适应梯度变化的普适在线凸优化算法

刘朗麒 张利军

(南京大学计算机软件新技术全国重点实验室 南京 210023)

(南京大学人工智能学院 南京 210023)

摘 要 普适在线凸优化算法能够自动适应多类损失函数并进行优化,这使得用户无须自行判别损失函数的类型,降低了在线凸优化技术的使用门槛.虽然现有的普适算法对于多类损失函数的理论保障均达到极小极大最优,但是它们难以针对一般凸函数获得问题相关的理论保障.为解决该问题,本文提出的 UAGV 算法不仅能够自动适应一般凸与强凸的损失函数,同时首次在平滑条件下对于一般凸损失函数保障了梯度变化界,即能够在损失函数梯度变化缓慢时取得更好的性能.算法整体采用元算法-专家算法的二层结构,在顶层本文创新性地采用具有乐观项的元算法,并针对梯度变化界的形式设计替代损失函数与乐观项,使得其在结合底层专家算法时能够获得相应保障.在多个数据集上的实验结果表明,UAGV 算法对于平滑一般凸函数产生的遗憾整体小于现有普适算法,在部分数据集上遗憾减小的幅度超过 14%.

关键词 在线凸优化;普适算法;平滑;梯度变化;问题相关界;乐观项

中图法分类号 TP181 **DOI 号** 10.11897/SP.J.1016.2024.02629

Universal Online Convex Optimization Algorithm with Adaptivity to Gradient-Variation

LIU Lang-Qi ZHANG Li-Jun

(*National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing* 210023)

(*School of Artificial Intelligence, Nanjing University, Nanjing* 210023)

Abstract Contrasting with online convex optimization algorithms designed for specific function types, universal algorithms are able to automatically adapt to various loss functions. This capability eliminates the need for users to correctly classify the type of loss function, thereby lowering the barrier to employing online convex optimization techniques. Previous studies have proposed algorithms that provide minimax optimal theoretical guarantees for several types of loss functions. However, they struggle to attain tighter theoretical guarantees that are correlated with the structure of the problem for general convex functions. To address this issue, we introduce the Universal Online Convex Optimization Algorithm with Adaptivity to Gradient-Variation (UAGV). This novel algorithm automatically adapts to both general convex and strongly convex loss functions. Furthermore, under the smooth condition, UAGV enjoys the gradient-variation bound for general convex loss functions, which is a problem-dependent bound. The algorithm adopts a two-layered structure, with a meta-algorithm in the upper layer and several expert-algorithms in the lower layer. We innovatively adopt the meta-algorithm with an optimism term, which can be interpreted as a prediction of the loss vector. When the optimism term closely matches the loss vector, the meta-algorithm achieves small regret. Thus, we carefully design the surrogate loss function and the optimism term according to the form of gradient-variation bound, enhancing the meta-algorithm's

收稿日期:2023-08-08;在线发布日期:2024-07-12.本课题得到国家自然科学基金(62122037,U23A20382)资助.**刘朗麒**,硕士研究生,中国计算机学会(CCF)会员,主要研究方向为在线学习、数据挖掘.E-mail:liulq@lamda.nju.edu.cn.**张利军**(通信作者),博士,教授,中国计算机学会(CCF)会员,主要研究领域为机器学习、优化.E-mail:zhanglj@lamda.nju.edu.cn.

ability of combining decisions generated by expert algorithms and helping to obtain the corresponding theoretical guarantee. Experimental results from several datasets indicate that the UAGV algorithm can effectively track the best expert-algorithm, and its optimization results for smooth general convex functions outperform those of existing universal algorithms. Specifically, the regret of UAGV is over 14% smaller than that of existing algorithms on certain datasets.

Keywords online convex optimization; universal algorithm; smoothness; gradient-variation; problem-dependent bound; optimism

1 引 言

随着通信技术、互联网技术与相关应用的快速发展,人类社会每天都在产生巨量的数据[1],其中如传感器数据、互联网数据、业务数据等很多数据呈现出流式的特征[2].为了能够及时分析与处理这类流式数据,研究者们往往不能将这些数据离线存储,而是需要在数据到来后直接在内存中进行计算[3].这一需求催生了机器学习领域中对于在线学习算法的研究.与传统的批量式学习相比,在线学习直接利用流式数据进行实时训练,在学习过程中模型能够根据反馈进行自动更新,时间与空间复杂度均较低,更加适用于大数据场景.

在线凸优化(Online Convex Optimization,OCO)是一种流行的在线学习框架,研究者们能够利用其建模很多具体的在线学习任务,例如在线回归、在线排序和在线分类[4].在线凸优化从博弈的视角描述在线学习过程:学习者与环境作为博弈的双方进行共计 T 轮的博弈;在第 t 轮中,学习者首先从 d 维的凸决策集 $\mathcal{X}\subseteq\mathbb{R}^d$ 中选择一个决策 $\boldsymbol{x}_t$ 并提交;与此同时,环境选择并公布一个凸的损失函数 $f_t:\mathcal{X}\mapsto\mathbb{R}$,学习者遭受损失 $f_t(\boldsymbol{x}_t)$,并根据损失函数来调整后续决策.为了更加形象地理解这一过程,我们以在线分类任务为例进行说明.在每一轮中,我们会得到一个样本的特征及其标签,此时我们可以将决策 $\boldsymbol{x}_t$ 视为一个以特征为输入的模型,同时我们可以根据模型输出与标签构造损失函数 $f_t(\boldsymbol{x}_t)$.随着新样本不断地到来,在线分类模型能够根据当前样本迭代地进行更新.衡量在线学习性能的准则是遗憾[5],其定义为学习者在学习过程中的累计损失与最优固定决策的累计损失之差.学习者的目标是最小化遗憾.

学术界针对不同类型的在线损失函数已经提出高效的在线凸优化算法,其中我们主要关注一般凸和强凸函数.当损失函数为一般凸函数时,采用在线梯度下降(Online Gradient Descent,OGD)算法能保障 $\mathcal{O}(\sqrt{T})$ 的遗憾界[5];当损失函数为 λ-强凸函数时,采用 SC-OGD 算法(SC 是强凸 Strongly Convex 的缩写,该算法通过在 OGD 算法的基础上调整步长而实现)能够保障 $\mathcal{O}(\log T/\lambda)$ 的遗憾界[6].这两个遗憾界均与理论下界相匹配,即实现了极小极大最优(Minimax Optimal)[7].需要指出的是,理论下界仅关注问题中最困难的情况,例如当损失函数具有对抗性时;而在很多现实问题中损失函数的性质会相对温和,若在线算法能利用这些性质则可以减小学习过程中的遗憾.对于这类情况,研究者们使用关于损失函数的累积量来描述在线函数所可能具有的性质,并希望用这些累积量来替换遗憾界中的 T 以获得问题相关的遗憾界[8-10].一种常见的问题相关界是梯度变化界(Gradient-Variation Bound)[8],其关注的累积量 V_T 是相邻两轮间损失函数梯度 $\nabla f_t(\cdot)$ 的最大变化量之和,即

$$V_T=\sum_{t=1}^{T}\max_{\boldsymbol{x}\in\mathcal{X}}\left\|\nabla f_t(\boldsymbol{x})-\nabla f_{t-1}(\boldsymbol{x})\right\|^2 \tag{1}$$

以优化平滑一般凸且梯度有界的损失函数为例,采用 OGD 算法能够保障 $\mathcal{O}(\sqrt{T})$ 的极小极大最优界,而采用 OEGD 算法[8]能够保障 $\mathcal{O}(\sqrt{V_T})$ 的梯度变化界.由于损失函数梯度有界,因此在最坏情况下有 $V_T=\mathcal{O}(T)$,即 $\mathcal{O}(\sqrt{V_T})$ 的遗憾界仍为极小极大最优;而当损失函数梯度变化缓慢时,梯度变化界将会较极小极大最优界更紧.

虽然上述在线凸优化算法具有较好的理论保障,但是在实际使用时对于用户的专业知识水平要求较高.首先,这些算法仅针对某一特定类型的损失函数,对于不同类型的函数需要选用对应的算法,否则将产生次优的结果,因此用户需要判断损失函数的类型.其次,部分算法要求输入关于损失函数的参数来调节步长,例如 SC-OGD 算法需要输入强凸函数

的量度 λ，而这些参数需要由用户进行估计. 为了降低在线凸优化技术的使用门槛，研究者们开始关注能够适应多种函数类型且无须问题先验知识的普适算法[11]. 在这一领域，比较具有代表性的工作包括 MetaGrad 算法[12-13]、Maler 算法[14]和 USC 算法[15]. 目前最先进的 USC 算法能够对于强凸和一般凸函数均实现极小极大最优界保障，同时对于强凸函数具有梯度变化界保障.

我们注意到，USC 算法对于现实问题中最为常见一般凸函数并没有提供梯度变化界保障，这使得 USC 算法的理论保障尚不令人满意：很多常用的损失函数满足平滑一般凸但不满足强凸，例如均方误差和交叉熵损失，此时 USC 算法无法在损失函数梯度变化缓慢时获得更紧的遗憾界. 针对这一局限性，我们重点关注如何设计对于一般凸函数具有梯度变化界保障的普适算法. 本文提出的 UAGV(Universal Online Convex Optimization Algorithm with Adaptivity to Gradient-Variation)算法借鉴了此前工作[12-15]中所采用的专家学习框架，该框架包含专家算法和元算法两部分. 专家算法由多种不同的优化算法组成，其中包括具有梯度变化界保障的在线凸优化算法，它们并行地运行. 元算法需要结合专家算法的决策来产生普适算法的决策，而如何使得元算法具有梯度变化界保障是本问题的难点. 为此，我们创新地采用具有乐观项的 MSMWC 算法[16]作为元算法，并针对梯度变化界的形式设计替代损失函数与乐观项. 通过理论分析，我们证明此时 UAGV 算法整体对于平滑一般凸函数具有梯度变化界保障. 最后，我们在多个数据集上进行实验，实验结果验证了 UAGV 算法的有效性.

综上所述，本文主要贡献如下：

(1) 为解决现有普适算法对于一般凸函数不具有问题相关界保障的问题，本文提出了采用专家学习框架的 UAGV 算法. 对于平滑一般凸函数，我们成功使 UAGV 算法获得梯度变化界，其遗憾界保障为 $\mathcal{O}(\sqrt{V_T \log T})$，当 $V_T \ll T$ 时将获得比极小极大最优界更紧的遗憾界.

(2) UAGV 算法也能够自动适应 λ-强凸函数，保障了 $\mathcal{O}(\log T/\lambda)$ 的遗憾界. 该遗憾界与极小极大最优界相匹配.

(3) 在多个标准数据集上的实验结果表明，当损失函数为平滑一般凸函数时，在运行过程中 UAGV 算法的遗憾整体小于现有普适算法；当损失函数为强凸函数时，在运行过程中 UAGV 算法与现有普适算法性能相当.

本文第 2 节介绍相关工作；第 3 节介绍在线凸优化问题中的一些概念定义与常用的假设；第 4 节详细介绍提出的 UAGV 算法，并分析其具有的理论保障；第 5 节中给出实验的细节与结果分析；第 6 节对本文进行总结.

2　相关工作

在本节中，我们首先将在线凸优化领域的相关工作分为传统在线凸优化算法和普适在线凸优化算法两类，并分别进行介绍. 接着，我们对于设计算法时使用的乐观项技术及相关算法作介绍.

2.1　传统在线凸优化算法

在线凸优化算法旨在最小化遗憾，其定义为 T 轮中在线算法决策 $\boldsymbol{x}_t$ 的累计损失与最优固定决策的累计损失之差，即

$$\mathrm{REG}_T = \underbrace{\sum_{t=1}^{T} f_t(\boldsymbol{x}_t)}_{\text{学习过程累计损失}} - \underbrace{\min_{\boldsymbol{x}\in\mathcal{X}} \sum_{t=1}^{T} f_t(\boldsymbol{x})}_{\text{最优决策累计损失}} \tag{2}$$

其中，$f_t(\cdot)$ 是第 t 轮中的损失函数. 研究者们主要对于三种类型的损失函数展开研究，分别是一般凸函数、λ-强凸函数和 α-指数凹函数. 对于一般凸函数，采用第 t 轮步长为 $\mathcal{O}(1/\sqrt{t})$ 的 OGD 算法能够保障 $\mathcal{O}(\sqrt{T})$ 的遗憾界[5]；对于 λ-强凸函数，采用第 t 轮步长为衰减更快的 $\mathcal{O}(1/(\lambda t))$ 的 SC-OGD 算法能够保障 $\mathcal{O}(\log T/\lambda)$ 的遗憾界[6]；对于 α-指数凹函数，采用在线牛顿法(Online Newton Step，ONS)能够保障 $\mathcal{O}(d\log T/\alpha)$ 的遗憾界[17]，其中 d 为决策的维度. 需要指出的是，SC-OGD 算法和 ONS 算法分别需要在提前得知强凸量度 λ 和指数凹量度 α 之后才能合适地调整算法步长并获得相应的理论保障. 通过对于理论下界进行分析[7]，以上三种算法分别对于对应类型的函数实现了极小极大最优，即在它们的表现在最坏情况下是无法提升的. 但在很多现实问题中，环境所公布的损失函数并不会像最坏情况下一样具有对抗性，而是存在与问题相关的良好性质. 如果算法能够利用这些性质，则可以获得比极小极大最优界更紧的遗憾界. 基于这个考量，研究者们提出了一些具有问题相关遗憾界的算法[8-10,18]. 这些算法所保障的遗憾界不再仅依赖于 T，而是与一个在 T 轮中累积的量相关. 在最坏情况下，这些累积量会

退化至$\mathcal{O}(T)$，此时问题相关遗憾界依然实现了极小极大最优；而当问题所具有的性质使得累积量的量级小于$\mathcal{O}(T)$时，算法就能获得更紧的遗憾界.

一个常见的问题相关界是梯度变化界[8,19]，此时遗憾界所相关的累积量是相邻两轮损失函数梯度变化的最大值，见式(1). 对于平滑一般凸的损失函数，OEGD 算法[8]保障了$\mathcal{O}(\sqrt{V_T})$的遗憾界. 如果相邻两轮损失函数的梯度在决策集内的变化很温和，那么具有梯度变化界的算法会自动获得更紧的遗憾界. 另一个常见的问题相关界是小损失界(Small-Loss Bound)[18,20-21]. 小损失界的意思是，算法所保障的遗憾界与最优决策在 T 轮中的累计损失相关，即

$$L_T^* = \min_{\boldsymbol{x}\in\mathcal{X}} \sum_{t=1}^{T} f_t(\boldsymbol{x}) \tag{3}$$

当损失函数非负且平滑一般凸时，SOGD 算法[20]保障了$\mathcal{O}(\sqrt{L_T^*})$的遗憾界. 如果最优决策的累计损失很小，那么具有小损失界保障的算法会自动获得更紧的遗憾界. 以上两种问题相关界需要在损失函数平滑的条件下才能获得，其中小损失界还额外要求损失函数非负.

2.2 普适在线凸优化算法

尽管传统在线凸优化算法存在丰富的研究成果，为了获得较好的性能，用户需要判别损失函数的类型以选择对应算法，同时可能还需要提前估计损失函数的量度以设置算法参数. 这要求用户具有一定的专业领域知识，拔高了使用在线凸优化技术的门槛. 普适算法的研究初衷正是减少人工的分析与预测，使得优化算法能够直接用于具体问题.

研究者们对于在线凸优化领域中普适算法的探索始于 AOGD 算法[11]. AOGD 算法能够自动适应一般凸和强凸函数，分别能够保障$\mathcal{O}(\sqrt{T})$和$\mathcal{O}(\log T)$的遗憾界. 但是，AOGD 算法要求输入每一轮中损失函数的强凸量度，而能够获得该信息是一个过于强的假设，在现实情况中难以成立.

为了使普适算法无需提前获得损失函数的相关信息，Van Erven 和 Koolen[12]在专家学习[22]框架下提出了 MetaGrad 算法. 专家学习框架具有两层结构，由底层中同时运行的多个在线凸优化算法(专家算法)和顶层中基于历史信息综合专家决策的算法(元算法)构成. 此时元算法所求解的问题可以看作是在线凸优化的一个特例：在第 t 轮中，元算法首先选择对于 N 个专家的权重 $\boldsymbol{w}_t \in \Delta_N$，然后得到一个损失向量 $\ell_t \in \mathbb{R}^N$ 并遭受线性损失$\langle \boldsymbol{w}_t, \ell_t \rangle$，算法希望能够最小化累计损失. 在 MetaGrad 算法中，每个专家算法并不是直接优化损失函数 $f_t(\cdot)$，而是运行带有特定学习率 η 的 ONS 算法来优化替代损失函数

$$\ell_t^{\eta}(\boldsymbol{x}) = -\eta\langle \nabla f_t(\boldsymbol{x}_t), \boldsymbol{x}_t - \boldsymbol{x}\rangle + \eta^2\langle \nabla f_t(\boldsymbol{x}_t), \boldsymbol{x}_t - \boldsymbol{x}\rangle^2 \tag{4}$$

它们产生的决策采用一个类似指数加权平均(Exponentially Weighted Average)[22]的元算法来进行结合. 通过同时运行$\mathcal{O}(\log T)$个具有不同学习率的专家算法来优化式(4)，MetaGrad 算法在无须输入 α-指数凹函数的量度 α 时仍然能够保障$\mathcal{O}(d\log T/\alpha)$的遗憾界，其中 d 为决策的维度. 同时，MetaGrad 算法对于一般凸函数具有$\mathcal{O}(\sqrt{T\log\log T})$的遗憾界保障. 在后续的工作中，他们进一步提出了计算复杂度更低的 $\text{MetaGrad}_{\text{freq}}$ 算法[13]. 但是，MetaGrad 与 $\text{MetaGrad}_{\text{freq}}$ 算法没有显式地支持强凸函数，因此只能将强凸损失函数弱化为指数凹函数进行处理，获得的$\mathcal{O}(d\log T)$遗憾界与极小极大最优界有着 d 倍的差距.

为了使普适算法对一般凸函数和强凸函数均能保障极小极大最优界，Wang 等人[14]提出的 Maler 算法对于一般凸与强凸函数也采用了类似式(4)的构造重新设计了替代损失函数，并分别使用 1 个一般凸专家和$\mathcal{O}(\log T)$个具有不同学习率的强凸专家进行优化. Maler 算法能够在 MetaGrad 算法对于 α-指数凹函数的$\mathcal{O}(d\log T/\alpha)$遗憾界之外，同时保障对于一般凸函数的$\mathcal{O}(\sqrt{T})$遗憾界和对于 λ-强凸函数的$\mathcal{O}(\log T/\lambda)$遗憾界，均实现了极小极大最优. Wang 等人[21]接着探索了普适算法在损失函数平滑条件下的遗憾界，通过调整三种替代损失函数来保障问题相关的小损失界.

Zhang 等人[15]随后指出，如果对于不同类型的函数分别设计专家算法所优化的替代损失函数，会使得普适算法难以获得除小损失界以外的问题相关遗憾界，如梯度变化界. 因此，在他们提出的 USC 算法中，专家算法直接在原损失函数 $f_t(\cdot)$ 上进行优化，而元算法采用结构简单的线性损失

$$L_t(\boldsymbol{x}) = \langle \nabla f_t(\boldsymbol{x}_t), \boldsymbol{x} - \boldsymbol{x}_t\rangle \tag{5}$$

来衡量专家决策 $\boldsymbol{x}$ 的好坏. 此时，令 $\boldsymbol{x}_t$ 和 $\boldsymbol{x}_t^i$ 分别表示第 t 轮中元算法和第 i 个专家算法的决策，那么可以将普适算法的遗憾界分解为元界与专家界，即

$$\sum_{t=1}^{T} f_t(\boldsymbol{x}_t) - \min_{\boldsymbol{x}\in\mathcal{X}} \sum_{t=1}^{T} f_t(\boldsymbol{x}) =$$

$$\underbrace{\sum_{t=1}^{T} f_t(\boldsymbol{x}_t)-\sum_{t=1}^{T} f_t(\boldsymbol{x}_t^i)}_{\text{元界}}+\underbrace{\sum_{t=1}^{T} f_t(\boldsymbol{x}_t^i)-\min_{\boldsymbol{x}\in\mathcal{X}}\sum_{t=1}^{T} f_t(\boldsymbol{x})}_{\text{专家界}} \tag{6}$$

其中元界为普适算法与专家算法累计损失之差的界，专家界为专家算法与最优决策累计损失之差的界. 如果专家算法包含多个优化不同类型损失函数的传统在线凸优化算法，那么专家界就能够适应多种类型的函数. 为了使 USC 算法的元界也能够适应多种类型的函数，他们要求元算法对于额外损失具有二阶界[23]，即

$$\sum_{t=1}^{T} l_t-\ell_t^i=O\left(\sqrt{\sum_{t=1}^{T}(l_t-\ell_t^i)^2}\right) \tag{7}$$

其中 l_t 是元算法的损失，ℓ_t^i是第 i 个专家算法的损失，两者均基于线性损失(5)进行设计. 在损失函数平滑的条件下，USC 算法对于指数凹函数和强凸函数能在 Wang 等人[21]结果的基础上额外获得梯度变化界，即$\mathcal{O}(d\log V_T/\alpha)$和$\mathcal{O}(\log V_T/\lambda)$的遗憾界保障，其中 V_T 的定义见式(1). 但是，AOGD、MetaGrad、Maler 和 USC 算法均无法对于最常见的一般凸函数获得梯度变化界，这使得它们在很多情况下难以利用问题相关性质得到更小的遗憾.

2.3　乐观项技术及相关算法

为了保障在线凸优化算法具有问题相关界，一些过往工作使用了乐观项技术[8,19]，其主要思想是在迭代过程中对于损失函数的相关信息进行预测，并在更新时利用这些预测信息. Rakhlin 等人[24]将这一技术总结为乐观镜像下降(Optimistic Mirror Descent)框架. 我们继续沿用描述在线凸优化问题的符号来进行介绍. 基于单步梯度下降的算法(如 OGD 算法和 SC-OGD 算法)的决策更新方式为

$$\boldsymbol{x}_{t+1}=\Pi_{\mathcal{X}}[\boldsymbol{x}_t-\eta_t\langle\boldsymbol{x}_t,\nabla f_t(\boldsymbol{x}_t)\rangle],$$

其中 η_t表示第 t 轮更新的步长，$\Pi_{\mathcal{X}}[\boldsymbol{x}]$表示将决策 $\boldsymbol{x}$ 投影到域$\mathcal{X}$内. 与单步梯度下降不同，采用乐观镜像下降的算法需要将更新拆分为两步：

$$\begin{aligned}\boldsymbol{x}_t&=\arg\min_{\boldsymbol{x}\in\mathcal{X}}\eta_t\langle\boldsymbol{x},\boldsymbol{m}_t\rangle+\mathcal{D}_\psi(\boldsymbol{x},\boldsymbol{x}_t'),\\ \boldsymbol{x}_{t+1}'&=\arg\min_{\boldsymbol{x}\in\mathcal{X}}\eta_t\langle\boldsymbol{x},\nabla f_t(\boldsymbol{x}_t)\rangle+\mathcal{D}_\psi(\boldsymbol{x},\boldsymbol{x}_t')\end{aligned} \tag{8}$$

其中，$\boldsymbol{m}_t$ 是第 t 轮的乐观项，$\{\boldsymbol{x}_t'\}$是用于计算$\{\boldsymbol{x}_t\}$的辅助序列，$\mathcal{D}_\psi(\boldsymbol{x},\boldsymbol{y})\triangleq\psi(\boldsymbol{x})-\psi(\boldsymbol{y})-\langle\nabla\psi(\boldsymbol{y}),\boldsymbol{x}-\boldsymbol{y}\rangle$是根据可微凸函数 $\psi(\cdot)$定义的 Bregman 散度. 根据 Rakhlin 等人的分析，当取$\mathcal{D}_\psi(\boldsymbol{x},\boldsymbol{y})=\frac{1}{2}\|\boldsymbol{x}-\boldsymbol{y}\|_2^2$并设置最优的步长 η_t时，采用式(8)进行更新的算法具有$\mathcal{O}\left(\sqrt{\sum_{t=1}^{T}\|\nabla f_t(\boldsymbol{x}_t)-\boldsymbol{m}_t\|_2^2}\right)$的遗憾界保障. 此时，乐观项 $\boldsymbol{m}_t$ 对于梯度 $\nabla f_t(\boldsymbol{x}_t)$的预测越准确，则该算法的遗憾界将越小.

由于专家学习框架中元算法所求解的问题是一个特殊的在线凸优化问题，因此也可以获得含有乐观项的理论保障. Syrgkanis 等人[25]对于该问题提出的 Optimistic Hedge 算法可以看作是乐观镜像下降框架的一个实例，其中设置在线损失函数为 $f_t(\boldsymbol{w}_t)=\langle\boldsymbol{w}_t,\boldsymbol{\ell}_t\rangle$，并取$\mathcal{D}_\psi(\boldsymbol{x},\boldsymbol{y})=\mathrm{KL}(\boldsymbol{x},\boldsymbol{y})$，即 KL 散度. Optimistic Hedge 算法所具有的遗憾界保障为$\mathcal{O}\left(\sqrt{\sum_{t=1}^{T}\|\boldsymbol{\ell}_t-\boldsymbol{m}_t\|_\infty^2\ln N}\right)$，即此时乐观项$\boldsymbol{m}_t\in\boldsymbol{R}^N$所预测的是第 t 轮的损失向量 $\boldsymbol{\ell}_t\in\boldsymbol{R}^N$. Chen 等人[16]提出的 MSMWC 算法则对于乐观镜像下降框架进行修改，引入了可变学习率向量 $\boldsymbol{\eta}_t\in\mathbb{R}_{\geq 0}^N$ 与修正项 $\boldsymbol{a}_t\in\mathbb{R}^N$，同时对于每一轮迭代设置了不同的 Bregman 散度$\mathcal{D}_{\psi_t}(\cdot,\cdot)$，其流程见算法 1. MSMWC 算法能够保障对于 $\forall i\in[N]$均有$\mathcal{O}\left(\sqrt{\ln(NT)\sum_{t=1}^{T}(\ell_t^i-m_t^i)^2}\right)$的遗憾界.

算法 1.　MSMWC.

输入：运行轮数 T，各轮的乐观项$\{\boldsymbol{m}_t\}_{t=1}^T$，各轮的损失向量$\{\boldsymbol{\ell}_t\}_{t=1}^T$

输出：各轮的权重$\{\boldsymbol{x}_t\}_{t=1}^T$

1. 初始化 $\boldsymbol{w}_1'\in\Delta_N$
2. FOR $t=1,2,\cdots,T$
3. 　设置乐观项 $\boldsymbol{m}_t$
4. 　选择一个紧的凸决策子集 Ω_t 和学习率 $\boldsymbol{\eta}_t$
5. 　计算 $\boldsymbol{w}_t=\arg\min_{\boldsymbol{w}\in\Omega_t}\langle\boldsymbol{w},\boldsymbol{m}_t\rangle+\mathcal{D}_{\psi_t}(\boldsymbol{w},\boldsymbol{w}_t')$，其中$\psi_t(\boldsymbol{w})=\sum_{i=1}^{N}(w_i\ln w_i)/\eta_t^i$
6. 　将$\boldsymbol{w}_t$ 作为决策，获得损失向量 $\boldsymbol{\ell}_t$，并构建修正项 $\boldsymbol{a}_t$，其中 $a_t^i=32\eta_t^i(\ell_t^i-m_t^i)^2$
7. 　计算 $\boldsymbol{w}_{t+1}'=\arg\min_{\boldsymbol{w}\in\Omega_t}\langle\boldsymbol{w},\boldsymbol{\ell}_t+\boldsymbol{a}_t\rangle+\mathcal{D}_{\psi_t}(\boldsymbol{w},\boldsymbol{w}_t')$
8. END FOR

使用乐观项技术的关键和难点在于针对算法需求进行乐观项的设计. Chiang 等人[8]提出的 OEGD 算法可以理解为在乐观镜像下降框架中取 $\boldsymbol{m}_t=\nabla f_{t-1}(\boldsymbol{x}_{t-1})$，即将上一轮的梯度作为乐观项. 此时若损失函数的梯度$\nabla f_t(\cdot)$在相邻两轮间变化缓慢，则 OEGD 算法具有较小的遗憾界. Zhao 等人[19]则使用 Optimistic Hedge 算法来结合多个具有不同学习率的子算法，他们所设置的乐观项最终能够推导出算法整体的问题相关遗憾界保障. 但是他们的算法仅针对一般凸函数，而无法对于强凸函数获得极小极大最优界保障.

3 定义与常用假设

我们首先介绍凸函数、强凸函数以及平滑函数的定义[18,22].

定义 1. 一个函数 $f:\mathcal{X}\mapsto\mathbb{R}$ 若满足 $\forall \boldsymbol{x},\boldsymbol{y}\in\mathcal{X}$,

$$f(\boldsymbol{y})\geq f(\boldsymbol{x})+\langle\nabla f(\boldsymbol{x}),\boldsymbol{y}-\boldsymbol{x}\rangle \tag{9}$$

则称 $f(\cdot)$为凸函数.

定义 2. 一个函数 $f:\mathcal{X}\mapsto\mathbb{R}$ 若满足 $\forall \boldsymbol{x},\boldsymbol{y}\in\mathcal{X}$,

$$f(\boldsymbol{y})\geq f(\boldsymbol{x})+\langle\nabla f(\boldsymbol{x}),\boldsymbol{y}-\boldsymbol{x}\rangle+\frac{\lambda}{2}\|\boldsymbol{y}-\boldsymbol{x}\|^2 \tag{10}$$

其中 $\lambda>0$,则称 $f(\cdot)$为强凸函数,其强凸量度为 λ.

定义 3. 一个函数 $f:\mathcal{X}\mapsto\mathbb{R}$ 若满足 $\forall \boldsymbol{x},\boldsymbol{y}\in\mathcal{X}$,

$$\|\nabla f(\boldsymbol{x})-\nabla f(\boldsymbol{y})\|\leq H\|\boldsymbol{x}-\boldsymbol{y}\| \tag{11}$$

则称 $f(\cdot)$为 H-平滑函数.

接着,我们引入在线凸优化领域中两个常见的假设[5,12,15,19].

假设 1. 假设所有损失函数 $f_t:\mathcal{X}\mapsto\mathbb{R}$ 均梯度有界,即 $\forall t\in[T]$,

$$\max_{\boldsymbol{x}\in\mathcal{X}}\|\nabla f_t(\boldsymbol{x})\|\leq G \tag{12}$$

假设 2. 假设决策集 $\mathcal{X}\subseteq\mathbb{R}^d$ 直径有界,即

$$\max_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{X}}\|\boldsymbol{x}-\boldsymbol{y}\|\leq D \tag{13}$$

同时 $\mathbf{0}$ 属于决策集 $\mathcal{X}$.

4 方 法

过去研究者们提出的普适在线凸优化算法[12-15]主要采用了专家学习[22]这一经典框架,本文将遵循同样的思路. UAGV 算法由专家算法和元算法构成:N 个专家算法同时运行不同设置的传统在线凸优化算法来优化损失函数 $f_t(\cdot)$,并向元算法传递决策 $\boldsymbol{x}_t^1,\boldsymbol{x}_t^2,\cdots,\boldsymbol{x}_t^N$;元算法在标准化后的线性损失上优化权重 $\boldsymbol{w}_t\in\Delta_N$,同时加权结合专家算法上传的决策来产生算法整体的决策 $\boldsymbol{x}_t=\sum_{i=1}^{N}w_t^i\boldsymbol{x}_t^i$. UAGV 算法的整体结构如图 1 所示.

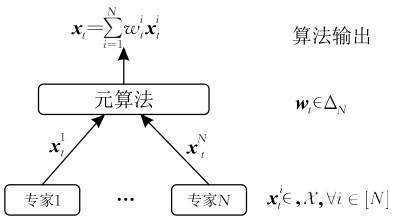


图 1 UAGV 算法的整体结构

根据式(6),当采用这样的算法结构时,普适算法的遗憾界可以分解为专家界与元界. 只有当专家界与元界均具有梯度变化界保障时,普适算法才能够适应梯度变化. 使得专家界具有梯度变化界保障是比较容易实现的,我们只需要令专家算法集合中包含具有梯度变化界保障的专家算法,如 OEGD 算法[8]即可. 但是,使得元界具有梯度变化界保障是一个难点.

Zhang 等人[15]通过使用具有二阶遗憾界保障的元算法来优化标准化后的线性损失,这使得 USC 算法能够适应强凸和指数凹函数并获得梯度变化界保障. 但是,他们的分析依赖于强凸与指数凹函数的性质,而对于性质更弱的一般凸函数无法得到该保障. 为了解决这个问题,我们提出了一个新型的元算法,并证明了该元算法在损失函数为最常见的一般凸函数时也能获得梯度变化界保障,从而使得 UAGV 算法适应梯度变化.

4.1 研究思路

Zhao 等人[19]的工作表明,如果在专家学习框架中使用具有乐观项的元算法,经过分析可以证明元算法具有梯度变化界保障. 但是,Zhao 等人[25]使用的元算法为 Optimistic Hedge 算法,它仅针对一般凸函数,而无法支持对于强凸函数的极小极大最优界保障. 因此我们考虑基于 Chen 等人[16]提出的 MSMWC 算法来设计元算法,采用该研究思路的原因有两点:一方面,MSMWC 算法具有乐观项,通过进行合理的设置,能够在损失函数为一般凸时推导出元算法的梯度变化界保障;另一方面,MSMWC 算法能提供式(7)形式的二阶界,从而支持元算法适应强凸函数.

我们在此引入 MSMWC 算法的遗憾界. 对于 $\forall t\in[T]$,当损失向量 $\boldsymbol{\ell}_t$ 与乐观项 $\boldsymbol{m}_t$ 的所有维度均有界时,通过合理地设置 MSMWC 算法的初始辅助量 $\boldsymbol{w}_1'$、凸决策子集 Ω_t 和学习率 $\boldsymbol{\eta}_t$,我们可以得到如下理论保障.

引理 1. 假设 $|\ell_t^i|,|m_t^i|\leq 1$ 对于 $\forall t\in[T],\forall i\in[N]$ 均成立. 设置 $\boldsymbol{w}_1'=\mathbf{1}/N,\Omega=\Omega_1=\cdots=\Omega_T=\left\{\boldsymbol{w}\in\Delta_N:w_i\geq\frac{1}{NT}\right\},\eta_t^i=\min\left\{\sqrt{\frac{\ln(NT)}{\sum_{s<t}(\ell_s^i-m_s^i)^2}},\frac{1}{64}\right\}$ 时,MSMWC 算法可以保障对于 $\forall j\in[N]$ 有

$$\sum_{t=1}^{T}\langle\boldsymbol{w}_t,\boldsymbol{\ell}_t\rangle-\ell_t^j=O\left(\ln(NT)+\sqrt{\ln(NT)\sum_{t=1}^{T}(\ell_t^j-m_t^j)^2}-\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2\right).$$

值得注意的是,引理 1 并不是对文献[16]中定

理的照搬，其证明见附录 A. 我们对于原先证明中被丢弃的一个负项进行仔细分析，使得 MSMWC 算法的遗憾界中出现一个额外的负项 $\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}'_t\|_1^2$，而它在推导元算法对于一般凸函数的梯度变化界保障时具有关键性作用.

4.2　算法实现

4.2.1　元算法设计

为了得到对一般凸函数有梯度变化界保障并且支持强凸函数的新型元算法，我们需要结合专家算法来设置元算法的损失向量 $\boldsymbol{\ell}_t$、元损失 l_t 和乐观项 $\boldsymbol{m}_t$.

我们基于线性损失来设计损失向量 $\boldsymbol{\ell}_t\in\mathbb{R}^N$ 和元损失 $l_t\in\mathbb{R}$，其值分别为

$$\ell_t^i=\frac{1}{3DG}\langle\nabla f_t(\boldsymbol{x}_t),\boldsymbol{x}_t^i\rangle,\ \forall i\in[N],$$

$$l_t=\frac{1}{3DG}\langle\nabla f_t(\boldsymbol{x}_t),\boldsymbol{x}_t\rangle \tag{14}$$

其中，$\boldsymbol{x}_t^i\in\mathcal{X}$ 是第 i 个专家算法所产生的决策；$\boldsymbol{x}_t=\sum_{i=1}^{N}w_t^i\boldsymbol{x}_t^i$ 是元算法所产生的决策，它通过以 $\boldsymbol{w}_t\in\Omega$ 的权重线性组合 N 个专家算法的决策 $\boldsymbol{x}_t^1,\boldsymbol{x}_t^2,\cdots,\boldsymbol{x}_t^N$ 而计算得到. 损失向量 $\boldsymbol{\ell}_t$ 和元损失 l_t 均有 $1/(3DG)$ 的标准化系数(该系数需要与乐观项 $\boldsymbol{m}_t$ 的相同)，在标准化后满足 $|\ell_t^i|,|l_t|\leq 1/3$. 通过这样的设置，我们可以直接将元损失 l_t 与 MSMWC 算法中的加权损失 $\langle\boldsymbol{w}_t,\boldsymbol{\ell}_t\rangle$ 建立联系，即

$$l_t=\frac{1}{3DG}\langle\nabla f_t(\boldsymbol{x}_t),\boldsymbol{x}_t\rangle=\frac{1}{3DG}\langle\boldsymbol{w}_t,\boldsymbol{\ell}_t\rangle \tag{15}$$

接着，我们借鉴了 Zhao 等人[19]的思路来设置乐观项 $\boldsymbol{m}_t$，从而使得元算法具有梯度变化界保障. 对于 $\forall i\in[N]$，我们分别设置

$$m_t^i=\frac{\langle\nabla f_t(\boldsymbol{x}_t),\boldsymbol{x}_t\rangle+\langle\nabla f_{t-1}(\bar{\boldsymbol{x}}_t),\boldsymbol{x}_t^i-\boldsymbol{x}_t\rangle}{3DG} \tag{16}$$

其中 $\bar{\boldsymbol{x}}_t=\sum_{i=1}^{N}w_t'^i\boldsymbol{x}_t^i$. 乐观项 $\boldsymbol{m}_t$ 也设置了 $1/(3DG)$ 的标准化系数，使得其在标准化后满足 $|m_t^i|\leq 1$.

需要说明的是，设计元算法的过程中最困难的部分在于设置乐观项 $\boldsymbol{m}_t$. 由于 $\boldsymbol{m}_t$ 是对于第 t 轮损失向量 $\boldsymbol{\ell}_t$ 的预测，因此我们希望对于 $\forall i\in[N]$ 能够使得 m_t^i 与 ℓ_t^i 尽量接近. 但是在构造 $\boldsymbol{m}_t$ 的阶段，我们甚至无法获取当前轮的决策 $\boldsymbol{x}_t$(根据算法 1 的第 5 步，我们需要在构造 $\boldsymbol{m}_t$ 之后再计算权重 $\boldsymbol{w}_t$，并与专家决策加权得到 $\boldsymbol{x}_t$)，则更加无法得知梯度 $\nabla f_t(\boldsymbol{x}_t)$，因此对于损失向量 $\boldsymbol{\ell}_t$ 的估计是很困难的. 此时，我们分两步进行乐观项的设计. 首先，我们使用第 $t-1$ 轮中计算得到的辅助序列 $\boldsymbol{w}'_t$ 与专家决策加权得到 $\bar{\boldsymbol{x}}_t$ 来作为 $\boldsymbol{x}_t$ 的近似，接着将其代入上一轮损失函数的梯度来构造 $\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)$ 作为 $\nabla f_t(\boldsymbol{x}_t)$ 的近似. 若 $\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)$ 与 $\nabla f_t(\boldsymbol{x}_t)$ 接近，则 m_t^i 与 ℓ_t^i 接近，从而使得元界更紧. 而式(16)中除 $\langle f_{t-1}(\bar{\boldsymbol{x}}_t),\boldsymbol{x}_t^i\rangle$ 以外的部分是对于 $\boldsymbol{m}_t$ 所有维度的整体偏移量，我们可以借助 MSMWC 算法的性质，通过拆分来避免直接计算该偏移量，详见第 4.2.2 节. 对于元算法进行遗憾界分析时，当前轮梯度的近似值与实际值的差距 $\|\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)-\nabla f_{t-1}(\boldsymbol{x}_t)\|^2$ 最终将产生元算法的梯度变化界保障. 综上，我们构造乐观项时的创新性在于：(1) 分步骤构造损失函数梯度的近似，从而使得算法能够获得问题相关界保障；(2) 针对普适算法的需求，选择合适的元算法来结合专家决策，使得乐观项中未知的整体偏移量不影响任何后续部分的计算.

当采用式(14)和(16)的设置时，$|\ell_t^i|$ 与 $|m_t^i|$ 的范围均符合引理 1 的条件. 因此我们结合式(15)与引理 1 得到元界

$$\sum_{t=1}^{T}l_t-\ell_t^i=\mathcal{O}\left(\sqrt{\ln(NT)\sum_{t=1}^{T}(\ell_t^i-m_t^i)^2}\right),$$

即此时的元算法具有形如式(7)的二阶界，通过分析可以证明其理论保障能够自动适应一般凸函数和强凸函数. 对于这两类函数的遗憾界保障分别见第 4.3 节和 4.4 节.

4.2.2　算法细节

UAGV 算法的整体结构如算法 2 所示. 我们需要考虑两种可能的凸函数类型：强凸函数和一般凸函数.

算法 2.　UAGV.

输入：运行轮数 T，在线输入的损失函数 $\{f_t(\cdot)\}_{t=1}^T$，强凸算法集合 $\mathcal{A}_{str}$，强凸量度集合 $\mathcal{P}_{str}$，一般凸算法集合 $\mathcal{A}_{con}$

输出：在线输出的各轮算法决策 $\{\boldsymbol{x}_t\}_{t=1}^T$

```
初始化专家集合 E = ∅
FOR A ∈ A_con
    创建一个专家 E(A)
    E = E ∪ E(A)
ENDFOR
FOR A ∈ A_str
    FOR λ ∈ P_str
        创建一个专家 E(A,λ)
        E = E ∪ E(A,λ)
    ENDFOR
ENDFOR
令 N = |E|，并设置 w'_1 = 1/N
FOR t = 1,2,…,T
    从专家集合 E 中获得每个专家 E^i 的决策 x_t^i
```

15. 构建乐观项 $\boldsymbol{m}_t$,规则见式(16)
16. 设置凸集 Ω 与学习率 $\boldsymbol{\eta}_t$,规则见引理 1
17. 计算 $\boldsymbol{w}_t=\arg\min_{\boldsymbol{w}\in\Omega}\langle\boldsymbol{w},\boldsymbol{m}_t\rangle+\mathcal{D}_{\psi_t}(\boldsymbol{w},\boldsymbol{w}'_t)$
18. 计算专家决策的加权平均 $\boldsymbol{x}_t=\sum_{i=1}^{N}w_t^i\boldsymbol{x}_t^i$ 作为元算法的决策
19. 观察损失函数 $f_t(\cdot)$并计算损失向量 ℓ_t 与元损失 l_t,规则见式(14)
20. 向每个专家发送其所需要的关于损失函数$f_t(\cdot)$的信息
21. 构建修正项 $\boldsymbol{a}_t\in\mathbb{R}^N$,其中$a_t^i=32\eta_t^i(\ell_t^i-m_t^i)^2$
22. 计算 $\boldsymbol{w}'_{t+1}=\arg\min_{\boldsymbol{w}\in\Omega}\langle\boldsymbol{w},\ell_t+\boldsymbol{a}_t\rangle+\mathcal{D}_{\psi_t}(\boldsymbol{w},\boldsymbol{w}'_t)$

23. END FOR

在一开始,专家集合$\mathcal{E}$为空(第 1 步). 我们首先选择能够优化一般凸函数的算法来构成候选一般凸算法集合$\mathcal{A}_{con}$,由于这些算法不需要额外的参数,因此我们对于每个在线凸优化算法 $A\in\mathcal{A}_{con}$ 只添加一个专家 $E(A)$到专家集合$\mathcal{E}$中(第 2～5 步). 我们接着选择能够优化强凸函数的算法来构成候选强凸算法集合$\mathcal{A}_{str}$,但是这些算法在运行时需要输入无法提前获得的强凸量度,所以我们选择有限个数的强凸量度作为近似,并构成候选强凸量度集合$\mathcal{P}_{str}$,具体规则见第 4.4 节. 对于每个在线强凸优化算法$A\in\mathcal{A}_{str}$和每个强凸量度$\lambda\in\mathcal{P}_{str}$,我们都添加一个专家 $E(A,\lambda)$到专家集合$\mathcal{E}$中,表示在假设强凸函数的量度为 λ 时运行算法 A(第 6～11 步).

然后,我们同时运行 $N=|\mathcal{E}|$ 个专家,并使用基于 MSMWC 算法设计的新型元算法来在线追踪最优专家. 为了表示上的方便,我们假设专家集合$\mathcal{E}$是有序的,并用 E^i来代表第 i 个专家. 在第 t 轮中,我们将专家 E^i的决策表示为 $\boldsymbol{x}_t^i\in\mathcal{X}$,元算法赋予各专家的权重记为 $\boldsymbol{w}_t\in\Omega=\{\boldsymbol{w}\in\Delta_N:w^i\geq1/(NT)\}$,而 $\boldsymbol{w}'_t$为计算权重$\boldsymbol{w}_t$时所需要的辅助序列. 我们首先从专家集合$\mathcal{E}$中获得每个专家E^i的决策$\boldsymbol{x}_t^i$(第14步),并根据式(16)构建乐观项$\boldsymbol{m}_t$(第15步),同时按照引理 1 设置学习率向量$\boldsymbol{\eta}_t$(第 16 步). 接着,采用 MSMWC 算法的方式更新权重$\boldsymbol{w}_t$(第 17 步),并计算加权平均 $\boldsymbol{x}_t$ 作为 UAGV 算法在本轮的决策(第 18 步). 随后 UAGV 算法提交决策 $\boldsymbol{x}_t$,观察损失函数 $f_t(\cdot)$从而计算专家的线性损失 ℓ_t 与元损失 l_t(第 19 步),并向每个专家发送其更新时所需要的关于 $f_t(\cdot)$的信息(第 20 步). 最后,构建修正项$\boldsymbol{a}_t$并计算辅助序列$\boldsymbol{w}'_{t+1}$(第 21～22 步).

但是,在第 t 轮中设计乐观向量 $\boldsymbol{m}_t$ 的阶段(第 16 步),元算法无从得知在第 18～19 步才能获取的 $\boldsymbol{x}_t$ 与$\nabla f_t(\boldsymbol{x}_t)$,因此我们需要解释为何根据式(16)设置的 $\boldsymbol{m}_t$ 是可行的:由于各专家的决策在第 $t-1$ 轮中接受所需信息完成更新(上一轮的第 20 步),因此在第 t 轮的开始时每个专家的决策 $\boldsymbol{x}_t^i$ 都是已知的;同时,由于 $\boldsymbol{w}'_t$也是在第 $t-1$ 轮中进行计算的(上一轮的第 22 步),因此其同样已知,从而$\bar{\boldsymbol{x}}_t=\sum_{i=1}^{N}w_t'^i\boldsymbol{x}_t^i$ 也已知;然后,我们考虑将乐观向量的每一维度 m_t^i 拆分为以下两项:

$$m_{t,1}^i=\frac{1}{3DG}\langle\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t),\boldsymbol{x}_t\rangle,$$

$$m_{t,2}^i=\frac{1}{3DG}\langle\nabla f_{t-1}(\bar{\boldsymbol{x}}_t),\boldsymbol{x}_t^i\rangle,$$

其中,$m_{t,1}^i$与 i 无关且存在未知的 $\boldsymbol{x}_t$ 和$\nabla f_t(\boldsymbol{x}_t)$,而 $m_{t,2}^i$与 i 有关且已知;我们能够说明 $\boldsymbol{m}_{t,1}$并不会影响 $\boldsymbol{w}_t$ 的求解(第 17 步),因为对于 $\boldsymbol{w}_t\in\Omega\subseteq\Delta_N$ 有

$$\begin{aligned}&\min_{\boldsymbol{w}\in\Omega}\langle\boldsymbol{w},\boldsymbol{m}_t\rangle+\mathcal{D}_{\psi_t}(\boldsymbol{w},\boldsymbol{w}'_t)\\=&\min_{\boldsymbol{w}\in\Omega}\langle\boldsymbol{w},\boldsymbol{m}_{t,1}+\boldsymbol{m}_{t,2}\rangle+\mathcal{D}_{\psi_t}(\boldsymbol{w},\boldsymbol{w}'_t)\\=&m_{t,1}^1+\min_{\boldsymbol{w}\in\Omega}\langle\boldsymbol{w},\boldsymbol{m}_{t,2}\rangle+\mathcal{D}_{\psi_t}(\boldsymbol{w},\boldsymbol{w}'_t),\end{aligned}$$

所以 $\boldsymbol{m}_{t,1}$的取值不会影响到第 17 步求解得到的 $\boldsymbol{w}_t$. 在实现该算法时,我们可以先在第 15 步时设置$\boldsymbol{m}_t=\boldsymbol{m}_{t,2}$(即暂时认为 $\boldsymbol{m}_{t,1}=\boldsymbol{0}$),直到完成第 16～19 步后再真正计算 $\boldsymbol{m}_{t,1}$,并重新设置 $\boldsymbol{m}_t=\boldsymbol{m}_{t,1}+\boldsymbol{m}_{t,2}$. 由于这样的拆分不改变第 16～19 步中的任何计算结果,因此不会影响算法的遗憾界.

通过结合引理 1、损失向量与元损失的定义(14)以及乐观项的设置(16),我们可以分析在线损失函数为一般凸或强凸函数时 UAGV 算法的遗憾界,分别见 4.3 节和 4.4 节.

在计算复杂度方面,单个专家算法一次迭代的计算复杂度一般与决策的维度 d 相关而与 T 无关[26],而元算法中对于权重 $\boldsymbol{w}_t$ 与辅助量 $\boldsymbol{w}'_t$的计算是仅与专家个数 $N=\mathcal{O}(\log T)$相关的单纯形上凸优化问题. 为了简化讨论,我们记单个专家算法进行一轮迭代的计算复杂度为$\mathcal{O}(d)$,记求解一次 n 维单纯形上凸优化问题复杂度为 $c(n)$,那么 UAGV 算法运行一轮的计算复杂度为$\mathcal{O}(d\log T+c(\log T))$.

4.3 一般凸函数

为了使得 UAGV 算法能够适应一般凸函数,我们需要指定候选在线凸优化算法集合$\mathcal{A}_{con}$. 可选的算法包括 OGD 算法[5]、对于平滑凸函数的 OEGD 算法[8]等.

由于产生梯度变化界的一个必要条件是在线函数平滑[18],为此我们需要引入一个额外的假设.

假设 3. 所有在线函数在决策域$\mathcal{X}$内均满足 H-平滑.

需要说明的是，假设 3 在很多现实问题中是成立的. 在线凸优化算法常用于在线机器学习任务，而机器学习中很多常用的损失函数，如均方误差(Mean Squared Error)、交叉熵损失(Cross-Entropy Loss)、平滑 ℓ_1 损失(Smooth ℓ_1 Loss)等，均为平滑凸函数.

当在线函数为平滑凸函数时，通过结合引理 1 以及损失与乐观项的设计方案(14)(16)，我们证明该新型元算法能够适应在线损失函数的梯度变化. 此时若将具有梯度变化界的 OEGD 算法加入专家算法中，就能够使得 UAGV 算法对于平滑凸函数具有梯度变化界保障，证明见附录 B.1.

定理 1.　假设对于 $\forall t\in[T]$，在线凸函数 $f_t(\cdot)$ 均满足假设 1、2 和 3. 当候选算法集合 $\mathcal{A}_{con}$ 中包含 OEGD 算法时，UAGV 算法能够保证

$$\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\min_{\boldsymbol{x}\in\mathcal{X}}\sum_{t=1}^{T}f_t(\boldsymbol{x})=\mathcal{O}(\sqrt{V_T\log T}).$$

相比于不具有梯度变化界保障的在线凸优化算法，UAGV 算法的优势在于将遗憾界保障中的主导项由与问题无关的 $\mathcal{O}(\sqrt{T})$ 替换为与在线函数梯度变化相关的 $\widetilde{\mathcal{O}}(\sqrt{V_T})$. 当 $V_T\ll T$ 时，UAGV 算法能够受益于在线函数梯度变化缓慢的良好性质，获得比极小极大最优界更紧的遗憾界，而不具有梯度变化界的算法仅能获得极小极大最优界.

而对于非平滑的一般凸在线函数，UAGV 算法同样具有理论保障. 用 $R(A)$ 表示专家 $E(A)$ 的遗憾界，那么可以给出 UAGV 算法对于一般凸函数的理论保障，证明见附录 B.2.

定理 2.　假设对于 $\forall t\in[T]$，在线凸函数 $f_t(\cdot)$ 均满足假设 1 和 2，UAGV 算法能够保证

$$\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\min_{\boldsymbol{x}\in\mathcal{X}}\sum_{t=1}^{T}f_t(\boldsymbol{x})\le\min_{A\in\mathcal{A}_{con}}R(A)+\mathcal{O}(\sqrt{T\log T}).$$

4.4　强凸函数

为了使得 UAGV 算法能够适应强凸函数，我们需要指定候选在线强凸优化算法集合 $\mathcal{A}_{str}$ 和候选强凸量度集合 $\mathcal{P}_{str}$. 我们将能够处理在线强凸优化的算法加入 $\mathcal{A}_{str}$，例如 SC-OGD 算法[6].

在构建 $\mathcal{P}_{str}$ 时，我们首先需要分析强凸量度 λ 可能的取值范围. 若 $\lambda<1/T$，由于强凸函数的遗憾界与 $1/\lambda$ 相关，在此情况下遗憾界至少为 $\Omega(T)$. 这个结果不低于一般凸函数的极小极大最优界，因此我们无法利用强凸性获得更小的遗憾界，不如直接将其当作一般凸函数进行处理. 若 $\lambda>1$，根据强凸函数的定义(10)，λ-强凸的函数也为 1-强凸的，因此我们可以将其当作 1-强凸函数进行处理，此时的遗憾界比理论最优值只相差一个常数因子 λ.

综合上述两种情况，我们认为强凸量度 λ 可能的取值范围是 $[1/T,1]$，并采用以 2 为底的指数间隔的方式构建候选强凸量度集合 $\mathcal{P}_{str}$，即

$$\mathcal{P}_{str}=\left\{\frac{1}{T},\frac{2}{T},\cdots,\frac{2^L}{T}\right\},\ L=\lceil\log_2 T\rceil \quad (17)$$

当采用这种构建方式时，对于任意 $\lambda\in[1/T,1]$，均能保证存在 $\hat{\lambda}\in\mathcal{P}_{str}$ 使得 $\hat{\lambda}\le\lambda\le2\hat{\lambda}$.

用 $R(A,\hat{\lambda})$ 表示 UAGV 算法中专家 $E(A,\hat{\lambda})$ 的遗憾界，那么可以给出 UAGV 算法对于强凸函数的理论保障，证明见附录 B.3.

定理 3.　假设对于 $\forall t\in[T]$，在线函数 $f_t(\cdot)$ 均满足假设 1 和 2，且为 λ-强凸，其中 $\lambda\in[1/T,1]$. 对于 $\hat{\lambda}\in\mathcal{P}_{str}$ 且满足 $\hat{\lambda}\le\lambda\le2\hat{\lambda}$，UAGV 算法能够保证

$$\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\min_{\boldsymbol{x}\in\mathcal{X}}\sum_{t=1}^{T}f_t(\boldsymbol{x})\le\min_{A\in\mathcal{A}_{str}}R(A,\hat{\lambda})+\mathcal{O}\left(\frac{\log T}{\lambda}\right).$$

由于在线强凸优化算法的极小极大最优界为 $\mathcal{O}(\log T/\lambda)$，因此 UAGV 算法对于强凸函数的理论保障取得了极小极大最优.

5　实　验

在本节中，我们通过在多个数据集上进行实验，并与现有普适算法进行对比，从而展示 UAGV 算法的性能.

5.1　实验设置

我们使用在线分类任务[4]进行实验. 在第 t 轮中，学习者首先提交一个线性分类器 $\boldsymbol{x}_t\in\mathcal{X}$；接着，学习者会收到从数据集中采样得到的 m 个样本 $\{\boldsymbol{v}_t^{(k)},y_t^{(k)}\}_{k=1}^m$，其中 $\boldsymbol{v}_t^{(k)}\in\mathbb{R}^d$ 是第 k 个样本的特征向量，$y_t^{(k)}$ 是该样本对应的标签；最后，学习者在此轮遭受了 $f_t(\boldsymbol{x}_t)$ 的损失并更新自身的模型. 为了展示 UAGV 算法能够适应不同类型的函数，我们使用两种损失函数进行测试，分别为交叉熵损失函数

$$f_t(\boldsymbol{x})=-\frac{1}{m}\sum_{k=1}^{m}(y_t^{(k)}\log\hat{y}_t^{(k)}+(1-y_t^{(k)})\log(1-\hat{y}_t^{(k)})),$$

其中 $\hat{y}_t^{(k)}=1/(1+e^{-\boldsymbol{x}^{\mathrm{T}}\boldsymbol{v}_t^{(k)}})$ 是采用逻辑回归时对于第 k 个样本的预测，当所有样本特征向量的二范数均不超过 D_s 时该损失函数为 $D_s^2/4$-平滑凸函数，其证明见附录 C；以及采用二范数正则化的合页损失函数

$$f_t(\boldsymbol{x})=\frac{1}{m}\sum_{k=1}^{m}\max\{0,1-y_t^{(k)}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{v}_t^{(k)}\}+\frac{\lambda}{2}\|\boldsymbol{x}\|^2,$$

该损失函数是 λ-强凸函数. 当使用交叉熵损失函数时，正样本标签为 1、负样本标签为 0；当使用采用二范数正则化的合页损失函数时，正样本标签为 1、负样本标签为 -1.

实验在特征标准化后的 LIBSVM 数据集[27]上进行. 在任务中,我们通过对于数据集进行采样来构造在线凸损失函数,这对于数据集类型没有特殊要求,因此我们不限定类型地选取了特征数由 60 至 780 的 5 个数据集进行实验. 对于多分类数据集,我们将其中的一类作为正样本,其余作为负样本. 每一轮从数据集中随机采样一批大小为 m 的样本. 实验所使用数据集及正样本设置的概况见表 1.

表 1 数据集概况

数据集	特征数	样本数(正样本数)
splice	60	3175(1648)
mushrooms	112	8124(3916)
a9a	123	48842(11687)
usps	256	9298(1553)
mnist	780	70000(6876)

在选择交叉熵损失函数时,我们设置一次采样的样本数 $m=10$、决策集直径 $D=10$、时间维度 $T=5000$,并根据 D 和特征矩阵来估计梯度的界 G. 在选择采用二范数正则化的合页损失函数时,我们额外设置正则化因子 $\lambda=0.02$ 并相应修改对 G 的估计,其余设置保持不变. 对于所有数据集,每个算法均更换不同的随机种子重复实验 10 次,对结果取平均值.

5.2 对比算法

我们将 UAGV 算法的实验结果与现有普适算法进行对比,对比算法包括:MetaGrad 算法[12]、$\text{MetaGrad}_{\text{freq}}$ 算法[13]、Maler 算法[14]和 USC 算法[15]. 其中,对于 $\text{MetaGrad}_{\text{freq}}$ 算法设置 $rank=20$,USC 算法和 UAGV 算法均使用了以下的候选算法:

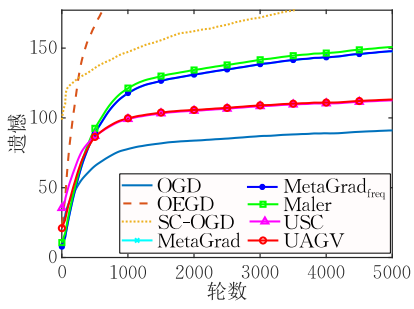
(1) 优化一般凸函数的候选算法集合 $\mathcal{A}_{con}$ 包括:设置步长为 $\eta_t=\dfrac{G}{D\sqrt{t}}$ 的 OGD 算法[5];设置步长为 $\eta_t=\min\left\{\dfrac{1}{4H},\dfrac{D}{\sqrt{1+\widetilde{V}_{t-1}}}\right\}$ 的 OEGD 算法[8],其中 $\widetilde{V}_{t-1}=\sum_{s=2}^{t-1}\left\|\nabla f_s(\boldsymbol{x}_{s-1})-\nabla f_{s-1}(\boldsymbol{x}_{s-1})\right\|^2$ 是对于由式(1)定义的 V_T 的经验估计.

(2) 优化强凸函数的候选算法集合 $\mathcal{A}_{str}$ 包括:设置步长为 $\eta_t=\dfrac{1}{\lambda t}$ 的 SC-OGD 算法[6],同时根据式(17)构建候选强凸量度集合 $\mathcal{P}_{str}$.

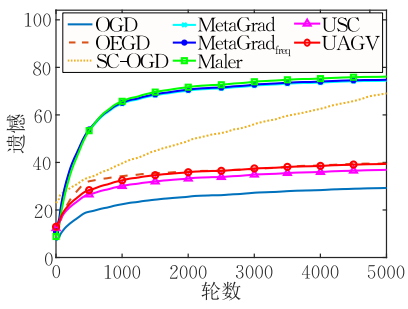
在实验结果中,我们同时将候选专家算法 OGD、OEGD 和 SC-OGD(取 $\lceil\log_2 T\rceil$ 个 SC-OGD 专家中遗憾最小的作为代表)的遗憾变化情况一并展示,使得我们能从侧面观察不同实验设置下在线损失函数所具有的性质.

5.3 实验结果

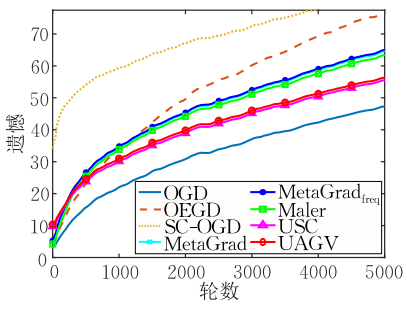
我们首先采用交叉熵损失函数进行实验,该损失函数是平滑一般凸的. 我们在 5 个数据集上分别比较 5 个普适算法 MetaGrad、$\text{MetaGrad}_{\text{freq}}$、Maler、USC 和 UAGV 的遗憾,并且与候选专家算法 OGD、OEGD 和 SC-OGD 的结果进行对照. 图 2(a)~(e)给出了所有算法运行时的遗憾变化情况,实验结果



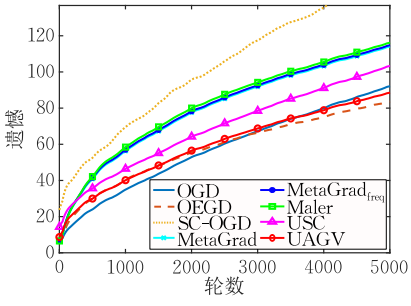
(a) splice上遗憾变化情况



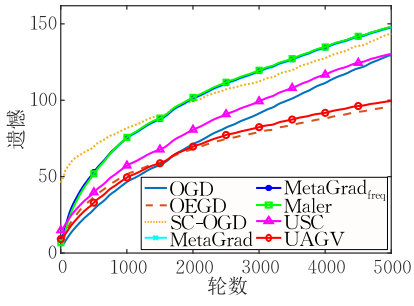
(b) mushrooms上遗憾变化情况



(c) a9a上遗憾变化情况



(d) usps上遗憾变化情况



(e) mnist上遗憾变化情况

图 2 使用交叉熵损失函数时,标准数据集上遗憾随回合数的变化情况

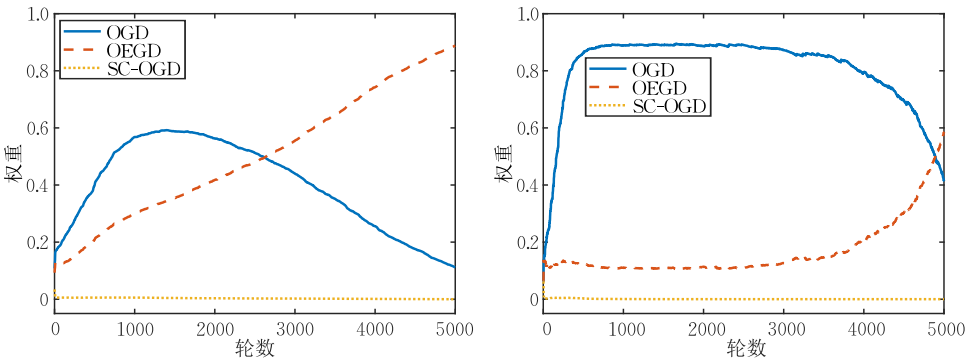
表明：

(1) 当采用交叉熵损失函数时，候选算法集合中最优的专家算法为针对一般凸函数的 OGD 算法或针对平滑凸函数的 OEGD 算法. 而若将损失函数作为强凸函数处理则施加了过强的假设，故而针对强凸函数的 SC-OGD 算法在实验中没有成为最优的专家.

(2) 在普适算法中，MetaGrad、$\text{MetaGrad}_{\text{freq}}$ 和 Maler 算法的遗憾比较接近且与最优的专家算法相差较大，USC 和 UAGV 算法的遗憾与最优的专家算法相差较小. 这是因为 MetaGrad 算法及其加速版本与 Maler 算法的专家算法都在优化形式相同的替代损失函数(4)而不是原损失函数，该方式引入了额外的遗憾，并且无法使得算法利用可能具有的问题相关性质.

需要说明的是，由于我们进行实验时选取的 5 个数据集并没有限定其具体的类型，因此不同数据内部的结构与所具有性质可能存在差异. 根据 OEGD 算法所具有梯度变化界保障，其能够在梯度变化缓慢时取得更好的性能，于是我们可以通过 OEGD 算法是否优于 OGD 算法来从侧面判断损失函数是否具有梯度变化缓慢的性质. 在 splice、mushrooms 和 a9a 数据集上，候选算法中 OEGD 算法的遗憾大于 OGD 算法，这说明在这些数据集上捕捉在线函数的梯度变化信息并没有给算法优化带来优势，相应地在这 3 个数据集上 UAGV 与 USC 算法的遗憾十分接近. 而在 usps 和 mnist 数据集上，OEGD 算法的遗憾小于 OGD 算法，这反映了在线函数具有梯度变化缓慢的性质且能够被优化算法所利用，此时我们提出的具有梯度变化界保障的 UAGV 算法相较于 USC 算法获得明显更小的遗憾，最终遗憾减小的幅度分别为 14.4%和 24.6%.

由于 USC 和 UAGV 算法均使用了相同的候选专家算法，两者的区别仅在于使用的元算法不同，因此我们对于两种算法的实验结果进行重点对比. 比较能反应两者差距的数据集为 usps 和 mnist，我们以 mnist 数据集上 USC 和 UAGV 算法的运行情况为例具体说明 UAGV 所使用的新型元算法的优势. 在 mnist 数据集上，不存在一种候选算法的遗憾在 T 轮中一直最小，根据图 2(e)所示，在约 1800 轮前 OGD 算法遗憾最小，而在此后 OEGD 算法遗憾最小. 这种较为复杂的情况考验了元算法跟踪最优专家的能力. 对比图 3(a)和(b)中 UAGV 与 USC 的元算法权重变化情况，可以发现 UAGV 的元算法一直在跟踪并逐渐提高对于 OEGD 算法的权重，而 USC 算法的权重直到接近 5000 轮时才倾向于 OEGD 算法. 这反映了 UAGV 算法在候选专家算法遗憾变化比较复杂的情况下的适应能力更强.



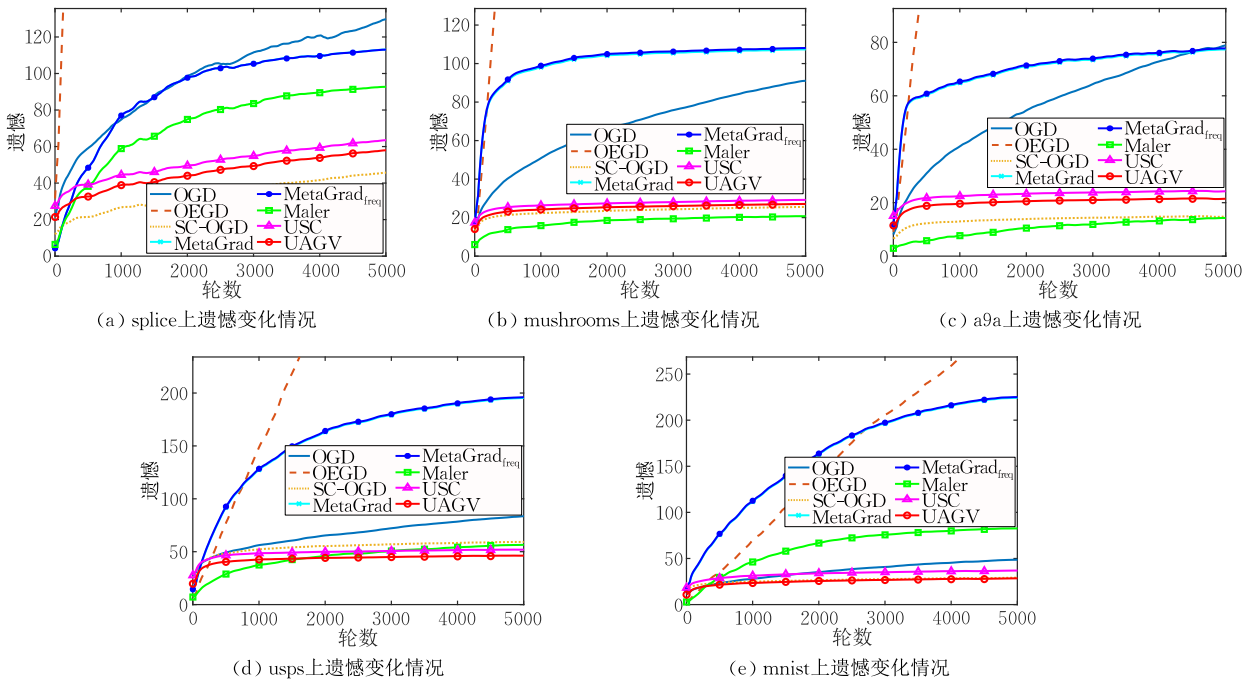
(a) UAGV的元算法权重变化　　(b) USC的元算法权重变化

图 3　使用交叉熵损失函数时，mnist 数据集上 UAGV 与 USC 元算法权重变化情况对比

接着我们使用二范数正则化的合页损失函数来进行实验，该损失函数是强凸的. 我们在 5 个数据集上分别比较 5 个普适算法 MetaGrad、$\text{MetaGrad}_{\text{freq}}$、Maler、USC 和 UAGV 的遗憾，并且与候选专家算法 OGD、OEGD 和 SC-OGD 的结果进行对照. 图 4(a)～(e)给出了采用不同算法时的遗憾，实验结果表明：

(1) 当采用二范数正则化的合页损失函数时，候选算法集合中最优的专家算法为针对强凸函数的 SC-OGD 算法，其次为针对一般凸函数的 OGD 算法. 由于损失函数并不平滑，因此 OEGD 算法不适用且表现最差.

(2) 在普适算法之间进行比较，MetaGrad 和 $\text{MetaGrad}_{\text{freq}}$ 算法由于对强凸损失函数遗憾界保障没有达到极小极大最优从而遗憾最大，同时 UAGV 算法与 Maler 和 USC 算法的遗憾比较接近. 这体现了 UAGV 算法能够像 Maler 和 USC 算法一样适应于强凸的损失函数.



(a) splice上遗憾变化情况 (b) mushrooms上遗憾变化情况 (c) a9a上遗憾变化情况

(d) usps上遗憾变化情况 (e) mnist上遗憾变化情况

图 4 使用二范数正则化的合页损失函数时,标准数据集上遗憾随回合数的变化情况

6 总 结

针对现有普适凸优化算法中对于一般凸函数难以获得问题相关界的缺陷,本文提出了在平滑条件下对于一般凸函数具有梯度变化界保障的 UAGV 算法. UAGV 算法采用专家学习框架,具有两层结构:底层同时运行多个专家算法,顶层使用元算法来结合专家算法的决策. 为了使得元算法具有梯度变化界保障,我们基于 MSMWC 算法设计了新型的元算法,其中的难点在于针对性地设置 MSMWC 算法中的替代损失函数与乐观项. 通过理论分析,我们证明了 UAGV 算法在平滑条件下对于一般凸函数具有梯度变化界保障,除此以外还能够适应强凸函数. 我们在标准数据集上进行实验,结果展示了 UAGV 算法对于平滑一般凸和强凸函数具有较好的优化效果.

UAGV 算法适用于具有凸损失函数的优化任务,其既可以在线利用流式数据进行优化,也可以对于离线数据小批量采样进行优化,应用场景较为广泛. 在使用时,无需用户对于凸损失函数的性质进行额外分析,算法能够自动适应于可能的强凸或梯度缓慢变化的性质以提高优化效率. 后续的研究可以考虑两方面的拓展,包括支持更多类型的损失函数以及利用更多种类的问题相关性质. 为了进行此方向的探索,我们可能需要对于元算法中乐观项的形式做出调整,或是采用更加复杂的算法结构进行遗憾界的进一步分解.

参 考 文 献

[1] Liang Ji-Ye, Feng Chen-Jiao, Song Peng. A survey on correlation analysis of big data. Chinese Journal of Computers, 2016, 39(1): 1-18(in Chinese)
(梁吉业, 冯晨娇, 宋鹏. 大数据相关分析综述. 计算机学报, 2016, 39(1): 1-18)

[2] Sun Da-Wei, Zhang Guang-Yan, Zheng Wei-Min. Big data stream computing: Technologies and instances. Journal of Software, 2014, 25(4): 839-862(in Chinese)
(孙大为, 张广艳, 郑纬民. 大数据流式计算:关键技术及系统实例. 软件学报, 2014, 25(4): 839-862)

[3] Li Zhi-Jie, Li Yuan-Xiang, Wang Feng, et al. Online learning algorithms for big data analytics: A survey. Journal of Computer Research and Development, 2015, 52(8): 1707-1721(in Chinese)
(李志杰, 李元香, 王峰等. 面向大数据分析的在线学习算法综述. 计算机研究与发展, 2015, 52(8): 1707-1721)

[4] Shalev-Shwartz S. Online learning and online convex optimization. Foundations and Trends® in Machine Learning, 2012, 4(2): 107-194

[5] Zinkevich M. Online convex programming and generalized infinitesimal gradient ascent//Proceedings of the 20th International Conference on Machine Learning. Washington, USA, 2003: 928-936

[6] Shalev-Shwartz S, Singer Y, Srebro N. Pegasos: Primal

estimated sub-gradient solver for SVM//Proceedings of the 24th International Conference on Machine Learning. Corvallis, USA, 2007: 807-814

[7] Abernethy J D, Bartlett P L, Rakhlin A, et al. Optimal stragies and minimax lower bounds for online convex games //Proceedings of the 21st Annual Conference on Learning Theory. Helsinki, Finland, 2008: 415-424

[8] Chiang C-K, Yang T, Lee C-J, et al. Online optimization with gradual variations//Proceedings of the 25th Annual Conference on Learning Theory. Edinburgh, UK, 2012: 6.1-6.20

[9] Kingma D P, Ba J. Adam: A method for stochastic optimization //Proceedings of the 3rd International Conference on Learning Representations. San Diego, USA, 2015

[10] Reddi S J, Kale S, Kumar S. On the convergence of Adam and beyond//Proceedings of the 6th International Conference on Learning Representations. Vancouver, Canada, 2018

[11] Bartlett P L, Hazan E, Rakhlin A. Adaptive online gradient descent//Proceedings of the 21st Annual Conference on Neural Information Processing Systems. Vancouver, Canada, 2007: 65-72

[12] Van Erven T, Koolen W M. MetaGrad: Multiple learning rates in online learning//Proceedings of the 30th Annual Conference on Neural Information Processing Systems. Barcelona, Spain, 2016: 3666-3674

[13] Van Erven T, Koolen W M, Van Der Hoeven D. MetaGrad: Adaptation using multiple learning rates in online learning. Journal of Machine Learning Research, 2021, 22(1): 7261-7321

[14] Wang G, Lu S, Zhang L. Adaptivity and optimality: A universal algorithm for online convex optimization//Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence. Tel Aviv, Israel, 2019: 659-668

[15] Zhang L, Wang G, Yi J, et al. A simple yet universal strategy for online convex optimization//Proceedings of the 39th International Conference on Machine Learning. Baltimore, USA, 2022: 26605-26623

[16] Chen L, Luo H, Wei C-Y. Impossible tuning made possible: A new expert algorithm and its applications//Proceedings of the 34th Conference on Learning Theory. Boulder, USA, 2021: 1216-1259

[17] Hazan E, Agarwal A, Kale S. Logarithmic regret algorithms for online convex optimization. Machine Learning, 2007, 69(2/3): 169-192

[18] Srebro N, Sridharan K, Tewari A. Smoothness, low noise and fast rates//Proceedings of the 24th Annual Conference on Neural Information Processing Systems. Vancouver, Canada, 2010: 2199-2207

[19] Zhao P, Zhang Y-J, Zhang L, et al. Adaptivity and non-stationarity: Problem-dependent dynamic regret for online convex optimization. arXiv preprint arXiv: 2112.14368, 2021

[20] Zhang L, Liu T-Y, Zhou Z-H. Adaptive regret of convex and smooth functions//Proceedings of the 36th International Conference on Machine Learning. Long Beach, USA, 2019: 7414-7423

[21] Wang G, Lu S, Hu Y, et al. Adapting to smoothness: A more universal algorithm for online convex optimization// Proceedings of the 34th AAAI Conference on Artificial Intelligence. New York, USA, 2020: 6162-6169

[22] Cesa-Bianchi N, Lugosi G. Prediction, Learning, and Games. Cambridge, UK: Cambridge University Press, 2006

[23] Gaillard P, Stoltz G, Van Erven T. A second-order bound with excess losses//Proceedings of the 27th Conference on Learning Theory. Barcelona, Spain, 2014: 176-196

[24] Rakhlin A, Sridharan K. Online learning with predictable sequences//Proceedings of the 26th Conference on Learning Theory. New Jersey, USA, 2013: 993-1019

[25] Syrgkanis V, Agarwal A, Luo H, et al. Fast convergence of regularized learning in games//Proceedings of the 28th Annual Conference on Neural Information Processing Systems. Montreal, Canada, 2015: 2989-2997

[26] Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, 2011, 12(7): 2121-2159

[27] Chang C-C, Lin C-J. LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2011, 2(3): 1-27

[28] Tsybakov A B. Introduction to Nonparametric Estimation. New York, USA: Springer, 2009

附录 A.

A.1. 对于引理 1 的证明.

Chen 等人[16]在分析 MSMWC 算法的遗憾界时，所用到的关键性引理(文献[16]中的引理 1)的第一步丢弃了下式中最后的一个负项

$$\sum_{t=1}^{T}\langle \boldsymbol{w}_t-\boldsymbol{u},\boldsymbol{\ell}_t+\boldsymbol{a}_t\rangle\leq\sum_{t=1}^{T}(\mathcal{D}_{\psi_t}(\boldsymbol{u},\boldsymbol{w}'_t)-\mathcal{D}_{\psi_t}(\boldsymbol{u},\boldsymbol{w}'_{t+1}))+\sum_{t=1}^{T}(\langle \boldsymbol{w}_t-\boldsymbol{w}'_{t+1},\boldsymbol{\ell}_t-\boldsymbol{m}_t+\boldsymbol{a}_t\rangle-\mathcal{D}_{\psi_t}(\boldsymbol{w}'_{t+1},\boldsymbol{w}_t))-\sum_{t=1}^{T}\mathcal{D}_{\psi_t}(\boldsymbol{w}_t,\boldsymbol{w}'_t).$$

我们将该负项保留，并将上式代回到文献[16]中的定理 2，能够得到

$$\sum_{t=1}^{T}\langle \boldsymbol{w}_t-\boldsymbol{w}_t^j,\boldsymbol{\ell}_t\rangle=\mathcal{O}\left(\ln(NT)+\sqrt{\ln(NT)\sum_{t=1}^{T}(\ell_t^j-m_t^j)^2}-\sum_{t=1}^{T}\mathcal{D}_{\psi_t}(\boldsymbol{w}_t,\boldsymbol{w}'_t)\right)\tag{18}$$

我们在此基础上对于该负项加以分析以证明引理 1. 根据算法 2 中的定义，有

$\mathcal{D}_{\psi_t}(\boldsymbol{x},\boldsymbol{y})\triangleq\sum_{i=1}^{N}\frac{1}{\eta_t^i}\mathrm{KL}(x_i,y_i)=\sum_{i=1}^{N}\frac{1}{\eta_t^i}\left(x_i\ln\frac{x_i}{y_i}-x_i+y_i\right)$，根据 $\boldsymbol{x},\boldsymbol{y}$ 满足 $\forall\boldsymbol{x},\boldsymbol{y}\in\Omega=\left\{\boldsymbol{w}\in\Delta_N:w_i\geq\frac{1}{NT}\right\}$，有 $x_i\ln\frac{x_i}{y_i}-x_i+y_i>0$. 同时 $\eta_t^i\triangleq\min\left\{\sqrt{\ln(NT)\Big/\sum_{s<t}(\ell_s^i-m_s^i)^2},C\right\}$，其中 $C>0$，那么有 $\frac{1}{\eta_t^i}=\max\left\{\sqrt{\sum_{s<t}(\ell_s^i-m_s^i)^2/\ln(NT)},\frac{1}{C}\right\}$，即 $\frac{1}{\eta_t^i}\geq\frac{1}{C}$，因此能够将 $\mathcal{D}_{\psi_t}(\boldsymbol{x},\boldsymbol{y})$ 放缩为

$$\mathcal{D}_{\psi_t}(\boldsymbol{x},\boldsymbol{y})=\sum_{i=1}^{N}\frac{1}{\eta_t^i}\left(x_i\ln\frac{x_i}{y_i}-x_i+y_i\right)\geq\sum_{i=1}^{N}\frac{1}{C}\left(x_i\ln\frac{x_i}{y_i}-x_i+y_i\right)=\frac{1}{C}\mathrm{KL}(\boldsymbol{x},\boldsymbol{y})\geq\frac{1}{2C}\|\boldsymbol{x}-\boldsymbol{y}\|_1^2,$$

其中，第三步利用了 Δ_N 的定义，最后一步利用了 Pinsker 不等式[28]，即 $\mathrm{KL}(\boldsymbol{x},\boldsymbol{y})\geq\frac{1}{2}\|\boldsymbol{x}-\boldsymbol{y}\|_1^2$. 代入 $\boldsymbol{x}=\boldsymbol{w}_t$、$\boldsymbol{y}=\boldsymbol{w}_t'$ 和 $C=\frac{1}{64}$ 能够得到 $-\mathcal{D}_{\psi_t}(\boldsymbol{w},\boldsymbol{w}_t')\leq-32\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2$. 将其代回式(18)即可完成引理 1 的证明. 证毕.

附录 B.

B.1. 对于定理 1 的证明.

首先分析元界. 根据引理 1，MSMWC 算法的理论保证

$$\sum_{t=1}^{T}(l_t-\ell_t^j)=\mathcal{O}\left(\ln(NT)+\sqrt{\ln(NT)\sum_{t=1}^{T}(\ell_t^j-m_t^j)^2}-\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2\right),$$

对于所有 $j\in[N]$ 同时成立. 因此对于使用线性损失的元算法，代入式(14)中损失向量 $\boldsymbol{\ell}_t$ 和元损失 l_t 的定义可以得到

$$\sum_{t=1}^{T}\langle\nabla f_t(\boldsymbol{x}_t),\boldsymbol{x}_t-\boldsymbol{x}_t^j\rangle\overset{(14)}{=}3DG\sum_{t=1}^{T}(l_t-\ell_t^j)=\mathcal{O}\left(\ln(NT)+\sqrt{\ln(NT)\sum_{t=1}^{T}(\ell_t^j-m_t^j)^2}-\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2\right)\tag{19}$$

当在线函数 $f_t(\cdot)$ 为一般凸时，我们在分析元界 $\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\sum_{t=1}^{T}f_t(\boldsymbol{x}_t^j)$ 时保留式(19)中的负项，并代入式(14)和(16)中损失向量 $\boldsymbol{\ell}_t$ 与乐观项 $\boldsymbol{m}_t$ 的定义得到

$$\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\sum_{t=1}^{T}f_t(\boldsymbol{x}_t^j)\overset{(9)}{\leq}\sum_{t=1}^{T}\langle\nabla f_t(\boldsymbol{x}_t),\boldsymbol{x}_t-\boldsymbol{x}_t^j\rangle\overset{(19)}{=}\mathcal{O}\left(\ln(NT)+\sqrt{\ln(NT)\sum_{t=1}^{T}(\ell_t^j-m_t^j)^2}-\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2\right)\overset{(14),(16)}{=}\mathcal{O}\left(\ln(NT)+\sqrt{\ln(NT)\sum_{t=1}^{T}\langle\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t),\boldsymbol{x}_t^j-\boldsymbol{x}_t\rangle^2}-\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2\right)\tag{20}$$

我们接着对式(20)中根号内的 $\sum_{t=1}^{T}\langle\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t),\boldsymbol{x}_t^j-\boldsymbol{x}_t\rangle^2$ 一项进行分析，有

$$\begin{aligned}&\sum_{t=1}^{T}\langle\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t),\boldsymbol{x}_t^j-\boldsymbol{x}_t\rangle^2\leq\\&\sum_{t=1}^{T}\|\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)\|^2\|\boldsymbol{x}_t^j-\boldsymbol{x}_t\|^2\leq\\&\sum_{t=1}^{T}\|\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)\|^2(\|\boldsymbol{x}_t^j\|+\|\boldsymbol{x}_t\|)^2\leq\\&\sum_{t=1}^{T}\|\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)\|^2\cdot2(\|\boldsymbol{x}_t^j\|^2+\|\boldsymbol{x}_t\|^2)\overset{(13)}{\leq}\\&4D^2\sum_{t=1}^{T}\|\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)\|^2\end{aligned}\tag{21}$$

其中，第一步用到了柯西-施瓦茨不等式 $\langle\boldsymbol{x},\boldsymbol{y}\rangle^2\leq\|\boldsymbol{x}\|^2\|\boldsymbol{y}\|^2$.

当在线函数 $f_t(\cdot)$ 均为 H-平滑函数时，我们能够利用平滑的性质(11)在式(21)的基础上进行更加仔细的分析，得到

$$\begin{aligned}&\sum_{t=1}^{T}\langle\nabla f_t(\boldsymbol{x}_t)-\nabla f_t(\bar{\boldsymbol{x}}_t),\boldsymbol{x}_t^j-\boldsymbol{x}_t\rangle^2\overset{(21)}{\leq}\\&4D^2\sum_{t=1}^{T}\|\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)\|^2\leq\\&8D^2\sum_{t=1}^{T}(\|\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\boldsymbol{x}_t)\|^2+\|\nabla f_{t-1}(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)\|^2)\overset{(11)}{\leq}\\&8D^2\sum_{t=1}^{T}\sup_{x\in\mathcal{X}}\|\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\boldsymbol{x}_t)\|^2+8D^2H^2\sum_{t=1}^{T}\|\boldsymbol{x}_t-\bar{\boldsymbol{x}}_t\|^2\leq8D^2V_T+8D^4H^2\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2\end{aligned}\tag{22}$$

其中最后一步利用了 $\|\boldsymbol{x}_t-\bar{\boldsymbol{x}}_t\|^2$ 能够被元算法的权重变化量约束住的性质，即

$$\|\boldsymbol{x}_t-\bar{\boldsymbol{x}}_t\|^2=\left\|\sum_{t=1}^{N}(w_t^i-w_t'^i)\boldsymbol{x}_t^i\right\|^2\leq\left(\sum_{t=1}^{N}|w_t^i-w_t'^i|\|\boldsymbol{x}_t^i\|\right)^2\leq D^2\|\boldsymbol{w}_t-\boldsymbol{w}'\|_1^2.$$

综合式(20)和(22)，我们可以推导得出 UAGV 算法对于平滑凸在线函数的元界

$$\begin{aligned}&\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\sum_{t=1}^{T}f_t(\boldsymbol{x}_t^j)\overset{(20),(22)}{=}\mathcal{O}\left(\ln(NT)+\sqrt{\ln(NT)\left(8D^2V_T+8D^4H^2\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2\right)}-\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2\right)=\\&\mathcal{O}\left(\ln(NT)+\sqrt{8D^2\ln(NT)V_T}+\sqrt{8D^4H^2\ln(NT)\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2}-\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2\right)=\\&\mathcal{O}\left(\ln(NT)+\sqrt{8D^2\ln(NT)V_T}+2D^4H^2\ln(NT)+\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2-\sum_{t=1}^{T}\|\boldsymbol{w}_t-\boldsymbol{w}_t'\|_1^2\right)=\mathcal{O}(\sqrt{V_T\log T})\end{aligned}\tag{23}$$

其中，第二步用到了不等式 $\sqrt{a+b}\leq\sqrt{a}+\sqrt{b}$，第三步用到了基本不等式 $2\sqrt{ab}\leq a+b$.

现在我们考虑专家界. 对于所有 $A\in\mathcal{A}_{con}$，均有专家界

$$\sum_{t=1}^{T}f_t(\boldsymbol{x}_t^j)-\min_{x\in\mathcal{X}}\sum_{t=1}^{T}f_t(\boldsymbol{x})\leq R(A)\tag{24}$$

同时，根据 OEGD 算法对于平滑凸函数的理论保障[8]，有

$$R(\mathrm{OEGD})=\mathcal{O}(\sqrt{1+V_T}) \tag{25}$$

于是当 OEGD$\in\mathcal{A}_{con}$ 时，结合元界(23)与专家界(24)、(25)，我们得到 UAGV 算法对于平滑凸在线函数的遗憾界

$$\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\min_{\boldsymbol{x}\in\mathcal{X}}\sum_{t=1}^{T}f_t(\boldsymbol{x})\overset{(23),(24)}{\leq}\min_{A\in\mathcal{A}_{con}}R(A)+\mathcal{O}(\sqrt{V_T\log T})\overset{(25)}{=}\mathcal{O}(\sqrt{V_T\log T}).$$

证毕.

B.2. 对于定理 2 的证明.

当在线函数不具有平滑性质时，我们在式(21)的基础上进行分析，得到

$$\begin{aligned}&\sum_{t=1}^{T}\langle\nabla f_t(\boldsymbol{x}_t)-\nabla f_t(\bar{\boldsymbol{x}}_t),\boldsymbol{x}_t^j-\boldsymbol{x}_t\rangle^2\overset{(21)}{\leq}\\&4D^2\sum_{t=1}^{T}\|\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)\|^2\leq\\&4D^2\sum_{t=1}^{T}(\|\nabla f_t(\boldsymbol{x}_t)\|+\|\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)\|)^2\leq\\&4D^2\sum_{t=1}^{T}2(\|\nabla f_t(\boldsymbol{x}_t)\|^2+\|\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)\|^2)\\&\overset{(12)}{\leq}16D^2G^2T\end{aligned} \tag{26}$$

将其代入式(20)得到元界

$$\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\sum_{t=1}^{T}f_t(\boldsymbol{x}_t^j)\overset{(20),(26)}{=}\mathcal{O}(\sqrt{T\log T}) \tag{27}$$

结合元界(27)与专家界(24)，最终得到 UAGV 算法对于一般凸在线函数的遗憾界

$$\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\min_{\boldsymbol{x}\in\mathcal{X}}\sum_{t=1}^{T}f_t(\boldsymbol{x})\leq\min_{A\in\mathcal{A}_{con}}R(A)+\mathcal{O}(\sqrt{T\log T}).$$

证毕.

B.3. 对于定理 3 的证明.

当在线函数 $f_t(\cdot)$ 强凸时，我们在分析元界 $\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\sum_{t=1}^{T}f_t(\boldsymbol{x}_t^j)$ 时丢弃式(19)最后的负项，得到

$$\begin{aligned}&\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\sum_{t=1}^{T}f_t(\boldsymbol{x}_t^j)\overset{(10)}{\leq}\\&\sum_{t=1}^{T}\langle\nabla f_t(\boldsymbol{x}_t),\boldsymbol{x}_t-\boldsymbol{x}_t^j\rangle-\frac{\lambda}{2}\|\boldsymbol{x}_t-\boldsymbol{x}_t^j\|^2\overset{(19)}{=}\\&\mathcal{O}\left(\ln(NT)+\sqrt{\ln(NT)\sum_{t=1}^{T}(\ell_t^j-m_t^j)^2}\right)-\frac{\lambda}{2}\|\boldsymbol{x}_t-\boldsymbol{x}_t^j\|^2\end{aligned} \tag{28}$$

附录 C.

C.1. 对于交叉熵损失函数平滑性的证明.

对于 m 个样本 $\boldsymbol{X}=(\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_m)^{\mathrm{T}}\in\mathbb{R}^{m\times d}$，设分类模型权重为 $\boldsymbol{w}\in\mathbb{R}^d$，真实标签为 $\boldsymbol{y}\in\{0,1\}^m$，作出的预测为 $\hat{\boldsymbol{y}}\in(0,1)^m$，其中 $\hat{y}_k=\dfrac{1}{1+e^{-\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_k}}$，那么交叉熵损失写作

$$f(\boldsymbol{w})=-\frac{1}{m}\sum_{k=1}^{m}(y_k\log\hat{y}_k+(1-y_k)\log(1-\hat{y}_k))=\frac{1}{m}\sum_{k=1}^{m}(1-y_k)\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_k+\frac{1}{m}\sum_{k=1}^{m}\log(1+\mathrm{e}^{-\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_k}),$$

其对应的黑塞矩阵为

$$\boldsymbol{H}=\frac{1}{m}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}\in\mathbb{R}^{d\times d},$$

其中 $\boldsymbol{A}\in\mathbb{R}^{m\times m}$ 为对角矩阵，$A_{k,k}=\hat{y}_k(1-\hat{y}_k)\leq1/4$.

对于 H-平滑函数 $f(\cdot)$，等价地有 $\nabla^2 f(\boldsymbol{x})\leq H\boldsymbol{I}$，即 $H\boldsymbol{I}-\nabla^2 f(\boldsymbol{x})$ 是正定矩阵. 由于交叉熵损失的黑塞矩阵 $\boldsymbol{H}$ 是实对称矩阵，因此 $H\boldsymbol{I}-\boldsymbol{H}$ 的埃尔米特矩阵 $(H\boldsymbol{I}-\boldsymbol{H})^H$ 等于自身. 而矩阵 $\boldsymbol{M}$ 正定等价于 $\mathrm{eig}\left(\dfrac{\boldsymbol{M}+\boldsymbol{M}^H}{2}\right)>0$，对于实对称的 $H\boldsymbol{I}-\boldsymbol{H}$ 则是 $\mathrm{eig}(H\boldsymbol{I}-\boldsymbol{H})>0$，因此我们需要分析 $\boldsymbol{H}$ 的最大特征值的上界.

我们接着对式(28)中的根号项进行分析. 代入式(14)和(16)中损失向量 ℓ_t 与乐观项 $\boldsymbol{m}_t$ 的定义，得到

$$\begin{aligned}&\sqrt{\ln(NT)\sum_{t=1}^{T}(\ell_t^j-m_t^j)^2}\overset{(14),(16)}{=}\\&\sqrt{\frac{1}{3DG}\ln(NT)\sum_{t=1}^{T}\langle\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t),\boldsymbol{x}_t^j-\boldsymbol{x}_t\rangle^2}\leq\\&\sqrt{\frac{1}{3DG}\ln(NT)\sum_{t=1}^{T}\|\nabla f_t(\boldsymbol{x}_t)-\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)\|^2\|\boldsymbol{x}_t^j-\boldsymbol{x}_t\|^2}\leq\\&\sqrt{\frac{1}{3DG}\ln(NT)\sum_{t=1}^{T}2(\|\nabla f_t(\boldsymbol{x}_t)\|^2+\|\nabla f_{t-1}(\bar{\boldsymbol{x}}_t)\|^2)\|\boldsymbol{x}_t^j-\boldsymbol{x}_t\|^2}\overset{(12)}{\leq}\\&\sqrt{\frac{4G}{3D}\ln(NT)\sum_{t=1}^{T}\|\boldsymbol{x}_t^j-\boldsymbol{x}_t\|^2}\leq\frac{2G}{3D\lambda}\ln(NT)+\frac{\lambda}{2}\|\boldsymbol{x}_t^j-\boldsymbol{x}_t\|^2\end{aligned} \tag{29}$$

其中，第二步用到了柯西-施瓦茨不等式 $\langle\boldsymbol{x},\boldsymbol{y}\rangle^2\leq\|\boldsymbol{x}\|^2\|\boldsymbol{y}\|^2$，最后一步用到了对于 $\forall a,b>0$ 的基本不等式 $2\sqrt{ab}\leq a+b$. 结合式(28)、(29)以及 N 的定义(17)，我们得到 UAGV 算法对于强凸在线函数的元界

$$\begin{aligned}&\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\sum_{t=1}^{T}f_t(\boldsymbol{x}_t^j)\overset{(28)}{=}\\&\mathcal{O}\left(\ln(NT)+\sqrt{\ln(NT)\sum_{t=1}^{T}(\ell_t^j-m_t^j)^2}\right)-\frac{\lambda}{2}\|\boldsymbol{x}_t-\boldsymbol{x}_t^j\|^2\overset{(29)}{=}\\&\mathcal{O}\left(\left(1+\frac{2G}{3D\lambda}\right)\ln(NT)\right)+\frac{\lambda}{2}\|\boldsymbol{x}_t^j-\boldsymbol{x}_t\|^2-\frac{\lambda}{2}\|\boldsymbol{x}_t-\boldsymbol{x}_t^j\|^2\overset{(17)}{=}\\&\mathcal{O}\left(\frac{\log T}{\lambda}\right)\end{aligned} \tag{30}$$

对于 $A\in\mathcal{A}_{str}$，存在 $\hat{\lambda}\in\mathcal{P}_{str}$ 满足 $\hat{\lambda}\leq\lambda\leq2\hat{\lambda}$. 由于 λ-强凸函数也是 $\hat{\lambda}$-强凸的，因此在所有采用算法 A 的强凸专家中，专家 $E(A,\hat{\lambda})$ 将给出具有最佳理论保障的遗憾界，为

$$\sum_{t=1}^{T}f_t(\boldsymbol{x}_t^j)-\min_{\boldsymbol{x}\in\mathcal{X}}\sum_{t=1}^{T}f_t(\boldsymbol{x})\leq R(A,\hat{\lambda}) \tag{31}$$

结合元界(30)与专家界(31)，最终得到 UAGV 算法对于强凸在线函数的遗憾界

$$\sum_{t=1}^{T}f_t(\boldsymbol{x}_t)-\min_{\boldsymbol{x}\in X}\sum_{t=1}^{T}f_t(\boldsymbol{x})\leq\min_{A\in\mathcal{A}_{str}}R(A,\hat{\lambda})+\mathcal{O}\left(\frac{\log T}{\lambda}\right).$$

证毕.

由于对称矩阵的迹等于其特征值之和，则有

$$\max(\mathrm{eig}(\boldsymbol{H}))\leq \mathrm{tr}(\boldsymbol{H})=\mathrm{tr}\left(\frac{1}{m}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}\right)\leq \mathrm{tr}\left(\frac{1}{4m}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right).$$

当样本的二范数有界，即 $\|\boldsymbol{x}_k\|\leq D_s$，$\forall k\in[m]$ 时，有

$$\mathrm{tr}\left(\frac{1}{4m}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)\leq \frac{1}{4m}\cdot mD_s^2=\frac{D_s^2}{4}.$$

因此我们得到 $\max(\mathrm{eig}(\boldsymbol{H}))\leq D_s^2/4$，即交叉熵损失为 $D_s^2/4$-平滑的. 需要注意的是，这是一个比较松的估计，因为并不是所有样本的二范数都能够达到上界 D_s；同时 $A_{k,k}=\hat{y}_k(1-\hat{y}_k)\leq 1/4$ 在 $\hat{y}_k=1/2$ 处才取到等号，而在进行了一段时间的模型更新后大部分预测 $\hat{y}_k$ 会倾向于 0 或 1，则有 $A_{k,k}<1/4$. 证毕.

LIU Lang-Qi, M. S. candidate. His research interests include online learning and data mining.

ZHANG Li-Jun, Ph. D., professor. His research interests include machine learning and optimization.

Background

Utilizing machine learning techniques allows for data analysis and the construction of associated mathematical models. Nonetheless, employing traditional batch-based machine learning methods for model training requires numerous complete iterations over data and retraining upon every data increment, posing challenges for rapidly evolving environments and settings of continuous learning. In response to these challenges, researchers have put forth the concept of online learning. This paradigm processes sequential data input via iterative updates, thereby reducing the complexity of updating models compared to its traditional counterparts.

Online convex optimization (OCO) represents an important branch of online learning. Traditional OCO algorithms are designed for specific convex function types and often require prior knowledge about the function-related moduli, posing a usage barrier. In contrast, universal OCO algorithms can automatically adapt to both the function type and moduli, proving to be more user-friendly. At present, while existing universal algorithms can achieve minimax optimal bound and problem-dependent bound for both strongly-convex and exponentially-concave functions, they fail to acquire problem-dependent bound, such as gradient-variation bound, for general convex functions. To remedy this issue, our paper introduces the Universal Online Convex Optimization Algorithm with Adaptivity to Gradient-Variation (UAGV), which leverages a meta-algorithm to integrate the predictions of individual expert algorithms. Through theoretical analysis, UAGV provides a gradient-variation bound guarantee for general convex functions under smooth condition and automatically adapts to strongly convex functions.

This work is supported by the National Natural Science Foundation of China (Nos. 62122037 and U23A20382).