Comparator-Adaptive Convex Bandits

$\|\mathbf{u}\|$ $\sqrt{T}$ $T^{3/4}$ $T^{2/3}\|\mathbf{u}\|$

Expected regret bounds for: Linear loss, Lipschitz loss, Smooth loss

Dirk van der Hoeven — Universiteit Leiden, The Netherlands

Ashok Cutkosky — Boston University

Haipeng Luo — USC University of Southern California

Setting: Bandit Convex Optimization

The Bandit Convex Optimization setting proceeds in rounds $t = 1, \dots T$. In each round t

1. the environment picks a Lipschitz loss function ℓ_t
2. the learner plays a randomized prediction $\boldsymbol{w}_t \in \mathcal{W} \subseteq \mathbb{R}^d$
3. the environment reveals $\ell_t(\boldsymbol{w}_t)$
4. the learner suffers $\ell_t(\boldsymbol{w}_t)$

Goal setting: control expected regret with respect to comparator $\boldsymbol{u} \in \mathcal{W}$

$$\mathcal{R}_T(\boldsymbol{u}) = \mathbb{E}\left[\sum_{t=1}^{T} \ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{u})\right]$$

Goal paper: regret that scales with $\|\boldsymbol{u}\|$.

Motivation

In the **Full Information setting** comparator-adaptive algorithms have been successfully applied in several problems:

- deal with unconstrained domains (McMahan and Orabona, 2014)
- combining online algorithms (Cutkosky, 2019)
- obtaining better strongly adaptive algorithms (Jun et al., 2017)
- adaptive local differential privacy (Van der Hoeven, 2019)

But no comparator-adaptive algorithms exist for the **Bandit Setting**!

Results

Loss functions (L-Lipschitz)	unconstrained settings	constrained settings
Linear	$\widetilde{O}\left(\|\boldsymbol{u}\|dL\sqrt{T}\right)$	$\widetilde{O}\left(\|\boldsymbol{u}\|cdL\sqrt{T}\right)$
Convex	$\widetilde{O}\left(\|\boldsymbol{u}\|L\sqrt{d}T^{\frac{3}{4}}\right)$	$\widetilde{O}\left(\|\boldsymbol{u}\|cL\sqrt{d}T^{\frac{3}{4}}\right)$
Convex and β-smooth	$\widetilde{O}\left(\max\{\|\boldsymbol{u}\|^2, \|\boldsymbol{u}\|\}\beta(dLT)^{\frac{2}{3}}\right)$	-

$1/c$ is radius of the largest ball contained by $\mathcal{W}$ and $c = O(d)$ at most. When $\mathcal{W}$ is a ball $c = 1$.

Basis of Algorithm and Analysis

- Play $w_t = v_t \boldsymbol{z}_t$, v_t controls the scale and $\boldsymbol{z}_t$ controls the direction.

In the **analysis** we:

- **split the analysis** for learning of the direction of $\boldsymbol{u}$, $\frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}$ and the scale of $\boldsymbol{u}$, $\|\boldsymbol{u}\|$ in two parts (Cutkosky and Orabona, 2018).

For linear losses:

$$\langle v_t\boldsymbol{z}_t - \boldsymbol{u}, \boldsymbol{g}_t\rangle = \langle v_t\boldsymbol{z}_t - \|\boldsymbol{u}\|\boldsymbol{z}_t, \boldsymbol{g}_t\rangle + \|\boldsymbol{u}\|\langle \boldsymbol{z}_t - \frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}, \boldsymbol{g}_t\rangle$$

Similar in spirit for convex losses, but with extra steps.

- **assume** that $\ell(\mathbf{0})$ is known

Reasonable assumption: loss for a nominal action often known.

Key Insights: Linear Loss

- With the split of direction and scale the scaling algorithm obtains **full information feedback**:

As feedback we receive $\ell_t(\boldsymbol{w}_t) = \langle v_t\boldsymbol{z}_t, \boldsymbol{g}_t\rangle$. We can simply send $\frac{\ell_t(\boldsymbol{w}_t)}{v_t}$ to the scaling algorithm to learn the appropriate scale of $\boldsymbol{z}_t$.

- In the constrained setting: constrain v_t to $[0, 1]$ rather than constraining $v_t\boldsymbol{z}_t$

If $z_t \in \mathcal{W}$, then so is $v_t\boldsymbol{z}_t$ if $v_t \in [0, 1]$. Result: you **don't have to project a randomized vector**.

Algorithm for Convex Losses

Input: Scaling algorithm $\mathcal{A}_{\mathcal{V}}$, $\delta \in (0, 1]$, $\alpha \in [0, 1]$, domain $\mathcal{Z} \subseteq \mathbb{B}$, and learning rate η
for $t = 1 \dots T$ **do**
  Get v_t from $\mathcal{A}_{\mathcal{V}}$
  Sample $\boldsymbol{s}_t \sim U(\mathbb{S})$
  Play $\boldsymbol{w}_t = v_t(\boldsymbol{z}_t + \delta\boldsymbol{s}_t)$
  Set $\hat{\boldsymbol{g}}_t = \frac{d}{v_t\delta}\ell_t(\boldsymbol{w}_t)s_t$
  if ℓ_t is β-smooth **then**
    Set $\bar{\ell}_t(v) = v\langle \boldsymbol{z}_t, \hat{\boldsymbol{g}}_t\rangle + \beta\delta^2 v^2$
  else
    Set $\bar{\ell}_t(v) = v\langle \boldsymbol{z}_t, \hat{\boldsymbol{g}}_t\rangle + 2\delta L|v|$
  end if
  Send $\partial\bar{\ell}_t(v_t)$ to algorithm $\mathcal{A}_{\mathcal{V}}$ as the t-th loss value
  Update $\boldsymbol{z}_{t+1} = \arg\min_{\boldsymbol{z}\in(1-\alpha)\mathcal{Z}} \eta\langle \boldsymbol{z}, \hat{\boldsymbol{g}}_t\rangle + \|\boldsymbol{z}_t - \boldsymbol{z}\|_2^2$
end for

Key Insights for Lipschitz Losses

- Modify the estimate for the gradient:

$$\hat{\boldsymbol{g}}_t = \frac{d}{v_t\delta}\ell_t(v_t\boldsymbol{z}_t + v_t\delta\boldsymbol{s}_t)s_t$$

What if v_t very small? Use $\ell_t(\mathbf{0}) = 0$ and Lipschitz assumption:

$$\frac{d}{v_t\delta}|\ell_t(\boldsymbol{w}_t)| = \frac{d}{v_t\delta}|\ell_t(v_t(\boldsymbol{z}_t + \delta\boldsymbol{s}_t)) - \ell_t(\mathbf{0})| \leq \frac{dL\|\boldsymbol{z}_t + \delta\boldsymbol{s}_t\|_2}{\delta}$$

- New surrogate loss for learning the scale:

$$\bar{\ell}_t(v) = v\langle \boldsymbol{z}_t, \hat{\boldsymbol{g}}_t\rangle + \underbrace{2\delta L|v|}_{\text{new part}}$$

Key Insights for Smooth Losses

- New surrogate loss for learning the scale:

$$\bar{\ell}_t(v) = v\langle \boldsymbol{z}_t, \hat{\boldsymbol{g}}_t\rangle + \underbrace{\beta\delta^2 v^2}_{\text{new part}}$$

Problem: $\bar{\ell}_t(v)$ is not Lipschitz if $v \in \mathbb{R}$!
Solution: choose $|v| \in [0, \frac{1}{\delta^3}]$ and exploit that $\mathcal{R}_T(0) = O(1)$.
If $\|\boldsymbol{u}\| \leq \frac{1}{\delta^3}$ then no problem if we constrain v. If $\|\boldsymbol{u}\| > \frac{1}{\delta^3}$ then

$$\mathbb{E}[\mathcal{R}_T(\boldsymbol{u})] = \mathbb{E}[\mathcal{R}_T(\mathbf{0})] + \sum_{t=1}^{T}\mathbb{E}[\ell_t(\mathbf{0}) - \ell_t(\boldsymbol{u})] = O(1 + \|\boldsymbol{u}\|_2 LT).$$

Future work

- Get rid of c constant in constrained algorithms
- Get rid of assumption that $\ell_t(\mathbf{0})$ = known
- Comparator-adaptive algorithm for the **smooth and constrained** case with $\widetilde{O}(\|\boldsymbol{u}\|T^{2/3})$ regret?
- Comparator-adaptive algorithm for **strongly convex** and smooth case with $\widetilde{O}(\|\boldsymbol{u}\|\sqrt{T})$ regret?