Statistical Techniques in Robotics (16-831, S22) Lecture #04 (Monday, January 31)

PWEA (WMA & RWMA)

Lecturer: Kris Kitani *Scribes: Mansi Agarwal, Ashwin Misra*

1 Review

Up till now, we have discussed Prediction with Expert Advice (PWEA) algorithms including their analysis. We calculated the mistake bounds for the Greedy, Majority and Randomized Greedy (Expected) algorithms, with the Randomized Greedy expected mistake bound being the tightest mistake bound. The concept of regret was introduced as a more general characteristic to describe the "goodness" of an online algorithm. This lecture introduces two algorithms which introduces relaxing the realizability assumption and allows even the best hypothesis to make mistakes. The Weighted Majority Algorithm (WMA) and the Randomized Weighted Majority Algorithm (RWMA) are such algorithms, where in the weights are updated when the prediction doesn't match the ground truth. RWMA is WMA with the difference that a multinomial distribution is created from the weights and the prediction is a sample from this distribution.

1.1 Strategy for Bound Derivation

This is a summary for the five steps required to derive bounds for any online learning algorithm.

1. **Potential Function** definition (The potential function is essentially the size of the version space)
2. **Upper Bound** the potential function
3. **Lower Bound** the potential function
4. **Combine** the upper and lower bounds
5. **Approximate/Use Algebra** to get the performance bound

1.2 Realizability

The assumption that a perfect hypothesis exists. All environmental responses are generated by a target mapping

$$h^* : \mathcal{X} \rightarrow \mathcal{Y}$$
$$h^* \in \mathcal{H}$$

It says that the optimal mapping is contained in the hypothesis class and the learner has access to this perfect hypothesis through the class. When we assume realizability, the performance is evaluated in terms of absolute mistake bounds. A mistake is the value of the loss function for not adopting the perfect hypothesis. The mistake bounds for Randomized Greedy algorithm and majority algorithm were logarithmic in the hypotheses space whereas for the Greedy algorithm, it was linear.

When we don't consider the realizability assumption, we evaluate the performance of an online learner in the terms of 'regret' bound. This is a more realistic performance characteristic as it allows the best hypothesis to even make a few mistakes, which is often true in real-life conditions.

Regret When we relax the realizability assumption

$$h^* \notin \mathcal{H}$$

This means that unlike mistake bounds, there is no guarantee on the mistake bound. The online learner should be competitive with the best fixed hypothesis in $\mathcal{H}$. The performance is evaluated in terms of relative regret bounds. In retrospect, for the learner, not following $h \in \mathcal{H}$. The definition of regret is :

$$\begin{aligned} R^{(T)}(H) &= \sum_{t=1}^{T} l(\hat{y}^{(t)}, y^{(t)}) - \min_h \sum_{t=1}^{T} l(h(x^{(t)}), y^{(t)}) && (1) \\ &= \max_h R^{(T)}(H) && (2) \end{aligned}$$

where, $\sum_{t=1}^{T} l(\hat{y}^{(t)}, y^{(t)})$ represents the cumulative loss of the learner, and $\min \sum_{t=1}^{T} l(h(x^{(t)}), y^{(t)})$ represents the best single hypothesis with the lowest loss. We define the average regret as:

$$\frac{1}{T} R^{(T)}(\mathcal{H}) \tag{3}$$

And we want the average regret to be as small as possible for the learner to perform optimally. We would also not want the regret to explode as $T \to \infty$. Hence the average regret should be bounded. To achieve this, it must grow sub-linearly in T.

The algorithm is a **no-regret** algorithm if and only if $\frac{R(h)}{T} \to 0$ as $T \to \infty$.

The above definition means that as $T \to \infty$, $\frac{R^{(T)}(h)}{T} \to 0$ (regret divided by T) or average regret over time goes to zero. To reiterate, the regret of the algorithm is the difference in the cumulative loss between the best expert and your algorithm.

2 Summary

Till now, we assumed that there exists one perfect hypothesis in the hypothesis set (realizability). In other words, the hypothesis set had one instance which made no mistakes at any time step. Also, in previous algorithms, if a hypothesis made a mistake, it was removed from the set and was not considered later.

In real world, there may or may not exist a perfect hypothesis. Therefore, removing a hypothesis from the set as soon as it made a mistake could eventually lead to an empty hypothesis set and would break the algorithm.

To avoid this, we relax the realizability assumption and allow even the best (note, not necessarily perfect) hypothesis to make a few mistakes. This leads to the question of how do we remove hypothesis from the set? More importantly, do we actually discard them? The following algorithms will help answer these questions. It will also help us estimate the mistake bounds (regret bounds) of the algorithms.

2.1 Weighted Majority Algorithm (WMA)

We now discuss Weighted Majority Algorithm that relaxes the assumption of realizability. The Weighted Majority Algorithm [4] is listed as Algorithm 1 below.

Step 1 initialises a vector of weights $\mathbf{w}^{(1)}$. In Step 2, we initialise a penalty rate η ($\eta > 0$). Step 3 suggests that the inner loop runs from timestep $t = 1$ to $t = T$ ($T \to \infty$). In Step 4, we receive experts' predictions as $\mathbf{x}^{(t)}$. $\mathbf{x}^{(t)}$ is a N-dimensional vector with values $\in \{-1, 1\}$. Step 5 is where the online learner decides which prediction to make. If we notice the equation, we realise that the vector $\mathbf{w}^{(t)}$ acts as weights (for every expert) and the algorithm gets its name from this step. The learner weighs each expert prediction with its associated weight and sums them. The sign of the sum will be positive if majority experts predict $+1$, else it will be negative. Therefore, we look at the sign of the dot product of the weight vector and the experts' prediction vector and predict $\hat{y}^{(t)}$ accordingly. In Step 6, we receive the actual ground truth $y^{(t)}$. Step 7 enlists the weight update rule. The update equation states that the weights are updated only when the learner's prediction doesn't match the actual answer. In fact, when the learner makes a mistake, the weights of the experts (with wrong predictions) are penalised by $-\eta$.

It is important to note the weight update rule as we will be using it in our regret bound analysis.

Algorithm 1 Weighted Majority Algorithm (WMA)

```
1: $\mathbf{w}^{(1)} \leftarrow \{w_n^{(1)} = 1\}$
2: $\eta \leq \frac{1}{2}$
3: for $t = 1, \cdots, T$ do
4:     RECEIVE ($\mathbf{x}^{(t)} \in \{-1, 1\}^N$)
5:     $\hat{y}^{(t)} = \text{sign}\Big(\sum_{n=1}^{N} x_n^{(t)} \cdot w_n^{(t)}\Big) \in \{-1, 1\}$
6:     RECEIVE ($y^{(t)} \in \{-1, 1\}$)
7:     $w_n^{(t+1)} \leftarrow w_n^{(t)}\big(1 - \eta \cdot \mathbf{1}[y^{(t)} \neq x_n^{(t)}]\big)$
8: end for
```

Earlier, our hypothesis class was a set of all experts. In this case, out hypothesis class is the space of all positive weight linear classifiers. We say linear as in this algorithm, we take the sign of the dot product of observations and weights to classify the space. Another significant difference is that earlier, our hypothesis set was finite (set of all experts), but now, the set is infinite as there can be infinite N dimensional weight vectors with positive values.

Moving ahead, we want to estimate the mistake bounds as well as regret bounds of WMA. In the following subsections, we will derive both of these results. But first we state a lemma, which will be useful in deriving the regret bound.

Lemma 1. *For $0 < x < 1$, following two inequalities hold:*

$$-x - x^2 \leq \log(1-x) \tag{4}$$

$$\log(1-x) \leq -x \tag{5}$$

Proof. Applying Taylor series to $\log(1-x)$.

Theorem 2. *(Mistake bound of WMA) Let $M^{(t)}$, $m_i^{(t)}$ respectively be the number of mistakes that have been made by the WMA learner and the i-th hypothesis until the time step t, and N, η be the number of experts and the penalty rate. Then, $M^{(t)}$ is upper-bounded as:*

$$M^{(t)} \leq 2(1+\eta)m_i^{(t)} + \frac{2\ln N}{\eta}$$

Proof. Lets define the potential function $\Phi^{(t)}$ for WMA as follows:

$$\Phi^{(t)} = \sum_{n=1}^{N} w_n^{(t)} \tag{6}$$

where, $w_n^{(t)}$ is the weight for the expert n at time step t. We choose this as the potential function as it is similar to the size of the version space.

Suppose the online learner made a mistake at the time step t. As we are using weighted majority rule, if the online learner makes a mistake then, at least half of weighted experts are mistaken. Therefore, at least half of the weights will accordingly be penalized by $(1-\eta)$, and this leads to the following inequality:

$$\Phi^{(t+1)} \leq \Phi^{(t)}\Big(\frac{1}{2} + \frac{1}{2}(1-\eta)\Big) = \Big(1 - \frac{\eta}{2}\Big)\Phi^{(t)} \tag{7}$$

Here, the first half corresponds to the experts that were correct and the other half that were incorrect and are penalised by $(1-\eta)$.

We know that $\Phi^{(1)} = N$ as the weights are initialized to 1 and the vector is N dimensional.

For $t = 2$,

$$\Phi^{(2)} \leq \Phi^{(1)}\Big(1 - \frac{\eta}{2}\Big)^{M^{(1)}} = N\Big(1 - \frac{\eta}{2}\Big)^{M^{(1)}} \tag{8}$$

Here, $M^{(1)}$ refers to the number of mistakes till timestep $t = 1$.

Recursively applying the above inequality, we can derive the upper bound of the potential function as follows:

$$\Phi^{(t+1)} \leq \Phi^{(1)}\Big(1 - \frac{\eta}{2}\Big)^{M^{(t)}} = N\Big(1 - \frac{\eta}{2}\Big)^{M^{(t)}} \tag{9}$$

From the definition of the potential function, we can also derive the lower-bound of the potential function as follows:

$$\Phi^{(t+1)} = \sum_{n=1}^{N} w_n^{(t+1)} \geq w_i^{(t+1)} \quad \text{for all } 1 \leq i \leq N \tag{10}$$

The above inequality comes from the fact that single element of a sum is less than or equal to the sum if all elements are positive. This aligns with our setup as every weight $w_i^{(t+1)}$ is positive, since they are both initialized and multiplied with positive values.

Moreover, by the weight update rule of WMA, we can get the lower bounds in desired variables as follows:

$$w_i^{(t+1)} = w_i^{(1)}(1-\eta)^{m_i^{(t)}} = 1 \cdot (1-\eta)^{m_i^{(t)}} \tag{11}$$

By combining Eq. (9) and Eq. (11), we can now derive the mathematical relation between $M^{(t)}$ and $m_n^{(t)}$:

$$(1-\eta)^{m_i^{(t)}} \leq \Phi^{(t+1)} \leq N\Big(1-\frac{\eta}{2}\Big)^{M^{(t)}} \tag{12}$$

Finally, we can compute the (upper) bound of mistake $M^{(t)}$ by combining the bounds of $\Phi^{(t+1)}$ (Eq. (12)) and Lemma 1 as follows:

$$\begin{aligned}
m_i^{(t)} \ln(1-\eta) &\leq \ln N + M^{(t)} \ln\Big(1-\frac{\eta}{2}\Big) \quad \text{(by applying ln on both sides in Eq. (12))} \\
m_i^{(t)}(-\eta-\eta^2) &\leq \ln N + M^{(t)} \ln\Big(1-\frac{\eta}{2}\Big) \quad (\because \text{ Lemma 1}) \\
m_i^{(t)}(-\eta-\eta^2) &\leq \ln N + M^{(t)}\Big(-\frac{\eta}{2}\Big) \quad (\because \text{ Lemma 1}) \\
M^{(t)}\Big(\frac{\eta}{2}\Big) &\leq \ln N + m_i^{(t)}(\eta+\eta^2) \\
M^{(t)} &\leq \frac{2\ln N}{\eta} + 2m_i^{(t)}(1+\eta) \quad \forall n
\end{aligned}$$

Therefore, we have obtained the upper bound of mistakes for WMA. □

Here, we see that the mistake bound of WMA depends on the number of experts (N) like previous algorithms, but also depends on the number of mistakes an expert i makes till timestep t ($m_i^{(t)}$). We also notice that this bound is relative as it depends on the performance of an expert. This comes from our relaxation of the realizability. If we assume a perfect expert exists then, $m_i^{(T)} = 0$.

As discussed earlier, we conduct our analysis using the *regret bound* instead of the *mistake bound* under the condition of non-realizability.

Theorem 3. *WMA is* ***not*** *a no-regret algorithm.*

Proof. By the definition of regret (Eqn 1) and Theorem 2, we can obtain the upper bound of *regret* of WMA, $R(h_n)$, as follows:

$$R(h_n) = M^{(T)} - m_n^{(T)} \leq m_n^{(T)} + 2\eta m_n^{(T)} + \frac{2\ln N}{\eta} \tag{13}$$

We now measure the Big-O complexity of regret. $m_n^{(T)}$ grows with time and follows $O(T)$. Therefore, the first two terms on RHS in Eqn 13 ($m_n^{(T)}$ and $2\eta m_n^{(T)}$) follow $O(T)$. The last term ($\frac{2\ln N}{\eta}$) follows $O(1)$. Therefore, the growth of RHS is $O(T)$. In other words, this means that regret grows linearly with time.

As discussed earlier, an algorithm is a "no-regret" algorithm if and only if $\frac{R(h)}{T} \to 0$ as $T \to \infty$. Here, we see that $\frac{R(h)}{T}$ does not converge to zero as $R(h)$ grows linearly with time. Thus, WMA is not a no-regret algorithm. $\square$

We can try and modify WMA to make it a no-regret algorithm by modifying the penalty factor η. If we make $\eta = \frac{1}{t}$, $m_n^{(T)}$ still grows linearly and follows $O(T)$. We see that $2\eta m_n^{(T)}$ does not grow linearly, in fact follows $O(1)$ (which is an improvement). However, the last term ($\frac{2\ln N}{\eta}$) now follows $O(T)$ and the overall complexity is still $O(T)$.

Let us consider $\eta = \frac{1}{\sqrt{t}}$. Now, $m_n^{(T)}$ still grows linearly and follows $O(T)$. We see that $2\eta m_n^{(T)}$ does not grow linearly, but follows $O(\sqrt{T})$ (which is an improvement). The last term ($\frac{2\ln N}{\eta}$) also follows $O(\sqrt{T})$. However, the overall complexity is still $O(T)$.

Therefore, we see that WMA is a bounded regret algorithm (has an upper bound), where error just grows linearly over time (on the optimistic side, the error does not explode). Further, as we will see, using any kind of deterministic prediction, we can't really get a no-regret algorithm, but just by adding some randomization, we will cut the number of mistakes in half. Actually, just by changing one single line of the algorithm, we will cut the number of mistakes in half. This brings us to the study of the Randomized Weighted Majority Algorithm (RWMA) in the next section.

2.2 Randomized Weighted Majority Algorithm (RWMA)

Now, we introduce the Randomized Weighted Majority Algorithm (RWMA) as shown in Algorithm 2, which introduces a random factor to the Weighted Majority Algorithm. It differs from WMA as it employs a multinomial distribution using the weights and then samples the prediction from this distribution.

Step 1 initialises a vector of weights $\mathbf{w}^{(1)}$. In Step 2, we initialise a penalty rate η ($\eta \leq 0.5$). Step 3 suggests that the inner loop runs from timestep $t = 1$ to $t = T$ ($T \to \infty$). In Step 4, we receive experts' predictions as $\mathbf{x}^{(t)}$. $\mathbf{x}^{(t)}$ is a N-dimensional vector with values $\in \{-1, 1\}$. Step 5 is the selector function, where the hypothesis is the multinomial function of the ratio of the weight at step t to the potential function which is the sum of weights at that timestep. In step 6, our prediction $\hat{y}^{(t)}$ is sampled from the multinomial distribution from the previous step. A hypothesis is sampled and the observation is passed through it to get the prediction. In Step 7, we receive the ground truth $y^{(t)}$. Step 7 enlists the weight update rule. The update equation states that the weights are updated only when the learner's prediction doesn't match the actual answer. In fact, when the learner makes a mistake, the weights of the experts (with wrong predictions) are reduced by $-\eta$.

Algorithm 2 Randomized Weighted Majority Algorithm (RWMA)

```
1: $\mathbf{w}^{(1)} \leftarrow \{w_n^{(1)} = 1\}$
2: $\eta \leq 0.5$
3: for $t = 1, \cdots, T$ do
4:     RECEIVE $(\mathbf{x}^{(t)} \in \{-1, 1\}^N)$
5:     $h \sim$ MULTINOMIAL$(\mathbf{w}^{(t)}/\Phi^{(t)})$, where $\Phi^{(t)} = \sum_{n=1}^{N} w_n^{(t)}$
6:     $\hat{y}^{(t)} = h_i(\mathbf{x}^{(t)})$
7:     RECEIVE $(y^{(t)} \in \{-1, 1\})$
8:     $w_n^{(t+1)} \leftarrow w_n^{(t)}\big(1 - \eta \cdot \mathbf{1}[y^{(t)} \neq h_n(\mathbf{x})^{(t)}]\big)$
9: end for
```

With this algorithm, we try to introduce randomization to improve the mistake and regret bounds, as we will see in the proofs below. We state a lemma to help with our further derivations

Lemma 4. *$e^x \geq 1 + x$ for all $x \in \mathbb{R}$.*

Proof. This inequality can be shown by applying Taylor expansion to e^x.

Theorem 5. *(Relative Mistake bound of RWMA) Let $M^{(t)}, m_i^{(t)}$ respectively be the number of mistakes that have been made by the RWMA learner and the n-th hypothesis until the time step t, and N, η be the number of experts and the penalty rate. Then, expected mistakes of learner $\mathbb{E}[M^{(t)}]$ is upper-bounded as:*

$$\mathbb{E}[M^{(t)}] \leq (1 + \eta) m_n^{(t)} + \frac{\log N}{\eta}$$

Proof. Lets define the potential function $\Phi^{(t+1)}$ as the sum of all weights:

$$\Phi^{(t+1)} = \sum_{n=1}^{N} w_n^{(t+1)} \tag{14}$$

With the weight updating step, $w_n^{(t+1)}$ can be written in terms of $w_n^{(t)}$, and to derive the mathematical relation between $\Phi^{(t+1)}$ and $\Phi^{(t)}$:

$$\begin{aligned}
\Phi^{(t+1)} = \sum_n w_n^{(t+1)} &= \sum_n w_n^{(t)}(1 - \eta\alpha_n^{(t)}) \\
&= \sum_n w_n^{(t)} - \sum_n \eta\alpha_n^{(t)} w_n^{(t)} \\
&= \Phi^{(t)} - \frac{\Phi^{(t)}}{\Phi^{(t)}} \sum_n \eta\alpha_n^{(t)} w_n^{(t)} \\
&= \Phi^{(t)} - \Phi^{(t)}\eta \sum_n \alpha_n^{(t)} \frac{w_n^{(t)}}{\Phi^{(t)}} \\
&= \Phi^{(t)} - \Phi^{(t)}\eta \sum_n \alpha_n^{(t)} p_n^{(t)} \quad (\text{where } p_n^{(t)} = \frac{w_n^{(t)}}{\Phi^{(t)}}) \\
&= \Phi^{(t)}\Big(1 - \eta \sum_n \alpha_n^{(t)} p_n^{(t)}\Big)
\end{aligned} \tag{15}$$

Where $p_n^{(t)}$ is the probability of selecting the expert n at step t. Assuming all weights are initialized to 1, $\Phi^{(T+1)}$ can be calculated by induction in Eq. (15) as follows:

$$\Phi^{(T+1)} = \Phi^{(1)} \prod_{t=1}^{T} \Big(1 - \eta \sum_n \alpha_n^{(t)} p_n^{(t)}\Big) \tag{16}$$

$$= N \prod_{t=1}^{T} \Big(1 - \eta \sum_n \alpha_n^{(t)} p_n^{(t)}\Big) \tag{17}$$

We can now derive the upper bound of the potential function $\Phi^{(T+1)}$ by combining Lemma 4 and Eq. (17) by the inequality:

$$\begin{aligned}
\Phi^{(T+1)} &= N \prod_{t=1}^{T} \Big(1 - \eta \sum_n \alpha_n^{(t)} p_n^{(t)}\Big) \\
&\leq N \prod_{t=1}^{T} \exp\Big(-\eta \sum_n \alpha_n^{(t)} p_n^{(t)}\Big) \quad (\because \text{ Lemma } 4) \\
&= N \exp\Big(-\eta \sum_{t=1}^{T} \mathbb{E}\big[\mathbf{1}[y^{(t)} \neq \hat{y}^{(t)}]\big]\Big) \quad (\because \ \sum_n \alpha_n^{(t)} p_n^{(t)} = \mathbb{E}_p\big[\mathbf{1}[y^{(t)} \neq \hat{y}^{(t)}]\big]) \\
&= N \exp\big(-\mathbb{E}[-\eta M^{(T)}]\big) \quad (\because \ ) \qquad (18)
\end{aligned}$$

By applying the same argument as in WMA, the lower bound of $\Phi^{(T+1)}$ is $(1-\eta)^{m_n^{(T)}}$.

$$\Phi^{(T+1)} \geq (1-\eta)^{m_n^{(T)}} \tag{19}$$

Using the strategy, now combining Eq. (18) & (19), we can now combine the bounds $\mathbb{E}[M^{(t)}]$ and $m_n^{(t)}$ to derive the mathematical relation as follows:

$$(1-\eta)^{m_n^{(T)}} \leq \Phi^{(T+1)} \leq N \exp\big(-\mathbb{E}[-\eta M^{(T)}]\big) \tag{20}$$

Finally, we can compute the (upper) bound of expected mistakes $\mathbb{E}[M^{(t)}]$ by combining the bounds of $\Phi^{(t+1)}$ (Eq. (20)) and Lemma 1 as follows:

$$\begin{aligned}
1 - \eta^{m_n^{(T)}} &\leq N \exp\big(-\mathbb{E}[-\eta M^{(T)}]\big) \\
&\qquad \text{(Taking natural log on both sides)} \\
m_n^{(T)} \ln(1-\eta) &\leq \ln N - \eta \mathbb{E}[M^{(T)}] \\
m_n^{(T)}(-\eta - \eta^2) &\leq \ln N - \eta \mathbb{E}[M^{(T)}] \quad (\because \ \text{Lemma } 1) \\
-m_n^{(T)}(1+\eta) &\leq \frac{\ln N}{\eta} - \mathbb{E}[M^{(T)}] \\
\mathbb{E}[M^{(T)}] &\leq \frac{\ln N}{\eta} + (1+\eta) m_n^{(T)}
\end{aligned}$$

We have obtained now the upper bound of the expected mistake bound for RWMA. We can see by comparison that the bound is better than WMA by a factor of 2. □

We are now interested in the performance (regret bound) of RWMA algorithm.

Theorem 6. *RWMA is a* ***no-regret*** *algorithm.*

Proof. By regret as we saw Theorem 5, we could obtain the upper bound of *expected regret* of RWMA, $\mathbb{E}[R]$, by rearranging the terms of the expected mistake bound:

$$\mathbb{E}[M^{(T)}] \leq \frac{\ln N}{\eta} + (1+\eta)m_n^{(T)} \tag{21}$$

$$\mathbb{E}[M^{(T)}] \leq \frac{\ln N}{\eta} + m_n^{(T)} +_n^{(T)} \tag{22}$$

$$\mathbb{E}[M^{(T)}] - m_n^{(T)} \leq \frac{\ln N}{\eta} + \eta m_n^{(T)} \tag{23}$$

$$\mathbb{E}[R] = \mathbb{E}[M^{(T)}] - m_n^{(T)} \leq \eta m_n^{(T)} + \frac{\ln N}{\eta} \tag{24}$$

As discussed earlier, an algorithm is a ”no-regret” algorithm if and only if

$$\mathbb{E}[R] \to 0 \text{ as } T \to \infty \tag{25}$$

Here, we see that $\mathbb{E}[R]$ does not converge to zero as $\mathbb{E}[R]$ grows linearly with time. Thus, RWMA is not a no-regret algorithm. This statement is false if we employ the correct adaptive penalty term η.

If we make $\eta = \frac{1}{t}$, We see that $\eta m_n^{(T)}$ does not grow linearly, in fact follows $O(1)$ (which is an improvement). However, the last term ($\frac{\ln N}{\eta}$) now follows $O(T)$ and the overall complexity is still $O(T)$.

Let us consider $\eta = \frac{1}{\sqrt{t}}$. We see that $\eta m_n^{(T)}$ does not grow linearly, but follows $O(\sqrt{T})$ (which is an improvement). The last term ($\frac{\ln N}{\eta}$) also follows $O(\sqrt{T})$. Therefore, the overall complexity is now $O(\sqrt{T})$. Hence, RWMA with correct adaptive penalty term is a no-regret algorithm as the expected value of the regret grow sub-linearly with time. □

3 Appendix

We use a simple example to see WMA in action. There is a True/False quiz where we are allowed to seek advice from n advisors throughout the game. Let the advisors' decisions at each round be based on tossing coins whose bias is sampled uniformly. Likewise, the game generates questions whose answers follows a Bernoulli distribution.

Following is a simple Python code [3] employing WMA to win the quiz (with least mistakes).

```python
import numpy as np
# let's try 100 games each with 1000 rounds,
num_runs = 100
T = 1000

diff = np.zeros((num_runs, T))
for run in range(num_runs):
    n = np.random.randint(1,10)
   ps = np.random.random(n)
   ws = np.ones(n)
   p_g = np.random.random()
   eta = 0.25

   # your (online learner) mistakes
   m = 0
   # advisors mistakes
   ms = np.zeros(n, dtype=np.int32)

   for t in range(T):
       # game correct answer
       z = np.random.random() >= p_g
       # advisors answers
       xs = np.random.random(n) >= ps
       # your answer
       y = np.sum(xs * ws) >= np.sum((1-xs) * ws)

       # your mistakes
       m += (y != z)
       # advistors mistakes
       ms[xs != z] += 1

       # theoretical bound
       th_ub = 2 * (1+eta) * min(ms) + 2. * np.log(n) / eta
       diff[run, t] = th_ub - m

       # update weights
       ws[xs != z] = (1-eta) * ws[xs != z]


assert np.min(diff) >= 0, "Theoretical gap should be non-negative"
```

For more intuitive running examples for both WMA and RWMA, refer to the slides by Sylvester et al [2].

In fact, these algorithms are not limited to Prediction with Expert Advice (PWEA) problems. They can be used to solve problems like:

1. Linear Classification
2. Non-Linear Classification
3. Linear Programs
4. Exponentiated Gradient Descent
5. Repeated Matrix Games

In addition to WMA and RWMA, there exists a meta algorithm known as the multiplicative weights algorithm. As discussed in the article [1], the decision i is chosen with probability proportional to the weight $w_i^{(t)}$. This is the same as distribution over decisions $p^{(t)} = \{w_i^{(t)}/\Phi^{(t)}, ..., w_n^{(t)}/\Phi^{(t)}\}$, with the weight updating and penalty steps being the same as WMA. This is not a majority algorithm, it substitutes the majority by calculating the proportional probability at each step.

References

[1] Decision-making under total uncertainty: the multiplicative weight algorithm, https://www.cs.princeton.edu/courses/archive/fall13/cos521/lecnotes/lec8.pdf.

[2] Online Learning using Expert Advice , https://www.cl.cam.ac.uk/teaching/1920/probablty/materials/lecture15.pdf.

[3] Weighted Majority Algorithm Code, https://ash-aldujaili.github.io/blog/2018/01/08/wma/.

[4] N. Littlestone, M. K. Warmuth, et al. *The weighted majority algorithm.*