮，我们选择其中一个，然后点击该按钮；之后，我们可以得到点击按钮的反馈，却无法观测其他按钮点击后的结果。对于这个问题，学习器只能观测到所选决策的结果，就属于赌博机在线学习。

在线学习的目的是最小化累计的损失。我们假设算法共执行T次迭代，那么累计损失就是$\sum_{t=1}^{T} f_t(\mathbf{w}_t)$。为了刻画在线算法的性能，我们通常将算法的在线损失和离线算法的最小损失进

行比较，将其差值定义为遗憾（Regret）：

$$\text{Regret} = \sum_{t=1}^{T} f_t(\mathbf{w}_t) - \min_{\mathbf{w}\in\mathcal{W}} \sum_{t=1}^{T} f_t(\mathbf{w})$$

最小化累计损失也就等价于最小化遗憾。在线学习算法希望达到次线性的遗憾，即Regret $= o(T)$。这时候我们称算法是Hannan一致的（Hannan Consistent）[Cesa-Bianchi and Lugosi, 2006].

3.2 基于专家建议的预测

下面，我们介绍一种最基本的在线学习框架[Cesa-Bianchi and Lugosi, 2006]：基于专家建议的预测（Prediction with Expert Advice）。在这个学习框架中，我们假设存在N个专家可以对未来进行预测，学习器可以观测到专家的预测结果，进而做出自己的判断。整体学习流程如下：

1: **for** $t = 1, \ldots, T$ **do**
2: 对手选择输出y_t，N个专家提供建议$\{f_{i,t} : i \in [N]\}$
3: 学习器观测到专家建议
4: 学习器做出自己的预测$\widehat{p}_t$
5: 学习器观测到输出y_t
6: 学习器遭受损失$\ell(\widehat{p}_t, y_t)$，第i个专家遭受损失$\ell(f_{i,t}, y_t)$
7: **end for**

在每一时刻t，对手或者环境选择了一个输出y_t，该输出学习器是看不到的；学习器能够看到N个专家对当前轮输出的预测$\{f_{i,t} : i \in [N]\}$；依据这些预测，学习器形成自己的判断$\widehat{p}_t$，之后观测到真实的输出y_t；最后，学习器遭受损失$\ell(\widehat{p}_t, y_t)$，每一个专家i遭受损失$\ell(f_{i,t}, y_t)$，其中$\ell(\cdot,\cdot)$为某损失函数。学习器的目标是最小化其累计损失和最优专家累计损失的差值，也就是遗憾。为了简化符号，令$\widehat{L}_T = \sum_{t=1}^{T} \ell(\widehat{p}_t, y_t)$表示学习器的累计损失，$L_{i,T} = \sum_{t=1}^{T} \ell(f_{i,t}, y_t)$表示第$i$个专家的损失。遗憾可以表示为：

$$\sum_{t=1}^{T} \ell(\widehat{p}_t, y_t) - \min_{i=1,\ldots,N} \sum_{t=1}^{T} \ell(f_{i,t}, y_t) = \widehat{L}_T - \min_{i=1,\ldots,N} L_{i,T}.$$

上述问题的核心是学习器如何从专家建议中得到自己的预测，最常用的方式是加权平均

$$\widehat{p}_t = \frac{\sum_{i=1}^{N} w_{i,t-1} f_{i,t}}{\sum_{j=1}^{N} w_{j,t-1}} \tag{27}$$

其中$w_{i,t-1}$代表第i个专家的权重。这样，算法的核心问题就变成了如何设置权重以及如何更新权重。权重设置的基本思想是依据专家的历史表现：对于累计损失小的专家，设置较大的权重。常用的权重设置方式有两种：多项式加权和指数加权。采用多项式加权，学习器用下面的方式设置权重

$$w_{i,t-1} = \frac{2(R_{i,t-1})_+^{p-1}}{\|(\mathbf{R}_{t-1})_+\|_p^{p-2}} \tag{28}$$

其中$p>0$为参数，$\mathbf{R}_{t-1}=[R_{1,t-1},\dots,R_{N,t-1}]$,

$$R_{i,t-1}=\sum_{t=1}^{t-1}\left(\ell(\widehat{p}_t,y_t)-\ell(f_{i,t},y_t)\right)=\widehat{L}_{t-1}-L_{i,t-1},$$

$(x)_+=\max(0,x)$为取正操作。将(28)代入(27)，可以得到学习器的预测结果为

$$\widehat{p}_t=\frac{\sum_{i=1}^{N}\left(\sum_{s=1}^{t-1}\left(\ell(\widehat{p}_s,y_s)-\ell(f_{i,s},y_s)\right)\right)_+^{p-1}f_{i,t}}{\sum_{j=1}^{N}\left(\sum_{s=1}^{t-1}\left(\ell(\widehat{p}_s,y_s)-\ell(f_{j,s},y_s)\right)\right)_+^{p-1}}.$$

采用指数加权，学习器用下面的方式设置权重

$$w_{i,t-1}=\frac{e^{\eta R_{i,t-1}}}{\sum_{j=1}^{N}e^{\eta R_{j,t-1}}}=\frac{e^{\eta(\widehat{L}_{t-1}-L_{i,t-1})}}{\sum_{j=1}^{N}e^{\eta(\widehat{L}_{t-1}-L_{j,t-1})}}=\frac{e^{-\eta L_{i,t-1}}}{\sum_{j=1}^{N}e^{-\eta L_{j,t-1}}} \tag{29}$$

其中$\eta>0$为参数。将(29)代入(27)，可以得到学习器的预测结果为

$$\widehat{p}_t=\frac{\sum_{i=1}^{N}e^{-\eta L_{i,t-1}}f_{i,t}}{\sum_{j=1}^{N}e^{-\eta L_{j,t-1}}}.$$

下面，我们给出两种加权方式的理论保障。对于多项式加权平均，学习器的遗憾界如下所示[Cesa-Bianchi and Lugosi, 2006, Corollary 2.1]。

定理3.1 (多项式加权遗憾界). *假设损失函数$\ell(\cdot,\cdot)$是关于第一个参数的凸函数，并且取值范围为$[0,1]$。多项式加权平均学习器的遗憾满足*

$$\widehat{L}_T-\min_{i=1,\dots,N}L_{i,T}\le\sqrt{T(p-1)N^{2/p}}.$$

通过设置合适的参数p，可以对上述定理进一步简化。比如当$p=2\ln N$时，遗憾界为

$$\widehat{L}_T-\min_{i=1,\dots,N}L_{i,T}\le\sqrt{Te(2\ln N-1)}=O\left(\sqrt{T\log N}\right).$$

也就意味着遗憾界随T是次线性增长的。另外，注意到遗憾界对专家的个数N的依赖非常弱—随N对数增长。这意味着在实际应用中，我们可以采用大量的专家使得$\min_{i=1,\dots,N}L_{i,T}$非常小，而遗憾界只会略微增加。

对于指数加权平均，学习器的遗憾界如下所示[Cesa-Bianchi and Lugosi, 2006, Corollary 2.2]。

定理3.2 (指数加权遗憾界). *假设损失函数$\ell(\cdot,\cdot)$是关于第一个参数的凸函数，并且取值范围为$[0,1]$。指数加权平均学习器的遗憾满足*

$$\widehat{L}_T-\min_{i=1,\dots,N}L_{i,T}\le\frac{\ln N}{\eta}+\frac{\eta T}{2}.$$

同样，通过选择合适的参数η，可以得到具体的遗憾界。令$\eta=\sqrt{2\ln N/T}$，可以得到

$$\widehat{L}_T-\min_{i=1,\dots,N}L_{i,T}\le\sqrt{2T\ln N}=O\left(\sqrt{T\log N}\right).$$

从上面的介绍可以看出，基于专家建议的预测模型简单，算法也很直观。但是它的应用非常广泛，因为专家是抽象的概念，可以用来表示任意复杂的算法。最近许多针对复杂环境的在线学习算法都是在这个模型的基础上提出，比如Hazan and Seshadhri [2009]提出的最小化自适应遗憾的学习算法，用专家来表示初始时间不同的在线算法；Hou et al. [2017]提出的针对特征演变的学习算法，用专家来表示定义在不同特征空间的在线算法。

除了前面最基本的两种加权平均算法及其理论保证之外，还存在一些扩展，主要为了获得更紧的遗憾界以及支持更复杂的环境。比如当最优专家累计损失非常小时，我们可以得到更紧的遗憾界。令$L_T^* = \min_{i=1,\dots,N} L_{i,T}$，通过设置合适的参数，指数加权平均学习器可以得到下面的遗憾界[Cesa-Bianchi and Lugosi, 2006, Corollary 2.4]

$$\widehat{L}_T - \min_{i=1,\dots,N} L_{i,T} \leq \sqrt{2L_T^* \ln N} + \ln N.$$

这时候的遗憾界并不直接依赖于迭代次数T，而是由L_T^*所决定。在最坏情况下，$L_T^* = O(T)$，遗憾界的量级依然是$O(\sqrt{T \log N})$。而当L_T^*和T呈次线性关系时，即$L_T^* = o(T)$，遗憾界和T的依赖关系得到改善。在理想情况下，L_T^*是常数，遗憾界甚至和T无关。另外，当损失函数是η指数凹时（其定义在下一节给出），指数加权平均学习器也可以得到与T无关的遗憾界[Cesa-Bianchi and Lugosi, 2006, Theorem 3.2 and Proposition 3.1]

$$\widehat{L}_T - \min_{i=1,\dots,N} L_{i,T} \leq \frac{\ln N}{\eta}.$$

前面的遗憾界告诉我们，通过设置合适的权重，可以保证学习器的累计损失和最优的专家损失接近。但是，当我们任务非常困难，所有专家的累计损失都很大，遗憾界就失去了意义。在这种情况下，我们可以对遗憾进行改进，定义新的性能度量准则—跟踪遗憾（Tracking Regret）。给定一个专家的序列$i_1, \dots, i_T \in [N]$，跟踪遗憾定义为

$$R(i_1, \dots, i_T) = \sum_{t=1}^{T} \ell(\widehat{p}_t, y_t) - \sum_{t=1}^{T} \ell(f_{i_t,t}, y_t).$$

跟踪遗憾将学习器的累计损失和专家序列的累计损失相比较。在现实问题中，不同的专家可能擅长不同的问题。通过选择合适的专家序列，能够使得累计损失$\sum_{t=1}^{T} \ell(f_{i_t,t}, y_t)$的数值很小。如果算法的跟踪遗憾界也小，就可以保证学习器有较小的累计损失。关于跟踪遗憾的具体算法和理论，可以参考书籍Cesa-Bianchi and Lugosi [2006]。

3.3 在线凸优化

下面，我们介绍在线凸优化。顾名思义，在线凸优化处理的函数为凸函数。具体而言，在线凸优化采取3.1节里描述的基本框架，并假设所有的损失函数$f_t(\cdot)$和定义域$\mathcal{W}$都是凸的。在线凸优化是非常强大的学习范式，可以用于求解经验风险最小化问题，降低计算复杂度。对于所有采用凸损失函数的经验风险最小化问题，都可以通过下面的方式在线求解：

- 在每一个时刻t，学习器从解空间$\mathcal{W}$选择模型$\mathbf{w}_t$；

- 学习器观测到样本和标记$(\mathbf{x}_t, y_t)$，并遭受损失

$$f_t(\mathbf{w}_t) = \ell(\mathbf{w}_t^\top \mathbf{x}_t, y_t)$$

- 学习器根据损失函数更新模型$\mathbf{w}_t$;

这里的损失函数$\ell(\cdot,\cdot)$用来衡量预测的标记$\mathbf{w}_t^\top \mathbf{x}_t$和真实标记$y_t$的差异。对于分类问题，我们可以选择Hinge损失：

$$\ell(\mathbf{w}_t^\top \mathbf{x}_t, y_t) = \max\left(0, 1 - y_t \mathbf{w}_t^\top \mathbf{x}_t\right),$$

或逻辑损失：

$$\ell(\mathbf{w}_t^\top \mathbf{x}_t, y_t) = \log\left(1 + \exp(-y_t \mathbf{w}_t^\top \mathbf{x}_t)\right). \tag{30}$$

对于回归问题，我们可以选择平方损失：

$$\ell(\mathbf{w}_t^\top \mathbf{x}_t, y_t) = \left(y_t - \mathbf{w}_t^\top \mathbf{x}_t\right)^2. \tag{31}$$

3.3.1 在线梯度下降（OGD）

针对在线凸优化，最经典的算法是在线梯度下降（Online Gradient Descent，OGD）。其基本流程与我们之前介绍过的梯度下降和随机梯度下降非常类似。算法整体流程如下

1: 任意初始化$\mathbf{w}_1 \in \mathcal{W}$
2: **for** $t = 1, \ldots, T$ **do**
3: 学习器从解空间$\mathcal{W}$选择决策$\mathbf{w}_t$；同时，对手选择一个损失函数$f_t(\cdot) : \mathcal{W} \mapsto \mathbb{R}$
4: 学习器观测到损失函数$f_t(\cdot)$，并遭受损失$f_t(\mathbf{w}_t)$
5: 学习器依照在线梯度下降更新决策：

$$\mathbf{w}_{t+1} = \Pi_{\mathcal{W}}\left[\mathbf{w}_t - \eta_t \nabla f_t(\mathbf{w}_t)\right]$$

6: **end for**

也就是说在每一轮迭代，算法使用当前时刻损失函数的梯度更新当前模型，因此被称为在线梯度下降。

虽然在线梯度下降非常简单，但是对于一般的凸函数，它可以达到最优的$O(\sqrt{T})$遗憾界[Abernethy et al., 2008a]。在线梯度下降的遗憾界如下所示[Zinkevich, 2003]。

定理3.3 (在线梯度下降遗憾界—凸函数). *假设所有在线函数是G-Lipschitz连续，定义域$\mathcal{W}$直径为D，即*

$$\|\nabla f_t(\mathbf{w})\|_2 \leq G,\ \forall t \in [t], \mathbf{w} \in \mathcal{W},\quad \|\mathbf{x} - \mathbf{y}\|_2 \leq D,\ \forall \mathbf{x}, \mathbf{y} \in \mathcal{W}.$$

将步长设置为$\eta_t = \frac{D}{G\sqrt{t}}$。对于在线凸优化，在线梯度下降满足

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \min_{\mathbf{w}\in\mathcal{W}} \sum_{t=1}^{T} f_t(\mathbf{w}) \leq \frac{3DG}{2}\sqrt{T} = O\left(\sqrt{T}\right).$$

证明. 令$\mathbf{w}'_{t+1} = \mathbf{w}_t - \eta_t \nabla f_t(\mathbf{w}_t)$。对于任意的$\mathbf{w} \in \mathcal{W}$，

$$\begin{aligned} f_t(\mathbf{w}_t) - f_t(\mathbf{w}) \leq & \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle = \frac{1}{\eta_t} \langle \mathbf{w}_t - \mathbf{w}'_{t+1}, \mathbf{w}_t - \mathbf{w} \rangle \\ = & \frac{1}{2\eta_t} \left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}'_{t+1} - \mathbf{w}\|_2^2 + \|\mathbf{w}_t - \mathbf{w}'_{t+1}\|_2^2 \right) \\ = & \frac{1}{2\eta_t} \left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}'_{t+1} - \mathbf{w}\|_2^2 \right) + \frac{\eta_t}{2} \|\nabla f_t(\mathbf{w}_t)\|_2^2 \\ \leq & \frac{1}{2\eta_t} \left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}_{t+1} - \mathbf{w}\|_2^2 \right) + \frac{\eta_t}{2} G^2. \end{aligned}$$

对上面的不等式从$t = 1$到T求和，得到

$$\begin{aligned} \sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}) \leq & \frac{1}{2\eta_1} \|\mathbf{w}_1 - \mathbf{w}\|_2^2 - \frac{1}{2\eta_T} \|\mathbf{w}_{T+1} - \mathbf{w}\|_2^2 \\ & + \frac{1}{2} \sum_{t=2}^{T} \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) \|\mathbf{w}_t - \mathbf{w}\|_2^2 + \frac{G^2}{2} \sum_{t=1}^{T} \eta_t. \end{aligned} \tag{32}$$

由于

$$\eta_t \leq \eta_{t-1}, \ \|\mathbf{w}_t - \mathbf{w}\|_2^2 \leq D^2,$$

(32)可以进一步化简为

$$\begin{aligned} \sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}) \leq & \frac{D^2}{2\eta_1} + \frac{D^2}{2} \sum_{t=2}^{T} \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) + \frac{G^2}{2} \sum_{t=1}^{T} \eta_t \\ = & \frac{D^2}{2\eta_T} + \frac{G^2}{2} \sum_{t=1}^{T} \eta_t = \frac{DG\sqrt{T}}{2} + \frac{DG}{2} \sum_{t=1}^{T} \frac{1}{\sqrt{t}} \leq \frac{3DG}{2} \sqrt{T}. \end{aligned}$$

□

接下来，我们介绍在线强凸优化，也就是所有在线函数都是λ强凸的情况。强凸的定义见公式(15)。在这种情况下，通过设定合适的步长，在线梯度下降可以达到$O(\log T)$的遗憾界[Hazan et al., 2007]。

定理3.4 (在线梯度下降遗憾界—强凸函数). *假设所有在线函数是λ强凸，即满足公式(15)，假设所有在线函数是G-Lipschitz连续，即*

$$\|\nabla f_t(\mathbf{w})\|_2 \leq G, \ \forall t \in [t], \mathbf{w} \in \mathcal{W}.$$

将步长设置为$\eta_t = \frac{1}{\lambda t}$。对于在线强凸优化，在线梯度下降满足

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \min_{\mathbf{w} \in \mathcal{W}} \sum_{t=1}^{T} f_t(\mathbf{w}) \leq \frac{G^2}{2\lambda} (\log T + 1) = O\left(\frac{\log T}{\lambda} \right).$$

证明. 对于任意的$\mathbf{w} \in \mathcal{W}$，利用强凸性质，可以证明

$$\begin{aligned} & f_t(\mathbf{w}_t) - f_t(\mathbf{w}) \\ \overset{(15)}{\leq} & \langle \nabla f_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w} \rangle - \frac{\lambda}{2}\|\mathbf{w}_t - \mathbf{w}\|_2^2 \\ \leq & \frac{1}{2\eta_t}\left(\|\mathbf{w}_t - \mathbf{w}\|_2^2 - \|\mathbf{w}_{t+1} - \mathbf{w}\|_2^2\right) + \frac{\eta_t}{2}G^2 - \frac{\lambda}{2}\|\mathbf{w}_t - \mathbf{w}\|_2^2. \end{aligned}$$

仿照公式(32)的推导过程，可以得到

$$\begin{aligned} \sum_{t=1}^{T} f_t(\mathbf{w}_t) - f_t(\mathbf{w}) \leq & \frac{1}{2\eta_1}\|\mathbf{w}_1 - \mathbf{w}\|_2^2 - \frac{\lambda}{2}\|\mathbf{w}_1 - \mathbf{w}\|_2^2 - \frac{1}{2\eta_T}\|\mathbf{w}_{T+1} - \mathbf{w}\|_2^2 \\ & + \frac{1}{2}\sum_{t=2}^{T}\left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} - \lambda\right)\|\mathbf{w}_t - \mathbf{w}\|_2^2 + \frac{G^2}{2}\sum_{t=1}^{T}\eta_t. \end{aligned}$$

将步长$\eta_t = \frac{1}{\lambda t}$代入上式，可以得到

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - f_t(\mathbf{w}) \leq \frac{G^2}{2\lambda}\sum_{t=1}^{T}\frac{1}{t} \leq \frac{G^2}{2\lambda}(\log T + 1).$$

□

3.3.2 在线牛顿法（ONS）

下面我们介绍一种特殊的凸函数—指数凹函数，其定义如下：

定义3.1 (指数凹函数). 对于函数$f(\cdot) : \mathcal{W} \mapsto \mathbb{R}$，如果$\exp(-\alpha f(\cdot))$在$\mathcal{W}$上是凹函数，我们称之为$\alpha$指数凹。

指数凹是一种比凸强，但是比强凸弱的条件。梯度有界的强凸函数一定是指数凹函数，但是指数凹函数未必是强凸的。根据定义，很容易验证我们之前介绍的逻辑损失(30)、平方损失(31)都是指数凹的，虽然它们都不是强凸函数。另外一个常见的指数凹函数是负对数损失

$$f(\mathbf{w}) = -\log(\mathbf{x}^\top \mathbf{w}).$$

对于指数凹函数，我们可以把它当做凸函数，利用在线梯度下降得到$O(\sqrt{T})$的遗憾界。但是，这样的结果对T的依赖不是最优的。针对指数凹函数，Hazan et al. [2007]提出了在线牛顿法（Online Newton Step，ONS），能够达到$O(d\log T)$的遗憾界，其中d是变量的维度。在线牛顿法的基本流程如下：

任意初始化$\mathbf{w}_1 \in \mathcal{W}$，$A_0 = \frac{1}{\beta^2 D^2}I$
for $t = 1, \ldots, T$ **do**
　学习器从解空间$\mathcal{W}$选择决策$\mathbf{w}_t$；同时，对手选择一个损失函数$f_t(\cdot) : \mathcal{W} \mapsto \mathbb{R}$
　学习器观测到损失函数$f_t(\cdot)$，并遭受损失$f_t(\mathbf{w}_t)$

5: 学习器依照在线牛顿法更新决策：

$$\mathbf{w}_{t+1} = \Pi_{\mathcal{W}}^{A_t}\left(\mathbf{w}'_{t+1}\right)$$

其中

$$\mathbf{w}'_{t+1} = \mathbf{w}_t - \frac{1}{\beta}A_t^{-1}\nabla f_t(\mathbf{w}_t)$$

$$A_t = A_{t-1} + \nabla f_t(\mathbf{w}_t)[\nabla f_t(\mathbf{w}_t)]^\top$$

6: **end for**

在上面的算法中，β是输入参数，其取值会在后面给出，D是定义域$\mathcal{W}$的直径。算法最大的特点是维持了梯度的外积之和A_t。计算中间解$\mathbf{w}'_{t+1}$的公式类似于牛顿法，区别在于使用矩阵A_t代替了黑森矩阵。此外，由于$\mathbf{w}'_{t+1}$可能会出界，因此利用投影操作将其投影到定义域$\mathcal{W}$中。但是，在这里我们并不是直接利用之前出现过的欧式投影，而是引入了广义投影操作$\Pi_{\mathcal{W}}^{A_t}$，其定义如下

$$\Pi_{\mathcal{W}}^{A_t}(\mathbf{x}) = \operatorname*{argmin}_{\mathbf{w}\in\mathcal{W}}(\mathbf{w}-\mathbf{x})^\top A_t(\mathbf{w}-\mathbf{x}).$$

对于在线牛顿法，其遗憾界如下定理所示[Hazan et al., 2007, Theorem 2]。

定理3.5 (在线牛顿法遗憾界). *假设所有在线函数是G-Lipschitz连续，定义域$\mathcal{W}$直径为D，即*

$$\|\nabla f_t(\mathbf{w})\|_2 \le G,\ \forall t\in[t], \mathbf{w}\in\mathcal{W},\quad \|\mathbf{x}-\mathbf{y}\|_2 \le D,\ \forall \mathbf{x},\mathbf{y}\in\mathcal{W}.$$

假设所有在线函数是α指数凹，设置$\beta = \frac{1}{2}\min\left\{\frac{1}{4GD}, \alpha\right\}$，在线牛顿法满足

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \min_{\mathbf{w}\in\mathcal{W}}\sum_{t=1}^{T} f_t(\mathbf{w}) \le 5\left(\frac{1}{\alpha}+GD\right)d\log T = O\left(\frac{d\log T}{\alpha}\right).$$

证明. 首先，为了简化标记，给定半正定矩阵M，我们定义$\|\mathbf{x}\|_M = \sqrt{\mathbf{x}^\top M\mathbf{x}}$。

根据指数凹函数的定义，可以推导出[Hazan et al., 2007, Lemma 3]

$$f_t(\mathbf{x}) \ge f_t(\mathbf{y}) + \langle\nabla f_t(\mathbf{y}), \mathbf{x}-\mathbf{y}\rangle + \frac{\beta}{2}(\mathbf{x}-\mathbf{y})^\top\nabla f_t(\mathbf{y})\nabla f_t(\mathbf{y})^\top(\mathbf{x}-\mathbf{y}).$$

因此，对于任意的$\mathbf{w}\in\mathcal{W}$，

$$\begin{aligned} f_t(\mathbf{w}_t) - f_t(\mathbf{w}) \le& \langle\nabla f_t(\mathbf{w}_t), \mathbf{w}_t-\mathbf{w}\rangle - \frac{\beta}{2}(\mathbf{w}_t-\mathbf{w})^\top\nabla f_t(\mathbf{w}_t)\nabla f_t(\mathbf{w}_t)^\top(\mathbf{w}_t-\mathbf{w}) \\ =& \langle\nabla f_t(\mathbf{w}_t), \mathbf{w}_t-\mathbf{w}\rangle + \frac{\beta}{2}\left(\|\mathbf{w}_t-\mathbf{w}\|_{A_{t-1}}^2 - \|\mathbf{w}_t-\mathbf{w}\|_{A_t}^2\right). \end{aligned} \tag{33}$$

根据广义投影

$$\begin{aligned} \mathbf{w}_{t+1} =& \Pi_{\mathcal{W}}^{A_t}\left[\mathbf{w}_t - \frac{1}{\beta}A_t^{-1}\nabla f_t(\mathbf{w}_t)\right] \\ =& \operatorname*{argmin}_{\mathbf{w}\in\mathcal{W}}\left[\mathbf{w} - \left(\mathbf{w}_t - \frac{1}{\beta}A_t^{-1}\nabla f_t(\mathbf{w}_t)\right)\right]^\top A_t\left[\mathbf{w} - \left(\mathbf{w}_t - \frac{1}{\beta}A_t^{-1}\nabla f_t(\mathbf{w}_t)\right)\right] \\ =& \operatorname*{argmin}_{\mathbf{w}\in\mathcal{W}}\frac{1}{\beta}\langle\nabla f_t(\mathbf{w}_t), \mathbf{w}-\mathbf{w}_t\rangle + \frac{1}{2}(\mathbf{w}-\mathbf{w}_t)^\top A_t(\mathbf{w}-\mathbf{w}_t) \end{aligned}$$

的性质[Mahdavi et al., 2015, Lemma 2]，可以证明

$$\langle \mathbf{w}_t - \mathbf{w}, \nabla f_t(\mathbf{w}_t) \rangle \leq \frac{\beta}{2} \left(\|\mathbf{w}_t - \mathbf{w}\|_{A_t}^2 - \|\mathbf{w}_{t+1} - \mathbf{w}\|_{A_t}^2 \right) + \frac{1}{2\beta} \|\nabla f_t(\mathbf{w}_t)\|_{A_t^{-1}}^2.$$

将上式代入(33)，得到

$$f_t(\mathbf{w}_t) - f_t(\mathbf{w}) \leq \frac{\beta}{2} \left(\|\mathbf{w}_t - \mathbf{w}\|_{A_{t-1}}^2 - \|\mathbf{w}_{t+1} - \mathbf{w}\|_{A_t} \right) + \frac{1}{2\beta} \|\nabla f_t(\mathbf{w}_t)\|_{A_t^{-1}}^2.$$

对上面的不等式从$t = 1$到T求和，得到

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}) \leq \frac{\beta}{2} \|\mathbf{w}_1 - \mathbf{w}\|_{A_0}^2 + \frac{1}{2\beta} \sum_{t=1}^{T} \|\nabla f_t(\mathbf{w}_t)\|_{A_t^{-1}}^2 \leq \frac{1}{2\beta} + \frac{1}{2\beta} \sum_{t=1}^{T} \|\nabla f_t(\mathbf{w}_t)\|_{A_t^{-1}}^2. \quad (34)$$

最后，根据Hazan et al. [2007, Lemma 11]，可以证明

$$\sum_{t=1}^{T} \|\nabla f_t(\mathbf{w}_t)\|_{A_t^{-1}}^2 \leq d \log T.$$

将上式代入(34)，可以得到

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}) \leq \frac{1}{2\beta}(1 + d \log T) \leq 5 \left(\frac{1}{\alpha} + GD \right) d \log T.$$

□

3.3.3 在线到批处理转换

在线学习可以帮助我们求解随机优化问题，被称为在线到批处理转换（Online-to-batch Conversion）。假设在线函数$f_1(\cdot), \ldots, f_T(\cdot)$是从同一分布$\mathcal{P}$独立采样得到，定义期望函数

$$F(\cdot) = \mathrm{E}_{f \sim \mathcal{P}}[f(\cdot)].$$

我们的目标是找到一个解$\bar{\mathbf{w}}$来尽可能最小化期望函数$F(\cdot)$。

首先，利用在线学习算法，比如在线梯度下降或在线牛顿法，能够得到如下形式的遗憾界

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}) \leq C,$$

其中C的具体取值依赖于具体的算法和函数类型。对上面式子两边求期望，可以得到

$$\mathrm{E}\left[\sum_{t=1}^{T} F(\mathbf{w}_t) \right] - TF(\mathbf{w}) \leq C.$$

定义$\bar{\mathbf{w}} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{w}_t$，根据Jensen不等式，可以得到如下的风险界

$$\mathrm{E}\left[F(\bar{\mathbf{w}}) \right] - F(\mathbf{w}) \leq \frac{1}{T} \left(\mathrm{E}\left[\sum_{t=1}^{T} F(\mathbf{w}_t) \right] - TF(\mathbf{w}) \right) \leq \frac{C}{T}.$$

下面，我们利用第3.3节的遗憾界，给出具体的风险界。当在线函数是凸函数，算法是在线梯度下降，通过定理3.3可以得到

$$\mathrm{E}\left[F(\bar{\mathbf{w}})\right] - F(\mathbf{w}) \leq \frac{3DG}{2\sqrt{T}} = O\left(\frac{1}{\sqrt{T}}\right).$$

上式表明，从期望意义上我们取得了$O(1/\sqrt{T})$的收敛速率，这与定理2.1中随机梯度下降的收敛速率是一致的。同样，利用集中不等式，我们也可以证明$O(1/\sqrt{T})$的收敛速率以大概率成立[Cesa-bianchi et al., 2002]。当在线函数是λ强凸，算法是在线梯度下降，通过定理3.4可以得到

$$\mathrm{E}\left[F(\bar{\mathbf{w}})\right] - F(\mathbf{w}) \leq \frac{G^2}{2\lambda T}(\log T + 1) = O\left(\frac{\log T}{\lambda T}\right).$$

上式表明，从期望意义上我们取得了$O(\log T/[\lambda T])$的收敛速率。同样，也可以证明该速率以大概率成立[Kakade and Tewari, 2009]。但是，这一次我们得到的速率比定理2.3中Epoch-GD的$O(1/[\lambda T])$速率慢。这个结果表明：虽然在线学习可以用来求解随机优化，但是这样得到的结果未必是最优的。同时，这也意味着在线学习比随机优化更加困难[Hazan and Kale, 2011]。

最后，我们考虑在线函数是α指数凹的情况。采用在线牛顿法，通过定理3.5可以得到

$$\mathrm{E}\left[F(\bar{\mathbf{w}})\right] - F(\mathbf{w}) \leq \left(\frac{1}{\alpha} + GD\right)\frac{5d\log T}{T} = O\left(\frac{d\log T}{\alpha T}\right).$$

上式表明，从期望意义上我们取得了$O(d\log T/[\alpha T])$的收敛速率。同样，也可以证明该速率以大概率成立[Mahdavi et al., 2015]。

除了前面介绍的三种情况之外，在线凸优化也存在许多扩展，从而得到更紧的遗憾界以及处理更加困难的问题。比如，Srebro et al. [2010] 分析了在线函数平滑的情况，证明了当最优损失很小时，可以得到比$\sqrt{T}$更紧的遗憾界。具体而言，通过设置合适的步长，在线梯度下降取得了如下遗憾界

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \min_{\mathbf{w}\in\mathcal{W}} \sum_{t=1}^{T} f_t(\mathbf{w}) = O(1 + \sqrt{f_*})$$

其中$f_* = \min_{\mathbf{w}\in\mathcal{W}} \sum_{t=1}^{T} f_t(\mathbf{w})$是最优的离线损失。当最优损失$f_*$和$T$呈次线性关系时，即$f_* = o(T)$，遗憾界变得更紧。

此外，针对动态变化的环境，研究人员提出了新的度量准则，包括动态遗憾[Zinkevich, 2003, Zhang et al., 2017b, 2018a]和自适应遗憾[Hazan and Seshadhri, 2007, Daniely et al., 2015, Zhang et al., 2019]。给定一个序列$\mathbf{u}_1, \ldots, \mathbf{u}_T \in \mathcal{W}$，动态遗憾将该序列的累计损失作为基准，定义下面的度量准则

$$R(\mathbf{u}_1, \ldots, \mathbf{u}_T) = \sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{u}_t).$$

给定一个区间长度τ，自适应遗憾旨在最小化在线算法在任意长度为τ区间上的遗憾，其度量准则如下

$$R(T, \tau) = \max_{[s, s+\tau-1]\subseteq[T]} \left(\sum_{t=s}^{s+\tau-1} f_t(\mathbf{w}_t) - \min_{\mathbf{w}\in\mathcal{W}} \sum_{t=s}^{s+\tau-1} f_t(\mathbf{w})\right).$$

动态遗憾和自适应遗憾是两种针对最优模型不固定的解决方案，它们之间的关系可以参考论文[Zhang et al., 2018b]。

最后，我们指出前面介绍的在线学习算法，比如在线梯度下降、在线牛顿法均缺乏普适性。这是由于这些算法只能处理一种函数类型。此外，在面临强凸函数和指数凹函数时，它们还需要知道强凸和指数凹的参数。因此，当我们将在线学习算法应用到实际问题中，通常需要领域专家来选择算法、设置参数，这极大地制约了在线学习的普及。针对该问题，最近的研究提出了普适性在线学习算法，能够同时处理多种类型的函数，并且可以自动选择合适的参数。代表性的算法包括Metagrad[van Erven and Koolen, 2016]和Maler[Wang et al., 2019]。

3.4 实例

在本节，我们介绍如何利用在线学习求解支持向量机[Shalev-Shwartz et al., 2007]。其学习流程如下：

- 每一时刻t，学习器选择分类面$\mathbf{w}_t \in \mathbb{R}^d$;
- 学习器观测到样本和标记$(\mathbf{x}_t, y_t)$，并遭受损失

$$f_t(\mathbf{w}_t) = \max(0, 1 - y_t \mathbf{w}_t^\top \mathbf{x}_t) + \frac{\lambda}{2}\|\mathbf{w}_t\|^2$$

其中$\lambda > 0$是给定的正则化参数，$\|\mathbf{x}_t\| \leq R$，$y_t \in [\pm 1]$。

因为损失函数是λ强凸，应该采用步长$\eta_t = 1/(\lambda t)$的在线梯度下降。

1: 初始化$\mathbf{w}_1 = 0$
2: **for** $t = 1, \ldots, T$ **do**
3: 设置步长$\eta_t = \frac{1}{\lambda t}$
4: 计算梯度

$$\nabla f_t(\mathbf{w}_t) = \begin{cases} -y_t \mathbf{x}_t + \lambda \mathbf{w}_t, & y_t \mathbf{w}_t^\top \mathbf{x}_t \leq 1 \\ \lambda \mathbf{w}_t, & y_t \mathbf{w}_t^\top \mathbf{x}_t > 1 \end{cases}$$

5: 学习器依照在线梯度下降更新决策：

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla f_t(\mathbf{w}_t)$$

6: **end for**

为了应用讲义中定理3.4，我们需要知道每一轮梯度的上界G。通过梯度的表达公式，可以看出

$$\|\nabla f_t(\mathbf{w}_t)\| \leq \|\mathbf{x}_t\| + \lambda \|\mathbf{w}_t\| \leq R + \lambda \|\mathbf{w}_t\|.$$

因此，我们需要知道$\|\mathbf{w}_t\|$的上界。接下来，我们通过数学归纳法证明$\|\mathbf{w}_t\| \leq R/\lambda$，$\forall t \geq 1$。

首先，我们知道$\|\mathbf{w}_1\| = 0 \leq R/\lambda$。假设$\|\mathbf{w}_k\| \leq R/\lambda$。根据在线梯度下降的更新公式，

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta_k \nabla f_k(\mathbf{w}_k) = \left(1 - \frac{1}{k}\right)\mathbf{w}_k + \frac{y_k}{\lambda k}\mathbf{x}_k\{\mathbf{w}_k^\top \mathbf{x}_k \leq 1\} = \frac{k-1}{k}\mathbf{w}_k + \frac{y_k}{\lambda k}\mathbf{x}_k\{\mathbf{w}_k^\top \mathbf{x}_k \leq 1\}.$$

因此

$$\|\mathbf{w}_{k+1}\| \leq \frac{k-1}{k}\|\mathbf{w}_k\| + \frac{1}{\lambda k}\|\mathbf{x}_k\| \leq \frac{k-1}{k}\frac{R}{\lambda} + \frac{R}{\lambda k} = \frac{R}{\lambda}.$$

最后，我们得到$\|\nabla f_t(\mathbf{w}_t)\| \leq 2R$，$\forall t \geq 1$。根据定理3.4，

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \min_{\mathbf{w}\in\mathcal{W}} \sum_{t=1}^{T} f_t(\mathbf{w}) \leq \frac{2R^2}{\lambda}(\log T + 1).$$

4 赌博机在线学习

4.1 引言

在本章，我们介绍赌博机在线学习。在这种设定中，学习器只能观测到部分答案，也就是只能观测到损失函数的部分信息，比如函数值。这类问题之所以被称为赌博机在线学习，是因为其最早被用来建模赌场中的多臂赌博机问题（Multi-Armed Bandits）[Robbins, 1952]。

在前一章，我们通过赛马和赌博机的例子阐述了完全信息在线学习和赌博机在线学习的区别。接下来，我们再用机器学习领域常见的多分类问题来介绍它们不同的应用场景[Kakade et al., 2008]。对于多分类问题，每一轮迭代，学习器观测到一个样本，然后预测其标记。在预测之后，如果学习器能观测到正确的标记，那么这个问题就属于完全信息在线学习；如果学习器只被告知预测结果是否正确，那么我们面临的就是赌博机在线学习。其区别在于，在赌博机设定下，学习器预测错误之后并不知道正确答案。

与完全信息在线学习相比，赌博机在线学习更加困难。在设计和分析赌博机在线学习算法时，需要对学习过程进行统计假设[Bubeck and Cesa-Bianchi, 2012]。首先，对手可以分为随机（Stochastic）和对抗（Adversarial）这两种类型[1]。在随机类型中，损失函数是从同一个分布中独立采样得到，这类问题相对容易。对抗类型又可以进一步分为健忘（Oblivious）和非健忘（Nonoblivious）两种。在健忘设定中，损失函数$f_1, \ldots, f_T$和学习器的决策无关，可以认为是固定的函数序列。而在非健忘设定中，损失函数f_t可以依赖于学习器之前的决策$\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}$。由于存在复杂的依赖关系，非健忘类型的问题最困难。

接下来，我们将分别介绍多臂赌博机、线性赌博机、凸赌博机三种常见的赌博机学习问题。

4.2 多臂赌博机

在多臂赌博机问题中，学习器面临K个手臂。在每一轮迭代，学习器选择1个手臂，摇动该手臂并获得对应的奖励。学习器的目的是最大化T轮迭代的累计收益。

在本节，我们考虑随机情况下的多臂赌博机问题，假设每一个手臂的奖励是独立同分布的[Auer et al., 2002a]。将第i个手臂的奖励均值记为μ_i，将学习器在第t轮选择的手臂记为i_t。学习器的遗憾定义

[1] 对手也可以分为随机和非随机两种类型。

为

$$\text{Regret} = T \max_{i \in [K]} \mu_i - \sum_{t=1}^{T} \mu_{i_t}.$$

由于观测信息不充分，学习器面临探索和利用之间的折中（Exploration and Exploitation Trade-off）。一方面，为了准确地估计每个手臂的奖励均值，学习器需要尝试不同的手臂；而另一方面，为了最小化遗憾，学习器又倾向于选择能得到最大收益的手臂。对于随机设定，解决探索和利用折中的典型算法是置信上界（Upper Confidence Bound，UCB）。其核心思想是为每一个手臂i维持一个置信上界$\hat{\mu}_i$，并保证以很大的概率使得均值$\mu_i \leq \hat{\mu}_i$。然后，算法通过选择具有最大上界的手臂自动实现探索和利用之间的折中。

由于我们假设每个手臂的奖励是独立同分布的，因此可以利用集中不等式构造置信上界。首先，我们引入Chernoff-Hoeffding不等式[Auer et al., 2002a]。

定理4.1 (Chernoff-Hoeffding不等式). 令$X_1, \ldots, X_T$为取值在$[0,1]$区间内的随机变量，并且

$$\mathrm{E}\left[X_t | X_1, \ldots, X_{t-1}\right] = \mu.$$

将均值记为$\bar{X} = \sum_{t=1}^{T} X_t / T$。那么，对于所有的$\alpha \geq 0$，

$$\Pr\left[\bar{X} \geq \mu + \alpha\right] \leq e^{-2T\alpha^2}, \ \Pr\left[\bar{X} \leq \mu - \alpha\right] \leq e^{-2T\alpha^2}.$$

下面我们阐述如何利用上述不等式构造置信上界。假设第i个手臂的奖励取值范围为$[0,1]$，并且算法摇动了该手臂n_i次。将n_i次奖励的均值记为$\bar{\mu}_i$，将置信上界定义为

$$\hat{\mu}_i = \bar{\mu}_i + \sqrt{\frac{2\log(\alpha)}{n_i}}.$$

根据定理4.1，可以得到以至少$1-\alpha^{-4}$的概率，$\mu_i \leq \hat{\mu}_i$。可以看出，置信上界$\hat{\mu}_i$是由两部分组成：样本均值$\bar{\mu}_i$和区间宽度$\sqrt{\frac{2\log(\alpha)}{n_i}}$。其中，样本均值反映了学习器当前的知识，而宽度则反映了知识的不确定性。置信上界取值大，那么是因为样本均值大，对应于利用；要么是因为区间宽度很大，对应于探索。因此，依据置信上界来选择手臂，就可以自动在探索和利用之间取得平衡。

基于置信上界的随机多臂赌博机算法整体流程如下：

1: **for** $t = 1, 2, \ldots, K$ **do**
2: 摇动第t个手臂，并观测到奖励X_t^t
3: **end for**
4: 初始化每个手臂被选择的次数：

$$n_i^K = 1, \ i = 1, \ldots, K$$

5: 初始化每个手臂的样本均值：

$$\bar{\mu}_i(n_i^K) = X_t^t, \ i = 1, \ldots, K$$

6: **for** $t = K+1, 2, \ldots, T$ **do**

7: 摇动手臂

$$i_t = \underset{i\in[K]}{\operatorname{argmax}}\, \bar{\mu}_i(n_i^{t-1}) + \sqrt{\frac{2\log(t-1)}{n_i^{t-1}}}$$

并且观测到奖励$X_{i_t}^t$

8: 更新每个手臂被摇动的次数：

$$n_{i_t}^t = n_{i_t}^{t-1} + 1, \quad n_i^t = n_i^{t-1},\ i \neq i_t$$

9: 更新手臂i_t的样本均值：

$$\bar{\mu}_{i_t}(n_{i_t}^t) = \frac{n_{i_t}^{t-1} \times \bar{\mu}_{i_t}(n_{i_t}^{t-1}) + X_{i_t}^t}{n_{i_t}^t}$$

10: **end for**

在上述算法中，我们令X_i^t表示在第t轮摇动手臂i得到的奖励；n_i^t表示在截止到第t轮手臂i被选择的次数；$\bar{\mu}_i(n)$表示手臂i摇动n次后得到的样本均值。由于没有先验知识，算法的前K轮是纯粹的探索阶段，摇动每个手臂1次。从$K+1$轮开始，算法依据置信上界选择手臂。

对于随机多臂赌博机，可以证明置信上界算法在期望意义上的遗憾界为$O(K\log T)$[Auer et al., 2002a, Theorem 1]。

定理4.2 (随机多臂赌博机遗憾界). *假设每一个手臂的奖励属于区间*$[0,1]$，*并且服从同样的分布。将最优手臂的索引记为*$*$，*即*$* = \operatorname{argmax}_{i\in[K]} \mu_i$。*置信上界算法满足*

$$T\max_{i\in[K]}\mu_i - \mathrm{E}\left[\sum_{t=1}^{T}\mu_{i_t}\right] \leq 8\sum_{i\neq *}\frac{\ln T}{\Delta_i} + \left(1+\frac{\pi^2}{3}\right)\sum_{i\neq *}\Delta_i = O\left(K\log T\right)$$

其中$\Delta_i = \mu_* - \mu_i$。

证明. 根据遗憾的定义，可以得到

$$\mathrm{Regret} = T\max_{i\in[K]}\mu_i - \sum_{t=1}^{T}\mu_{i_t} = T\mu_* - \sum_{t=1}^{T}\mu_{i_t} = \sum_{i\neq *}(\mu_* - \mu_i)n_i^T = \sum_{i\neq *}\Delta_i n_i^T.$$

上式表明，通过计算第i个手臂在T轮迭代中被摇动的次数n_i^T，就可以得到遗憾的上界。

首先，可以得到

$$
\begin{aligned}
n_i^T =& 1 + \sum_{t=K+1}^{T} \{i_t = i\} \\
\leq& \ell + \sum_{t=K+1}^{T} \left\{ i_t = i, n_i^{t-1} \geq \ell \right\} \\
\leq& \ell + \sum_{t=K+1}^{T} \left\{ \bar{\mu}_*(n_*^{t-1}) + \sqrt{\frac{2\log(t-1)}{n_*^{t-1}}} \leq \bar{\mu}_i(n_i^{t-1}) + \sqrt{\frac{2\log(t-1)}{n_i^{t-1}}}, n_i^{t-1} \geq \ell \right\} \\
\leq& \ell + \sum_{t=K+1}^{T} \left\{ \min_{0<p<t} \bar{\mu}_*(p) + \sqrt{\frac{2\log(t-1)}{p}} \leq \max_{\ell \leq q < t} \bar{\mu}_i(q) + \sqrt{\frac{2\log(t-1)}{q}} \right\} \\
\leq& \ell + \sum_{t=1}^{T-1} \sum_{p=1}^{t-1} \sum_{q=\ell}^{t-1} \left\{ \bar{\mu}_*(p) + \sqrt{\frac{2\log(t)}{p}} \leq \bar{\mu}_i(q) + \sqrt{\frac{2\log(t)}{q}} \right\} \\
\leq& \ell + \sum_{t=1}^{T-1} \sum_{p=1}^{t-1} \sum_{q=\ell}^{t-1} \left(\left\{ \bar{\mu}_*(p) + \sqrt{\frac{2\log(t)}{p}} \leq \mu_* \right\} \right. \\
& \left. + \left\{ \mu_* \leq \mu_i + 2\sqrt{\frac{2\log(t)}{q}} \right\} + \left\{ \mu_i + \sqrt{\frac{2\log(t)}{q}} \leq \bar{\mu}_i(q) \right\} \right).
\end{aligned}
$$

然后，对上式求期望，得到

$$
\begin{aligned}
\mathrm{E}[n_i^T] \leq& \ell + \sum_{t=1}^{T-1} \sum_{p=1}^{t-1} \sum_{q=\ell}^{t-1} \left(\Pr\left[\bar{\mu}_*(p) + \sqrt{\frac{2\log(t)}{p}} \leq \mu_* \right] \right. \\
& \left. + \Pr\left[\mu_* \leq \mu_i + 2\sqrt{\frac{2\log(t)}{q}} \right] + \Pr\left[\mu_i + \sqrt{\frac{2\log(t)}{q}} \leq \bar{\mu}_i(q) \right] \right) \\
\leq& \ell + \sum_{t=1}^{T-1} \sum_{p=1}^{t-1} \sum_{q=\ell}^{t-1} \left(t^{-4} + \Pr\left[\mu_* \leq \mu_i + 2\sqrt{\frac{2\log(t)}{q}} \right] + t^{-4} \right).
\end{aligned}
$$

其中最后一个不等式利用了定理4.1。

令$\ell = \left\lceil \frac{8\ln T}{\Delta_i^2} \right\rceil$，可以使得

$$
\Pr\left[\mu_* \leq \mu_i + 2\sqrt{\frac{2\log(t)}{q}} \right] = 0, \ q \geq \ell.
$$

因此，我们有

$$
\begin{aligned}
\mathrm{E}[n_i^T] \leq& \left\lceil \frac{8\ln T}{\Delta_i^2} \right\rceil + 2\sum_{t=1}^{\infty} \sum_{p=1}^{t-1} \sum_{q=\lceil 8\ln T/\Delta_i^2 \rceil}^{t-1} t^{-4} \\
\leq& \frac{8\ln T}{\Delta_i^2} + 1 + 2\sum_{t=1}^{\infty} t^{-2} \leq \frac{8\ln T}{\Delta_i^2} + 1 + \frac{\pi^2}{3}.
\end{aligned}
$$

最后，我们得到

$$\mathrm{E}\left[\mathrm{Regret}\right] = \sum_{i \neq *} \Delta_i \mathrm{E}\left[n_i^T\right] \leq 8 \sum_{i \neq *} \frac{\ln T}{\Delta_i} + \left(1 + \frac{\pi^2}{3}\right) \sum_{i \neq *} \Delta_i.$$

□

定理4.2告诉我们，从期望意义上，随机多臂赌博机具备$O(K \log T)$的遗憾界。同时，也可以证明该遗憾界以大概率成立[Abbasi-yadkori et al., 2011]。从遗憾界的具体形式可以看出，遗憾界的取值依赖于Δ_i，即不同手臂均值之差。如果最优手臂和次优手臂之间的差值很小，遗憾界中的常数会很大。为了解决这个问题，我们可以建立与分布无关的遗憾界[Bubeck and Cesa-Bianchi, 2012]。另外，非随机情况下的多臂赌博机也有相应的算法和遗憾界，具体可以参考文献[Auer et al., 2002b]。

4.3 线性赌博机

前面介绍的多臂赌博机可以应用到在线商品推荐问题中：将商品建模为手臂，每次向用户推荐特定商品就等价于选择1个手臂；用户对商品的反馈，对应于奖励。利用前面的置信上界法，在随机情况下可以得到$O(K \log T)$的遗憾界。虽然该遗憾界随迭代次数T增长非常缓慢，但是它和手臂的个数K呈线性关系。因此，当商品的数量特别大时，多臂赌博机的遗憾界也会很大，算法的实际效果并不理想。

随机多臂赌博机将手臂建模为抽象的概念，没有利用手臂之间的关联，因此其遗憾界和手臂个数成线性关系。而在实际应用中，手臂是有物理意义的，往往存在辅助信息可以用来估计手臂的奖励。比如，对于每一个商品，我们可以利用商品描述、用户评价等信息得到一个d维的向量来描述该商品。这样，每一个手臂就变成了一个d维空间内的向量，而奖励就可以建模为该向量的函数。对于手臂$\mathbf{x} \in \mathbb{R}^d$，随机线性赌博机假设其奖励均值$\mu_{\mathbf{x}}$是$\mathbf{x}$的线性函数[Dani et al., 2008a, Abbasi-yadkori et al., 2011]，即

$$\mu_{\mathbf{x}} = \mathbf{x}^\top \mathbf{w}_*,$$

其中$\mathbf{w}_*$是未知的参数。通过这样的假设，不同的手臂共享同一组参数$\mathbf{w}_*$，也就建立起手臂之间的关联。当学习器选择了手臂$\mathbf{x}$后，观测到奖励

$$y = \mathbf{x}^\top \mathbf{w}_* + \epsilon,$$

其中ϵ为均值为0的随机噪声。令$\mathcal{D} \subseteq \mathbb{R}^d$表示手臂组成的集合，学习器的遗憾被定义为：

$$T \max_{\mathbf{x} \in \mathcal{D}} \mathbf{x}^\top \mathbf{w}_* - \sum_{t=1}^{T} \mathbf{x}_t^\top \mathbf{w}_*,$$

其中$\mathbf{x}_t \in \mathcal{D}$表示学习器在第$t$轮选择的手臂。

对于随机线性赌博机，我们同样可以利用置信上界法求解。通过上面的描述可以知道，如果学习器能够估计参数$\mathbf{w}_*$，也就可以估计每一个手臂$\mathbf{x}$的奖励均值$\mu_{\mathbf{x}}$。假设我们可以证明以很大的概率$\mathbf{w}_* \in \mathcal{C}$，其中$\mathcal{C}$表示置信区域。那么对于每一个手臂$\mathbf{x}$，我们可以构造置信上界

$$\hat{\mu}_{\mathbf{x}} = \max_{\mathbf{w} \in \mathcal{C}_t} \mathbf{x}^\top \mathbf{w},$$

并且以很大的概率$\mu_{\mathbf{x}} = \mathbf{x}^\top \mathbf{w}_* \leq \hat{\mu}_{\mathbf{x}}$。接下来，算法就可以依据置信上界$\hat{\mu}_{\mathbf{x}}$来选择最优的手臂，得到奖励后更新置信区域。

假设算法已经运行了t轮，学习器选择了手臂$\mathbf{x}_1, \ldots, \mathbf{x}_t$，并观测到奖励$y_1, \ldots, y_t$。由于我们假设

$$y_t = \mathbf{x}_t^\top \mathbf{w}_* + \epsilon_t,$$

其中ϵ_t为均值为0的随机噪声。因此，可以通过求解岭回归问题来估计参数$\mathbf{w}_*$：

$$\min_{\mathbf{w} \in \mathbb{R}^d} \sum_{i=1}^{t} \left(y_i - \mathbf{x}_i^\top \mathbf{w}\right)^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2.$$

将其最优解记为

$$\mathbf{w}_t = \left(\lambda I + \sum_{i=1}^{t} \mathbf{x}_i \mathbf{x}_i^\top\right)^{-1} \left(\sum_{i=1}^{t} y_i \mathbf{x}_i\right).$$

根据Sherman–Morrison公式[Golub and Van Loan, 1996]，学习算法可以在线地计算$\mathbf{w}_t$:

$$\begin{aligned} \mathbf{w}_t =& \left(Z_{t-1}^{-1} - \frac{Z_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top Z_{t-1}^{-1}}{1 + \mathbf{x}_t^\top Z_{t-1}^{-1}\mathbf{x}_t}\right)(\mathbf{z}_{t-1} + y_t\mathbf{x}_t) \\ =& \mathbf{w}_{t-1} + \left(y_t - \frac{\mathbf{x}_t^\top Z_{t-1}^{-1}\mathbf{z}_t}{1 + \mathbf{x}_t^\top Z_{t-1}^{-1}\mathbf{x}_t}\right) Z_{t-1}^{-1}\mathbf{x}_t, \\ Z_t^{-1} =& Z_{t-1}^{-1} - \frac{Z_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top Z_{t-1}^{-1}}{1 + \mathbf{x}_t^\top Z_{t-1}^{-1}\mathbf{x}_t}, \\ \mathbf{z}_t =& \mathbf{z}_{t-1} + y_t\mathbf{x}_t, \end{aligned} \tag{35}$$

其中$Z_{t-1} = \lambda I + \sum_{i=1}^{t-1} \mathbf{x}_i\mathbf{x}_i^\top$ and $\mathbf{z}_{t-1} = \sum_{i=1}^{t-1} y_i\mathbf{x}_i$。这样，学习器就不需要保存历史数据$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_t, y_t)$，只需要在线维护$\mathbf{z}_t$、$Z_t^{-1}$和$\mathbf{w}_t$即可。

基于$\mathbf{w}_t$，可以利用集中不等式构造参数$\mathbf{w}_*$的置信区域[Abbasi-yadkori et al., 2011, Theorem 2]。

定理4.3 (置信区域). *假设观测数据满足*

$$y_t = \mathbf{x}_t^\top \mathbf{w}_* + \epsilon_t$$

其中噪声ϵ_t均值为0，并且是条件R次高斯（Conditionally R-sub-Gaussian），即

$$\mathrm{E}_t\left[e^{\lambda\epsilon_t}\right] \leq \exp\left(\frac{\lambda^2 R^2}{2}\right), \ \forall \lambda \in \mathbb{R}.$$

假设$\|\mathbf{w}_*\| \leq S$，并且$\|\mathbf{x}_t\| \leq L$，$\forall t \in [T]$。那么以至少$1-\delta$的概率，

$$\mathbf{w}_* \in \mathcal{C}_t = \left\{\mathbf{w} \in \mathbb{R}^d \middle| \|\mathbf{w} - \mathbf{w}_t\|_{Z_t} \leq R\sqrt{2\log\frac{1}{\delta} + d\log\left(1 + \frac{tL^2}{\lambda d}\right)} + S\sqrt{\lambda}\right\}, \tag{36}$$

其中$Z_t = \lambda I + \sum_{i=1}^{t} \mathbf{x}_i\mathbf{x}_i^\top$。

定理4.3表明，$\mathbf{w}_*$以大概率属于一个中心为$\mathbf{w}_t$的椭圆$\mathcal{C}_t$内。

基于置信上界的随机线性赌博机算法整体流程如下：

1: **for** $t = 1, 2, \ldots, T$ **do**
2: 根据置信上界选择选择手臂

$$\mathbf{x}_t = \operatorname*{argmax}_{\mathbf{x}\in\mathcal{D}} \max_{\mathbf{w}\in\mathcal{C}_{t-1}} \mathbf{x}^\top \mathbf{w}$$

并观测到奖励y_t
3: 依据(35)计算$\mathbf{w}_t$，并依据(36)更新置信区域$\mathcal{C}_t$
4: **end for**

算法的主要计算代价在于选择手臂的过程，该优化问题等价于

$$\max_{(\mathbf{x},\mathbf{w})\in\mathcal{D}\times\mathcal{C}_{t-1}} \mathbf{x}^\top \mathbf{w}.$$

当$\mathcal{D}$包含有限个手臂时，可以通过枚举所有的手臂来求解上述问题。当$\mathcal{D}$包含无穷多个手臂时，上述问题非常难以求解，在某些情况下甚至是NP困难的。但是，对于特殊的问题，比如$\mathcal{D}$是球体时，该问题可以在多项式时间内求解[Dani et al., 2008a, Zhang et al., 2016]。

对于随机线性赌博机，可以证明置信上界算法的遗憾界为$\widetilde{O}(d\sqrt{T})$[Abbasi-yadkori et al., 2011, Theorem 3]。

定理4.4 (随机线性赌博机遗憾界). *假设定理4.3的前提条件成立，假设*$-1 \le \mathbf{x}^\top \mathbf{w}_* \le 1$，$\forall \mathbf{x} \in \mathcal{D}$。*以至少*$1-\delta$*的概率，置信上界算法满足*

$$\begin{aligned} & T \max_{\mathbf{x}\in\mathcal{D}} \mathbf{x}^\top \mathbf{w}_* - \sum_{t=1}^{T} \mathbf{x}_t^\top \mathbf{w}_* \\ \le & 2\sqrt{2Td\log\left(1+\frac{TL^2}{\lambda d}\right)}\left(R\sqrt{2\log\frac{1}{\delta}+d\log\left(1+\frac{TL^2}{\lambda d}\right)}+S\sqrt{\lambda}\right) = \widetilde{O}\left(d\sqrt{T}\right). \end{aligned}$$

证明. 令$\mathbf{x}_* = \operatorname{argmax}_{\mathbf{x}\in\mathcal{D}} \mathbf{x}^\top \mathbf{w}_*$。算法在第$t$轮的遗憾为

$$r_t = \mathbf{x}_*^\top \mathbf{w}_* - \mathbf{x}_t^\top \mathbf{w}_*.$$

令$(\mathbf{x}_t, \widetilde{\mathbf{w}}_t) = \operatorname{argmax}_{(\mathbf{x},\mathbf{w})\in\mathcal{D}\times\mathcal{C}_{t-1}} \mathbf{x}^\top \mathbf{w}$，以$1-\delta$的概率

$$\mathbf{w}_* \in \mathcal{C}_{t-1} \Rightarrow \mathbf{x}_*^\top \mathbf{w}_* \le \mathbf{x}_t^\top \widetilde{\mathbf{w}}_t.$$

因此，以$1-\delta$的概率

$$r_t \le \mathbf{x}_t^\top \widetilde{\mathbf{w}}_t - \mathbf{x}_t^\top \mathbf{w}_* = \mathbf{x}_t^\top (\widetilde{\mathbf{w}}_t - \mathbf{w}_*) = \mathbf{x}_t^\top (\widetilde{\mathbf{w}}_t - \mathbf{w}_{t-1}) + \mathbf{x}_t^\top (\mathbf{w}_{t-1} - \mathbf{w}_*).$$

根据柯西-施瓦茨不等式，

$$r_t \le \|\widetilde{\mathbf{w}}_t - \mathbf{w}_{t-1}\|_{Z_{t-1}} \|\mathbf{x}_t\|_{Z_{t-1}^{-1}} + \|\mathbf{w}_{t-1} - \mathbf{w}_*\|_{Z_{t-1}} \|\mathbf{x}_t\|_{Z_{t-1}^{-1}}.$$

根据定理4.3，可以进一步得到

$$r_t \le 2\gamma_{t-1} \|\mathbf{x}_t\|_{Z_{t-1}^{-1}} \tag{37}$$

其中

$$\gamma_t = R\sqrt{2\log\frac{1}{\delta} + d\log\left(1+\frac{tL^2}{\lambda d}\right)} + S\sqrt{\lambda}.$$

根据条件$-1 \le \mathbf{x}^\top \mathbf{w}_* \le 1$，可以得到$r_t \le 2$。结合(37)，可以得到

$$r_t \le 2\min\left(\gamma_{t-1}\|\mathbf{x}_t\|_{Z_{t-1}^{-1}}, 1\right) \le 2\gamma_{t-1}\min\left(\|\mathbf{x}_t\|_{Z_{t-1}^{-1}}, 1\right).$$

因此，以$1-\delta$的概率

$$\begin{aligned} &T\max_{\mathbf{x}\in\mathcal{D}} \mathbf{x}^\top \mathbf{w}_* - \sum_{t=1}^{T} \mathbf{x}_t^\top \mathbf{w}_* = \sum_{t=1}^{T} r_t \\ \le &\sqrt{T\sum_{t=1}^{T} r_t^2} \le 2\sqrt{T\sum_{t=1}^{T}\gamma_{t-1}^2 \min\left(\|\mathbf{x}_t\|_{Z_{t-1}^{-1}}^2, 1\right)} \le 2\gamma_T\sqrt{T\sum_{t=1}^{T}\min\left(\|\mathbf{x}_t\|_{Z_{t-1}^{-1}}^2, 1\right)}. \end{aligned} \tag{38}$$

根据$Z_{t-1} = \lambda I + \sum_{i=1}^{t-1}\mathbf{x}_i\mathbf{x}_i^\top$的表达式，可以证明[Abbasi-yadkori et al., 2011, Lemma 11]

$$\sum_{t=1}^{T}\min\left(\|\mathbf{x}_t\|_{Z_{t-1}^{-1}}^2, 1\right) \le 2\log\frac{\det(Z_T)}{\det(\lambda I)} \le 2d\log\left(1+\frac{TL^2}{\lambda d}\right).$$

将上式代入(38)，可以得到以$1-\delta$的概率

$$T\max_{\mathbf{x}\in\mathcal{D}} \mathbf{x}^\top \mathbf{w}_* - \sum_{t=1}^{T} \mathbf{x}_t^\top \mathbf{w}_* \le 2\sqrt{2Td\log\left(1+\frac{TL^2}{\lambda d}\right)}\left(R\sqrt{2\log\frac{1}{\delta} + d\log\left(1+\frac{TL^2}{\lambda d}\right)} + S\sqrt{\lambda}\right).$$

□

上面介绍了随机线性赌博机的算法和遗憾界，对于非随机情况，可以参考文献[Dani et al., 2008b, Abernethy et al., 2008b, Bartlett et al., 2008]。线性赌博机假设用户的反馈是实数值，然而实际应用中的反馈往往是1位的（二值的），比如是否购买、是否喜欢、是否点击等，1位反馈下的在线线性学习可以参考文献[Zhang et al., 2016]。另外，在赌博机设定下，还可以把线性模型推广到广义线性模型[Filippi et al., 2010, Jun et al., 2017]。

4.4 凸赌博机

在这一节，我们研究赌博机设定下的在线凸优化[Flaxman et al., 2005]。对于在线凸优化，解空间$\mathcal{W}$以及所有的损失函数$f_t(\cdot): \mathcal{W} \mapsto \mathbb{R}$都是凸的。我们在第3.3节，讨论了完全信息设定下的在线凸优化问题，并介绍了在线梯度下降算法。完全信息设定下，学习器可以观测到完整的损失函数，因此可以求梯度。但是在赌博机设定下，学习器只能观测到损失函数f_t在决策$\mathbf{w}_t$上的值$f_t(\mathbf{w}_t)$，因此无法直接应用在线梯度下降。针对这一问题，我们需要引入从函数值估计梯度的技术。

首先，我们定义单位球体$\mathbb{B}$和单位球面$\mathbb{S}$:

$$\mathbb{B} = \left\{\mathbf{w}\in\mathbb{R}^d \middle| \|\mathbf{w}\| \le 1\right\},\ \mathbb{S} = \left\{\mathbf{w}\in\mathbb{R}^d \middle| \|\mathbf{w}\| = 1\right\}.$$

令$\mathbf{v}$表示均匀分布在单位球体$\mathbb{B}$内的随机变量，给定某函数$f(\cdot)$，定义

$$\hat{f}(\mathbf{w}) = \mathrm{E}_{\mathbf{v}\in\mathbb{B}}\left[f(\mathbf{w}+\delta\mathbf{v})\right],$$

其中$\delta > 0$为参数。$\hat{f}(\cdot)$可以当做函数$f(\cdot)$的平滑版本，并且当δ很小时，$\hat{f}(\cdot)$和$f(\cdot)$非常接近。函数$\hat{f}(\cdot)$具备非常好的性质—可以通过采样获得它的随机梯度[Flaxman et al., 2005, Lemma 2.1]。

引理4.1. *令$\mathbf{u}$表示均匀分布在单位球面$\mathbb{S}$内的随机变量，我们有*

$$\mathrm{E}_{\mathbf{u}\in\mathbb{S}}\left[f(\mathbf{w}+\delta\mathbf{u})\mathbf{u}\right] = \frac{\delta}{d}\nabla\hat{f}(\mathbf{w}).$$

依据引理4.1，我们知道$\frac{d}{\delta}f(\mathbf{w}+\delta\mathbf{u})\mathbf{u}$是函数$\hat{f}(\cdot)$在$\mathbf{w}$处的随机梯度。同时，当$\delta$很小时，$\hat{f}(\mathbf{w}) \approx f(\mathbf{w})$。因此，当$\delta$很小时，可以用$\frac{d}{\delta}f(\mathbf{w}+\delta\mathbf{u})\mathbf{u}$来近似函数$f(\cdot)$在$\mathbf{w}$处的梯度，也就可以执行梯度下降算法。

依据上述思路设计的在线算法流程如下：

1: 初始化$\mathbf{z}_1 = 0 \in \mathcal{W}$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: 学习器随机选择一个单位向量$\mathbf{u}_t \in \mathbb{R}^d$
4: 学习器选择决策$\mathbf{w}_t = \mathbf{z}_t + \delta\mathbf{u}_t \in \mathcal{W}$; 同时，对手选择一个损失函数$f_t(\cdot) : \mathcal{W} \mapsto \mathbb{R}$
5: 学习器遭受损失$f_t(\mathbf{w}_t)$，并更新$\mathbf{z}_t$：

$$\mathbf{z}_{t+1} = \Pi_{(1-\alpha)\mathcal{W}}\left[\mathbf{z}_t - \eta f_t(\mathbf{w}_t)\mathbf{u}_t\right]$$

6: **end for**

对上述算法，我们有几点说明：

1. 首先，算法引入了一个辅助向量序列$\mathbf{z}_1, \ldots$，并且最后在变量$\mathbf{z}_t$上执行了梯度下降。这是因为根据引理4.1，$\frac{d}{\delta}f_t(\mathbf{w}_t)\mathbf{u}_t$是$\hat{f}(\cdot)$在$\mathbf{z}_t$处的随机梯度，并不是在$\mathbf{w}_t$处的随机梯度。
2. 在最后执行投影操作时，算法将中间解投影到了$(1-\alpha)\mathcal{W}$，而不是$\mathcal{W}$。这样做的目的是使得$\mathbf{z}_{t+1}$处于定义域$\mathcal{W}$的内部，从而保证$\mathbf{w}_{t+1} = \mathbf{z}_{t+1} + \delta\mathbf{u}_{t+1}$依然属于$\mathcal{W}$。

在健忘设定下，也就是函数序列$f_1, \ldots, f_T$和学习器的决策无关时，可以证明上述算法从期望意义上达到了$O(T^{3/4})$的遗憾界[Flaxman et al., 2005, Theorem 3.3]。

定理4.5 (凸赌博机遗憾界)**.** *假设下面的条件成立：*

- *函数$f_1, \ldots, f_T : \mathcal{W} \mapsto [-C, C]$是固定的函数序列；*
- *每一个损失函数f_t都是G-Lipschitz连续的，即*

$$|f_t(\mathbf{w}) - f_t(\mathbf{w}')| \le G\|\mathbf{w} - \mathbf{w}'\|,\ \forall \mathbf{w}, \mathbf{w}' \in \mathcal{W}.$$

- *定义域$\mathcal{W}$满足$r\mathbb{B} \subseteq \mathcal{W} \subseteq R\mathbb{B}$。*

那么，当T足够大时，令$\eta = \frac{R}{C\sqrt{T}}$、$\alpha = \frac{\delta}{r}$、$\delta = T^{-1/4}\sqrt{\frac{RdCr}{3(Lr+C)}}$，可以得到

$$\mathrm{E}\left[\sum_{t=1}^{T} f_t(\mathbf{w}_t)\right] - \min_{\mathbf{w}\in\mathcal{W}} \sum_{t=1}^{T} f_t(\mathbf{w}) \leq 2T^{3/4}\sqrt{3RdC\left(G + \frac{C}{r}\right)}.$$

可以证明，在非健忘情况下，我们同样可以取得$O(T^{3/4})$的遗憾界。当损失函数具有额外的性质，比如平滑和强凸，遗憾界还可以进一步提升。在赌博机设定下，针对平滑损失函数的工作可以参考文献[Saha and Tewari, 2011]；针对强凸函数、平滑且强凸函数的工作可以参考文献[Agarwal et al., 2010]。此外，如果学习器在每一轮迭代能够查询函数值多次，遗憾界也可以得到提升[Agarwal et al., 2010]。

参考文献

Y. Abbasi-yadkori, D. Pál, and C. Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems 24*, pages 2312–2320, 2011.

J. Abernethy, P. L. Bartlett, A. Rakhlin, and A. Tewari. Optimal stragies and minimax lower bounds for online convex games. In *Proceedings of the 21st Annual Conference on Learning Theory*, pages 415–423, 2008a.

J. Abernethy, E. Hazan, and A. Rakhlin. Competing in the dark: An efficient algorithm for bandit linear optimization. In *Proceedings of the 21st Annual Conference on Learning*, pages 263–274, 2008b.

A. Agarwal, O. Dekel, and L. Xiao. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *Proceedings of the 23rd Annual Conference on Learning Theory*, pages 28–40, 2010.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002a.

P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002b.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

P. L. Bartlett, O. Bousquet, and S. Mendelson. Local rademacher complexities. *The Annals of Statistics*, 33(4):1497–1537, 2005.

P. L. Bartlett, V. Dani, T. P. Hayes, S. M. Kakade, A. Rakhlin, and A. Tewari. High-probability regret bounds for bandit online linear optimization. In *Proceedings of the 21st Annual Conference on Learning*, pages 335–341, 2008.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

N. Cesa-bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. In *Advances in Neural Information Processing Systems 14*, pages 359–366, 2002.

V. Dani, T. P. Hayes, and S. M. Kakade. Stochastic linear optimization under bandit feedback. In *Proceedings of the 21st Annual Conference on Learning*, pages 355–366, 2008a.

V. Dani, T. P. Hayes, and S. M. Kakade. The price of bandit information for online optimization. In *Advances in Neural Information Processing Systems 20*, pages 345–352, 2008b.

A. Daniely, A. Gonen, and S. Shalev-Shwartz. Strongly adaptive online learning. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1405–1411, 2015.

S. Filippi, O. Cappe, A. Garivier, and C. Szepesvári. Parametric bandits: The generalized linear case. In *Advances in Neural Information Processing Systems 23*, pages 586–594, 2010.

A. D. Flaxman, A. T. Kalai, and H. B. McMahan. Online convex optimization in the bandit setting: Gradient descent without a gradient. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 385–394, 2005.

G. H. Golub and C. F. Van Loan. *Matrix computations, 3rd Edition*. Johns Hopkins University Press, 1996.

E. Hazan and S. Kale. Beyond the regret minimization barrier: an optimal algorithm for stochastic strongly-convex optimization. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 421–436, 2011.

E. Hazan and S. Kale. Beyond the regret minimization barrier: Optimal algorithms for stochastic strongly-convex optimization. *Journal of Machine Learning Research*, 15:2489–2512, 2014.

E. Hazan and C. Seshadhri. Adaptive algorithms for online decision problems. *Electronic Colloquium on Computational Complexity*, 88, 2007.

E. Hazan and C. Seshadhri. Efficient learning algorithms for changing environments. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 393–400, 2009.

E. Hazan, A. Agarwal, and S. Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

B.-J. Hou, L. Zhang, and Z.-H. Zhou. Learning with feature evolvable streams. In *Advances in Neural Information Processing Systems 30*, 2017.

K.-S. Jun, A. Bhargava, R. Nowak, and R. Willett. Scalable generalized linear bandits: Online computation and hashing. In *Advances in Neural Information Processing Systems 30*, pages 99–109, 2017.

S. M. Kakade and A. Tewari. On the generalization ability of online strongly convex programming algorithms. In *Advances in Neural Information Processing Systems 21*, pages 801–808, 2009.

S. M. Kakade, S. Shalev-Shwartz, and A. Tewari. Efficient bandit algorithms for online multiclass prediction. In *Proceedings of the 25th International Conference on Machine Learning*, pages 440–447, 2008.

V. Koltchinskii. *Oracle Inequalities in Empirical Risk Minimization and Sparse Recovery Problems*. Springer, 2011.

H. J. Kushner and G. G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, second edition, 2003.

M. Mahdavi, L. Zhang, and R. Jin. Lower and upper bounds on the generalization of stochastic exponentially concave optimization. In *Proceedings of the 28th Annual Conference on Learning Theory*, 2015.

A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

Y. Nesterov. *Introductory lectures on convex optimization: a basic course*, volume 87 of *Applied optimization*. Kluwer Academic Publishers, 2004.

Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

Y. Nesterov. Gradient methods for minimizing composite objective function. Core discussion papers, 2007.

Y. Nesterov. Random gradient-free minimization of convex functions. Core discussion papers, 2011.

A. Rakhlin, O. Shamir, and K. Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. In *Proceedings of the 29th International Conference on Machine Learning*, pages 449–456, 2012.

H. Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

A. Saha and A. Tewari. Improved regret guarantees for online smooth convex optimization with bandit feedback. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, pages 636–642, 2011.

S. Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: primal estimated sub-gradient solver for SVM. In *Proceedings of the 24th International Conference on Machine Learning*, pages 807–814, 2007.

S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan. Stochastic convex optimization. In *Proceedings of the 22nd Annual Conference on Learning Theory*, 2009.

N. Srebro, K. Sridharan, and A. Tewari. Optimistic rates for learning with a smooth loss. *ArXiv e-prints*, arXiv:1009.3896, 2010.

K. Sridharan, S. Shalev-shwartz, and N. Srebro. Fast rates for regularized objectives. In *Advances in Neural Information Processing Systems 21*, pages 1545–1552, 2009.

P. Tseng. On acclerated proximal gradient methods for convex-concave optimization. Technical report, University of Washington, 2008.

T. van Erven and W. M. Koolen. Metagrad: Multiple learning rates in online learning. In *Advances in Neural Information Processing Systems 29*, pages 3666–3674, 2016.

V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

G. Wang, S. Lu, and L. Zhang. Adaptivity and optimality: A universal algorithm for online convex optimization. In *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence*, 2019.

L. Zhang and Z.-H. Zhou. Stochastic approximation of smooth and strongly convex functions: Beyond the $O(1/T)$ convergence rate. In *Proceedings of the 32nd Annual Conference on Learning Theory*, 2019.

L. Zhang, T. Yang, R. Jin, Y. Xiao, and Z.-H. Zhou. Online stochastic linear optimization under one-bit feedback. In *Proceedings of the 33rd International Conference on Machine Learning*, 2016.

L. Zhang, T. Yang, and R. Jin. Empirical risk minimization for stochastic convex optimization: $O(1/n)$- and $O(1/n^2)$-type of risk bounds. In *Proceedings of the 30th Annual Conference on Learning Theory*, pages 1954–1979, 2017a.

L. Zhang, T. Yang, J. Yi, R. Jin, and Z.-H. Zhou. Improved dynamic regret for non-degenerate functions. In *Advances in Neural Information Processing Systems 30*, pages 732–741, 2017b.

L. Zhang, S. Lu, and Z.-H. Zhou. Adaptive online learning in dynamic environments. In *Advances in Neural Information Processing Systems 31*, pages 1330–1340, 2018a.

L. Zhang, T. Yang, R. Jin, and Z.-H. Zhou. Dynamic regret of strongly adaptive methods. In *Proceedings of the 35th International Conference on Machine Learning*, 2018b.

L. Zhang, T.-Y. Liu, and Z.-H. Zhou. Adaptive regret of convex and smooth functions. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning*, pages 928–936, 2003.