带记忆成本的在线优化鲁棒化学习

Pengfei Li JianyiYang ShaoleiRen加州大学河滨分校 加州大学河滨分校 加州大学河滨分校

arXiv:2305.00677v1 [cs.LG] 1 May 2023

摘要——**带记忆成本的在线优化具有许多现实世界中的应用场景，其中需要在不知道未来输入的情况下依次做出动作。然而，记忆成本会使各时刻的动作相互耦合，从而带来巨大的挑战。传统上，人们通过各种专家设计的在线算法来解决这一问题，目标是实现有界的最坏情况竞争比，但由此获得的平均性能往往并不理想。另一方面，新兴的基于机器学习（ML）的优化器能够提升平均性能，却缺乏最坏情况性能鲁棒性。本文提出了一种新颖的专家鲁棒化学习（ERL）方法，同时实现良好的平均性能和鲁棒性。具体而言，在鲁棒性方面，ERL 引入了一种新颖的投影算子，通过利用专家在线算法使机器学习动作具备鲁棒性；在平均性能方面，ERL 基于循环架构训练机器学习优化器，并明确考虑下游专家鲁棒化。我们证明，对于任意 $\lambda \geq 1$，ERL 相对于专家算法可以达到 λ-竞争比，相对于最优离线算法可以达到 $\lambda \cdot C$-竞争比（其中 C 是专家的竞争比）。此外，我们还将分析扩展到多步记忆成本这一新颖场景。最后，我们通过能源调度应用中的实证实验验证了分析结果。**

I. 引言

在线优化是一类经典的序贯决策问题，智能体在运行时选择不可撤销动作，却不知道未来的输入。此外，在许多实际应用中，人们非常希望动作随时间平滑变化。例如，在运动规划中，机器人受到速度和/或加速度限制，无法随意移动；在数据中心容量配置中，为避免过度的磨损成本和设置延迟，服务器不能频繁开关；在智能电网中的能源调度中，快速调整能源生产可能代价高昂 [23], [27], [32],[35]。因此，众多现实应用催生了带有记忆成本的在线优化问题，这类成本会对随时间频繁改变动作进行惩罚。

加入记忆成本可以为在线动作的平滑性提供关键的正则化，但也带来了显著的算法挑战。更具体地说，记忆成本实质上将多个时间步上的在线动作耦合起来，因此，在不知道未来信息的情况下，想要获得最优动作极具挑战性，甚至可能无法实现。传统上，人们通过专家针对各种设定设计的在线算法来应对这一挑战 [24], [27], [28], [35], [42]。即使面对对抗性输入，这些专家算法通常也能凭借有保证的竞争比实现最坏情况下的性能鲁棒性，但其保守性也ನ್ನಡ这意味着，在许多典型情况下，它们的平均表现可能并不理想。

近来，实际应用中历史数据的日益丰富，推动了人们采用机器学习（ML）方法解决优化问题 [3], [23], [32]。尤其是，基于离线训练的循环神经网络或强化学习的优化器不断涌现，用于解决各种在线优化问题，包括在线资源分配 [15], 、在线背包 [21], 等。这些基于机器学习的优化器利用有关问题输入的统计信息以及神经网络强大的预测能力，在经验上实现了前所未有的平均性能。然而，它们也存在一个重大缺陷——缺乏性能鲁棒性。具体而言，与具有鲁棒性保证的专家在线算法不同，基于机器学习的优化器的竞争比可能糟糕到任意程度，例如训练测试分布不同时、测试输入具有对抗性时，和/或模型容量受到严格限制时[6], [23], [32]。因此，缺乏鲁棒性使现有的基于机器学习的在线优化器无法用于许多实际应用，尤其是那些风险极高的应用。

为了充分发挥机器学习和专家设计的优势，近期已有研究提出机器学习增强在线算法 [7],[11], ，其中也包括我们关注的带内存成本的在线优化场景 [6], [23], [32]。这些研究最常见的目标是实现有限的竞争比（即鲁棒性），以限制机器学习输出任意糟糕时的最坏情况性能；同时实现较低的竞争比（即一致性），从而大致保留机器学习模型带来的良好平均性能。然而，对于我们的问题设置，要同时实现良好的鲁棒性和一致性仍面临重大挑战（参见广泛相关的算法 [6], [23], [32]），更不用说良好的一致性并不总能转化为良好的平均性能。此外，现有的机器学习增强算法通常将机器学习模型视为外生的黑箱，并将其作为独立模型进行预训练，而不考虑下游专家算法。这实质上造成了训练与测试之间的不匹配——机器学习模型单独训练，却与下游算法流程共同接受测试——从而可能不必要地损害最终的平均性能。

本文聚焦于带内存成本的在线优化，并提出一种新颖的专家鲁棒化学习（ERL）方法，兼具良好的平均性能和有保证的鲁棒性。ERL 的核心思想是让专家

和机器学习各自发挥所长：为保证鲁棒性，ERL 使用专家在线算法，通过将机器学习动作投影到精心设计的鲁棒动作空间中，对其进行鲁棒化；为实现良好的平均性能，ERL 在训练机器学习模型时显式考虑下游专家鲁棒化过程，从而避免训练与测试之间的不匹配。我们证明，对于任意控制机器学习动作灵活程度的信任超参数 $\lambda \geq 1$，ERL 相对于专家在线算法可达到 λ 竞争比，因此相对于最优离线算法可达到 $\lambda \cdot C$ 竞争比（其中 C 是专家的竞争比）。新增的鲁棒化步骤是一个隐式层，因此反向传播并非易事。为实现高效的端到端训练并提升平均情况下的性能，我们还推导了鲁棒化步骤相对于其输入的梯度。随后，我们将分析扩展到一种新颖的设定，其中记忆成本跨越多个步骤。最后，我们通过能源调度应用实验对 ERL 进行实证验证，结果表明，该方法能够在平均成本和竞争比之间实现最佳权衡。

II. 相关工作

具有（单步）记忆成本的在线优化问题，已在各种设定下通过专家算法得到了广泛研究，例如在线梯度下降（OGD）[43], 、在线平衡下降（OBD）[10], 以及正则化 OBD（R-OBD）[17]。此外，利用未来输入信息的专家算法还包括滚动时域控制（RHC）[13] 、承诺时域控制（CHC）[9],以及滚动时域梯度下降（RHGD）[24], [26]。这些算法经过精心设计，能够实现有界的竞争比和/或遗憾值，但它们在平均意义上的表现可能并不理想。

机器学习增强算法设计也开始出现在带内存成本的在线优化的上下文中 [6],[32]。然而，这些算法只是将外生机器学习优化器产生的动作作为额外输入；它们仍然侧重于手动设计，无法同时实现良好的最坏情况性能和平均性能。例如，为了通过设置超参数 $\delta \to 0$ 保留机器学习动作的良好平均性能，当机器学习动作任意糟糕时，对于 α-多面体成本函数 [32]，竞争比最高可达 $12+o\left(\frac{1}{\delta}\left(\frac{2}{\alpha+\delta}\right)^{(1+\alpha)}\right)^{2/(\delta\alpha)}$。研究 [23] 考虑了平方单步切换成本，并训练机器学习模型来规范化在线动作，但其最坏情况竞争比是无界的。在正交的上下文中，通过假设给定的下游算法，[16] 针对 Count-Min Sketch 问题重新训练机器学习模型。因此，新颖的专家鲁棒化（用于可调且有界的性能鲁棒性）、端到端训练（用于良好的平均性能）以及全新的问题设定，使我们的工作与现有文献截然不同。

基于离线训练的循环神经网络或强化学习的学习优化（L2O） [22]近期已应用于在线优化，包括在线资源分配和在线二分图匹配 [15], [21]。尽管有对抗训练的帮助， [14],现有基于机器学习的优化器仍有一项关键缺陷：缺乏有保证的性能鲁棒性，因此不适用于高风险应用。由于未来输入未知且动作不可撤销，在学习优化（L2O）和传统求解器之间择优选用的朴素方法 [20] 并不适用于在线优化。

ERL 与近期的决策聚焦学习框架 [38]相关。不过，ERL 不仅仅是训练机器学习模型，还提出了一种新颖的专家鲁棒化框架。此外，ERL 直接使用鲁棒化动作来确定训练损失，而现有的决策聚焦学习则要求训练损失中包含真实标签。

最后，ERL 与多臂老虎机和强化学习中的保守探索 [40]也有交集。保守探索关注未知奖励函数（以及在适用时的状态转移模型），并利用现有策略引导探索过程，以确保鲁棒性。但在设计上，两者有显著差异：保守探索在每一步选择动作（或在回合式强化学习中为每个回合选择策略）时，无需考虑未来输入不确定性；也就是说，它只关注累积奖励。相比之下，ERL 必须保留预留成本，以确保无论未来输入如何，都始终存在可行解，从而实现有保证的确定性最坏情况竞争比（而非概率保证）。即使假设奖励函数（以及状态转移模型）完全已知，现有保守多臂老虎机／强化学习[40] 所用的鲁棒化规则也无法应用于我们的问题以实现有保证的竞争比，这进一步凸显了这一关键点。其他相关问题包括约束策略优化和安全强化学习 [30],[39]。这些研究致力于约束平均安全成本，和／或避免某些危险状态（可能以较高概率实现）。相比之下，ERL 的目标不同，它通过引入一种新颖的专家鲁棒化步骤，在任何情况下都能保证竞争比有界。

III. 单步记忆成本的表述

为便于读者理解，我们先从单步记忆成本（又称切换成本 [27], [28], [42]）入手。将包含 T 个时间步的序列视为一个问题实例。在每个时间步 $t = 1, \cdots, T$，智能体会收到命中成本的上下文向量/参数 $y_t \in \mathcal{Y} \in \mathbb{R}^m$，采取一项不可撤销的 动作 $x_t \in \mathcal{X} \subseteq \mathbb{R}^d$，随后产生 $f(x_t, y_t) \geq 0$的命中成本。为鼓励动作随时间平滑变化，智能体还会产生记忆成本 $d(x_t, x_{t-1}) \geq 0$，该成本根据度量空间中相邻两个动作之间的距离来定义。具体而言，我们考虑 $d(x_t, x_{t-1}) = \|x_t - x_{t-1}\|$，其中 $\|\cdot\|$ 表示带有 $p \geq 1$的 l_p 范数。因此，智能体的目标是最小化序列中 T 个时间步内的命中成本与记忆成本之和，如下所示：

$$\min_{x_1,\cdots x_T} \sum_{t=1}^{T} f(x_t, y_t) + d(x_t, x_{t-1}), \tag{1}$$

其中，初始动作 x_0 作为额外输入提供。虽然也可以对总记忆成本施加约束，但将记忆成本作为在线动作的平滑正则项加入目标函数的表述，与现有文献 [23], [27], [32], [42]一致。

要最优地求解式 (1)，关键挑战在于记忆成本会将不同时刻的在线动作耦合起来，但 y_t 要到每一步 $t = 1, \cdots, T$ 开始时才会向智能体揭示。给定任意在线算法 π，我们将其针对输入上下文为 $\boldsymbol{s} = (x_0, \boldsymbol{y}) \in \mathcal{S} = \mathcal{X} \times \mathcal{Y}^T$的问题实例所产生的总成本记为 $\text{cost}(\pi, \boldsymbol{s}) = \sum_{t=1}^{T} f(x_t^\pi, y_t) + d(x_t^\pi, x_{t-1}^\pi)$，其中 $x_t^\pi, t = 1, \cdots, T$ 是算法 π 生成的动作。尽管 $\boldsymbol{s}$ 服从一种基于机器学习的优化器能够妥善处理的一般分布，但其中仍可能包含对抗性情形。为简便起见，我们将省略上下文参数，并将 $\text{cost}(x_{i:j}^\pi) = \sum_{t=i}^{j} f(x_t^\pi, y_t) + d(x_t^\pi, x_{t-1}^\pi)$ 记为，其中 $x_{i:j}^\pi = (x_i^\pi, \cdots, x_j^\pi)$ 是算法 π 下针对$t = i, \cdots, j$ 的动作。

定义 *1*（α-**多面体**）：给定上下文参数 $y \in \mathcal{Y}$，若命中成本函数 $f(x, y)$：$\mathcal{X} \mapsto \mathbb{R}^+$具有唯一极小值点 $x^* \in \mathcal{X}$，并且对于任意 $x \in \mathcal{X}$满足 $f(x, y) - f(x^*, y) \geq \alpha \cdot d(x, x^*)$，则称其对于 $\alpha > 0$ 是 α-多面体的。

定义 *2*（**竞争比**）：对于 $\lambda \geq 1$，若在线算法 ALG 针对算法π，在加性因子 $B \geq 0$ 下满足其总成本 $\text{cost}(ALG, \boldsymbol{s}) \leq \lambda \cdot \text{cost}(\pi, \boldsymbol{s}) + B$，则称其为 λ-竞争的，其中对于任意输入 $\boldsymbol{s} = (x_0, \boldsymbol{y})$均成立。

α-多面体的定义在文献中通常用于推导相对于最优离线算法的竞争比。定义 2 中的确定性竞争比是一般性的，加性因子 B 与问题输入 $\boldsymbol{s} = (x_0, \boldsymbol{y})$无关。令$B = 0$，即可得到严格竞争比 [19], [31], [32]。此外，当 $\lambda = 1$时，定义 2 中的加性因子 B 刻画了 ALG相对于算法 π所产生的遗憾。当未指定 π 时，竞争比默认是相对于最优离线算法 OPT。

IV. ERL：专家鲁棒化学习

本节中，我们考虑单步记忆成本，并展示 ERL 的设计。

A. 纯基于机器学习的优化器简介

要解决带内存成本的在线优化，一个自然的想法是利用机器学习的能力，找出可用在线信息到动作的映射。更具体地说，我们可以基于循环神经网络（RNN）或等价地通过强化学习，在线下预训练机器学习模型。我们将时刻 t 的机器学习动作记作$\tilde{x}_t = h_w(\tilde{x}_{t-1}, y_t)$，其中 w 是机器学习模型参数。其循环特性源于带记忆成本的序列式在线优化：给定先前的动作 $\tilde{x}_{t-1}$ 和当前输入 y_t，我们循环地产生在线动作 $\tilde{x}_t$。给定一组训练问题实例，可以通过最小化一个损失函数来学习机器学习模型参数 w；该损失函数可以是式（1）中的成本总和 [3]。

缺点：众所周知，这类基于机器学习的优化器存在显著缺点——缺乏鲁棒性。具体而言，由于多种原因，竞争比可能变得任意糟糕，例如分布偏移、困难的问题实例，甚至对抗性输入，和/或有限的机器学习容量 [5],[6], [32]。虽然分布鲁棒学习可以在平均意义上部分缓解鲁棒性不足的问题 [36], [41],，但仍无法保证机器学习模型在任何问题实例上都具有有界竞争比。

B. 专家鲁棒化

在不同设定下，已有多种专家算法可用于解决带内存成本的在线优化问题 [25],[42]。由于这些算法具有保守性，平均表现可能不佳，但它们能针对任意输入提供最坏情况性能鲁棒性。因此，我们希望借助一种专家算法 π 来对机器学习动作进行鲁棒化。对于每个 t，我们将鲁棒化前的机器学习动作记为 $\tilde{x}_t$，专家动作记为 x_t^π，鲁棒化后的动作记为 x_t。

一种简单的想法是在训练过程中加入适当的正则项，当总成本超过专家成本的 λ 倍时施加惩罚。但这行不通，因为在线推理期间出现糟糕的问题实例时，机器学习动作仍可能违反鲁棒性要求。另一种做法是约束鲁棒化后的动作，使得对于任意 $t = 1, \cdots, T$，截至时间 t 的累积成本满足 $\text{cost}(x_{1:t}) \leq \lambda\text{cost}(x_{1:t}^\pi) + B$，其中 $\text{cost}(x_{1:t})$ 和 $\text{cost}(x_{1:t}^\pi)$ 分别是 ERL 和专家的累积成本（假设专家独自运行其算法）。然而，这很容易导致 ERL 的可行行动集合为空，因为记忆成本会使不同时刻的动作相互耦合。为说明这一点，假设 $\text{cost}(x_{1:t}) = \lambda\text{cost}(x_{1:t}^\pi) + B$，但在时间 t时 $x_t \neq x_t^\pi$。那么，在时间 $t + 1$，专家的总成本可能低至 $f(x_{t+1}^\pi, y_{t+1}) + d(x_{t+1}^\pi, x_t^\pi)$，以至于即使令 $x_{t+1} = x_{t+1}^\pi$（即跟随专家），也会因较大的记忆成本 $d(x_{t+1}^\pi, x_t)$ 而违反约束 $\text{cost}(x_{1:t+1}) \leq \lambda\text{cost}(x_{1:t+1}^\pi) + B$。因此，在这种情况下，没有任何动作能够保证鲁棒性。

现在，我们介绍一种新颖的鲁棒化框架，称为 ERL。为实现鲁棒性，ERL 的关键在于防范偏离专家动作的风险，以应对未来不确定性。具体而言，在每一步 t，我们通过求解以下问题，将机器学习动作 $\tilde{x}_t$ 投影到由专家 π 指定的鲁棒动作空间中：

$$\begin{aligned} x_t = \arg\min_{x \in \mathcal{X}} \frac{1}{2}\|x - \tilde{x}_t\|^2 \\ s.t. \quad \text{cost}(x_{1:t-1}) + f(x, y_t) + d(x, x_{t-1}) + d(x, x_t^\pi) \\ \leq \lambda\text{cost}(x_{1:t}^\pi) + B \end{aligned} \tag{2}$$

其中，$\lambda \geq 1$ 和 $B \geq 0$ 是表示鲁棒性要求程度的超参数。我们将此投影步骤表示为 $x_t = \text{proj}(\tilde{x}_t, x_t^\pi, \text{cost}(x_{1:t-1}), \text{cost}(x_{1:t}^\pi))$。重要的是，在公式（2）中约束时间 t 的鲁棒化后累积成本时，关键在于添加预留成本 $d(x_t, x_t^\pi)$。这样做可以确保，如果该约束在

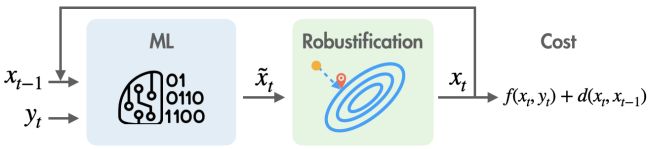


图 1：ERL。对于每个在线输入，我们首先运行前向推理以获得机器学习动作，然后将其投影到经过专家鲁棒化的动作空间中，作为实际动作。

算法 1 专家鲁棒化学习（ERL）

要求： $\lambda \geq 1$, $B \geq 0$, 初始 x_0, 训练好的机器学习模型（第 IV-C 节），以及专家在线算法 π

1: 对于 $t = 1, \cdots, T$
2: 接收上下文 y_t
3: 专家选择 x_t^π，机器学习选择 $\tilde{x}_t \leftarrow h(x_{t-1}, y_t)$
4: $x_t \leftarrow$ 基于 proj($\tilde{x}_t, x_t^\pi$,cost($x_{1:t-1}$),cost($x_{1:t}^\pi$)) on 公式（2） //鲁棒化

时间 t满足，那么无论输入如何，在时间 $t+1$ 也都将满足——通过选择 $x_{t+1} = x_{t+1}^\pi$ 遵循专家始终是一个可行的解决方案。

ERL 推理过程如图 1 所示，并在算法 1 中进行了描述。在每个步骤 t，我们运行机器学习模型以生成一个动作 $\tilde{x}_t$，获取专家的动作 x_t^π，然后通过求解公式（2），将 $\tilde{x}_t$ 投影到鲁棒化动作空间中。注意，专家在线算法将上下文 y_t 作为输入，并独立地遵循自身轨迹输出其动作 x_t^π，不受机器学习动作的影响。接下来，我们将正式给出 ERL 的鲁棒性分析。

定理 4.1： 设 π 是方程 (1) 中问题的任意专家在线算法。对于任意 $\lambda \geq 1$ 和 $B \geq 0$，ERL 相对于 π 具有 λ 竞争比，且有一个加性因子 B，即对于任意输入 $\boldsymbol{s} = (x_0, \boldsymbol{y})$，cost(ERL $\boldsymbol{s}$) $\leq \lambda \cdot$ cost($\pi, \boldsymbol{s}$) $+ B$。

定理 4.1 在附录 B 中得到证明，它通过表明 ERL 能够针对任意专家算法 π，实现相对于 $B \geq 0$ 的任意竞争比 $\lambda \geq 1$，展示了 ERL 的强大能力。这里，对于任意 $B \geq 0$，超参数 $\lambda \geq 1$ 可以视为信任 参数：λ 越高，我们对机器学习动作 $\tilde{x}_t$ 的信任程度就越高，因此有可能以更高的竞争比为代价，实现更低的平均成本。加性因子 $B \geq 0$ 表示松弛量，而 $B = 0$ 则退化为严格竞争比定义。如果我们将 $\lambda = 1$ 和 B 设置为关于 T 的次线性函数，那么即使在最坏情况下，随着 $T \to \infty$ 的增长，ERL 也能保证渐近地不劣于专家 π。

ERL 的竞争比 相对于 OPT。 人们也可能希望获得相对于最优离线算法 OPT 的有界竞争比。为此，我们考虑一种最先进的专家在线算法，称为 Robust，该算法在每一步最小化命中成本，而不考虑记忆成本。这个简单的在线算法却出人意料地实现了良好的竞争比 $\max\left(\frac{2}{\alpha}, 1\right)$ 相对于 OPT，适用于 α-多面体命中成本函数 [42]。应用定理 4.1，我们得到以下推论。

推论 4.1： 考虑 Robust 作为算法 π，它对任意 t，选择 $x_t = \arg\min_{x \in \mathcal{X}} f(x, y_t)$。假设命中成本函数 $f(x, y_t)$ 是 α-多面体的。对于任意$\lambda \geq 1$和 $B \geq 0$，在加性因子为 B 的条件下，ERL 相对于 OPT 是 $\lambda \cdot \max\left(\frac{2}{\alpha}, 1\right)$-竞争的，即对于任意输入 $\boldsymbol{s} = (x_0, \boldsymbol{y})$，cost(ERL $\boldsymbol{s}$) $\leq \lambda \cdot \max\left(\frac{2}{\alpha}, 1\right) \cdot$ cost(OPT $\boldsymbol{s}$) $+ B$。

C. 端到端训练

在传统的机器学习增强算法 [6], [12], 中，机器学习模型经过训练后会自主生成良好的动作，而不会考虑下游的修改（即 ERL 中的专家鲁棒化）。虽然设计的算法有时仍能保留良好的机器学习动作（即所谓的一致性 [6]），但训练过程与测试过程之间仍存在不匹配——训练过程产生的是良好的鲁棒化前机器学习动作，但实际用于测试的却是鲁棒化后的动作 [3], [23], [29], [38]。因此，为了提升 ERL 的平均 性能，我们需要在训练机器学习模型时明确考虑投影步骤。

端到端训练颇具挑战性，因为公式（2）中的投影步骤本身就是一个优化问题，因此是一个隐式层。此外，与典型的可微优化器不同， [1], [4], [38], 由于我们的在线优化问题具有循环特性，我们需要推导梯度，以执行时间反向传播。令 w 表示 RNN 中每个基础机器学习模型 $\tilde{x}_t = h_w(x_{t-1}, y_t)$ 的权重，如图 1 所示。我们需要推导成本函数($x_{1:T}$)相对于 w 的梯度，如下所示：

$$\nabla_w \text{cost}(x_{1:T}) = \sum_{t=1}^{T} \nabla_w \big(f(x_t, y_t) + d(x_t, x_{t-1})\big), \qquad (3)$$

其中，$x_t = \text{proj}(\tilde{x}_t, x_t^\pi, \text{cost}(x_{1:t-1}), \text{cost}(x_{1:t}^\pi))$ 是步骤 $t = 1, \cdots, T$ 中鲁棒化后的动作。因此，总梯度可通过对 T 个步骤的所有梯度求和来计算。对步骤 t 应用链式法则，可得 $\nabla_w\big(f(x_t, y_t) + d(x_t, x_{t-1})\big) = \nabla_{x_t}\big(f(x_t, y_t) + d(x_t, x_{t-1})\big)\nabla_w x_t + \nabla_{x_{t-1}} d(x_t, x_{t-1}) \cdot \nabla_w x_{t-1}$，其中 $\nabla_w x_t = \big(\nabla_{\tilde{x}_t} x_t \nabla_w \tilde{x}_t + \nabla_{\text{cost}(x_{1:t-1})} x_t \nabla_w \text{cost}(x_{1:t-1})\big)$。给定 f 和 d 的显式形式，即可得到梯度 $\nabla_{x_t}\big(f(x_t, y_t) + d(x_t, x_{t-1})\big)$ 和 $\nabla_{x_{t-1}} d(x_t, x_{t-1})$； $\nabla_w \tilde{x}_t$ 可通过机器学习模型（例如神经网络）中的反向传播轻松计算，而 $\nabla_w \text{cost}(x_{1:t-1})$ 则递归计算至 $t = 1$。因此，关键在于推导投影算子 $x_t = \text{proj}(\tilde{x}_t, x_t^\pi, \text{cost}(x_{1:t-1}), \text{cost}(x_{1:t}^\pi))$ 关于机器学习动作 $\tilde{x}_t$ 和 cost($x_{1:t-1}$) 的梯度。我们在以下命题中基于 KKT条件 [8] 给出结果。

Proposition 4.2 (Gradient by KKT conditions): Assume that x_t and μ are the primal and dual solutions to Eqn. (2), respectively. Let $\Delta_{11} = I + \mu\Big(\nabla_{x_t, x_t}\big(f(x_t, y_t) + d(x_t, x_{t-1})\big) + \nabla_{x_t, x_t} d(x_t, x_t^\pi)\Big)$, $\Delta_{12} = \nabla_{x_t}\big(f(x_t, y_t) + d(x_t, x_{t-1})\big) + \nabla_{x_t} d(x_t, x_t^\pi)$, $\Delta_{21} = \mu\Big(\nabla_{x_t}\big(f(x_t, y_t) + d(x_t, x_{t-1})\big) + \nabla_{x_t} d(x_t, x_t^\pi)\Big)^\top$, $\Delta_{22} = f(x_t, y_t) + d(x_t, x_{t-1}) + d(x_t, x_t^\pi) +$

$\text{cost}(x_{1:t-1}) - [\lambda\text{cost}(x^{\pi}_{1:t}) + B]$。投影操作 $x_t = \text{proj}(\tilde{x}_t,\ x^{\pi}_t,\ \text{cost}(x_{1:t-1}),\ \text{cost}(x^{\pi}_{1:t}))$ 关于 $\tilde{x}_t$ 和 $\text{cost}(x_{1:t-1})$ 的梯度为

$$\nabla_{\tilde{x}_t} x_t = \Delta_{11}^{-1}[I + \Delta_{12} Sc(\Delta, \Delta_{11})^{-1}\Delta_{21}\Delta_{11}^{-1}],$$

$$\nabla_{\text{cost}(x_{1:t-1})} x_t = \Delta_{11}^{-1}\Delta_{12} Sc(\Delta, \Delta_{11})^{-1}\mu,$$

其中，$Sc(\Delta, \Delta_{11}) = \Delta_{22} - \Delta_{21}\Delta_{11}^{-1}\Delta_{12}$ 是分块矩阵 $\Delta = [[\Delta_{11}, \Delta_{12}], [\Delta_{21}, \Delta_{22}]]$ 中 Δ_{11} 的舒尔补。

我们指出，如果舒尔补 $Sc(\Delta, \Delta_{11})$ 不满秩（例如，机器学习动作 $\tilde{x}_t$ 位于公式（2）中动作空间的边界上），或者命中成本函数 f 或记忆成本 d 对某些 x_t 不可微，我们仍然可以根据命题 4.2 近似计算梯度，以进行反向传播。具体而言，如果 $Sc(\Delta, \Delta_{11})$ 不满秩，可以使用 $Sc(\Delta, \Delta_{11})$ 的伪逆；如果 f 或 d 在 x_t 处不可微，可以用其次梯度代替。这也是训练机器学习模型（尤其是神经网络 [18]）时处理不可微点的一种常见技巧。例如，我们通常将 0 用作 $ReLu(x)$ 在 $x = 0$ 处的次梯度。重要的是，命题 4.2 提供了一种实用且便捷的反向传播方法。

训练。与典型的基于机器学习的在线优化方法一样 [2], [3], [14], [21], ，我们利用命题 4.2 中的梯度，并明确考虑投影过程，基于预先收集的历史问题实例训练机器学习模型。此外，我们也可以在在线推理期间收集新问题实例的批次，从而在线更新机器学习模型。训练过程可以使用总成本 $\sum_i \text{cost}_i(x_{1:T})$ 作为损失进行监督，其中 i 是训练问题实例的索引。

V. 扩展至多步记忆成本

受高阶动力学平滑性的启发，我们现在转向一个更一般的情形，其中记忆成本可以跨越多个时间步：$d(x_t,\ x_{t-q:t-1}) = \|x_t - \sum_{i=1}^{q} C_i x_{t-i}\|$，其中 $q \geq 1$ 是记忆长度，$C_i \in \mathbb{R}^{d\times d}$ 则因问题而异。例如，考虑一个机器人运动规划问题，其中 x_t 表示时间 t 的位置，并且我们非常重视加速度平滑性。在这种情况下，记忆成本可以写为 $d(x_t, x_{t-2:t-1}) = \|(x_t - x_{t-1}) - (x_{t-1} - x_{t-2})\| = \|x_t - 2x_{t-1} + x_{t-2}\|$，对此我们可以设定 $C_1 = -2\cdot I$、$C_2 = I$ 和 $q = 2$，其中 I 是 $\mathbb{R}^{d\times d}$ 中的单位矩阵。请注意，[35] 中的专家算法采用相同形式的多步记忆结构，但考虑的是平方 记忆成本，并附带其他强假设（例如强凸命中成本），因此需要采用完全不同的技术 [42]。据我们所知，我们的工作是首个研究度量空间中多步记忆成本的工作。

专家鲁棒化。鉴于存在多步记忆成本，我们的机器学习模型的输入包括 y_t 和 $x_{t-q:t-1}$，并输出 $\tilde{x}_t$，随后通过求解以下问题使其具备 Robust 性：

$$\begin{aligned} x_t = \arg\min_{x\in\mathcal{X}} \frac{1}{2}\|x - \tilde{x}_t\|^2 \\ s.t. \quad \text{cost}(x_{1:t-1}) + f(x, y_t) + d(x, x_{t-q:t-1}) \\ + G(x, x_{t-q:t-1}, x^{\pi}_{t-q:t}) \leq \lambda\text{cost}(x^{\pi}_{1:t}) + B, \end{aligned} \tag{4}$$

其中，预留成本 $G(x,\ x_{t-q:t-1},\ x^{\pi}_{t-q:t})$ 定义为

$$\begin{aligned} & G\left(x, x_{t-q:t-1}, x^{\pi}_{t-q:t}\right) \\ = & \sum_{k=1}^{\min(q, T-t)} \left\| C_k x + \sum_{i=1}^{q-k} C_{k+i} x_{t-i} - \sum_{i=0}^{q-k} C_{k+i} x^{\pi}_{t-i} \right\|. \end{aligned} \tag{5}$$

方程（5）的关键见解是，我们需要考虑 ERL 相较于专家算法 π 在未来最多 q 步内可能产生的更高记忆成本。通过在每一步为累积成本预留相应成本，我们可以确保 ERL 在未来始终能够回退到专家的动作，而不会违反鲁棒性要求。ERL 推理过程仍遵循算法 1，唯一的区别是，第 5 行中用于专家鲁棒化的投影步骤基于方程（4）。

Robust 的竞争比 `Robust`。Robust 在单步记忆设置下具有有界竞争比 [42], ，但在多步设置下是否如此尚不明确。这里，我们证明 Robust 在多步记忆情形下同样具有竞争性。证明见附录 A。

定理 5.1： 假设 $f(\cdot,\ y_t)\colon \mathcal{X} \mapsto \mathbb{R}$ 是 α-多面体的，且记忆成本定义为 $d(x_t,\ x_{t-q:t-1}) = \|x_t - \sum_{i=1}^{q} C_i x_{t-i}\|$，其中 $t = 1,\ \cdots,\ T$，$C_i \in \mathbb{R}^{d\times d}$ 且 $\sum_{i=1}^{q}\|C_i\| = \beta$，其中 $\|C_i\|$ 是由 l_p 向量范数诱导的矩阵范数。对于任意 $t = 1,\ \cdots,\ T$，选择 $x_t = \arg\min_{x\in\mathcal{X}} f(x,\ y_t)$ 的 Robust 算法相对于 OPT 严格具有最大 $\binom{\beta+1}{\alpha}, 1)$-竞争性，即对于任意输入 $\boldsymbol{s} = (x_0,\ \boldsymbol{y})$，$\text{cost}(\text{Robust}\ \boldsymbol{s}) \leq \max\binom{\beta+1}{\alpha}, 1) \cdot \text{cost}(\text{OPT}\ \boldsymbol{s})$。

ERL 的竞争比 `ERL`。在多步记忆成本情形下，定理 4.1 仍然成立。也就是说，对于任意 $\lambda \geq 1$ 和 $B \geq 0$，ERL 仍能以 λ 的竞争比与任意专家在线算法 π 竞争，但需加上加性因子 B。此外，将这一结果与定理 5.1 结合，可得以下推论（证明见附录 B）。

推论 5.1： 设专家 π 为 Robust，选择 $x_t = \arg\min_{x\in\mathcal{X}} f(x,\ y_t)$，其中任意 $t = 1,\ \cdots,\ T$ 均成立。在与定理 5.1 相同的假设下，对于任意 $\lambda \geq 1$ 和 $B \geq 0$，ERL 相对于 OPT 的竞争比为 $\lambda\max\binom{\beta+1}{\alpha}, 1)$，但需加上加性因子 B，其中 $\beta = \sum_{i=1}^{q}\|C_i\|$；也就是说，对于任意输入 $\boldsymbol{s} = (x_0,\ \boldsymbol{y})$，$\text{cost}(\text{ERL}\ \boldsymbol{s}) \leq \lambda\max\binom{\beta+1}{\alpha}, 1) \cdot \text{cost}(\text{OPT}\ \boldsymbol{s}) + B$。

最后，对于端到端训练，可以推导方程（4）中投影关于 $\tilde{x}_t$ 和 $\text{cost}(x_{1:t-1})$ 的梯度，并按照第 IV-C 节中的步骤训练机器学习模型。为简洁起见，这里省略相关内容。

VI. 实验结果

为了通过实证验证 ERL，我们考虑存在不确定的可再生能源时的动态能源调度应用。具体而言，为实现可持续发展，风能和太阳能等可再生能源正大规模接入电网。然而，其可用性极不稳定，受天气状况和设备效率等多种因素影响。另一方面，平衡电力需求与发电量对于确保电网稳定性至关重要——一旦两者不匹配，就需要迅速利用其他可能更昂贵的能源进行补偿。因此，电网运营商面临的一项挑战是如何根据实时可再生能源可用性动态安排能源生产，以满足净需求。生产量 x_t 与净需求 y_t 之间的不匹配需要使用昂贵的能源/储能进行补偿，因此会产生即时成本 $f(x_t, y_t) = \alpha\|x_t - y_t\|$；而随时间改变生产水平则会产生记忆 成本 $d(x_t, x_{t-1}) = \|x_t - x_{t-1}\|$（这是由于发电机爬坡/降坡成本所致）。因此，这是一个典型的具有记忆成本 [23], [25], [42]的在线优化问题。

A. 数据集

我们考虑利用轨迹数据和经验方程生成的间歇性可再生能源。具体而言，对于风力发电，步骤 t 产生的能量根据 [33] 建模为 $P_{\text{wind},\ t} = \frac{1}{2}\kappa_{\text{wind}}\varrho A_{\text{swept}}V^3_{\text{风}\ t}$。

符号说明如下：κ_{wind} 是风能的转换效率（%），ϱ 是空气密度（kg/m^3），A_{swept} 是涡轮机的扫掠面积（m^2），$V_{\text{wind}、\ t}$ 是时间步 t的风速（kW/m^2）。第 t 步产生的太阳能电量根据 [37]给出，为$P_{\text{solar}、\ t} = \frac{1}{2}\kappa_{\text{solar}}A_{\text{a\=rray}}I_{\text{rad}、\ t}(1 - 0.05 * (\text{温度}_t - 25))$。符号说明如下：$\kappa_{\text{solar}}$ 是太阳能电池板的转换效率（%），A_{array} 是阵列面积（m^2），$I_{\text{rad}、\ t}$ 是太阳辐射（kW/m^2），而 Temp$_t$是第 t步的温度（℃）。因此，在时间步 t，可再生能源产生的总能源$P_{\text{r}、\ t} = P_{\text{wind}、\ t} + P_{\text{solar}、\ t}$。假设在时间步 t，净能源需求为$y_t = \max(P_{\text{s}, \ t} - P_{\text{r},t}, 0)$，其中$P_{\text{s}、\ t}$是可再生能源并网前的需求。能源发电量是智能体的在线动作 x_t。我们将命中成本建模为动作 x_t 与上下文 y_t之差的缩放 l_2-范数，即 $f(x_t, y_t) = \alpha\|x_t - y_t\|$。此外，我们将切换成本建模为两个连续动作之差的 l_2-范数，即 $c(x_t, x_{t-1}) = \|x_t - x_{t-1}\|$。命中成本参数设为 $\alpha = 0.2$。风能参数设为 $\kappa_{\text{wind}} = 30\%$、$\varrho = 1.23kg/m^3$、$A_{\text{swept}} = 500、\ 000m^2$。太阳能参数设为 $\kappa_{\text{wind}} = 10\%$、$A_{\text{array}} = 10、\ 000m^2$。其他参数，如风速、太阳辐射和温度数据，均收集自国家太阳辐射数据库 [34]，其中包含2015年的详细每小时数据。

为了生成训练和测试用的数据集，我们使用滑动窗口生成多个每小时数据序列，每个序列长度为 25（即 24 个动作步加 1 个初始步）。对于每组连续的 25 个每小时数据，我们可以计算每个步骤/小时的上下文信息。我们将第一小时的能源发电量定义为初始动作 x_0。该问题可表述为：$\min_{x_1, \ \cdots x_T} \sum_{t=1}^{T} \alpha\|x_t - y_t\| + \|x_t - x_{t-1}\|$。我们使用 CVXPY 库求得离线最优解。

B. 实验设置

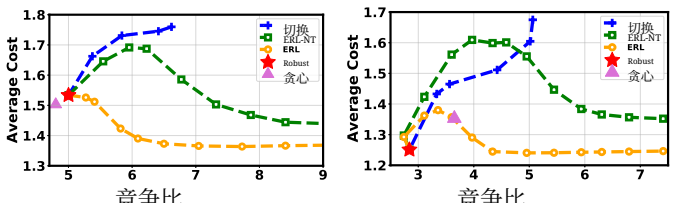
我们使用一个具有 2 个隐藏层的 RNN，每层包含 8 个神经元。训练该模型时，我们使用 2015 年前两个月（一月至二月）的数据，共包含 1440 个按小时采样的天气数据样本。具体而言，我们使用滑动窗口生成 1416 个数据序列。我们以批量大小为 50 对 RNN 模型训练 140 个训练轮次。该模型使用 PyTorch 库实现，在配备 8GB 内存和 M1 芯片的 2020 款 MacBook Air 上，训练过程通常需要约 3 分钟。在 ERL 中，我们设置松弛参数 $B = 0$，使其遵循竞争比的严格定义。默认情况下，我们使用 $\lambda = 1.4$ 训练 ERL。

为了评估不同算法的性能，我们将 2015 年剩余的 10 个月划分为五个时段，每个时段为两个月。共有三种不同情况：机器学习在平均性能和最坏情况性能方面均优于 Robust；机器学习的平均性能优于 Robust，但最坏情况性能较差；以及机器学习的平均性能和最坏情况性能均不如 Robust。第一种情况出现在三月至四月的测试时段，因为训练数据集和测试数据集中的数据彼此一致，这归因于它们的天气模式相似。接下来，我们重点关注另外两种情况，因为它们更有趣，也更具典型性：在实际应用中，训练数据集与测试数据集之间的数据分布偏移十分常见。这也与我们的主要贡献一致——使基于机器学习的优化器具备鲁棒性。

尽管我们也可以基于在线收集数据重新训练或更新机器学习模型（在 ERL、机器学习和 ERL-NT 中），但现有的基于机器学习的优化器通常是在离线环境下预训练的[2], [15]。因此，在测试其性能时，我们保持机器学习模型不变，以突出 ERL 中专家鲁棒化步骤的作用——无论测试分布如何，ERL 都能提供针对专家的、可证明的最坏情况竞争比保证。

C. 基线方法

我们将 ERL 与以下基线进行比较。**离线最优（OPT）：** OPT 拥有求解式 (1) 所需的全部上下文信息，可得到最优解；**鲁棒专家（Robust）：** Robust 是最先进的专家，对于 $t = 1, \cdots, T$，它选择 $x_t = \arg\min_{x \in \mathcal{X}} f(x, y_t)$，并保证竞争比为 [42]；**简单贪心（贪心）：** 贪心在每一步都贪心地最小化命中成本和记忆成本之和；**纯基于机器学习的优化器（机器学习）：** 机器学习使用与 ERL 相同的循环神经网络，但训练和推理时都不使用专家鲁棒化；**动态切换（切换）：** 切换根据阈值超参数 [6]在 Robust 和机器学习之间动态切换；**ERL-NoTraining（ERL-NT）：** ERL-NT 使用算法 1 进行推理，但机器学习模型作为独立优化器进行训练，不采用端到端训练。



(a) 九月至十二月测试　　(b) 五月至八月测试

图 2：归一化平均成本与经验竞争比。机器学习的数值远超图表范围：（左）：平均成本 1.45，竞争比 50+；（右）：平均成本 1.367，竞争比 11.167。

尽管贪心在经验上可能优于 Robust，但其竞争比并无可证明的上界，而 Robust 则有（单步记忆情形见 [42]，多步记忆情形见我们的定理 5.1）。因此，我们在 ERL 中使用 Robust 作为专家。

D. 结果

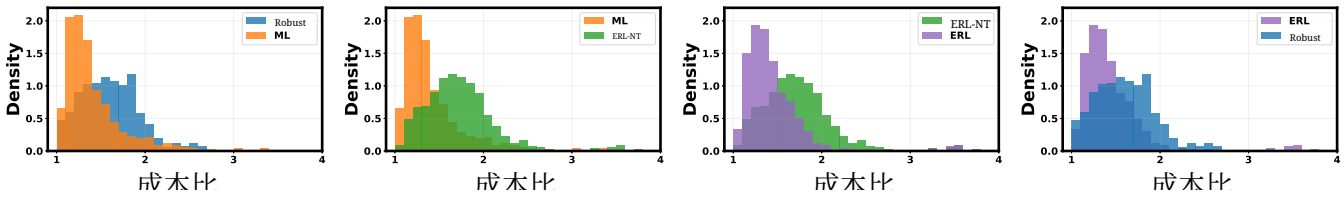
9 月至 12 月测试。我们在图 2 中展示平均成本与竞争比的实证结果。所有平均成本均以 OPT 的平均成本为基准进行归一化。ML 的平均成本低于 Robust，但其经验竞争比却高得多，这是由于基于机器学习的优化器普遍存在的缺陷——缺乏性能鲁棒性。具体而言，训练分布和测试分布在实践中很少完全相同，这可能导致 ML 的竞争比极差。在这种情况下，尽管贪心在实证上的表现优于 Robust，但它并不具备任何竞争比保证。我们看到，与 Robust 相比，切换的表现很差，因为它不考虑 Robust 或 ML 的实际表现，而是根据预设阈值进行硬切换。与 ML 相比，ERL-NT 通过专家鲁棒化可以实现低得多的竞争比，但平均成本也会大幅增加，甚至可能高于 Robust（因为 ERL-NT 中的机器学习模型训练并未考虑鲁棒化步骤）。另一方面，ERL 能够保证竞争比，且平均成本远低于 ERL-NT。这凸显了 ERL 通过明确考虑下游专家鲁棒化过程来训练机器学习模型的优势。有趣的是，我们还发现，通过适当设置超参数 λ（在我们的案例中约为 $1.4 \sim 1.8$），ERL 的平均成本甚至可以低于 ML。这是因为对于 ML 无法很好解决的那些困难问题实例，ERL 可以借助 Robust 提供指导，从而给出相当不错的解。

成本比分布。为了进一步深入了解，我们还在图 3 中展示了不同算法对的详细比较，比较依据是成本比的分布密度。观察图 3(a) 中的 Robust 与机器学习，可以看到，在更多情况下，机器学习的成本比较低于 Robust，尽管它有一条长尾（因坐标轴范围限制，图中未显示）。这解释了为什么当训练与测试分布差异不大时，机器学习的平均性能可能优于专家算法 Robust，不过，机器学习仍然缺乏鲁棒性，而 Robust 则不然。比较图 3(b) 中的机器学习与 ERL-NT，可以看到，专家鲁棒化会使成本比向右移动（即平均成本增加），但 ERL-NT 能保证鲁棒性。接下来，从图 3(c) 可以观察到，与 ERL-NT 相比，ERL 的成本比向左移动，这表明在训练 ERL 时明确考虑专家鲁棒化过程的重要性。图 3(d) 显示，ERL 的成本比远低于 Robust。再次说明，在训练过程中考虑专家鲁棒化至关重要。

……的影响 λ. 尽管 ERL 和 ERL-NT 都能通过推理阶段的专家鲁棒化步骤保证鲁棒性，但 ERL 中的机器学习模型在训练时会明确考虑专家，而 ERL-NT 则只是将机器学习模型作为独立优化器进行训练。因此，与 ERL-NT 相比，ERL 能进一步提升平均性能。为进一步凸显 ERL 训练过程中考虑专家鲁棒化步骤的必要性，我们在表 I 中展示了不同算法的结果。通过使用与测试时相同的 λ 来训练 ERL，我们在实证结果中同时获得了最低的平均成本和最佳的竞争比。具体而言，当 $\lambda = 1.4$ 时，平均性能方面的差异比 $\lambda = 1.2$时更加明显。这可以解释为：随着 $\lambda \geq 1$增大，专家通过降低对鲁棒性的重视程度，为机器学习模型提供了更大的自由度，因而发挥的作用也更小。随后，当$\lambda = 1.4$时，平均性能优于 $\lambda = 1.2$时，尽管其保证的竞争比更高（表 I 中的实证结果也验证了这一点）。作为参考，我们还展示了其他不受 $\lambda \geq 1$影响的算法的性能。

五月至八月测试。接下来，我们转向一个更具挑战性的情形：由于天气模式不同，训练与测试分布偏移较大，在五月至八月期间，Robust 的表现无论在平均情况下还是最坏情况下都优于机器学习。这在实践中并不少见，因为机器学习模型可能因缺乏鲁棒性而表现极差。结果如图 2(b) 所示。所有平均值均以 OPT 的平均成本为基准进行归一化。

同样，机器学习的表现极为突出，其竞争比为 11.167，平均成本为 1.367（两者均相对于 OPT 归一化）。与前一种情况类似，Switch 不如 Robust，因为它会在达到预先设定的阈值时，在 Robust 和机器学习之间进行硬切换，而不考察 Robust 或机器学习的实际表现。由于缺乏鲁棒性保证，在这种情况下，贪心的表现也不如 Robust。通过改变 $\lambda \geq 1$，我们发现 ERL-NT 的平均成本可能非常高（甚至高于机器学习），尽管其竞争比仍保证相对于专家 Robust 达到 λ-竞争比。另一方面，ERL 使用 $\lambda = 1.4$ 进行训练，并使用不同的 $\lambda \geq 1$ 进行测试，其平均成本远低于 ERL-NT，同时也能保证竞争比。这表明，在训练阶段意识到专家鲁棒化的重要性。



(a) (b) (c) (d)

图 3：九月至十二月测试的成本比概率分布。密度值的计算方法是，将每个区间内的测试实例数量除以实例总数和区间宽度。我们使用 $\lambda = 1.4$ 测试 ERL-NT 和 ERL。为便于观察，横轴范围限制为 4，成本比大于 4 的数据未显示在图中。

	ERL-NT		ERL (λ = 1.4)		ERL (λ = 1.2)		切换		Robust		贪心	
λ用于测试	Avg	CR	Avg	CR	Avg	CR	Avg	CR	Avg	CR	Avg	CR
λ = 1.4	1.6977	6.0912	**1.3903**	6.0910	1.4343	6.0910	1.7454	6.4130	1.5336	5.000	1.5030	4.800
λ = 1.2	1.6457	5.5457	1.4832	5.5456	**1.4587**	5.5456	1.7454	6.4130	1.5336	5.000	1.5030	4.800

表 I：九月至十二月测试。“平均值”和“成本比”分别表示经验平均成本和竞争比（相对于 OPT 归一化）。粗体文字表示最佳平均性能。ERL（$\lambda = x$）表示我们使用 $\lambda = x$ 对 ERL 进行训练。

此外，ERL 的平均成本低于机器学习：即使训练与测试分布差异很大，专家也能帮助纠正许多鲁棒化前的不良动作，从而显著提升 ERL 相较于机器学习的平均性能。有趣的是，ERL 的平均性能并非随用于测试的 $\lambda \geq 1$ 参数单调变化。部分原因是训练和测试所用的 λ 不同，部分原因是训练与测试分布差异较大，导致 ERL 所用机器学习模型的平均性能不规则。通过 $\lambda = 1$，我们实际上完全不信任 ERL 中的机器学习模型，因此 ERL 会在每一步都遵循专家 Robust。

总结。 我们的实验凸显了以下关键点：ERL 通过利用专家鲁棒化，在竞争比方面保证了最坏情况鲁棒性，同时发挥机器学习的能力来提升平均性能。自然地，当训练测试分布 reasonably similar 时，我们预计 ERL（以及其他基于机器学习的优化器，如 ML）的平均性能会优于 Robust。然而，即使纯机器学习方法的表现可能任意糟糕，ERL 仍能凭借引入专家鲁棒化而提供良好的平均成本性能。最后但同样重要的是，由于明确考虑了专家鲁棒化过程，ERL 的平均性能远优于未考虑该过程的情况（即 ERL-NT）。

VII. 结论

本文提出了 ERL，这是一种新颖的专家鲁棒化学习方法，用于解决带记忆成本的在线优化问题。为保证鲁棒性，ERL 引入了投影算子，利用专家在线算法对机器学习动作进行鲁棒化；为获得良好的平均性能，ERL 基于循环架构训练机器学习优化器，并明确考虑下游专家鲁棒化过程。我们证明，对于任意 $\lambda \geq 1$，ERL 都能针对任意问题输入实现相对于专家算法的 λ-竞争比。我们还将分析扩展到多步记忆成本这一新颖设定。最后，我们针对能源调度应用开展实验以验证 ERL，结果表明，就平均性能和最差性能而言，ERL 能够提供最佳权衡。

致谢

本研究部分由美国国家科学基金会通过 CNS-1910208 项目资助。

附录

在更一般的情况下，记忆成本可能跨越多个步骤（例如加速度平滑性），而这一问题尚未得到充分研究。我们首先在附录 A 中给出多步记忆设置下 Robust 的竞争比，从而证明 Robust 仍然是一个有竞争力的专家。然后，在附录 B 中，我们证明 ERL 对任何专家仍然是 λ-竞争的，这也自动证明了单步记忆情况下的定理 4.1。

A. 定理 5.1 的证明

当 $t \geq q$ 时，Robust 满足以下条件：

$$
\begin{aligned}
& f(x_t^\pi, y_t) + ||x_t^\pi - \sum_{i=1}^{q} C_i x_{t-i}^\pi|| \\
\leq & f(x_t^\pi, y_t) + ||x_t^* - \sum_{i=1}^{q} C_i x_{t-i}^*|| + ||x_t^\pi - x_t^*|| \\
& + \sum_{i=1}^{q} ||C_i|| \cdot ||x_{t-i}^\pi - x_{t-i}^*|| \\
\leq & f(x_t^\pi, y_t) + ||x_t^* - \sum_{i=1}^{q} C_i x_{t-i}^*|| + \frac{1}{\alpha}\big(f(x_t^*, y_t) - f(x_t^\pi, y_t)\big) \\
& + \frac{1}{\alpha} \sum_{i=1}^{q} ||C_i|| \big(f(x_{t-i}^*, y_{t-i}) - f(x_{t-i}^\pi, y_{t-i})\big).
\end{aligned}
$$

第一个和第二个不等式来自 l_p 范数的三角不等式，第三个不等式则来自

命中成本函数的 α-多面体假设。对于 $t < q$，由于 $x_t^\pi = x_t^* = x_t,\ \forall t \in [-q+1, 0]$，上述不等式同样成立。我们将所有单步成本相加：

$$
\begin{aligned}
&\sum_{t=1}^{T} f(x_t^\pi, y_t) + \left\| x_t^\pi - \sum_{i=1}^{q} C_i x_{t-i}^\pi \right\| \\
\leq &\sum_{t=1}^{T} f(x_t^\pi, y_t) + \sum_{t=1}^{T} \left\| x_t^* - \sum_{i=1}^{q} C_i x_{t-i}^* \right\| \\
&+ \frac{1}{\alpha} \sum_{t=1}^{T} \big(f(x_t^*, y_t) - f(x_t^\pi, y_t) \big) \\
&+ \frac{1}{\alpha} \sum_{i=1}^{q} \|C_i\| \sum_{t=1}^{T} \big(f(x_{t-i}^*, y_{t-i}) - f(x_{t-i}^\pi, y_{t-i}) \big) \\
\leq &\sum_{t=1}^{T} f(x_t^\pi, y_t) + \sum_{t=1}^{T} \left\| x_t^* - \sum_{i=1}^{q} C_i x_{t-i}^* \right\| \\
&+ \frac{1}{\alpha} \sum_{t=1}^{T} \big(f(x_t^*, y_t) - f(x_t^\pi, y_t) \big) \\
&+ \frac{1}{\alpha} \sum_{i=1}^{q} \|C_i\| \sum_{t=1}^{T} \big(f(x_t^*, y_t) - f(x_t^\pi, y_t) \big) \\
= &\sum_{t=1}^{T} f(x_t^\pi, y_t) + \sum_{t=1}^{T} \left\| x_t^* - \sum_{i=1}^{q} C_i x_{t-i}^* \right\| \\
&+ \frac{1}{\alpha} (1 + \sum_{i=1}^{q} \|C_i\|) \sum_{t=1}^{T} \big(f(x_t^*, y_t) - f(x_t^\pi, y_t) \big),
\end{aligned}
$$

其中第二个不等式成立，是因为 $x_t^\pi = x_t^* = x_t,\ \forall t \in [-q+1, 0]$ 且 $f(x_t^*, y_t) - f(x_t^\pi, y_t) \geq 0$。因此，我们有

$$
\begin{aligned}
&\sum_{t=1}^{T} f(x_t^\pi, y_t) + \left\| x_t^\pi - \sum_{i=1}^{q} C_i x_{t-i}^\pi \right\| \\
\leq &(1 - \frac{1+\beta}{\alpha}) \sum_{t=1}^{T} f(x_t^\pi, y_t) + \sum_{t=1}^{T} \left\| x_t^* - \sum_{i=1}^{q} C_i x_{t-i}^* \right\| \\
&+ \frac{1+\beta}{\alpha} \sum_{t=1}^{T} f(x_t^*, y_t)
\end{aligned} \tag{6}
$$

若 $\alpha \leq 1+\beta$，则 $1 - \frac{1+\beta}{\alpha} \leq 0$，不等式 (6) 变为

$$
\begin{aligned}
&\sum_{t=1}^{T} f(x_t^\pi, y_t) + \left\| x_t^\pi - \sum_{i=1}^{q} C_i x_{t-i}^\pi \right\| \\
\leq &\frac{1+\beta}{\alpha} \sum_{t=1}^{T} f(x_t^*, y_t) + \sum_{t=1}^{T} \left\| x_t^* - \sum_{i=1}^{q} C_i x_{t-i}^\pi \right\|.
\end{aligned}
$$

如果 $\alpha > 1+\beta$，由于 $x_t^\pi = v_t$ 最小化了 $f(\cdot, y_t)$，那么 $f(x_t^\pi, y_t) \leq f(x_t^*, y_t)$，并且根据 (6)，Robust 是最优的。证明完成。

B. 定理 *4.1* 和推论 *5.1* 的证明

我们将前 t_1 步的累计成本记为 $\text{cost}(x_{1:t_1}) = \sum_{t=1}^{t_1} \big(f(x_t, y_t) + \tilde{d}(x_t, \sum_{i=1}^{q} C_i x_{t-i}) \big)$。当 $t = 1$, x_1^π 显然是 (4) 的一个可行解。接着，假设对于 $t \geq 1$, $x_{1:t-1}$ 满足约束，即 $\text{cost}(x_{1:t-1}) + G(x_{t-1}, x_{t-q-1:t-2}, x_{t-q-1:t-1}^\pi) - \big(\lambda \text{cost}(x_{1:t-1}^\pi) + B \big) \leq 0$.我们需要证明 x_t^π 是投影 (4) 的一个可行解。根据投影中的约束，我们有

$$
\begin{aligned}
&\big(\text{cost}(x_{1:t-1}) + f(x_t^\pi, y_t) + \tilde{d}(x_t^\pi, \sum_{i=1}^{q} C_i x_{t-i}) \big) \\
&+ G(x_t^\pi, x_{t-q:t-1}, x_{t-q:t}^\pi) - \big(\lambda \text{cost}(x_{1:t}^\pi) + B \big) \\
= &\text{cost}(x_{1:t-1}) - \big(\lambda \text{cost}(x_{1:t-1}^\pi) + B \big) \\
&+ G(x_t^\pi, x_{t-q:t-1}, x_{t-q:t}^\pi) \\
&+ \tilde{d}(x_t^\pi, \sum_{i=1}^{q} C_i x_{t-i}) - \tilde{d}(x_t^\pi, \sum_{i=1}^{q} C_i x_{t-i}^\pi).
\end{aligned}
$$

根据三角不等式，我们有 $\tilde{d}(x_t^\pi, \sum_{i=1}^{q} C_i x_{t-i}) - \tilde{d}\big(x_t^\pi, \sum_{i=1}^{q} C_i x_{t-i}^\pi\big) \leq \tilde{d}\big(\sum_{i=1}^{q} C_i x_{t-i}, \sum_{i=1}^{q} C_i x_{t-i}^\pi\big)$。(7) 和

$$
\begin{aligned}
&\big(\text{cost}(x_{1:t-1}) + f(x_t^\pi, y_t) + \tilde{d}(x_t^\pi, \sum_{i=1}^{q} C_i x_{t-i}) \big) \\
&+ G(x_t^\pi, x_{t-q:t-1}, x_{t-q:t}^\pi) - \big(\lambda \text{cost}(x_{1:t}^\pi) + B \big) \\
\leq &\text{cost}(x_{1:t-1}) - \big(\lambda \text{cost}(x_{1:t-1}^\pi) + B \big) \\
&+ G(x_t^\pi, x_{t-q:t-1}, x_{t-q:t}^\pi) \\
&+ \tilde{d}(\sum_{i=1}^{q} C_i x_{t-i}, \sum_{i=1}^{q} C_i x_{t-i}^\pi) \\
= &\text{cost}(x_{1:t-1}) - \big(\lambda \text{cost}(x_{1:t-1}^\pi) + B \big) \\
&+ \sum_{k=1}^{q} \tilde{d}(\sum_{i=1}^{q-k} C_{k+i} x_{t-i}, \sum_{i=1}^{q-k} C_{k+i} x_{t-i}^\pi) \\
&+ \tilde{d}(\sum_{i=1}^{q} C_i x_{t-i}, \sum_{i=1}^{q} C_i x_{t-i}^\pi) \\
= &\text{cost}(x_{1:t-1}) + G(x_{t-1}, x_{t-q-1:t-2}, x_{t-q-1:t-1}^\pi) \\
&- \big(\lambda \text{cost}(x_{1:t-1}^\pi) + B \big) \leq 0,
\end{aligned} \tag{7}
$$

其中，第一个不等式是由 l_p 范数的三角不等式得出的，第一个等式来自

$$
\begin{aligned}
&G(x_t^\pi, x_{t-q:t-1}, x_{t-q:t}^\pi) \\
= &\sum_{k=1}^{q} \tilde{d}(\sum_{i=1}^{q-k} C_{k+i} x_{t-i}, \sum_{i=1}^{q-k} C_{k+i} x_{t-i}^\pi),
\end{aligned} \tag{8}
$$

第二个等式来自

$$
\begin{aligned}
&\sum_{k=1}^{q} \tilde{d}(\sum_{i=1}^{q-k} C_{k+i} x_{t-i}, \sum_{i=1}^{q-k} C_{k+i} x_{t-i}^\pi) \\
&+ \tilde{d}(\sum_{i=1}^{q} C_i x_{t-i}, \sum_{i=1}^{q} C_i x_{t-i}^\pi) \\
= &\sum_{k=0}^{q} \tilde{d}(\sum_{i=1}^{q-k} C_{k+i} x_{t-i}, \sum_{i=1}^{q-k} C_{k+i} x_{t-i}^\pi) \\
= &G(x_{t-1}, x_{t-q-1:t-2}, x_{t-q-1:t-1}^\pi)
\end{aligned} \tag{9}
$$

因此，令 $q = 1$ 即可证明定理 4.1。根据定理 5.1，我们也证明了推论 5.1。

参考文献

[1] Akshay Agrawal、Brandon Amos、Shane Barratt、Stephen Boyd、Steven Diamond 和 J. Zico Kolter。可微凸优化层。载于 H. Wallach、H. Larochelle、A. Beygelzimer、F. d'Alché-Buc、E. Fox 和 R. Garnett 编，《神经信息处理系统进展》，第 32 卷。Curran Associates, Inc.，2019。[2] Mohammad Ali Alomrani、Reza Moravej 和 Elias B. Khalil。用于在线二分图匹配的深度策略：一种强化学习方法。CoRR，abs/2109.10380，2021。[3] Brandon Amos。关于连续域学习优化的摊销优化教程。CoRR，abs/2202.00665，2022。[4] Brandon Amos、Ivan Jimenez、Jacob Sacks、Byron Boots 和 J. Zico Kolter。用于端到端规划与控制的可微模型预测控制。载于 S. Bengio、H. Wallach、H. Larochelle、K. Grauman、N. Cesa-Bianchi 和 R. Garnett 编，《神经信息处理系统进展》，第 31 卷。Curran Associates, Inc.，2018。[5] Keerti Anand、Rong Ge、Amit Kumar 和 Debmalya Panigrahi。学习增强型在线算法的回归方法。载于 A. Beygelzimer、Y. Dauphin、P. Liang 和 J. Wortman Vaughan 编，《神经信息处理系统进展》，2021。[6] Antonios Antoniadis、Christian Coester、Marek Elias、Adam Polak 和 Bertrand Simon。具有不可信预测的在线度量算法。载于国际机器学习会议，2020。[7] Joan Boyar、Lene M. Favrholdt、Christian Kudahl、Kim S. Larsen 和 Jesper W. Mikkelsen。带建议的在线算法：综述。SIGACT 新闻，47(3):93–129，2016 年 8 月。[8] S. Boyd 和 L. Vandenberghe。《凸优化》。剑桥大学出版社，2004。[9] Niangjun Chen、Joshua Comden、Zhenhua Liu、Anshul Gandhi 和 Adam Wierman。在在线优化中使用预测：立足过去，展望未来。SIGMETRICS 性能评估评论，44(1):193–206，2016 年 6 月。[10] Niangjun Chen、Gautam Goel 和 Adam Wierman。通过在线平衡下降实现高维平滑在线凸优化。载于学习理论会议，2018。[11] Jakub Chłedowski、Adam Polak、Bartosz Szabucki 和 Konrad Tomasz Zołna。鲁棒学习增强型缓存：一项实验研究。载于国际机器学习会议，2021。[12] Nicolas Christianson、Tinashe Handina 和 Adam Wierman。利用黑箱建议追逐凸体与凸函数。载于学习理论会议，2022。[13] Joshua Comden、Sijie Yao、Niangjun Chen、Haipeng Xing 和 Zhenhua Liu。云资源配置中的在线优化：预测、遗憾值与算法。ACM 测量、分析与计算系统会议录，3(1)，2019 年 3 月。[14] Bingqian Du、Zhiyi Huang 和 Chuan Wu。用于在线资源分配的对抗性深度学习。ACM 建模、性能评估与计算系统汇刊，6(4)，2022 年 2 月。[15] Bingqian Du、Chuan Wu 和 Zhiyi Huang。面向云利润最大化的资源分配与定价学习。载于第三十三届 AAAI 人工智能会议、第三十一届人工智能创新应用会议和第九届 AAAI 人工智能教育进展研讨会论文集，AAAI'19/IAAI'19/EAAI'19。AAAI 出版社，2019。[16] Elbert Du、Franklyn Wang 和 Michael Mitzenmacher。将"学习"融入用于频率估计的学习增强型算法。载于 Marina Meila 和 Tong Zhang 编，《第 38 届国际机器学习会议论文集》，《机器学习研究会议录》第 139 卷，第 2860–2869 页。PMLR，2021 年 7 月 18–24 日。[17] Gautam Goel、Yiheng Lin、Haoyuan Sun 和 Adam Wierman。超越在线平衡下降：一种用于平滑在线优化的最优算法。载于 NeurIPS，第 32 卷，2019。[18] Ian Goodfellow、Yoshua Bengio 和 Aaron Courville。《深度学习》。麻省理工学院出版社，2016。[19] Elad Hazan。在线凸优化导论。优化基础与趋势，2(3-4):157–325，2016。[20] Howard Heaton、Xiaohan Chen、Zhangyang Wang 和 Wotao Yin。带保护机制的学习型凸优化，2020。[21] Weiwei Kong、Christopher Liaw、Aranyak Mehta 和 D. Sivakumar。一只新狗学会旧把戏：强化学习找到了经典优化算法。载于国际学习表征会议，2019。[22] Ke Li 和 Jitendra Malik。学习优化。载于国际学习表征会议，2017。[23] Pengfei Li、Jianyi Yang 和 Shaolei Ren。带切换成本的在线优化专家校准学习。载于 SIGMETRICS，2022 年。[24] Yingying Li、Xin Chen 和 Na Li。具有线性动力学和预测的在线最优控制：算法与遗憾分析。载于 NeurIPS，美国纽约州雷德胡克，2019 年。Curran Associates 公司。[25] Yingying Li 和 Na Li。通过基于梯度的算法利用平滑在线凸优化中的预测。载于 NeurIPS，第 33 卷，2020 年。[26] Yingying Li、Guannan Qu 和 Na Li。带预测和切换成本的在线优化：快速算法与基本极限。IEEE《自动控制汇刊》，2020 年。[27] M. Lin、A. Wierman、L. L. H. Andrew 和 E. Thereska。功率成比例数据中心的动态容量调整。载于 INFOCOM，2011 年。[28] Yiheng Lin、Gautam Goel 和 Adam Wierman。带预测和非凸损失的在线优化。《ACM 测量、分析与计算系统会议录》，4(1)，2020 年 5 月。[29] Heyuan Liu 和 Paul Grigas。智能预测后优化方法的风险界与校准。载于 A. Beygelzimer、Y. Dauphin、P. Liang 和 J. Wortman Vaughan（编），《神经信息处理系统进展》，2021 年。[30] Yecheng Jason Ma、Dinesh Jayaraman 和 Osbert Bastani。保守离线分布强化学习。载于 A. Beygelzimer、Y. Dauphin、P. Liang 和 J. Wortman Vaughan（编），《神经信息处理系统进展》，2021 年。[31] Aranyak Mehta。在线匹配与广告分配。《理论计算机科学基础与趋势》，8(4)：265–368，2013 年。[32] Daan Rutten、Nico Christianson、Debankur Mukherjee 和 Adam Wierman。带不可信预测的在线优化。CoRR，abs/2202.03519，2022 年。[33] Asis Sarkar 和 Dhiren Kumar Behera。基于电学类比的风力涡轮机叶片效率与功率计算。《国际科学与研究出版物期刊》，2(2)：1–5，2012 年。[34] Manajit Sengupta、Yu Xie、Anthony Lopez、Aron Habte、Galen Maclaurin 和 James Shelby。国家太阳辐射数据库（NSRDB）。《可再生能源与可持续能源评论》，89：51–60，2018 年。[35] Guanya Shi、Yiheng Lin、Soon-Jo Chung、Yisong Yue 和 Adam Wierman。带记忆的在线优化与竞争性控制。载于 NeurIPS，第 33 卷。Curran Associates, Inc.，2020 年。[36] Matthew Staib 和 Stefanie Jegelka。分布鲁棒优化与核方法中的泛化。《神经信息处理系统进展》，32：9134–9144，2019 年。[37] Can Wan、Jian Zhao、Yonghua Song、Zhao Xu、Jin Lin 和 Zechun Hu。面向智能电网能源管理的光伏与太阳能发电预测。《中国电机工程学会电力与能源系统学报》，1 46，2015 4 38( )：–。[38] Kai Wang、Sanket Shah、Haipeng Chen、Andrew Perrault、Finale Doshi-Velez 和 Milind Tambe。从特征中学习马尔可夫决策过程：通过强化学习进行序贯决策的预测后优化。载于 A. Beygelzimer、Y. Dauphin、P. Liang 和 J. Wortman Vaughan（编），《神经信息处理系统进展》，2021 年。[39] Tsung-Yen Yang、Justinian Rosca、Karthik Narasimhan 和 Peter J. Ramadge。基于投影的约束策略优化。载于国际学习表征会议，2020 年。[40] Yunchang Yang、Tianhao Wu、Han Zhong、Evrard Garcelon、Matteo Pirotta、Alessandro Lazaric、Liwei Wang 和 Simon Shaolei Du。保守多臂老虎机与强化学习的基于归约的框架。载于国际学习表征会议，2022 年。[41] Jingzhao Zhang、Aditya Krishna Menon、Andreas Veit、Srinadh Bhojanapalli、Sanjiv Kumar 和 Suvrit Sra。通过分布鲁棒优化应对标签偏移。载于国际学习表征会议，2021 年。[42] Lijun Zhang、Wei Jiang、Shiyin Lu 和 Tianbao Yang。重新审视平滑在线学习。载于 A. Beygelzimer、Y. Dauphin、P. Liang 和 J. Wortman Vaughan（编），《神经信息处理系统进展》，2021 年。[43] Martin Zinkevich。在线凸规划与广义无穷小梯度上升。载于第 20 届国际机器学习会议（ICML-03）会议录，第 928–936 页，2003 年。