Near-optimal guarantees on both “Easy” and “Hard” data!

What is "Easy" Data?

Examples:
- Non-IID Adversarial Losses
- Non-Stationary distributions
- Small gaps between means
- Non-strongly convex losses
- See Setting 1 in Figure 1.

Algorithms:
- Vowants of Follow-the-leader (FTL)
- Typical regret guarantees: $O(\log T)$

Problems:
- Horrible performance on "Hard" data!
- See Setting 1 in Figure 1.

What is "Hard" Data?

Examples:
- Highly predictable sequences
- IID losses with large gaps between means
- Strongly convex losses
- See Settings 2, 3 and 4 in Figure 1.

Algorithms:
- Variants of Follow-the-leader (FTL)
- Typical regret guarantees: $O(\sqrt{T})$

Problems:
- Horrible performance on "Easy" data.
- See Settings 2 and 4 in Figure 1.

Online Optimization (OO) [3]

Competing against a Benchmark

Our method guarantees a constant regret w.r.t. any existing benchmark strategy together with small regret against the best strategy in hindsight. This is particularly useful in domains where the learning algorithm should be safe and never worsen the performance of an existing strategy (e.g., portfolio optimization with benchmark reference).

Theoretical Results

Theorem 1 (cf. Lemma 1 in [2])

For any assignment of the loss sequence, the total expected loss of $(\mathcal{A})$-Prod initialized with weights $w_a \in [0,1]$ and $w_a = 1 - w_a$ simultaneously satisfies

$$L_T((\mathcal{A})\text{-Prod}) \leq L_T(A) + \eta \sum_{i=1}^{T} \left( f_i(b_i) - f_i(a_i) \right) - \log w_a - \log 1 - \log w_a,$$

and

$$L_T((\mathcal{A})\text{-Prod}) \leq L_T(B) + \log w_a.$$

Corollary 1

Let $C \geq 0$ be an upper bound on the total benchmark loss $L_T(B)$. Then setting $\eta = 1/\sqrt{\log C}/C < 1/2$ and $w_a = 1 - w_a = 1 - \eta$ simultaneously guarantees

$$\eta \left( L_T((\mathcal{A})\text{-Prod}) \right) \leq \eta \left( L_T(A) \right) + 2 \log \left( \frac{1}{1 - \eta} \right),$$

for any $A \in \mathcal{S}$ and

$$\eta \left( L_T((\mathcal{A})\text{-Prod}) \right) \leq 4 \log 2.$$

against any assignment of the loss sequence.

Parameters

- Decision set $S = \{1,2,\ldots,T\}$
- Family of loss function $F \subseteq \mathcal{S}^{[N]}$

Empirical Results

- Summary
  - Given a learning algorithm $A$, with worst-case performance guarantees, and an opportunistic strategy $B$, exploiting a specific structure within the loss sequence, smoothly adapts to "Easy" and "Hard" problems.
  - Guarantees best performance between benchmark $B$ and a worst-case algorithm $A$.

Discussion

- General-purpose, Implementable, Simple
- Open Problems
  - Learning with temporal constraints (e.g., switching costs, MDPs)?
  - What are good benchmark strategies for easy data?/ Learning with partial feedback?

References