Online Convex Optimization

Presented by: Sharan Vaswani
Introduction

• **Online learning:** Make a sequence of accurate predictions given knowledge of the correct answer to previous prediction tasks and possibly additional available information.

• **Applications:** Online advertisement placement, web ranking, spam filtering, online shortest paths, portfolio selection, recommender systems
Notation

\( X_t \) Decision/point chosen on timestamp \( t \)

\( \mathcal{K} \) Bounded convex, decision set

\( f_t \in \mathcal{F} : \mathcal{K} \rightarrow \mathbb{R} \) Bounded convex function available at timestamp \( t \)

\( T \) Number of iterations

\( \mathcal{A} \) Online algorithm
Protocol

For $t = 1: T$

1. Learner chooses $X_t$
2. Environment / Adversary chooses $f_t$
3. Learner suffers loss $f_t(X_t)$

Aim: To minimize cumulative loss across rounds measured using regret

$$\text{regret}_T(A) = \sup_{\{f_1, \ldots, f_T\} \subseteq \mathcal{F}} \left\{ \sum_{t=1}^{T} f_t(x_t) - \min_{x \in \mathcal{K}} \sum_{t=1}^{T} f_t(x) \right\}$$

Chosen by our online algorithm

Chosen by the offline algorithm
Learning from expert advice

• Need to make a decision (e.g.: whether to invest or not), helped by N experts who make predictions
• Decide which expert to follow at each step
• Receive feedback for each expert
Learning from expert advice

Algorithm 1 Hedge

1: Initialize: \( \forall i \in [N], \ W_1(i) = 1 \)
2: for \( t = 1 \) to \( T \) do
3: \( \quad \) Pick \( i_t \sim_R W_t \), i.e., \( i_t = i \) with probability \( p_t(i) = \frac{W_t(i)}{\sum_j W_t(j)} \)
4: \( \quad \) Observe loss \( \ell_t(i_t) \).
5: \( \quad \) Update weights \( W_{t+1}(i) = W_t(i)e^{-\varepsilon\ell_t(i)} \)
6: end for

Theorem 1.5. Let \( \ell_t^2 \) denote the \( N \)-dimensional vector of square losses, i.e., \( \ell_t^2(i) = \ell_t(i)^2 \), and let \( \varepsilon < \frac{1}{2} \). The Hedge algorithm satisfies for any expert \( i^* \in [N] \):

\[
\sum_{t=1}^{T} p_t^\top \ell_t \leq \sum_{t=1}^{T} f_t(i^*) + \varepsilon \sum_t p_t^\top \ell_t^2 + \frac{\ln N}{\varepsilon}
\]
Asymptotic Regret Bounds

Theorem 3.2. Any algorithm for online convex optimization incurs \( \Omega(DG\sqrt{T}) \) regret in the worst case. This is true even if the cost functions are generated from a fixed stationary distribution.

- Achieved by first order methods like online gradient descent and second order methods like online newton

<table>
<thead>
<tr>
<th>( \alpha )-strongly convex</th>
<th>( \beta )-smooth</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{1}{\alpha} \log T ) (^1)</td>
<td>( \sqrt{T} ) (^2)</td>
</tr>
</tbody>
</table>

- Smoothness doesn’t help (unlike in offline gradient descent), strong convexity does

1 - Hazan, 2007; 2 - Zinkevich, 2003
Follow the leader

**Idea:** Use loss information in previous iterations to choose point in current iteration.

\[
\forall t, \quad w_t = \text{argmin}_{w \in S} \sum_{i=1}^{t-1} f_i(w) \quad \text{(break ties arbitrarily)}
\]

**Special case:** Obtains \( \log(T) \) regret with quadratic functions (shooting game)

**Problem:** Predictions are not stable and may fluctuate drastically

---

**Example 2.2 (Failure of FTL).** Let \( S = [-1, 1] \subset \mathbb{R} \) and consider the sequence of linear functions such that \( f_t(w) = z_t w \) where

\[
z_t = \begin{cases} 
-0.5 & \text{if } t = 1 \\
1 & \text{if } t \text{ is even} \\
-1 & \text{if } t > 1 \land t \text{ is odd}
\end{cases}
\]

Then, the predictions of FTL will be to set \( w_t = 1 \) for \( t \) odd and \( w_t = -1 \) for \( t \) even. The cumulative loss of the FTL algorithm will therefore be \( T \) while the cumulative loss of the fixed solution \( u = 0 \in S \) is 0. Thus, the regret of FTL is \( T \)!
Follow the regularized leader

- **Idea:** Add regularization to make prediction more stable.

**Algorithm 10 Regularized Follow The Leader**

1: Input: $\eta > 0$, strongly convex regularization function $R$, and a convex compact set $\mathcal{K}$.
2: Let $x_1 = \arg\min_{x \in \mathcal{K}} \{R(x)\}$.
3: for $t = 1$ to $T$ do
4:     Predict $x_t$.
5:     Observe the payoff function $f_t$ and let $\nabla_t = \nabla f_t(x_t)$.
6:     Update

$$
x_{t+1} = \arg\min_{x \in \mathcal{K}} \left\{ \eta \sum_{s=1}^{t} \nabla_{s}^\top x + R(x) \right\}
$$

(5.1)

7: end for
Follow the regularized leader

**Theorem 5.1.** The RFTL algorithm attains for every \( u \in \mathcal{K} \) the following bound on the regret:

\[
\text{regret}_T \leq \sum_{t=1}^{T} \nabla_t^T (x_t - u) \leq 2\eta \sum_{t} \|
abla_t^*\|^2_t + \frac{1}{\eta} D_R
\]

Local norm of gradient at \( t \)
Can be upper bounded

Diameter of the decision set relative to \( R \)

Special Cases:
- **Online Gradient Descent** with L2 regularization i.e. \( R(x) = \frac{1}{2} \|x\|^2_2 \)
- **Hedge** with negative entropy regularization i.e. \( R(x) = x \log x \)
Online gradient descent

Algorithm 6 Online Gradient Descent

1: Input: convex set $\mathcal{K}$, $T$, $x_1 \in \mathcal{K}$, step sizes $\{\eta_t\}$
2: for $t = 1$ to $T$ do
3: Play $x_t$ and observe cost $f_t(x_t)$.
4: Update and project:
   \[
   y_{t+1} = x_t - \eta_t \nabla f_t(x_t)
   \]
   \[
   x_{t+1} = \Pi_{\mathcal{K}}[y_{t+1}]
   \]
5: end for
Online gradient descent

**Theorem 3.1.** Online Gradient Descent with step sizes \( \{\eta_t = \frac{D}{G\sqrt{t}}, t = 1, \ldots, T\} \) guarantees the following for all \( T \geq 1 \).

\[
\text{regret}_T = \sum_{t=1}^{T} f_t(x_t) - \min_{x^* \in \mathcal{K}} \sum_{t=1}^{T} f_t(x^*) \leq 3GD\sqrt{T}
\]

**Theorem 3.3.** For \( \alpha \)-strongly convex loss functions, Online Gradient Descent with step sizes \( \eta_t = \frac{1}{\alpha t} \) achieves the following guarantee, for all \( T \geq 1 \)

\[
\text{regret}_T \leq \frac{G^2}{2\alpha} (1 + \log T).
\]
Online to Batch reduction

Suppose we have a batch of data, can we OGD to look one point at a time and get a rate as good as SGD?

**Algorithm 7** Stochastic Gradient Descent

1. Input: $f, \mathcal{K}, T, \mathbf{x}_1 \in \mathcal{K}$, step sizes $\lbrace \eta_t \rbrace$
2. for $t = 1$ to $T$ do
3.   Let $\tilde{\nabla}_t \leftarrow \mathcal{O}(\mathbf{x}_t)$ and define: $f_t(\mathbf{x}) \triangleq \langle \tilde{\nabla}_t, \mathbf{x} \rangle$
4.   Update and project:

$$y_{t+1} = \mathbf{x}_t - \eta_t \tilde{\nabla}_t$$

$$x_{t+1} = \Pi_{\mathcal{K}}[y_{t+1}]$$

5. end for
6. return $\bar{x}_T \triangleq \frac{1}{T} \sum_{t=1}^{T} \mathbf{x}_t$

Use linearized functions with noisy gradient as input and return the average point over the iterations
Online to Batch reduction

Regret bound:

\[
\mathbb{E}[f(\overline{x}_T)] \leq \min_{x^* \in \mathcal{K}} f(x^*) + \frac{3GD}{\sqrt{T}}
\]

- For general convex function, achieves 1/sqrt(T) rate
- For strongly convex functions, online to batch conversion can give a log(T)/T rate later improved to 1/T (Hazan’14)
ADAGRAD

• Different regularizations in RTFL lead to different algorithms. Can we learn an optimal regularization to use?

**Algorithm 16 AdaGrad**

1: Input: parameters $\eta, \delta > 0, x_1 \in K$.
2: Initialize: $S_0 = G_0 = \delta I$.
3: for $t = 1$ to $T$ do
4:   Predict $x_t$, suffer loss $f_t(x_t)$.
5:   Update: $S_t \leftarrow S_{t-1} + \nabla_t \nabla_t^T$, $G_t = S_t^{1/2}$
6:      \[ y_{t+1} \leftarrow x_t - \eta G_t^{-1} \nabla_t \]
7:      \[ x_{t+1} \leftarrow \operatorname{arg min}_{x \in K} \| y_{t+1} - x \|^2_{G_t} \]
6: end for
\( \forall x \in \mathcal{K}. \nabla^2 R(x) = \nabla^2 \in \mathcal{H} \triangleq \{ X \in \mathbb{R}^{d \times d}, \text{Tr}(X) \leq 1, X \succeq 0 \} \)

**Theorem 5.9.** Let \( \{x_t\} \) be defined by Algorithm 16 with parameters \( \delta = 1, \eta = \frac{1}{D} \), where

\[
D = \max_{u \in \mathcal{K}} \max_{H \in \mathcal{H}} \| u - x_1 \|_H
\]

Then for any \( x^* \in \mathcal{K} \),

\[
\text{regret}_T(\text{AdaGrad}) \leq 2D \cdot \sqrt{\min_{H \in \mathcal{H}} \sum_t \| \nabla_t \|^*_H} \quad (5.6)
\]

\[
\text{regret}(\text{AdaGrad}) \leq 2D \text{Tr}(G_T)
\]
Variants

• Follow the perturbed leader (FPL) – form of randomized regularization
• Bandit feedback (observe loss only for the selected point) – can be analyzed using OCO by estimating the gradient at each point
• Use Frank Wolfe when projection is expensive