Regret Bound for Online Mirror Descent: Lazy and Agile Versions

Based on Bregman Divergence Analysis

Mathematical Proof

Introduction

Objective: Prove a regret bound for both Lazy and Agile Online Mirror Descent (OMD) algorithms using Bregman divergence.

- We analyze the regret in an online convex optimization setting.
- Bregman divergence serves as a key tool in the proof.
- We derive bounds on cumulative regret over T iterations.

Bregman Divergence Definition

Given a strictly convex differentiable function $R : K \to \mathbb{R}$, the Bregman divergence is defined as:

$$B_R(x||y) = R(x) - R(y) - \nabla R(y)^T(x - y). \qquad (1)$$

- Measures the difference between $R(x)$ and its first-order approximation at y.
- Plays a crucial role in the analysis of mirror descent algorithms.

Lazy Online Mirror Descent Algorithm

Algorithm 1 Lazy Online Mirror Descent

Input: Learning rate $\eta > 0$, regularization function $R(x)$, convex set K.

Initialize y_1 such that $\nabla R(y_1) = 0$ and $x_1 = \arg\min_{x \in K} B_R(x \| y_1)$.

for $t = 1$ to T **do**

  Play x_t.

  Observe payoff function f_t and compute $\nabla_t = \nabla f_t(x_t)$.

  Update y_t:

$$\nabla R(y_{t+1}) = \nabla R(y_t) - \eta \nabla_t. \tag{2}$$

  Compute: $x_{t+1} = \arg\min_{x \in K} B_R(x \| y_{t+1})$.

end for

Agile Online Mirror Descent Algorithm

Algorithm 2 Agile Online Mirror Descent

Input: Learning rate $\eta > 0$, regularization function $R(x)$, convex set K.
Initialize $x_1 = \arg\min_{x \in K} R(x)$.
for $t = 1$ to T **do**
  Play x_t.
  Observe payoff function f_t and compute $\nabla_t = \nabla f_t(x_t)$.
  Update:
$$\nabla R(x_{t+1}) = \nabla R(x_t) - \eta \nabla_t. \tag{3}$$
end for

Regret Bound for Lazy OMD

Theorem: The regret of Lazy OMD for any $u \in K$ satisfies:

$$\sum_{t=1}^{T}(f_t(x_t) - f_t(u)) \leq \frac{B_R(u||y_1)}{\eta} + \frac{1}{2\eta}\sum_{t=1}^{T} \|\nabla_t\|_{*t}^2. \tag{4}$$

Regret Bound for Agile OMD

Theorem: The regret of Agile OMD for any $u \in K$ satisfies:

$$\sum_{t=1}^{T} (f_t(x_t) - f_t(u)) \leq \frac{R(u) - R(x_1)}{\eta} + \frac{1}{2\eta} \sum_{t=1}^{T} \|\nabla_t\|_{*t}^2. \qquad (5)$$

Proof of Regret Bound for Lazy OMD

Step 1: Expanding the Bregman Divergence Recursion

$$B_R(u||y_{t+1}) = B_R(u||y_t)+(\nabla R(y_t)-\nabla R(y_{t+1}))^T(u-y_t)-B_R(y_{t+1}||y_t). \tag{6}$$

Step 2: Substituting the Update Rule

$$\nabla R(y_t) - \nabla R(y_{t+1}) = \eta \nabla_t. \tag{7}$$

Step 3: Bounding the Sum

$$\sum_{t=1}^{T} \nabla_t^T (x_t - u) \leq \frac{B_R(u||y_1)}{\eta} + \frac{1}{2\eta} \sum_{t=1}^{T} \|\nabla_t\|_{*t}^2. \tag{8}$$

Proof of Regret Bound for Agile OMD

Step 1: Expanding the Bregman Divergence for Agile Updates

$$R(u) - R(x_{t+1}) = R(u) - R(x_t) + \eta \nabla_t^T (u - x_t) - B_R(x_{t+1} || x_t). \quad (9)$$

Step 2: Summing Over All Iterations

$$\sum_{t=1}^{T} (f_t(x_t) - f_t(u)) \leq \frac{R(u) - R(x_1)}{\eta} + \frac{1}{2\eta} \sum_{t=1}^{T} \|\nabla_t\|_{*t}^2. \quad (10)$$

Introduction

Objective: Show how projection relates to Fixed Share and Variable Share algorithms.

- Projection in online learning ensures stability.
- Fixed Share and Variable Share algorithms perform weight updates over experts.
- Both can be derived using Bregman divergence.

Projection in Online Learning

Given a convex function $R(x)$, the Bregman projection onto a convex set K is:

$$x_{t+1} = \arg\min_{x \in K} B_R(x||x_t), \tag{11}$$

where Bregman divergence is:

$$B_R(x||y) = R(x) - R(y) - \nabla R(y)^T(x - y). \tag{12}$$

- Projection maintains feasibility in optimization.
- It ensures stability when tracking changing solutions.

Mathematical Derivation of Fixed Share Algorithm

The Fixed Share algorithm updates weights as:

$$w_t^i = (1-\alpha) w_{t-1}^i + \frac{\alpha}{N} \sum_j w_{t-1}^j. \tag{13}$$

The derivation follows from minimizing the objective function:

$$\arg\min_{w \in \Delta} D_{KL}(w || w_{t-1}) + \alpha \sum_i w^i. \tag{14}$$

where KL divergence is defined as:

$$D_{KL}(w || w_{t-1}) = \sum_i w^i \log \frac{w^i}{w_{t-1}^i}. \tag{15}$$

Applying first-order conditions, we obtain the Fixed Share update rule.

Mathematical Derivation of Variable Share Algorithm

The Variable Share algorithm modifies the share rate adaptively:

$$w_t = \arg\min_{w \in \Delta} D_{KL}(w || w_{t-1}) + \alpha_t \sum_i w^i. \tag{16}$$

Solving for optimal weights:

$$w^i = \frac{w_{t-1}^i e^{-\alpha_t}}{\sum_j w_{t-1}^j e^{-\alpha_t}}. \tag{17}$$

Key Insights:

- When the environment is stable, α_t is small (less adaptation).
- When the environment is changing, α_t increases (faster adaptation).

Generalizing to Bregman Projection

Both algorithms follow the framework:

$$w_t = \arg\min_{w \in \Delta} B_R(w||w_{t-1}) + \eta_t \sum_i w^i L_t^i. \tag{18}$$

where:

- $B_R(w||w_{t-1})$ ensures stability.
- η_t adapts over time (Variable Share).

Conclusion: Fixed and Variable Share algorithms are special cases of Bregman projection.