1 Online Convex Programming Problems

Convex Optimization Problems are a key framing choice for Online Learning Problems since these problems have a structure that makes them easy to solve.

1.1 Definition

Online Convex Programming was proposed by Martin Zinkevich in 2003. Problems are framed in the same context of time steps, loss function, experts as the standard weighted majority algorithm. Algorithms for Online Convex Programming preserve many of the same qualities of the weighted majority algorithm while being computationally tractable. An online convex programming problem consists of:

- A set of experts that are elements of some convex set.
- A loss function $l_t(w_t)$ that is a convex over the set of experts
- The goal at each time step of predicting an expert $w_t$ to minimize the regret given by:

$$R(w) = \sum_{t=0}^{T} l_t(w_t) - l_t(w^*).$$  \hfill(1)

where $w_t$ is an expert and $w^*$ is the best expert in retrospect.

The instantaneous regret for some expert $w_t$ at time $t$ is:

$$R_{\text{inst}}(w) = l_t(w_t) - l_t(w^*)$$  \hfill(2)

1.2 Instantaneous regret

Let $l_t = (w_t^T f_i - y_i)^2$ be our loss functions, where $w$ is an expert and $f$ is a feature. We want to minimize the total regret in retrospect with respect to the best expert $w^*$:

$$R(w) = \sum_{t=0}^{T} l_t(w_t) - l_t(w^*).$$  \hfill(3)

We call $l_t(w_t) - l_t(w^*)$ the instantaneous regret for some $w_t$ at time $t$. Additionally, we have

$$l_t(w^*) \geq l_t(w_t) + \nabla l_t(w_t)^T (w^* - w_t)$$

$$l_t(w_t) - l_t(w^*) \leq \nabla l_t(w_t)^T (w_t - w^*).$$  \hfill(4)
The left hand side is the instantaneous regret, and the right hand side is some linear function times 
$(w_t - w^*)$. Thus our total regret will be bounded by $\sum_{t=0}^{T} \nabla l_t(w_t)^T(w_t - w^*)$.

1.3 Algorithm for projected online subgradient descent

This algorithm is a method to minimize the regret for an online convex optimization problem.

Algorithm 1 Projected Subgradient Descent():
1: choose $w_0$
2: for $t = 1...T$ do
3: Incur loss $l(w_t)$ and receive any $\nabla l_t(w_t)$
4: $\hat{w}_{t+1} \leftarrow w_t - \alpha \nabla l_t(w_t)$
5: $w_{t+1} \leftarrow \text{Proj}_C[\hat{w}_{t+1}]$
6: end for

Line 5 projects $\hat{w}_{t+1}$ back into the convex set $C$, and $\alpha$ in line 4 is the learning rate. Smaller $\alpha$ pays a larger upfront cost but is more likely to converge and has a lower regret over time. $\alpha$ can also be dependent on $t$.

Note that the projection will not cause the loss to grow, because it will bring $\hat{w}_{t+1}$ closer to any member of $C$, and thus closer to the optimal expert $w^*$ too.

2 Regret bounds for projected subgradient descent

2.1 Distance between $w_t$ and $w^*$

The distance between $w_t$ and $w^*$ at time $t$ is defined as

$$D(w_t, w^*) = (w_t - w^*)^T(w_t - w^*)$$

(5)

Now we look at

$$D(w_{t+1}, w^*) - D(w_t, w^*)$$

$$= (w_t - \alpha \nabla l_t(w_t) - w^*)^2 - (w_t - w^*)^2$$

$$= (z_t - \alpha \nabla l_t(w_t))^2 - z_t^2$$

$$= \alpha^2(\nabla l_t(w_t))^2 - 2\alpha \nabla l_t^T(w_t)z_t,$$

(6)
where \( z_t = w_t - w^* \). If we sum all of the terms over time, the intermediate terms will all cancel out and leave just \( D(w_T, w^*) - D(w_0, w^*) \).

\[
D(w_T, w^*) - D(w_0, w^*) = \sum_t D(w_{t+1}, w^*) - D(w_t, w^*) \nonumber
\]

\[
= -2\alpha \sum_t (w_t - w^*) \nabla l_t + \alpha^2 \sum_t |\nabla l_t|^2 \nonumber
\]

\[
= D(w_T, w^*) - D(w_0, w^*) \nonumber
\]

\[
\leq -2\alpha \sum_t (w_t - w^*) \nabla l_t + \alpha^2 GT, \quad (7)
\]

where \( |\nabla l_t|^2 \leq G \). Thus we have

\[
2\alpha R_T \leq 2\alpha \sum_t (w_t - w^*) \nabla l_t \leq D(w_0, w^*) - D(w_T, w^*) + \alpha^2 GT \quad (8)
\]

Since the distance between \( w_T \) and \( w^* \) is always non-negative, we can throw away the \( D(w_T, w^*) \) term and still keep the inequality valid.

\[
R_T \leq \sum_t (w_t - w^*) \nabla l_t \leq \frac{D(w_0, w^*)}{2\alpha} + \frac{\alpha GT}{2} \leq \frac{\alpha GT}{2} + \frac{F}{2\alpha^2}, \quad (9)
\]

where \( F \) is the largest distance between any two experts in the set.

Suppose we set \( \alpha = \sqrt{\frac{F}{GT}} \), then the upper bound for total regret is bounded by \( \sqrt{GTF} \), growing sub linearly of \( T \).

### 3 Bayes Rule as Generalized Weighted Majority Update

- **Bayes’ Rule** is a special case of weighted majority
- **Expert** \( e_i \)’s prediction is a probability distribution: \( p_i(y) \)
- **Predict:**
  - Choose expert \( e_i \) in proportion to \( \frac{w_i}{\sum_j w_j} \)
  - Predict the same as what expert \( e_i \) predicts
- **Standard loss for making a probabilistic prediction is log-loss:**

\[
l_t(i) = \log \frac{1}{p_i(y_t)} \quad (10)
\]

Where \( y_t \) is the true observation

- **Plugging the log-loss into the weight update rule:**

\[
w_{t+1}^{i} = w_t^i e^{\alpha \log (p_i(y_t))} \quad (11)
\]

- This simplifies to:

\[
w_{t+1}^i = w_t^i [p_i(y_t)]^\alpha \quad (12)
\]

- Which, when \( \alpha = 1 \), is Bayes’ Rule exactly. According to Bayes’ Rule: \( p(i|y) \propto p(y|i)p(i) \). In this case, \( p(i|y) \) is equivalent to \( w_{t+1}^i \), \( p(i) \) is equivalent to \( w_t^i \), and \( p(y|i) \) is \( p_i(y_t) \).