Experts and Adversarial Bandits

Algorithms and Analysis

[1]ECE
Technical University of Crete

March 31, 2023

So Far: assumed arm rewards are IID ("stochastic")

- Q: Let reward for arm i take value 1 with probability p_i or 0 with probability $1 - p_i$. Does this satisfy the stochastic MAB environment assumptions?

So Far: assumed arm rewards are IID ("stochastic")

- Q: Let reward for arm i take value 1 with probability p_i or 0 with probability $1 - p_i$. Does this satisfy the stochastic MAB environment assumptions?
- A: Yes! This was the typical example we considered in our code.

So Far: assumed arm rewards are IID ("stochastic")

- Q: Let reward for arm i take value 1 with probability p_i or 0 with probability $1 - p_i$. Does this satisfy the stochastic MAB environment assumptions?
- A: Yes! This was the typical example we considered in our code.
- Q: What if reward for i is uniformly chosen in the interval $[0.2, 0.7]$ at every round?

So Far: assumed arm rewards are IID (“stochastic”)

- Q: Let reward for arm i take value 1 with probability p_i or 0 with probability $1 - p_i$. Does this satisfy the stochastic MAB environment assumptions?
- A: Yes! This was the typical example we considered in our code.
- Q: What if reward for i is uniformly chosen in the interval $[0.2, 0.7]$ at every round?
- A: Yes! This is also part of our framework so far (and is what is asked in your 1st assignment).

- Q: Assume arm i is a simple 2-state Markov chain: Its reward is either 0 or 1 again, but consecutive rewards work as follows:

- Q: Assume arm i is a simple 2-state Markov chain: Its reward is either 0 or 1 again, but consecutive rewards work as follows:
  - When previous reward was 0, next reward is 1 with probability p, or 0 with probability $1 - p$.
  - When previous reward was 1, next reward is 0 with probability q, or 1 with probability $1 - q$.
  - The first reward (at time 0) is 0.

- Q: Assume arm i is a simple 2-state Markov chain: Its reward is either 0 or 1 again, but consecutive rewards work as follows:
    - When previous reward was 0, next reward is 1 with probability p, or 0 with probability $1 - p$.
    - When previous reward was 1, next reward is 0 with probability q, or 1 with probability $1 - q$.
    - The first reward (at time 0) is 0.

  Is *this* a "stochastic" MAB?

- Q: Assume arm i is a simple 2-state Markov chain: Its reward is either 0 or 1 again, but consecutive rewards work as follows:
    - When previous reward was 0, next reward is 1 with probability p, or 0 with probability $1-p$.
    - When previous reward was 1, next reward is 0 with probability q, or 1 with probability $1-q$.
    - The first reward (at time 0) is 0.

  Is *this* a "stochastic" MAB?
- A: Not really...(food for thought: where does the proof break?)

Non-IID rewards (cnt'd)

- Q: How about if arm i has reward 1 for the first $T/2$ rounds, then reward 0 for the rest $T/2$.

Non-IID rewards (cnt'd)

- Q: How about if arm i has reward 1 for the first $T/2$ rounds, then reward 0 for the rest $T/2$.
- A: No! unfortunately reward process is not stationary.

Non-IID rewards (cnt'd)

- Q: How about if arm i has reward 1 for the first $T/2$ rounds, then reward 0 for the rest $T/2$.
- A: No! unfortunately reward process is not stationary.
- Q: What if an adversary knows our algorithm and thus which arm we will play in the next round, then sets all other rewards to 1 and the chosen arm only to 0??

Non-IID rewards (cnt'd)

- Q: How about if arm i has reward 1 for the first $T/2$ rounds, then reward 0 for the rest $T/2$.
- A: No! unfortunately reward process is not stationary.
- Q: What if an adversary knows our algorithm and thus which arm we will play in the next round, then sets all other rewards to 1 and the chosen arm only to 0??
- A: This is an "adversarial" bandit $\rightarrow$ seems like a more difficult (hopeless?) environment

Stochastic MAB algorithms for Adversarial Bandits

- We would like algorithms that have *sublinear regret* even for adversarial bandits.

Stochastic MAB algorithms for Adversarial Bandits

- We would like algorithms that have *sublinear regret* even for adversarial bandits.
- Q: Would deterministic algorithms (e.g., Explore-then-Exploit, UCB) work in such environments?

Stochastic MAB algorithms for Adversarial Bandits

- We would like algorithms that have *sublinear regret* even for adversarial bandits.
- Q: Would deterministic algorithms (e.g., Explore-then-Exploit, UCB) work in such environments?

Example

- Assume 2 arms only
- Adversary knows our algorithm. Since it's deterministic, it also knows the arm it will pick next.

Stochastic MAB algorithms for Adversarial Bandits

- We would like algorithms that have *sublinear regret* even for adversarial bandits.
- Q: Would deterministic algorithms (e.g., Explore-then-Exploit, UCB) work in such environments?

Example

- Assume 2 arms only
- Adversary knows our algorithm. Since it's deterministic, it also knows the arm it will pick next.
- It sets reward for chosen arm to 0, and other arm to 1.

Stochastic MAB algorithms for Adversarial Bandits

- We would like algorithms that have *sublinear regret* even for adversarial bandits.
- Q: Would deterministic algorithms (e.g., Explore-then-Exploit, UCB) work in such environments?

Example

- Assume 2 arms only
- Adversary knows our algorithm. Since it's deterministic, it also knows the arm it will pick next.
- It sets reward for chosen arm to 0, and other arm to 1.
- Quiz: What is the best regret we can hope for?

Stochastic MAB algorithms for Adversarial Bandits

- We would like algorithms that have *sublinear regret* even for adversarial bandits.
- Q: Would deterministic algorithms (e.g., Explore-then-Exploit, UCB) work in such environments?

Example

- Assume 2 arms only
- Adversary knows our algorithm. Since it's deterministic, it also knows the arm it will pick next.
- It sets reward for chosen arm to 0, and other arm to 1.
- Quiz: What is the best regret we can hope for?
- Reward $R(T) = 0$ for the algorithm

Stochastic MAB algorithms for Adversarial Bandits

- We would like algorithms that have *sublinear regret* even for adversarial bandits.
- Q: Would deterministic algorithms (e.g., Explore-then-Exploit, UCB) work in such environments?

Example

- Assume 2 arms only
- Adversary knows our algorithm. Since it's deterministic, it also knows the arm it will pick next.
- It sets reward for chosen arm to 0, and other arm to 1.
- Quiz: What is the best regret we can hope for?
- Reward $R(T) = 0$ for the algorithm
- Reward for other arm (which is optimal) $R^*(T) \geq T/2$

Stochastic MAB algorithms for Adversarial Bandits

- We would like algorithms that have *sublinear regret* even for adversarial bandits.
- Q: Would deterministic algorithms (e.g., Explore-then-Exploit, UCB) work in such environments?

Example

- Assume 2 arms only
- Adversary knows our algorithm. Since it's deterministic, it also knows the arm it will pick next.
- It sets reward for chosen arm to 0, and other arm to 1.
- Quiz: What is the best regret we can hope for?
- Reward $R(T) = 0$ for the algorithm
- Reward for other arm (which is optimal) $R^*(T) \geq T/2$

Theorem

Any deterministic algorithm fails: it achieves $\mathbf{\Omega(T)}$ (i.e. linear) regret!

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).
- Claim: $Q_{i^*} \leq T/k$.

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).
- Claim: $Q_{i^*} \leq T/k$.
    - Proof by contradiction: Assume $Q_{i^*} > T/k$.

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).
- Claim: $Q_{i^*} \leq T/k$.
    - Proof by contradiction: Assume $Q_{i^*} > T/k$.
    - $Q_j \geq Q_{i^*} > T/k, \forall j$ (because of (1))

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).
- Claim: $Q_{i^*} \leq T/k$.
  - Proof by contradiction: Assume $Q_{i^*} > T/k$.
  - $Q_j \geq Q_{i^*} > T/k, \forall j$ (because of (1))
  - Hence $\sum_{i=1}^{k} Q_i > \sum_{i=1}^{k} T/k$

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).
- Claim: $Q_{i^*} \leq T/k$.
  - Proof by contradiction: Assume $Q_{i^*} > T/k$.
  - $Q_j \geq Q_{i^*} > T/k, \forall j$ (because of (1))
  - Hence $\sum_{i=1}^{k} Q_i > \sum_{i=1}^{k} T/k > T$ (not possible)

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).
- Claim: $Q_{i^*} \leq T/k$.
  - Proof by contradiction: Assume $Q_{i^*} > T/k$.
  - $Q_j \geq Q_{i^*} > T/k, \forall j$ (because of (1))
  - Hence $\sum_{i=1}^k Q_i > \sum_{i=1}^k T/k > T$ (not possible)
  - Optimal reward $R_*^T \geq T - T/k$

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).
- Claim: $Q_{i^*} \leq T/k$.
  - Proof by contradiction: Assume $Q_{i^*} > T/k$.
  - $Q_j \geq Q_{i^*} > T/k, \forall j$ (because of (1))
  - Hence $\sum_{i=1}^{k} Q_i > \sum_{i=1}^{k} T/k > T$ (not possible)
  - Optimal reward $R_*^T \geq T - T/k$ (always gets 1 when not chosen)

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).
- Claim: $Q_{i^*} \leq T/k$.
  - Proof by contradiction: Assume $Q_{i^*} > T/k$.
  - $Q_j \geq Q_{i^*} > T/k, \forall j$ (because of (1))
  - Hence $\sum_{i=1}^{k} Q_i > \sum_{i=1}^{k} T/k > T$ (not possible)
  - Optimal reward $R_*^T \geq T - T/k$ (always gets 1 when not chosen)
  - $Regret^T = R_*^T - R_{algo}^T$

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \dots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).
- Claim: $Q_{i^*} \leq T/k$.
  - Proof by contradiction: Assume $Q_{i^*} > T/k$.
  - $Q_j \geq Q_{i^*} > T/k, \forall j$ (because of (1))
  - Hence $\sum_{i=1}^{k} Q_i > \sum_{i=1}^{k} T/k > T$ (not possible)
  - Optimal reward $R_*^T \geq T - T/k$ (always gets 1 when not chosen)
  - $Regret^T = R_*^T - R_{algo}^T \geq T - T/k - 0$

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).
- Claim: $Q_{i^*} \leq T/k$.
  - Proof by contradiction: Assume $Q_{i^*} > T/k$.
  - $Q_j \geq Q_{i^*} > T/k, \forall j$ (because of (1))
  - Hence $\sum_{i=1}^k Q_i > \sum_{i=1}^k T/k > T$ (not possible)
  - Optimal reward $R_*^T \geq T - T/k$ (always gets 1 when not chosen)
  - $Regret^T = R_*^T - R_{algo}^T \geq T - T/k - 0 = O(T)$

Adversarial Bandits: A Negative Result

Proof of linear regret

- Assume the previous adversarial environment.
- Let Q_i denote again the number of times arm i was played by the algorithm
- (1) Optimal arm $i^* = argmin_i\{Q_1, Q_2, \ldots, Q_k\}$ (arm gets reward 1, unless it is played $\Rightarrow$ best arm is the one least chosen by our algorithm).
- Claim: $Q_{i^*} \leq T/k$.
  - Proof by contradiction: Assume $Q_{i^*} > T/k$.
  - $Q_j \geq Q_{i^*} > T/k, \forall j$ (because of (1))
  - Hence $\sum_{i=1}^{k} Q_i > \sum_{i=1}^{k} T/k > T$ (not possible)
  - Optimal reward $R_*^T \geq T - T/k$ (always gets 1 when not chosen)
  - $Regret^T = R_*^T - R_{algo}^T \geq T - T/k - 0 = O(T)$

- In other words *even UCB would not give sublinear regret in this setup*

Experts Framework

Adverserial Bandit problem have two difficulties:

Experts Framework

Adverserial Bandit problem have two difficulties:

1. **Hard to predict:** Rewards per arm harder to “predict” than stochastic bandits

Experts Framework

Adverserial Bandit problem have two difficulties:

1. **Hard to predict:** Rewards per arm harder to "predict" than stochastic bandits
2. **Partially Observable:** Only learn reward for played arm.

Experts Framework

Adverserial Bandit problem have two difficulties:

1. **Hard to predict:** Rewards per arm harder to "predict" than stochastic bandits
2. **Partially Observable:** Only learn reward for played arm.

Experts framework: address 1st difficulty, and relax the 2nd (for now).

Experts Framework

Adverserial Bandit problem have two difficulties:

1. **Hard to predict:** Rewards per arm harder to "predict" than stochastic bandits
2. **Partially Observable:** Only learn reward for played arm.

Experts framework: address 1st difficulty, and relax the 2nd (for now).

Experts: simple classification setup

- k experts: expert i predicts $x_i(t) \in \{0, 1\}$ (these are binary classifiers predicting e.g., spam, elections, image class, football score)

Experts Framework

Adverserial Bandit problem have two difficulties:

1. **Hard to predict:** Rewards per arm harder to "predict" than stochastic bandits
2. **Partially Observable:** Only learn reward for played arm.

Experts framework: address 1st difficulty, and relax the 2nd (for now).

Experts: simple classification setup

- k experts: expert i predicts $x_i(t) \in \{0, 1\}$ (these are binary classifiers predicting e.g., spam, elections, image class, football score)
- Their goal is to predict the true label at time t, $y(t) \in \{0, 1\}$

Experts Framework

Adverserial Bandit problem have two difficulties:

1. **Hard to predict:** Rewards per arm harder to “predict” than stochastic bandits
2. **Partially Observable:** Only learn reward for played arm.

Experts framework: address 1st difficulty, and relax the 2nd (for now).

Experts: simple classification setup

- k experts: expert i predicts $x_i(t) \in \{0, 1\}$ (these are binary classifiers predicting e.g., spam, elections, image class, football score)
- Their goal is to predict the true label at time t, $y(t) \in \{0, 1\}$
- Algorithm chooses $\hat{y}(t) = f\left(x_1(0 \ldots t), x_2(0 \ldots t), \ldots, x_k(0 \ldots t), y(0 \ldots t-1)\right)$

Experts Framework

Adverserial Bandit problem have two difficulties:

1. **Hard to predict:** Rewards per arm harder to "predict" than stochastic bandits
2. **Partially Observable:** Only learn reward for played arm.

Experts framework: address 1st difficulty, and relax the 2nd (for now).

Experts: simple classification setup

- k experts: expert i predicts $x_i(t) \in \{0, 1\}$ (these are binary classifiers predicting e.g., spam, elections, image class, football score)
- Their goal is to predict the true label at time t, $y(t) \in \{0, 1\}$
- Algorithm chooses $\hat{y}(t) = f\left(x_1(0 \ldots t), x_2(0 \ldots t), \ldots, x_k(0 \ldots t), y(0 \ldots t-1)\right)$ (i.e., as some function of **current and past** expert predictions

Experts Framework

Adverserial Bandit problem have two difficulties:

1. **Hard to predict:** Rewards per arm harder to "predict" than stochastic bandits
2. **Partially Observable:** Only learn reward for played arm.

Experts framework: address 1st difficulty, and relax the 2nd (for now).

Experts: simple classification setup

- k experts: expert i predicts $x_i(t) \in \{0, 1\}$ (these are binary classifiers predicting e.g., spam, elections, image class, football score)
- Their goal is to predict the true label at time t, $y(t) \in \{0, 1\}$
- Algorithm chooses $\hat{y}(t) = f\left(x_1(0 \ldots t), x_2(0 \ldots t), \ldots, x_k(0 \ldots t), y(0 \ldots t-1)\right)$ (i.e., as some function of **current and past** expert predictions **and past** outcomes.)

Experts Framework

Adverserial Bandit problem have two difficulties:

1. **Hard to predict:** Rewards per arm harder to "predict" than stochastic bandits
2. **Partially Observable:** Only learn reward for played arm.

Experts framework: address 1st difficulty, and relax the 2nd (for now).

Experts: simple classification setup

- k experts: expert i predicts $x_i(t) \in \{0, 1\}$ (these are binary classifiers predicting e.g., spam, elections, image class, football score)
- Their goal is to predict the true label at time t, $y(t) \in \{0, 1\}$
- Algorithm chooses $\hat{y}(t) = f\left(x_1(0 \ldots t), x_2(0 \ldots t), \ldots, x_k(0 \ldots t), y(0 \ldots t-1)\right)$ (i.e., as some function of **current and past** expert predictions **and past** outcomes.)
- True label $y(t)$ is revealed after $\hat{y}(t)$ is chosen, and a **loss** is incurred (e.g., -1) for every misclassification

Algorithm 1: Majority rule

Assume first: there exists an expert i^* that never makes mistakes

Algorithm 1: Majority rule

Assume first: there exists an expert i^* that never makes mistakes

Majority Rule (MR) Algorithm

- Start with set $K^0 = \{1, 2, \ldots, k\}$ (experts with no mistakes yet)

Algorithm 1: Majority rule

Assume first: there exists an expert i^* that never makes mistakes

Majority Rule (MR) Algorithm

- Start with set $K^0 = \{1, 2, \ldots, k\}$ (experts with no mistakes yet)
- At round t:

Algorithm 1: Majority rule

Assume first: there exists an expert i^* that never makes mistakes

Majority Rule (MR) Algorithm

- Start with set $K^0 = \{1, 2, \ldots, k\}$ (experts with no mistakes yet)
- At round t:
  - Let $N_0^t = \{i \in K^t : x_i(t) = 0\}$

Algorithm 1: Majority rule

Assume first: there exists an expert i^* that never makes mistakes

Majority Rule (MR) Algorithm

- Start with set $K^0 = \{1, 2, \ldots, k\}$ (experts with no mistakes yet)
- At round t:
  - Let $N_0^t = \{i \in K^t : x_i(t) = 0\}$ (experts predicting 0)

Algorithm 1: Majority rule

Assume first: there exists an expert i^* that never makes mistakes

Majority Rule (MR) Algorithm

- Start with set $K^0 = \{1, 2, \ldots, k\}$ (experts with no mistakes yet)
- At round t:
  - Let $N_0^t = \{i \in K^t : x_i(t) = 0\}$ (experts predicting 0)
  - Let $N_1^t = \{i \in K^t : x_i(t) = 1\}$ (experts predicting 1)

Algorithm 1: Majority rule

Assume first: there exists an expert i^* that never makes mistakes

Majority Rule (MR) Algorithm

- Start with set $K^0 = \{1, 2, \ldots, k\}$ (experts with no mistakes yet)
- At round t:
  - Let $N_0^t = \{i \in K^t : x_i(t) = 0\}$ (experts predicting 0)
  - Let $N_1^t = \{i \in K^t : x_i(t) = 1\}$ (experts predicting 1)
  - Choose:
    - $\hat{y}(t) = 0$, if $\|N_0^t\| \geq \|N_1^t\|$

Algorithm 1: Majority rule

Assume first: there exists an expert i^* that never makes mistakes

Majority Rule (MR) Algorithm

- Start with set $K^0 = \{1, 2, \ldots, k\}$ (experts with no mistakes yet)
- At round t:
  - Let $N_0^t = \{i \in K^t : x_i(t) = 0\}$ (experts predicting 0)
  - Let $N_1^t = \{i \in K^t : x_i(t) = 1\}$ (experts predicting 1)
  - Choose:
    - $\hat{y}(t) = 0$, if $\|N_0^t\| \geq \|N_1^t\|$
    - $\hat{y}(t) = 1$, if $\|N_0^t\| < \|N_1^t\|$

Algorithm 1: Majority rule

Assume first: there exists an expert i^* that never makes mistakes

Majority Rule (MR) Algorithm

- Start with set $K^0 = \{1, 2, \ldots, k\}$ (experts with no mistakes yet)
- At round t:
  - Let $N_0^t = \{i \in K^t : x_i(t) = 0\}$ (experts predicting 0)
  - Let $N_1^t = \{i \in K^t : x_i(t) = 1\}$ (experts predicting 1)
  - Choose:
    - $\hat{y}(t) = 0$, if $\|N_0^t\| \geq \|N_1^t\|$
    - $\hat{y}(t) = 1$, if $\|N_0^t\| < \|N_1^t\|$
  - $K^{t+1} = K^t \setminus \{i : x_i(t) \neq y(t)\}$

Algorithm 1: Majority rule

Assume first: there exists an expert i^* that never makes mistakes

Majority Rule (MR) Algorithm

- Start with set $K^0 = \{1, 2, \ldots, k\}$ (experts with no mistakes yet)
- At round t:
  - Let $N_0^t = \{i \in K^t : x_i(t) = 0\}$ (experts predicting 0)
  - Let $N_1^t = \{i \in K^t : x_i(t) = 1\}$ (experts predicting 1)
  - Choose:
    - $\hat{y}(t) = 0$, if $\|N_0^t\| \geq \|N_1^t\|$
    - $\hat{y}(t) = 1$, if $\|N_0^t\| < \|N_1^t\|$
  - $K^{t+1} = K^t \setminus \{i : x_i(t) \neq y(t)\}$ (remove experts that made a mistake)

Majority Rule (MR) Performance

- Quiz: how many mistakes does MR make?

Majority Rule (MR) Performance

- Quiz: how many mistakes does MR make?

Theorem

Majority rule algorithm makes $\mathbf{log_2 k}$ mistakes (if a perfect expert exists)

Majority Rule (MR) Performance

- Quiz: how many mistakes does MR make?

Theorem

Majority rule algorithm makes $\mathbf{log_2 k}$ mistakes (if a perfect expert exists)

Proof:

1. If $\hat{y}(t) \neq y(t)$ (mistake), then $\|K^{t+1}\| \leq \frac{\|K^t\|}{2}$

Majority Rule (MR) Performance

- Quiz: how many mistakes does MR make?

Theorem

Majority rule algorithm makes $\mathbf{log_2 k}$ mistakes (if a perfect expert exists)

Proof:

1. If $\hat{y}(t) \neq y(t)$ (mistake), then $\|K^{t+1}\| \leq \frac{\|K^t\|}{2}$ (majority - i.e., more than half of experts - made a mistake)

Majority Rule (MR) Performance

- Quiz: how many mistakes does MR make?

Theorem

Majority rule algorithm makes $\mathbf{log_2 k}$ mistakes (if a perfect expert exists)

Proof:

1. If $\hat{y}(t) \neq y(t)$ (mistake), then $\|K^{t+1}\| \leq \frac{\|K^t\|}{2}$ (majority - i.e., more than half of experts - made a mistake)
2. Assume the N^{th} mistake took place at round t.

Majority Rule (MR) Performance

- Quiz: how many mistakes does MR make?

Theorem

Majority rule algorithm makes $\mathbf{log_2 k}$ mistakes (if a perfect expert exists)

Proof:

1. If $\hat{y}(t) \neq y(t)$ (mistake), then $\|K^{t+1}\| \leq \frac{\|K^t\|}{2}$ (majority - i.e., more than half of experts - made a mistake)
2. Assume the N^{th} mistake took place at round t. (1) $\Rightarrow \|K^{t+1}\| \leq \frac{k}{2^N}$

Majority Rule (MR) Performance

- Quiz: how many mistakes does MR make?

Theorem

Majority rule algorithm makes $\mathbf{log_2 k}$ mistakes (if a perfect expert exists)

Proof:

1. If $\hat{y}(t) \neq y(t)$ (mistake), then $\|K^{t+1}\| \leq \frac{\|K^t\|}{2}$ (majority - i.e., more than half of experts - made a mistake)
2. Assume the N^{th} mistake took place at round t. (1) $\Rightarrow \|K^{t+1}\| \leq \frac{k}{2^N}$ (every mistakes cuts set K by at least $1/2$).

Majority Rule (MR) Performance

- Quiz: how many mistakes does MR make?

Theorem

Majority rule algorithm makes $\mathbf{log_2 k}$ mistakes (if a perfect expert exists)

Proof:

1. If $\hat{y}(t) \neq y(t)$ (mistake), then $\|K^{t+1}\| \leq \frac{\|K^t\|}{2}$ (majority - i.e., more than half of experts - made a mistake)
2. Assume the N^{th} mistake took place at round t. (1) $\Rightarrow \|K^{t+1}\| \leq \frac{k}{2^N}$ (every mistakes cuts set K by at least $1/2$).
3. But $\|K^t\| \geq 1, \forall t$ (since there is a perfect expert)

Majority Rule (MR) Performance

- Quiz: how many mistakes does MR make?

Theorem

Majority rule algorithm makes $\mathbf{log_2 k}$ mistakes (if a perfect expert exists)

Proof:

1. If $\hat{y}(t) \neq y(t)$ (mistake), then $\|K^{t+1}\| \leq \frac{\|K^t\|}{2}$ (majority - i.e., more than half of experts - made a mistake)
2. Assume the N^{th} mistake took place at round t. $(1) \Rightarrow \|K^{t+1}\| \leq \frac{k}{2^N}$ (every mistakes cuts set K by at least $1/2$).
3. But $\|K^t\| \geq 1, \forall t$ (since there is a perfect expert)
$(2) \Rightarrow \frac{k}{2^N} \geq 1$

Majority Rule (MR) Performance

- Quiz: how many mistakes does MR make?

Theorem

Majority rule algorithm makes $\mathbf{log_2 k}$ mistakes (if a perfect expert exists)

Proof:

1. If $\hat{y}(t) \neq y(t)$ (mistake), then $\|K^{t+1}\| \leq \frac{\|K^t\|}{2}$ (majority - i.e., more than half of experts - made a mistake)
2. Assume the N^{th} mistake took place at round t. (1) $\Rightarrow \|K^{t+1}\| \leq \frac{k}{2^N}$ (every mistakes cuts set K by at least $1/2$).
3. But $\|K^t\| \geq 1, \forall t$ (since there is a perfect expert) (2) $\Rightarrow \frac{k}{2^N} \geq 1 \Rightarrow N \leq log_2 k$

Non-perfect Experts

- Assume now that every expert can make mistakes.

Non-perfect Experts

- Assume now that every expert can make mistakes.
- Quiz: What is the expected number of mistakes the MR algorithms makes in T steps?

Non-perfect Experts

- Assume now that every expert can make mistakes.
- Quiz: What is the expected number of mistakes the MR algorithms makes in T steps?
- Answer: Normally, MR will eliminate all experts at some point.

Non-perfect Experts

- Assume now that every expert can make mistakes.
- Quiz: What is the expected number of mistakes the MR algorithms makes in T steps?
- Answer: Normally, MR will eliminate all experts at some point. Even if we assume it stops when $\|K^t\| = 1$, MR makes $O(T)$ mistakes if no expert is perfect. Why?

Non-perfect Experts

- Assume now that every expert can make mistakes.
- Quiz: What is the expected number of mistakes the MR algorithms makes in T steps?
- Answer: Normally, MR will eliminate all experts at some point. Even if we assume it stops when $\|K^t\| = 1$, MR makes $O(T)$ mistakes if no expert is perfect. Why?
- Finite probability that the best expert will be eliminated prematurely. Then regret will accumulate linearly.

Non-perfect Experts

- Assume now that every expert can make mistakes.
- Quiz: What is the expected number of mistakes the MR algorithms makes in T steps?
- Answer: Normally, MR will eliminate all experts at some point. Even if we assume it stops when $\|K^t\| = 1$, MR makes $O(T)$ mistakes if no expert is perfect. Why?
- Finite probability that the best expert will be eliminated prematurely. Then regret will accumulate linearly. (in fact, this probability goes to 1 as $k \to \infty$)

Non-perfect Experts

- Assume now that every expert can make mistakes.
- Quiz: What is the expected number of mistakes the MR algorithms makes in T steps?
- Answer: Normally, MR will eliminate all experts at some point. Even if we assume it stops when $\|K^t\| = 1$, MR makes $O(T)$ mistakes if no expert is perfect. Why?
- Finite probability that the best expert will be eliminated prematurely. Then regret will accumulate linearly. (in fact, this probability goes to 1 as $k \to \infty$)
- Solution: Need to keep track of "error-prone" experts, without eliminating!

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t.

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$
- Let $\eta \in (0, 1/2]$

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$
- Let $\eta \in (0, 1/2]$
- At round t:

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$
- Let $\eta \in (0, 1/2]$
- At round t:
  - Let $W_0^{(t)} = \sum_{i:x_i(t)=0} w_i^{(t)}$

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$
- Let $\eta \in (0, 1/2]$
- At round t:
  - Let $W_0^{(t)} = \sum_{i:x_i(t)=0} w_i^{(t)}$ (total weight of experts predicting 0)

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$
- Let $\eta \in (0, 1/2]$
- At round t:
  - Let $W_0^{(t)} = \sum_{i:x_i(t)=0} w_i^{(t)}$ (total weight of experts predicting 0)
  - Let $W_1^{(t)} = \sum_{i:x_i(t)=1} w_i^{(t)}$ (total weight of experts predicting 1)

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$
- Let $\eta \in (0, 1/2]$
- At round t:
  - Let $W_0^{(t)} = \sum_{i:x_i(t)=0} w_i^{(t)}$ (total weight of experts predicting 0)
  - Let $W_1^{(t)} = \sum_{i:x_i(t)=1} w_i^{(t)}$ (total weight of experts predicting 1)
  - Choose:
    - $\hat{y}(t) = 0$, if $W_0^{(t)} \geq W_1^{(t)}$

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$
- Let $\eta \in (0, 1/2]$
- At round t:
  - Let $W_0^{(t)} = \sum_{i:x_i(t)=0} w_i^{(t)}$ (total weight of experts predicting 0)
  - Let $W_1^{(t)} = \sum_{i:x_i(t)=1} w_i^{(t)}$ (total weight of experts predicting 1)
  - Choose:
    - $\hat{y}(t) = 0$, if $W_0^{(t)} \geq W_1^{(t)}$
    - $\hat{y}(t) = 1$, if $W_0^{(t)} < W_1^{(t)}$

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$
- Let $\eta \in (0, 1/2]$
- At round t:
  - Let $W_0^{(t)} = \sum_{i:x_i(t)=0} w_i^{(t)}$ (total weight of experts predicting 0)
  - Let $W_1^{(t)} = \sum_{i:x_i(t)=1} w_i^{(t)}$ (total weight of experts predicting 1)
  - Choose:
    - $\hat{y}(t) = 0$, if $W_0^{(t)} \geq W_1^{(t)}$
    - $\hat{y}(t) = 1$, if $W_0^{(t)} < W_1^{(t)}$
  - $w_i^{(t+1)} = (1 - \eta) w_i^{(t)}$

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$
- Let $\eta \in (0, 1/2]$
- At round t:
  - Let $W_0^{(t)} = \sum_{i:x_i(t)=0} w_i^{(t)}$ (total weight of experts predicting 0)
  - Let $W_1^{(t)} = \sum_{i:x_i(t)=1} w_i^{(t)}$ (total weight of experts predicting 1)
  - Choose:
    - $\hat{y}(t) = 0$, if $W_0^{(t)} \geq W_1^{(t)}$
    - $\hat{y}(t) = 1$, if $W_0^{(t)} < W_1^{(t)}$
  - $w_i^{(t+1)} = (1-\eta) w_i^{(t)}, \forall i : x_i(t) \neq y(t)$

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$
- Let $\eta \in (0, 1/2]$
- At round t:
  - Let $W_0^{(t)} = \sum_{i:x_i(t)=0} w_i^{(t)}$ (total weight of experts predicting 0)
  - Let $W_1^{(t)} = \sum_{i:x_i(t)=1} w_i^{(t)}$ (total weight of experts predicting 1)
  - Choose:
    - $\hat{y}(t) = 0$, if $W_0^{(t)} \geq W_1^{(t)}$
    - $\hat{y}(t) = 1$, if $W_0^{(t)} < W_1^{(t)}$
  - $w_i^{(t+1)} = (1-\eta) w_i^{(t)}, \forall i : x_i(t) \neq y(t)$ (discount weight if wrong)

Algorithm 2: Weighted Majority Rule (WMR)

Assume: every expert makes mistakes (but one makes the fewest)

Weighted Majority Rule (MR) Algorithm

- Let $w_i^{(t)}$ denote the weight of expert i at round t. Set $w_i^{(0)} = 1, \forall i$
- Let $W^{(t)} = \sum_{i=1}^{k} w_i^{(t)}$
- Let $\eta \in (0, 1/2]$
- At round t:
    - Let $W_0^{(t)} = \sum_{i:x_i(t)=0} w_i^{(t)}$ (total weight of experts predicting 0)
    - Let $W_1^{(t)} = \sum_{i:x_i(t)=1} w_i^{(t)}$ (total weight of experts predicting 1)
    - Choose:
        - $\hat{y}(t) = 0$, if $W_0^{(t)} \geq W_1^{(t)}$
        - $\hat{y}(t) = 1$, if $W_0^{(t)} < W_1^{(t)}$
    - $w_i^{(t+1)} = (1-\eta) w_i^{(t)}, \forall i : x_i(t) \neq y(t)$ (discount weight if wrong)
    - $w_i^{(t+1)} = w_i^{(t)}, \forall i : x_i(t) = y(t)$ (same weight if correct)

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2 + 2\eta) \cdot M^* + 2\frac{\ln k}{\eta}}$

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2+2\eta) \cdot M^* + 2\frac{\ln k}{\eta}}$

Proof: If a mistake is made at round t, then:

1. $W^{(t+1)} = \sum_i w_i^{(t+1)}$

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG}} \leq (\mathbf{2}+\mathbf{2}\eta)\cdot \mathbf{M}^* + \mathbf{2}\frac{\ln k}{\eta}$

Proof: If a mistake is made at round t, then:

1. $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t)\neq y(t)} w_i^{(t+1)}$

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2+2\eta) \cdot M^* + 2\frac{\ln k}{\eta}}$

Proof: If a mistake is made at round t, then:

1. $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t)\neq y(t)} w_i^{(t+1)} + \sum_{i:x_i(t)=y(t)} w_i^{(t+1)}$

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2 + 2\eta) \cdot M^* + 2\frac{\ln k}{\eta}}$

Proof: If a mistake is made at round t, then:

1. $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t)\neq y(t)} w_i^{(t+1)} + \sum_{i:x_i(t)=y(t)} w_i^{(t+1)}$
2. $\overset{(\text{def. of } w_i^{(t+1)})}{=} \sum_{i:x_i(t)\neq y(t)} (1-\eta) w_i^{(t)}$

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2 + 2\eta) \cdot M^* + 2\frac{\ln k}{\eta}}$

Proof: If a mistake is made at round t, then:

1. $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t)\neq y(t)} w_i^{(t+1)} + \sum_{i:x_i(t)=y(t)} w_i^{(t+1)}$
2. $\overset{\text{(def. of } w_i^{(t+1)})}{=} \sum_{i:x_i(t)\neq y(t)} (1-\eta) w_i^{(t)} + \sum_{i:x_i(t)=y(t)} w_i^{(t)}$

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2 + 2\eta) \cdot M^* + 2\frac{\ln k}{\eta}}$

Proof: If a mistake is made at round t, then:

1. $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t) \neq y(t)} w_i^{(t+1)} + \sum_{i:x_i(t)=y(t)} w_i^{(t+1)}$
2. $\overset{\text{(def. of } w_i^{(t+1)})}{=} \sum_{i:x_i(t) \neq y(t)} (1-\eta) w_i^{(t)} + \sum_{i:x_i(t)=y(t)} w_i^{(t)}$
3. $\leq (1-\eta) W^{(t)}/2 + W^{(t)}/2$

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2+2\eta)\cdot M^* + 2\frac{\ln k}{\eta}}$

Proof: If a mistake is made at round t, then:

1. $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t)\neq y(t)} w_i^{(t+1)} + \sum_{i:x_i(t)=y(t)} w_i^{(t+1)}$
2. $\overset{(\text{def. of } w_i^{(t+1)})}{=} \sum_{i:x_i(t)\neq y(t)}(1-\eta)w_i^{(t)} + \sum_{i:x_i(t)=y(t)} w_i^{(t)}$
3. $\leq (1-\eta)W^{(t)}/2 + W^{(t)}/2$ (mistake $\Rightarrow$ majority weight $\geq W^{(t)}/2$)

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2 + 2\eta) \cdot M^* + 2\frac{\ln k}{\eta}}$

Proof: If a mistake is made at round t, then:

1. $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t)\neq y(t)} w_i^{(t+1)} + \sum_{i:x_i(t)=y(t)} w_i^{(t+1)}$
2. $\overset{\text{(def. of } w_i^{(t+1)})}{=} \sum_{i:x_i(t)\neq y(t)} (1-\eta) w_i^{(t)} + \sum_{i:x_i(t)=y(t)} w_i^{(t)}$
3. $\leq (1-\eta) W^{(t)}/2 + W^{(t)}/2$ (mistake $\Rightarrow$ majority weight $\geq W^{(t)}/2$)
4. $\Rightarrow W^{(t+1)} \leq (1-\eta/2) W^{(t)}$

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2 + 2\eta) \cdot M^* + 2\frac{lnk}{\eta}}$

Proof: If a mistake is made at round t, then:

1. $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t)\neq y(t)} w_i^{(t+1)} + \sum_{i:x_i(t)=y(t)} w_i^{(t+1)}$
2. $\overset{(\text{def. of } w_i^{(t+1)})}{=} \sum_{i:x_i(t)\neq y(t)} (1-\eta) w_i^{(t)} + \sum_{i:x_i(t)=y(t)} w_i^{(t)}$
3. $\leq (1-\eta) W^{(t)}/2 + W^{(t)}/2$ (mistake $\Rightarrow$ majority weight $\geq W^{(t)}/2$)
4. $\Rightarrow W^{(t+1)} \leq (1-\eta/2) W^{(t)}$

$(5) \Rightarrow W^{(T+1)} \leq (1-\eta/2)^{M_{ALG}} \cdot W^{(0)}$

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2+2\eta) \cdot M^* + 2\frac{\ln k}{\eta}}$

Proof: If a mistake is made at round t, then:

1 $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t)\neq y(t)} w_i^{(t+1)} + \sum_{i:x_i(t)=y(t)} w_i^{(t+1)}$

2 $\overset{(\text{def. of } w_i^{(t+1)})}{=} \sum_{i:x_i(t)\neq y(t)} (1-\eta) w_i^{(t)} + \sum_{i:x_i(t)=y(t)} w_i^{(t)}$

3 $\leq (1-\eta) W^{(t)}/2 + W^{(t)}/2$ (mistake $\Rightarrow$ majority weight $\geq W^{(t)}/2$)

4 $\Rightarrow W^{(t+1)} \leq (1-\eta/2) W^{(t)}$

(5) $\Rightarrow W^{(T+1)} \leq (1-\eta/2)^{M_{ALG}} \cdot W^{(0)} = k(1-\eta/2)^{M_{ALG}}$

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2+2\eta) \cdot M^* + 2\frac{\ln k}{\eta}}$

Proof: If a mistake is made at round t, then:

1 $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t)\neq y(t)} w_i^{(t+1)} + \sum_{i:x_i(t)=y(t)} w_i^{(t+1)}$

2 $\overset{\text{(def. of } w_i^{(t+1)})}{=} \sum_{i:x_i(t)\neq y(t)} (1-\eta) w_i^{(t)} + \sum_{i:x_i(t)=y(t)} w_i^{(t)}$

3 $\leq (1-\eta)W^{(t)}/2 + W^{(t)}/2$ (mistake $\Rightarrow$ majority weight $\geq W^{(t)}/2$)

4 $\Rightarrow W^{(t+1)} \leq (1-\eta/2)W^{(t)}$

(5) $\Rightarrow W^{(T+1)} \leq (1-\eta/2)^{M_{ALG}} \cdot W^{(0)} = k(1-\eta/2)^{M_{ALG}}$ (Key Ineq.)

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG} \leq (2+2\eta) \cdot M^* + 2\frac{\ln k}{\eta}}$

Proof: If a mistake is made at round t, then:

1. $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t)\neq y(t)} w_i^{(t+1)} + \sum_{i:x_i(t)=y(t)} w_i^{(t+1)}$
2. $\overset{(\text{def. of } w_i^{(t+1)})}{=} \sum_{i:x_i(t)\neq y(t)} (1-\eta) w_i^{(t)} + \sum_{i:x_i(t)=y(t)} w_i^{(t)}$
3. $\leq (1-\eta)W^{(t)}/2 + W^{(t)}/2$ (mistake $\Rightarrow$ majority weight $\geq W^{(t)}/2$)
4. $\Rightarrow W^{(t+1)} \leq (1-\eta/2)W^{(t)}$

(5) $\Rightarrow W^{(T+1)} \leq (1-\eta/2)^{M_{ALG}} \cdot W^{(0)} = k(1-\eta/2)^{M_{ALG}}$ (Key Ineq.)

(6) Also: $W^{(T+1)} \geq w_{i^*}^{(T+1)}$

Performance of Algorithm 2 (WMR)

Quiz: How many more mistakes than best expert does WMR make?

Theorem

If M^* is the number of mistakes of the best expert, and M_{ALG} the number of mistakes of Weighted Majority Rule, then $\mathbf{M_{ALG}} \leq (\mathbf{2} + \mathbf{2}\eta) \cdot \mathbf{M}^* + \mathbf{2}\frac{\ln k}{\eta}$

Proof: If a mistake is made at round t, then:

1. $W^{(t+1)} = \sum_i w_i^{(t+1)} = \sum_{i:x_i(t)\neq y(t)} w_i^{(t+1)} + \sum_{i:x_i(t)=y(t)} w_i^{(t+1)}$
2. $\overset{(\text{def. of } w_i^{(t+1)})}{=} \sum_{i:x_i(t)\neq y(t)} (1-\eta) w_i^{(t)} + \sum_{i:x_i(t)=y(t)} w_i^{(t)}$
3. $\leq (1-\eta)W^{(t)}/2 + W^{(t)}/2$ (mistake $\Rightarrow$ majority weight $\geq W^{(t)}/2$)
4. $\Rightarrow W^{(t+1)} \leq (1-\eta/2)W^{(t)}$

(5) $\Rightarrow W^{(T+1)} \leq (1-\eta/2)^{M_{ALG}} \cdot W^{(0)} = k(1-\eta/2)^{M_{ALG}}$ (Key Ineq.)

(6) Also: $W^{(T+1)} \geq w_{i^*}^{(T+1)} = (1-\eta)^{M^*}$ (Key Ineq.)

Performance of Algorithm 2 (WMR) - cnt'd

Proof(cnt'd):

7 by (5),(6) $\Rightarrow (1-\eta)^{M^*} \leq k(1-\eta/2)^{M_{ALG}}$

Performance of Algorithm 2 (WMR) - cnt'd

Proof(cnt'd):

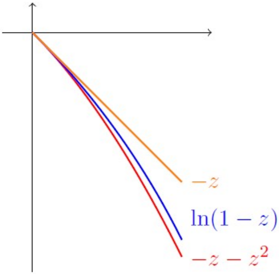
7. by (5),(6) $\Rightarrow (1-\eta)^{M^*} \leq k(1-\eta/2)^{M_{ALG}}$
8. $\overset{\text{taking logs}}{\Rightarrow} M^* \cdot ln(1-\eta) \leq lnk + M_{ALG} \cdot ln(1-\eta/2)$

Proof(cnt'd):

7 by (5),(6) $\Rightarrow (1-\eta)^{M^*} \leq k(1-\eta/2)^{M_{ALG}}$

8 $\overset{\text{taking logs}}{\Rightarrow} M^* \cdot ln(1-\eta) \leq lnk + M_{ALG} \cdot ln(1-\eta/2)$

9 Will use known formula: $-z - z^2 \leq ln(1 - z) \leq -z$.

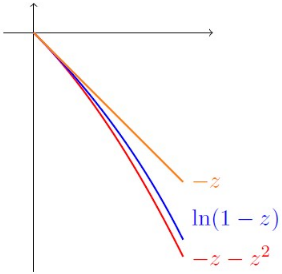


Proof(cnt'd):

7 by (5),(6) $\Rightarrow (1-\eta)^{M^*} \leq k(1-\eta/2)^{M_{ALG}}$

8 $\stackrel{\text{taking logs}}{\Rightarrow} M^* \cdot ln(1-\eta) \leq lnk + M_{ALG} \cdot ln(1-\eta/2)$

9 Will use known formula: $-z - z^2 \leq ln(1-z) \leq -z$.



10 $\stackrel{(9)}{\Rightarrow} M^* \cdot (-\eta - \eta^2)$

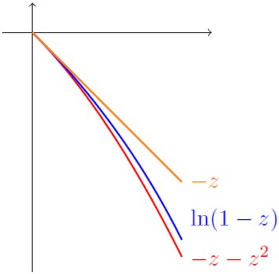
Performance of Algorithm 2 (WMR) - cnt'd

Proof(cnt'd):

7 by (5),(6) $\Rightarrow (1-\eta)^{M^*} \leq k(1-\eta/2)^{M_{ALG}}$

8 $\overset{\text{taking logs}}{\Rightarrow} M^* \cdot ln(1-\eta) \leq lnk + M_{ALG} \cdot ln(1-\eta/2)$

9 Will use known formula: $-z - z^2 \leq ln(1-z) \leq -z$.



10 $\overset{(9)}{\Rightarrow} M^* \cdot (-\eta - \eta^2) \leq lnk - \eta/2 \cdot M_{ALG}$

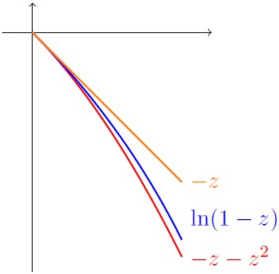
Performance of Algorithm 2 (WMR) - cnt'd

Proof(cnt'd):

7 by (5),(6) $\Rightarrow (1-\eta)^{M^*} \leq k(1-\eta/2)^{M_{ALG}}$

8 $\stackrel{\text{taking logs}}{\Rightarrow} M^* \cdot ln(1-\eta) \leq lnk + M_{ALG} \cdot ln(1-\eta/2)$

9 Will use known formula: $-z - z^2 \leq ln(1-z) \leq -z$.



10 $\stackrel{(9)}{\Rightarrow} M^* \cdot (-\eta - \eta^2) \leq lnk - \eta/2 \cdot M_{ALG}$

11 $M_{ALG} \leq (2+2\eta) \cdot M^* + 2\frac{lnk}{\eta}$

Observations about WMR performance & Proof

Methodology Observations

- Before: how fast $K^{(t)}$ reduces? Now: how fast $W^{(t)}$ reduces?

Observations about WMR performance & Proof

Methodology Observations

- Before: how fast $K^{(t)}$ reduces? Now: how fast $W^{(t)}$ reduces?
- MR (Algo 1) was setting $\eta = 1 \Rightarrow w_i(t)$ goes to 0 after 1 mistake.

Observations about WMR performance & Proof

Methodology Observations

- Before: how fast $K^{(t)}$ reduces? Now: how fast $W^{(t)}$ reduces?
- MR (Algo 1) was setting $\eta = 1 \Rightarrow w_i(t)$ goes to 0 after 1 mistake.
- Key Inequality (5): captures how fast the weight $W^{(t)}$ of our WMR can drop, as mistakes are made.

Observations about WMR performance & Proof

Methodology Observations

- Before: how fast $K^{(t)}$ reduces? Now: how fast $W^{(t)}$ reduces?
- MR (Algo 1) was setting $\eta = 1 \Rightarrow w_i(t)$ goes to 0 after 1 mistake.
- Key Inequality (5): captures how fast the weight $W^{(t)}$ of our WMR can drop, as mistakes are made.
- Key Inequality (6): lower bounds this as a function of best expert's mistakes

Observations about WMR performance & Proof

Methodology Observations

- Before: how fast $K^{(t)}$ reduces? Now: how fast $W^{(t)}$ reduces?
- MR (Algo 1) was setting $\eta = 1 \Rightarrow w_i(t)$ goes to 0 after 1 mistake.
- Key Inequality (5): captures how fast the weight $W^{(t)}$ of our WMR can drop, as mistakes are made.
- Key Inequality (6): lower bounds this as a function of best expert's mistakes $\Rightarrow$ if $W^{(t)}$ were lower the best expert would be chosen

Observations about WMR performance & Proof

Methodology Observations

- Before: how fast $K^{(t)}$ reduces? Now: how fast $W^{(t)}$ reduces?
- MR (Algo 1) was setting $\eta = 1 \Rightarrow w_i(t)$ goes to 0 after 1 mistake.
- Key Inequality (5): captures how fast the weight $W^{(t)}$ of our WMR can drop, as mistakes are made.
- Key Inequality (6): lower bounds this as a function of best expert's mistakes $\Rightarrow$ if $W^{(t)}$ were lower the best expert would be chosen

Result Observations: $M_{ALG} \leq (2+2\eta) \cdot M^* + 2\frac{lnk}{\eta}$

- small $\eta(\rightarrow 0) \Rightarrow M_{ALG} \leq 2M^* + 2\frac{lnk}{\eta}$:

Observations about WMR performance & Proof

Methodology Observations

- Before: how fast $K^{(t)}$ reduces? Now: how fast $W^{(t)}$ reduces?
- MR (Algo 1) was setting $\eta = 1 \Rightarrow w_i(t)$ goes to 0 after 1 mistake.
- Key Inequality (5): captures how fast the weight $W^{(t)}$ of our WMR can drop, as mistakes are made.
- Key Inequality (6): lower bounds this as a function of best expert's mistakes $\Rightarrow$ if $W^{(t)}$ were lower the best expert would be chosen

Result Observations: $M_{ALG} \leq (2 + 2\eta) \cdot M^* + 2\frac{lnk}{\eta}$

- small $\eta(\rightarrow 0) \Rightarrow M_{ALG} \leq 2M^* + 2\frac{lnk}{\eta}$: WMR makes *at least* twice as many errors

Observations about WMR performance & Proof

Methodology Observations

- Before: how fast $K^{(t)}$ reduces? Now: how fast $W^{(t)}$ reduces?
- MR (Algo 1) was setting $\eta = 1 \Rightarrow w_i(t)$ goes to 0 after 1 mistake.
- Key Inequality (5): captures how fast the weight $W^{(t)}$ of our WMR can drop, as mistakes are made.
- Key Inequality (6): lower bounds this as a function of best expert's mistakes $\Rightarrow$ if $W^{(t)}$ were lower the best expert would be chosen

Result Observations: $M_{ALG} \leq (2 + 2\eta) \cdot M^* + 2\frac{lnk}{\eta}$

- small $\eta(\rightarrow 0) \Rightarrow M_{ALG} \leq 2M^* + 2\frac{lnk}{\eta}$: WMR makes *at least* twice as many errors
- large $\eta(\rightarrow 1/2) \Rightarrow M_{ALG} \leq 3M^* + lnk$:

Observations about WMR performance & Proof

Methodology Observations

- Before: how fast $K^{(t)}$ reduces? Now: how fast $W^{(t)}$ reduces?
- MR (Algo 1) was setting $\eta = 1 \Rightarrow w_i(t)$ goes to 0 after 1 mistake.
- Key Inequality (5): captures how fast the weight $W^{(t)}$ of our WMR can drop, as mistakes are made.
- Key Inequality (6): lower bounds this as a function of best expert's mistakes $\Rightarrow$ if $W^{(t)}$ were lower the best expert would be chosen

Result Observations: $M_{ALG} \leq (2 + 2\eta) \cdot M^* + 2\frac{lnk}{\eta}$

- small $\eta(\rightarrow 0) \Rightarrow M_{ALG} \leq 2M^* + 2\frac{lnk}{\eta}$: WMR makes *at least* twice as many errors
- large $\eta(\rightarrow 1/2) \Rightarrow M_{ALG} \leq 3M^* + lnk$: could be better if horizon T is small and/or number of experts k is large.

Observations about WMR performance & Proof

Methodology Observations

- Before: how fast $K^{(t)}$ reduces? Now: how fast $W^{(t)}$ reduces?
- MR (Algo 1) was setting $\eta = 1 \Rightarrow w_i(t)$ goes to 0 after 1 mistake.
- Key Inequality (5): captures how fast the weight $W^{(t)}$ of our WMR can drop, as mistakes are made.
- Key Inequality (6): lower bounds this as a function of best expert's mistakes $\Rightarrow$ if $W^{(t)}$ were lower the best expert would be chosen

Result Observations: $M_{ALG} \leq (2+2\eta) \cdot M^* + 2\frac{lnk}{\eta}$

- small $\eta(\rightarrow 0) \Rightarrow M_{ALG} \leq 2M^* + 2\frac{lnk}{\eta}$: WMR makes *at least* twice as many errors
- large $\eta(\rightarrow 1/2) \Rightarrow M_{ALG} \leq 3M^* + lnk$: could be better if horizon T is small and/or number of experts k is large.
- optimal η? $\Rightarrow \eta_{opt} = \sqrt{\frac{lnK}{2M^*}}$:

Observations about WMR performance & Proof

Methodology Observations

- Before: how fast $K^{(t)}$ reduces? Now: how fast $W^{(t)}$ reduces?
- MR (Algo 1) was setting $\eta = 1 \Rightarrow w_i(t)$ goes to 0 after 1 mistake.
- Key Inequality (5): captures how fast the weight $W^{(t)}$ of our WMR can drop, as mistakes are made.
- Key Inequality (6): lower bounds this as a function of best expert's mistakes $\Rightarrow$ if $W^{(t)}$ were lower the best expert would be chosen

Result Observations: $M_{ALG} \leq (2+2\eta) \cdot M^* + 2\frac{lnk}{\eta}$

- small $\eta(\rightarrow 0) \Rightarrow M_{ALG} \leq 2M^* + 2\frac{lnk}{\eta}$: WMR makes *at least* twice as many errors
- large $\eta(\rightarrow 1/2) \Rightarrow M_{ALG} \leq 3M^* + lnk$: could be better if horizon T is small and/or number of experts k is large.
- optimal η? $\Rightarrow \eta_{opt} = \sqrt{\frac{lnK}{2M^*}}$: See any problem?

Algorithm 3: Multiplicative Weights or "Hedge"

So far: Assumed $x_i(t), y(t) \in \{0, 1\}$ (binary classification)

Algorithm 3: Multiplicative Weights or "Hedge"

<u>So far:</u> Assumed $x_i(t), y(t) \in \{0, 1\}$ (binary classification)
<u>Generalise:</u> At round t: expert i incurs a loss $l_i^t \in [0, 1]$ (will further generalize later)

Algorithm 3: Multiplicative Weights or “Hedge”

So far: Assumed $x_i(t), y(t) \in \{0, 1\}$ (binary classification)
Generalise: At round t: expert i incurs a loss $l_i^t \in [0, 1]$ (will further generalize later)
Applications: (a) any prediction (regression) task, (b) stock price loss, (c) channel selection performance $l_i^t = -log\,(1 + SNR_i^t)$, many others...

Algorithm 3: Multiplicative Weights or "Hedge"

So far: Assumed $x_i(t), y(t) \in \{0, 1\}$ (binary classification)
Generalise: At round t: expert i incurs a loss $l_i^t \in [0, 1]$ (will further generalize later)
Applications: (a) any prediction (regression) task, (b) stock price loss, (c) channel selection performance $l_i^t = -log\,(1 + SNR_i^t)$, many others...

Multiplicative Weights Algorithm

- Let $w_i^{(0)} = 1$ (as before)

Algorithm 3: Multiplicative Weights or "Hedge"

So far: Assumed $x_i(t), y(t) \in \{0, 1\}$ (binary classification)
Generalise: At round t: expert i incurs a loss $l_i^t \in [0, 1]$ (will further generalize later)
Applications: (a) any prediction (regression) task, (b) stock price loss, (c) channel selection performance $l_i^t = -log\,(1 + SNR_i^t)$, many others...

Multiplicative Weights Algorithm

- Let $w_i^{(0)} = 1$ (as before)
- At round t:

Algorithm 3: Multiplicative Weights or "Hedge"

So far: Assumed $x_i(t), y(t) \in \{0, 1\}$ (binary classification)
Generalise: At round t: expert i incurs a loss $l_i^t \in [0, 1]$ (will further generalize later)
Applications: (a) any prediction (regression) task, (b) stock price loss, (c) channel selection performance $l_i^t = -log\,(1 + SNR_i^t)$, many others...

Multiplicative Weights Algorithm

- Let $w_i^{(0)} = 1$ (as before)
- At round t:
  - Pick expert i with probability $p_i = \frac{w_i^t}{\sum_i w_i^t} = \frac{w_i^t}{W^t}$

Algorithm 3: Multiplicative Weights or "Hedge"

So far: Assumed $x_i(t), y(t) \in \{0, 1\}$ (binary classification)
Generalise: At round t: expert i incurs a loss $l_i^t \in [0, 1]$ (will further generalize later)
Applications: (a) any prediction (regression) task, (b) stock price loss, (c) channel selection performance $l_i^t = -log\,(1 + SNR_i^t)$, many others...

Multiplicative Weights Algorithm

- Let $w_i^{(0)} = 1$ (as before)
- At round t:
  - Pick expert i with probability $p_i = \frac{w_i^t}{\sum_i w_i^t} = \frac{w_i^t}{W^t}$
  - Algorithm incurs loss l_i^t

Algorithm 3: Multiplicative Weights or "Hedge"

So far: Assumed $x_i(t), y(t) \in \{0,1\}$ (binary classification)
Generalise: At round t: expert i incurs a loss $l_i^t \in [0,1]$ (will further generalize later)
Applications: (a) any prediction (regression) task, (b) stock price loss, (c) channel selection performance $l_i^t = -log\,(1 + SNR_i^t)$, many others...

Multiplicative Weights Algorithm

- Let $w_i^{(0)} = 1$ (as before)
- At round t:
  - Pick expert i with probability $p_i = \frac{w_i^t}{\sum_i w_i^t} = \frac{w_i^t}{W^t}$
  - Algorithm incurs loss l_i^t
  - $w_i^{(t+1)} = (1-\eta)^{l_i^t} \cdot w_i^{(t)}, \forall i$

Multiplicative Weights: Some Observations

- Key difference is that discount factor $(1-\eta)^{l_i^t}$ is applied to every expert. Bigger reduction of weight for experts seeing large loss at that round.

Multiplicative Weights: Some Observations

- Key difference is that discount factor $(1-\eta)^{l_i^t}$ is applied to every expert. Bigger reduction of weight for experts seeing large loss at that round.
- **Note:** We are talking about *losses* and not *rewards* (as in bandits). Hence, we want to minimize losses (like we minimized mistakes).

Multiplicative Weights: Some Observations

- Key difference is that discount factor $(1-\eta)^{l_i^t}$ is applied to every expert. Bigger reduction of weight for experts seeing large loss at that round.
- **Note:** We are talking about *losses* and not *rewards* (as in bandits). Hence, we want to minimize losses (like we minimized mistakes).
- Algorithm does *not* make a (weighted) *majority* decision anymore $\rightarrow$ it is now randomized and picks one expert at a time (common theme later).

Multiplicative Weights: Some Observations

- Key difference is that discount factor $(1-\eta)^{l_i^t}$ is applied to every expert. Bigger reduction of weight for experts seeing large loss at that round.
- **Note:** We are talking about *losses* and not *rewards* (as in bandits). Hence, we want to minimize losses (like we minimized mistakes).
- Algorithm does *not* make a (weighted) *majority* decision anymore $\rightarrow$ it is now randomized and picks one expert at a time (common theme later).
- Proof follows very similar methodology like WMR proof.

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1 $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}$,

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1 $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}, \forall i (\text{including } i^*)$

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1. $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}, \forall i (\text{including } i^*)$

Upper bound on W^{T+1} (similar to (5) in slide 12):

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1 $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}, \forall i$(including i^*)

Upper bound on W^{T+1} (similar to (5) in slide 12):

2 $W^{T+1} = \sum_i w_i^T \cdot (1-\eta)^{l_i^t}$

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1. $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}, \forall i(\text{including } i^*)$

Upper bound on W^{T+1} (similar to (5) in slide 12):

2. $W^{T+1} = \sum_i w_i^T \cdot (1-\eta)^{l_i^t}$
3. but $(1-\eta)^x \leq (1-\eta x)$ for $x \in [0,1]$

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1. $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}, \forall i(\text{including } i^*)$

Upper bound on W^{T+1} (similar to (5) in slide 12):

2. $W^{T+1} = \sum_i w_i^T \cdot (1-\eta)^{l_i^t}$
3. but $(1-\eta)^x \leq (1-\eta x)$ for $x \in [0,1]$
4. (2),(3) $\Rightarrow W^{T+1} \leq \sum_i w_i^T \cdot (1-\eta l_i^t)$

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1 $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}, \forall i$(including i^*)

Upper bound on W^{T+1} (similar to (5) in slide 12):

2 $W^{T+1} = \sum_i w_i^T \cdot (1-\eta)^{l_i^t}$

3 but $(1-\eta)^x \leq (1-\eta x)$ for $x \in [0,1]$

4 (2),(3) $\Rightarrow W^{T+1} \leq \sum_i w_i^T \cdot (1-\eta l_i^t)$

5 $= W^T - \eta \cdot \sum_i w_i^T \cdot l_i^t$

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1. $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}, \forall i$(including i^*)

Upper bound on W^{T+1} (similar to (5) in slide 12):

2. $W^{T+1} = \sum_i w_i^T \cdot (1-\eta)^{l_i^t}$
3. but $(1-\eta)^x \leq (1-\eta x)$ for $x \in [0,1]$
4. (2),(3) $\Rightarrow W^{T+1} \leq \sum_i w_i^T \cdot (1-\eta l_i^t)$
5. $= W^T - \eta \cdot \sum_i w_i^T \cdot l_i^t \overset{\text{def. of } p_i}{=} W^T - \eta W^T \cdot \sum_i p_i^T \cdot l_i^t$

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1. $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}, \forall i(\text{including } i^*)$

Upper bound on W^{T+1} (similar to (5) in slide 12):

2. $W^{T+1} = \sum_i w_i^T \cdot (1-\eta)^{l_i^t}$
3. but $(1-\eta)^x \leq (1-\eta x)$ for $x \in [0,1]$
4. (2),(3) $\Rightarrow W^{T+1} \leq \sum_i w_i^T \cdot (1-\eta l_i^t)$
5. $= W^T - \eta \cdot \sum_i w_i^T \cdot l_i^t \overset{\text{def. of } p_i}{=} W^T - \eta W^T \cdot \sum_i p_i^T \cdot l_i^t$
6. $W^T(1-\eta \cdot l_{ALG}^t)$

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1 $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}, \forall i (\text{including } i^*)$

Upper bound on W^{T+1} (similar to (5) in slide 12):

2 $W^{T+1} = \sum_i w_i^T \cdot (1-\eta)^{l_i^t}$

3 but $(1-\eta)^x \leq (1-\eta x)$ for $x \in [0,1]$

4 (2),(3) $\Rightarrow W^{T+1} \leq \sum_i w_i^T \cdot (1-\eta l_i^t)$

5 $= W^T - \eta \cdot \sum_i w_i^T \cdot l_i^t \overset{\text{def. of } p_i}{=} W^T - \eta W^T \cdot \sum_i p_i^T \cdot l_i^t$

6 $W^T(1-\eta \cdot l_{ALG}^t)$ (expected loss $l_{ALG}^t = \sum_i p_i^T \cdot l_i^t$)

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1. $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}, \forall i(\text{including } i^*)$

Upper bound on W^{T+1} (similar to (5) in slide 12):

2. $W^{T+1} = \sum_i w_i^T \cdot (1-\eta)^{l_i^t}$
3. but $(1-\eta)^x \leq (1-\eta x)$ for $x \in [0,1]$
4. (2),(3) $\Rightarrow W^{T+1} \leq \sum_i w_i^T \cdot (1-\eta l_i^t)$
5. $= W^T - \eta \cdot \sum_i w_i^T \cdot l_i^t \overset{\text{def. of } p_i}{=} W^T - \eta W^T \cdot \sum_i p_i^T \cdot l_i^t$
6. $W^T(1-\eta \cdot l_{ALG}^t)$ (expected loss $l_{ALG}^t = \sum_i p_i^T \cdot l_i^t$)
7. $\Rightarrow W^{T+1} \leq k \cdot \Pi_{t=1}^{T}(1-\eta \cdot l_{ALG}^t)$

Performance of Multiplicative Weights

Proof:

Lower bound on W^{T+1} (similar to (6) in slide 12):

1 $W^{T+1} \geq w_i^{T+1} = \Pi_{t=1}^{T}(1-\eta)^{l_i^t}, \forall i(\text{including } i^*)$

Upper bound on W^{T+1} (similar to (5) in slide 12):

2 $W^{T+1} = \sum_i w_i^T \cdot (1-\eta)^{l_i^t}$

3 but $(1-\eta)^x \leq (1-\eta x)$ for $x \in [0,1]$

4 (2),(3) $\Rightarrow W^{T+1} \leq \sum_i w_i^T \cdot (1-\eta l_i^t)$

5 $= W^T - \eta \cdot \sum_i w_i^T \cdot l_i^t \overset{\text{def. of } p_i}{=} W^T - \eta W^T \cdot \sum_i p_i^T \cdot l_i^t$

6 $W^T(1-\eta \cdot l_{ALG}^t)$ (expected loss $l_{ALG}^t = \sum_i p_i^T \cdot l_i^t$)

7 $\Rightarrow W^{T+1} \leq k \cdot \Pi_{t=1}^{T}(1-\eta \cdot l_{ALG}^t)$ (since $W^0 = k$)

Performance of Multiplicative Weights

Proof (cnt'd):

8. Combine lower and upper bound:
   $\Pi_{t=1}^{T}(1-\eta)^{l_i^t} \leq k \cdot \Pi_{t=1}^{T}(1-\eta \cdot l_{ALG}^t)$

Performance of Multiplicative Weights

Proof (cnt'd):

8 Combine lower and upper bound:
$\Pi_{t=1}^{T}(1-\eta)^{l_i^t} \leq k \cdot \Pi_{t=1}^{T}(1-\eta \cdot l_{ALG}^t)$

9 $\Rightarrow ln(1-\eta)\sum_t l_i^t \leq lnk + \sum_t ln(1-\eta \cdot l_{ALG}^t)$ (logarithms both sides)

Performance of Multiplicative Weights

Proof (cnt'd):

8. Combine lower and upper bound:
   $\Pi_{t=1}^{T}(1-\eta)^{l_i^t} \leq k \cdot \Pi_{t=1}^{T}(1-\eta \cdot l_{ALG}^t)$
9. $\Rightarrow ln(1-\eta)\sum_t l_i^t \leq lnk + \sum_t ln(1-\eta \cdot l_{ALG}^t)$ (logarithms both sides)
10. $\Rightarrow ln(1-\eta)L_{OPT}^T \leq lnk + \sum_t ln(1-\eta \cdot l_{ALG}^t)$

Performance of Multiplicative Weights

Proof (cnt'd):

8 Combine lower and upper bound:
$\Pi_{t=1}^{T}(1-\eta)^{l_i^t} \leq k \cdot \Pi_{t=1}^{T}(1-\eta \cdot l_{ALG}^t)$

9 $\Rightarrow ln(1-\eta)\sum_t l_i^t \leq lnk + \sum_t ln(1-\eta \cdot l_{ALG}^t)$ (logarithms both sides)

10 $\Rightarrow ln(1-\eta)L_{OPT}^T \leq lnk + \sum_t ln(1-\eta \cdot l_{ALG}^t)$
(use again) $-z - z^2 \leq ln(1-z) \leq -z$

Performance of Multiplicative Weights

Proof (cnt'd):

8 Combine lower and upper bound:
$\Pi_{t=1}^{T}(1-\eta)^{l_i^t} \leq k \cdot \Pi_{t=1}^{T}(1-\eta \cdot l_{ALG}^t)$

9 $\Rightarrow ln(1-\eta)\sum_t l_i^t \leq lnk + \sum_t ln(1-\eta \cdot l_{ALG}^t)$ (logarithms both sides)

10 $\Rightarrow ln(1-\eta)L_{OPT}^T \leq lnk + \sum_t ln(1-\eta \cdot l_{ALG}^t)$
(use again) $-z-z^2 \leq ln(1-z) \leq -z$

11 $\Rightarrow (-\eta-\eta^2)L_{OPT}^T \leq lnk - \sum_t \eta \cdot l_{ALG}^t)$

Performance of Multiplicative Weights

Proof (cnt'd):

8 Combine lower and upper bound:
$\Pi_{t=1}^{T}(1-\eta)^{l_i^t} \leq k \cdot \Pi_{t=1}^{T}(1-\eta \cdot l_{ALG}^t)$

9 $\Rightarrow ln(1-\eta)\sum_t l_i^t \leq lnk + \sum_t ln(1-\eta \cdot l_{ALG}^t)$ (logarithms both sides)

10 $\Rightarrow ln(1-\eta)L_{OPT}^T \leq lnk + \sum_t ln(1-\eta \cdot l_{ALG}^t)$
(use again) $-z-z^2 \leq ln(1-z) \leq -z$

11 $\Rightarrow (-\eta-\eta^2)L_{OPT}^T \leq lnk - \sum_t \eta \cdot l_{ALG}^t)$

12 (rearranging) $L_{ALG}^T \leq \frac{lnk}{\eta} + (1+\eta)L_{OPT}^T$

Performance of Multiplicative Weights

Theorem

- The total loss of the MW algorithm is upper bounded $\mathbf{L}^{\mathbf{T}}_{\mathbf{ALG}} \leq \frac{\mathbf{ln k}}{\eta} + (\mathbf{1} + \eta)\mathbf{L}^{\mathbf{T}}_{\mathbf{OPT}}$

Performance of Multiplicative Weights

Theorem

- The total loss of the MW algorithm is upper bounded $\mathbf{L}^{\mathbf{T}}_{\mathbf{ALG}} \leq \frac{\mathbf{ln k}}{\eta} + (\mathbf{1} + \eta)\mathbf{L}^{\mathbf{T}}_{\mathbf{OPT}}$
- Choosing $\eta = \sqrt{\frac{\ln k}{T}}$

Performance of Multiplicative Weights

Theorem

- The total loss of the MW algorithm is upper bounded $\mathbf{L}_{\mathbf{ALG}}^{\mathbf{T}} \leq \frac{\mathbf{lnk}}{\eta} + (\mathbf{1} + \eta)\mathbf{L}_{\mathbf{OPT}}^{\mathbf{T}}$
- Choosing $\eta = \sqrt{\frac{lnk}{T}}$
  $\Rightarrow \mathbf{Regret}^{\mathbf{T}} = \mathbf{L}_{\mathbf{ALG}}^{\mathbf{T}} - \mathbf{L}_{\mathbf{OPT}}^{\mathbf{T}} \leq \mathbf{2}\sqrt{\mathbf{Tlnk}} = \mathbf{O}(\sqrt{\mathbf{T}})$

Performance of Multiplicative Weights

Theorem

- The total loss of the MW algorithm is upper bounded
  $\mathbf{L}^{\mathbf{T}}_{\mathbf{ALG}} \leq \frac{\mathbf{ln k}}{\eta} + (\mathbf{1} + \eta)\mathbf{L}^{\mathbf{T}}_{\mathbf{OPT}}$
- Choosing $\eta = \sqrt{\frac{\ln k}{T}}$
  $\Rightarrow \mathbf{Regret}^{\mathbf{T}} = \mathbf{L}^{\mathbf{T}}_{\mathbf{ALG}} - \mathbf{L}^{\mathbf{T}}_{\mathbf{OPT}} \leq \mathbf{2}\sqrt{\mathbf{T ln k}} = \mathbf{O}(\sqrt{\mathbf{T}})$

- Comparing the first formula with that of Weight Majority Rule, we observe that the factor $(2 + \eta)$ has been cut to $(1 + \eta) \Rightarrow$ we can get close to optimal expert.

Performance of Multiplicative Weights

Theorem

- The total loss of the MW algorithm is upper bounded $\mathbf{L^T_{ALG}} \leq \frac{\mathbf{lnk}}{\eta} + (\mathbf{1} + \eta)\mathbf{L^T_{OPT}}$
- Choosing $\eta = \sqrt{\frac{lnk}{T}}$ $\Rightarrow \mathbf{Regret^T} = \mathbf{L^T_{ALG}} - \mathbf{L^T_{OPT}} \leq \mathbf{2}\sqrt{\mathbf{Tlnk}} = \mathbf{O}(\sqrt{\mathbf{T}})$

- Comparing the first formula with that of Weight Majority Rule, we observe that the factor $(2+\eta)$ has been cut to $(1+\eta) \Rightarrow$ we can get close to optimal expert.
- "optimal" η can be found by minimizing RHS $\Rightarrow \eta^* = \sqrt{\frac{lnk}{L^T_{OPT}}}$.

Performance of Multiplicative Weights

Theorem

- The total loss of the MW algorithm is upper bounded $\mathbf{L^T_{ALG} \leq \frac{lnk}{\eta} + (1+\eta)L^T_{OPT}}$
- Choosing $\eta = \sqrt{\frac{lnk}{T}}$ $\Rightarrow \mathbf{Regret^T = L^T_{ALG} - L^T_{OPT} \leq 2\sqrt{Tlnk} = O(\sqrt{T})}$

- Comparing the first formula with that of Weight Majority Rule, we observe that the factor $(2+\eta)$ has been cut to $(1+\eta) \Rightarrow$ we can get close to optimal expert.
- "optimal" η can be found by minimizing RHS $\Rightarrow \eta^* = \sqrt{\frac{lnk}{L^T_{OPT}}}$.
- We do not know L^T_{OPT} (as in WMR) but we can assume $L^T_{OPT} \sim T$, hence the choice of η

Performance of Multiplicative Weights

Theorem

- The total loss of the MW algorithm is upper bounded
  $\mathbf{L_{ALG}^T \leq \frac{lnk}{\eta} + (1+\eta)L_{OPT}^T}$
- Choosing $\eta = \sqrt{\frac{lnk}{T}}$
  $\Rightarrow \mathbf{Regret^T = L_{ALG}^T - L_{OPT}^T \leq 2\sqrt{Tlnk} = O(\sqrt{T})}$

- Comparing the first formula with that of Weight Majority Rule, we observe that the factor $(2+\eta)$ has been cut to $(1+\eta) \Rightarrow$ we can get close to optimal expert.
- "optimal" η can be found by minimizing RHS $\Rightarrow \eta^* = \sqrt{\frac{lnk}{L_{OPT}^T}}$.
- We do not know L_{OPT}^T (as in WMR) but we can assume $L_{OPT}^T \sim T$, hence the choice of η (unfortunately, same trick does not hold for WMR)

Performance of Multiplicative Weights

Theorem

- The total loss of the MW algorithm is upper bounded $\mathbf{L^T_{ALG} \leq \frac{\ln k}{\eta} + (1+\eta) L^T_{OPT}}$
- Choosing $\eta = \sqrt{\frac{\ln k}{T}}$
  $\Rightarrow \mathbf{Regret^T = L^T_{ALG} - L^T_{OPT} \leq 2\sqrt{T \ln k} = O(\sqrt{T})}$

- Comparing the first formula with that of Weight Majority Rule, we observe that the factor $(2+\eta)$ has been cut to $(1+\eta) \Rightarrow$ we can get close to optimal expert.
- "optimal" η can be found by minimizing RHS $\Rightarrow \eta^* = \sqrt{\frac{\ln k}{L^T_{OPT}}}$.
- We do not know L^T_{OPT} (as in WMR) but we can assume $L^T_{OPT} \sim T$, hence the choice of η (unfortunately, same trick does not hold for WMR)
- **Note:** An alternative but **equivalent** formulation requires $w_i^{t+1} = w_i^t \cdot e^{-\eta \cdot l_i^t}$. This is known as **Hedge**.

Performance of Multiplicative Weights or Hedge

Theorem

- The total loss of the MW algorithm is upper bounded
  $\mathbf{L^T_{ALG}} \leq \frac{\mathbf{lnk}}{\eta} + (\mathbf{1} + \eta)\mathbf{L^T_{OPT}}$
- Choosing $\eta = \sqrt{\frac{lnk}{T}}$
  $\Rightarrow \mathbf{Regret^T} = \mathbf{L^T_{ALG}} - \mathbf{L^T_{OPT}} \leq \mathbf{2}\sqrt{\mathbf{Tlnk}} = \mathbf{O}(\sqrt{\mathbf{T}})$

Key Remarks

1. Key Remark 1: Randomization is **necessary** for sublinear regret!

Performance of Multiplicative Weights or Hedge

Theorem

- The total loss of the MW algorithm is upper bounded $\mathbf{L_{ALG}^T} \leq \frac{\mathbf{lnk}}{\eta} + (\mathbf{1} + \eta)\mathbf{L_{OPT}^T}$
- Choosing $\eta = \sqrt{\frac{lnk}{T}}$
$\Rightarrow \mathbf{Regret^T} = \mathbf{L_{ALG}^T} - \mathbf{L_{OPT}^T} \leq \mathbf{2}\sqrt{\mathbf{Tlnk}} = \mathbf{O}(\sqrt{\mathbf{T}})$

Key Remarks

1. Key Remark 1: Randomization is **necessary** for sublinear regret!
2. Key Remark 2: MW/Hedge algorithm is order optimal! Every expert algorithm is $\Omega(\sqrt{T})$

Adversarial Bandits

- like Experts: l_i^t can change arbitrarily (not IID)

Adversarial Bandits

- like Experts: l_i^t can change arbitrarily (not IID)
- unlike Experts: *only* l_i^t for played arm i is revealed

Adversarial Bandits

- like Experts: l_i^t can change arbitrarily (not IID)
- unlike Experts: *only* l_i^t for played arm i is revealed
- We know that we cannot use previous algorithms (e.g., UCB)

Adversarial Bandits

- like Experts: l_i^t can change arbitrarily (not IID)
- unlike Experts: *only* l_i^t for played arm i is revealed
- We know that we cannot use previous algorithms (e.g., UCB)
- Hence we need to play arms probabilistically for bandits as well

Adversarial Bandits

- like Experts: l_i^t can change arbitrarily (not IID)
- unlike Experts: *only* l_i^t for played arm i is revealed
- We know that we cannot use previous algorithms (e.g., UCB)
- Hence we need to play arms probabilistically for bandits as well
- (how) can we use expert-like algorithms without full info?

How to use an expert algorithm (e.g. MW) for Bandits?

Observation: We cannot apply Multipl. Weights (or Hedge) directly!

How to use an expert algorithm (e.g. MW) for Bandits?

Observation: We cannot apply Multipl. Weights (or Hedge) directly! $\rightarrow$ Requires knowing all rewards/losses at time t to discount all weights.

Applying MW to Bandits: Attempt 1

at round t:

1. Pick arm i according to $p_i = \frac{w_i}{\sum_{i=1}^{k} w_i}$

How to use an expert algorithm (e.g. MW) for Bandits?

Observation: We cannot apply Multipl. Weights (or Hedge) directly! $\rightarrow$ Requires knowing all rewards/losses at time t to discount all weights.

Applying MW to Bandits: Attempt 1

at round t:

1. Pick arm i according to $p_i = \frac{w_i}{\sum_{i=1}^{k} w_i}$ (same as experts)

How to use an expert algorithm (e.g. MW) for Bandits?

Observation: We cannot apply Multipl. Weights (or Hedge) directly! $\rightarrow$ Requires knowing all rewards/losses at time t to discount all weights.

Applying MW to Bandits: Attempt 1

at round t:

1. Pick arm i according to $p_i = \frac{w_i}{\sum_{i=1}^{k} w_i}$ (same as experts)
2. Receive **only** loss l_i^t

How to use an expert algorithm (e.g. MW) for Bandits?

<u>Observation:</u> We cannot apply Multipl. Weights (or Hedge) directly! $\rightarrow$ Requires knowing all rewards/losses at time t to discount all weights.

Applying MW to Bandits: Attempt 1

at round t:

1. Pick arm i according to $p_i = \frac{w_i}{\sum_{i=1}^{k} w_i}$ (same as experts)
2. Receive **only** loss l_i^t (key difference from experts)

How to use an expert algorithm (e.g. MW) for Bandits?

<u>Observation:</u> We cannot apply Multipl. Weights (or Hedge) directly! $\rightarrow$ Requires knowing all rewards/losses at time t to discount all weights.

Applying MW to Bandits: Attempt 1

at round t:

1. Pick arm i according to $p_i = \frac{w_i}{\sum_{i=1}^{k} w_i}$ (same as experts)
2. Receive **only** loss l_i^t (key difference from experts)
3. Set

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

How to use an expert algorithm (e.g. MW) for Bandits?

Observation: We cannot apply Multipl. Weights (or Hedge) directly! $\rightarrow$ Requires knowing all rewards/losses at time t to discount all weights.

Applying MW to Bandits: Attempt 1

at round t:

1. Pick arm i according to $p_i = \frac{w_i}{\sum_{i=1}^{k} w_i}$ (same as experts)
2. Receive **only** loss l_i^t (key difference from experts)
3. Set

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

4. Use $\hat{l}_i^t$ to discount *all* weights

How to use an expert algorithm (e.g. MW) for Bandits?

Observation: We cannot apply Multipl. Weights (or Hedge) directly! $\rightarrow$ Requires knowing all rewards/losses at time t to discount all weights.

Applying MW to Bandits: Attempt 1

at round t:

1. Pick arm i according to $p_i = \frac{w_i}{\sum_{i=1}^{k} w_i}$ (same as experts)
2. Receive **only** loss l_i^t (key difference from experts)
3. Set
$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$
4. Use $\hat{l}_i^t$ to discount *all* weights $\Rightarrow w_i^{t+1} = (1-\eta)^{\hat{l}_i^t} \cdot w_i^t$

How to use an expert algorithm (e.g. MW) for Bandits?

Observation: We cannot apply Multipl. Weights (or Hedge) directly! $\rightarrow$ Requires knowing all rewards/losses at time t to discount all weights.

Applying MW to Bandits: Attempt 1

at round t:

1. Pick arm i according to $p_i = \frac{w_i}{\sum_{i=1}^{k} w_i}$ (same as experts)
2. Receive **only** loss l_i^t (key difference from experts)
3. Set

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

4. Use $\hat{l}_i^t$ to discount *all* weights $\Rightarrow w_i^{t+1} = (1-\eta)^{\hat{l}_i^t} \cdot w_i^t \Rightarrow$

$$w_i^{t+1} = \begin{cases} (1-\eta)^{\hat{l}_i^t} \cdot w_i^t, & \text{for picked arm } i \\ w_i^t, & \text{for all other } i \end{cases}$$

How to use an expert algorithm (e.g. MW) for Bandits?

Observation: We cannot apply Multipl. Weights (or Hedge) directly! $\rightarrow$ Requires knowing all rewards/losses at time t to discount all weights.

Applying MW to Bandits: Attempt 1

at round t:

1. Pick arm i according to $p_i = \frac{w_i}{\sum_{i=1}^{k} w_i}$ (same as experts)
2. Receive **only** loss l_i^t (key difference from experts)
3. Set
$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$
4. Use $\hat{l}_i^t$ to discount *all* weights $\Rightarrow w_i^{t+1} = (1-\eta)^{\hat{l}_i^t} \cdot w_i^t \Rightarrow$
$$w_i^{t+1} = \begin{cases} (1-\eta)^{\hat{l}_i^t} \cdot w_i^t, & \text{for picked arm } i \\ w_i^t, & \text{for all other } i \end{cases}$$

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

- Q: Why does this make sense?

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

- Q: Why does this make sense?
- A1: $\hat{l}_i^t$ is an **unbiased** estimate of l_i^t

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

- Q: Why does this make sense?
- A1: $\hat{l}_i^t$ is an **unbiased** estimate of l_i^t
  - Q: What does this even mean?

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

- Q: Why does this make sense?
- A1: $\hat{l}_i^t$ is an **unbiased** estimate of l_i^t
  - Q: What does this even mean?
  - A: $E[\hat{l}_i^t] = l_i^t$

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

- Q: Why does this make sense?
- A1: $\hat{l}_i^t$ is an **unbiased** estimate of l_i^t
  - Q: What does this even mean?
  - A: $E[\hat{l}_i^t] = l_i^t$
    Proof: $E[\hat{l}_i^t] = \sum_{j=1}^{k} p_j \cdot \hat{l}_j^t$

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

- Q: Why does this make sense?
- A1: $\hat{l}_i^t$ is an **unbiased** estimate of l_i^t
  - Q: What does this even mean?
  - A: $E[\hat{l}_i^t] = l_i^t$
    Proof: $E[\hat{l}_i^t] = \sum_{j=1}^k p_j \cdot \hat{l}_j^t = p_i \cdot \frac{l_i^t}{p_i} + \sum_{j \neq i} p_j \cdot 0$

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

- Q: Why does this make sense?
- A1: $\hat{l}_i^t$ is an **unbiased** estimate of l_i^t
    - Q: What does this even mean?
    - A: $E[\hat{l}_i^t] = l_i^t$

      Proof: $E[\hat{l}_i^t] = \sum_{j=1}^k p_j \cdot \hat{l}_j^t = p_i \cdot \frac{l_i^t}{p_i} + \sum_{j \neq i} p_j \cdot 0 = l_i^t$

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

- Q: Why does this make sense?
- A1: $\hat{l}_i^t$ is an **unbiased** estimate of l_i^t
  - Q: What does this even mean?
  - A: $E[\hat{l}_i^t] = l_i^t$
    Proof: $E[\hat{l}_i^t] = \sum_{j=1}^k p_j \cdot \hat{l}_j^t = p_i \cdot \frac{l_i^t}{p_i} + \sum_{j \neq i} p_j \cdot 0 = l_i^t$
- A2: The above method is also called **Importance Sampling**

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

- Q: Why does this make sense?
- A1: $\hat{l}_i^t$ is an **unbiased** estimate of l_i^t
  - Q: What does this even mean?
  - A: $E[\hat{l}_i^t] = l_i^t$
    Proof: $E[\hat{l}_i^t] = \sum_{j=1}^k p_j \cdot \hat{l}_j^t = p_i \cdot \frac{l_i^t}{p_i} + \sum_{j \neq i} p_j \cdot 0 = l_i^t$
- A2: The above method is also called **Importance Sampling** (We will encounter again in Deep RL methods)

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

- Q: Why does this make sense?
- A1: $\hat{l}_i^t$ is an **unbiased** estimate of l_i^t
  - Q: What does this even mean?
  - A: $E[\hat{l}_i^t] = l_i^t$
    Proof: $E[\hat{l}_i^t] = \sum_{j=1}^k p_j \cdot \hat{l}_j^t = p_i \cdot \frac{l_i^t}{p_i} + \sum_{j \neq i} p_j \cdot 0 = l_i^t$
- A2: The above method is also called **Importance Sampling** (We will encounter again in Deep RL methods)

Note: Signal Processing/Estimation Theory: Having "good" estimates usually means estimates with **no/low bias** and **low variance**

Multiplicative Weights to Bandits: Attempt 1

- Key step/idea is

$$\hat{l}_i^t = \begin{cases} \frac{l_i^t}{p_i}, & \text{for picked arm } i \\ 0, & \forall j \neq i \end{cases}$$

- Q: Why does this make sense?
- A1: $\hat{l}_i^t$ is an **unbiased** estimate of l_i^t
  - Q: What does this even mean?
  - A: $E[\hat{l}_i^t] = l_i^t$
    Proof: $E[\hat{l}_i^t] = \sum_{j=1}^{k} p_j \cdot \hat{l}_j^t = p_i \cdot \frac{l_i^t}{p_i} + \sum_{j \neq i} p_j \cdot 0 = l_i^t$
- A2: The above method is also called **Importance Sampling** (We will encounter again in Deep RL methods)

Note: Signal Processing/Estimation Theory: Having “good” estimates usually means estimates with **no/low bias** and **low variance** (Variance concerns will come back to “haunt” us in Deep RL methods)

Multiplicative Weights to Bandits: Attempt 2

BUT: MW requires loss to be in $[0, 1]$.

Multiplicative Weights to Bandits: Attempt 2

<u>**BUT:**</u> MW requires loss to be in $[0, 1]$. Unfortunately, $\hat{l}_i^t$ can be > 1 (why??).

Multiplicative Weights to Bandits: Attempt 2

<u>**BUT:**</u> MW requires loss to be in $[0,1]$. Unfortunately, $\hat{l}_i^t$ can be > 1 (why??). In fact it can even go to infinity (if $p_i \to 0$ at some round).

Multiplicative Weights to Bandits: Attempt 2

BUT: MW requires loss to be in $[0, 1]$. Unfortunately, $\hat{l}_i^t$ can be > 1 (why??). In fact it can even go to infinity (if $p_i \to 0$ at some round).

"Black Box" Method for Experts with generic reward $l_i^t \in [0, \rho]$

- Assume $l_i^t \in [0, \rho], \forall i \;\; (\rho > 1)$

Multiplicative Weights to Bandits: Attempt 2

BUT: MW requires loss to be in $[0, 1]$. Unfortunately, $\hat{l}_i^t$ can be > 1 (why??). In fact it can even go to infinity (if $p_i \to 0$ at some round).

"Black Box" Method for Experts with generic reward $l_i^t \in [0, \rho]$

- Assume $l_i^t \in [0, \rho], \forall i \;\; (\rho > 1)$
- Can feed (any) Experts Algorithm with $\hat{l}_i^t = \frac{l_i^t}{\rho}$

Multiplicative Weights to Bandits: Attempt 2

BUT: MW requires loss to be in $[0,1]$. Unfortunately, $\hat{l}_i^t$ can be > 1 (why??). In fact it can even go to infinity (if $p_i \to 0$ at some round).

"Black Box" Method for Experts with generic reward $l_i^t \in [0, \rho]$

- Assume $l_i^t \in [0, \rho], \forall i \;\; (\rho > 1)$
- Can feed (any) Experts Algorithm with $\hat{l}_i^t = \frac{l_i^t}{\rho}$ $(\Rightarrow \hat{l}_i^t \in [0,1])$

Multiplicative Weights to Bandits: Attempt 2

BUT: MW requires loss to be in $[0, 1]$. Unfortunately, $\hat{l}_i^t$ can be > 1 (why??). In fact it can even go to infinity (if $p_i \to 0$ at some round).

"Black Box" Method for Experts with generic reward $l_i^t \in [0, \rho]$

- Assume $l_i^t \in [0, \rho], \forall i \;\; (\rho > 1)$
- Can feed (any) Experts Algorithm with $\hat{l}_i^t = \frac{l_i^t}{\rho}$ $(\Rightarrow \hat{l}_i^t \in [0, 1])$.

Lemma

1. Let an expert algorithm (with $l_i^t \in [0, 1]$) achieve $L_{ALG} = E[\sum_{t=1}^{T} l_{I_t}^t] \leq \alpha \cdot \min_i \sum_{t=1}^{T} l_i^t + \beta$

Multiplicative Weights to Bandits: Attempt 2

BUT: MW requires loss to be in $[0, 1]$. Unfortunately, $\hat{l}_i^t$ can be > 1 (why??). In fact it can even go to infinity (if $p_i \to 0$ at some round).

"Black Box" Method for Experts with generic reward $l_i^t \in [0, \rho]$

- Assume $l_i^t \in [0, \rho], \forall i \ \ (\rho > 1)$
- Can feed (any) Experts Algorithm with $\hat{l}_i^t = \frac{l_i^t}{\rho}$ $(\Rightarrow \hat{l}_i^t \in [0, 1])$.

Lemma

1. Let an expert algorithm (with $l_i^t \in [0, 1]$) achieve $L_{ALG} = E[\sum_{t=1}^{T} l_{I_t}^t] \leq \alpha \cdot \min_i \sum_{t=1}^{T} l_i^t + \beta$ (for some constants α, β)

Multiplicative Weights to Bandits: Attempt 2

BUT: MW requires loss to be in $[0,1]$. Unfortunately, $\hat{l}_i^t$ can be > 1 (why??). In fact it can even go to infinity (if $p_i \to 0$ at some round).

"Black Box" Method for Experts with generic reward $l_i^t \in [0,\rho]$

- Assume $l_i^t \in [0,\rho], \forall i \;\; (\rho > 1)$
- Can feed (any) Experts Algorithm with $\hat{l}_i^t = \frac{l_i^t}{\rho}$ $(\Rightarrow \hat{l}_i^t \in [0,1])$.

Lemma

1. Let an expert algorithm (with $l_i^t \in [0,1]$) achieve $L_{ALG} = E[\sum_{t=1}^T l_{I_t}^t] \leq \alpha \cdot \min_i \sum_{t=1}^T l_i^t + \beta$ (for some constants α, β)
2. Then the same algorithm with $l_i^t \in [0,\rho] \;\; (\rho > 1)$ achieves $L_{ALG} \leq \alpha \cdot \min_i \sum_{t=1}^T l_i^t + \rho \cdot \beta$

Multiplicative Weights to Bandits: Attempt 2

BUT: MW requires loss to be in $[0,1]$. Unfortunately, $\hat{l}_i^t$ can be >1 (why??). In fact it can even go to infinity (if $p_i \to 0$ at some round).

"Black Box" Method for Experts with generic reward $l_i^t \in [0,\rho]$

- Assume $l_i^t \in [0,\rho], \forall i \;\; (\rho > 1)$
- Can feed (any) Experts Algorithm with $\hat{l}_i^t = \frac{l_i^t}{\rho}$ $(\Rightarrow \hat{l}_i^t \in [0,1])$.

Lemma

1. Let an expert algorithm (with $l_i^t \in [0,1]$) achieve $L_{ALG} = E[\sum_{t=1}^T l_{I_t}^t] \leq \alpha \cdot \min_i \sum_{t=1}^T l_i^t + \beta$ (for some constants α, β)
2. Then the same algorithm with $l_i^t \in [0,\rho] \;\; (\rho > 1)$ achieves $L_{ALG} \leq \alpha \cdot \min_i \sum_{t=1}^T l_i^t + \rho \cdot \beta$
3. The Multiplicative Weights algorithm for experts with rewards in $[0,\rho] \;\; (\rho > 1)$ achieves regret $\mathbf{Regret^T} = \mathbf{2}\rho\sqrt{\mathbf{T \ln k}}$

Multiplicative Weights to Bandits: Attempt 2

BUT: MW requires loss to be in $[0,1]$. Unfortunately, $\hat{l}_i^t$ can be > 1 (why??). In fact it can even go to infinity (if $p_i \to 0$ at some round).

"Black Box" Method for Experts with generic reward $l_i^t \in [0, \rho]$

- Assume $l_i^t \in [0, \rho], \forall i \;\; (\rho > 1)$
- Can feed (any) Experts Algorithm with $\hat{l}_i^t = \frac{l_i^t}{\rho}$ $(\Rightarrow \hat{l}_i^t \in [0, 1])$.

Lemma

1. Let an expert algorithm (with $l_i^t \in [0,1]$) achieve $L_{ALG} = E[\sum_{t=1}^{T} l_{I_t}^t] \leq \alpha \cdot \min_i \sum_{t=1}^{T} l_i^t + \beta$ (for some constants α, β)
2. Then the same algorithm with $l_i^t \in [0, \rho] \;\; (\rho > 1)$ achieves $L_{ALG} \leq \alpha \cdot \min_i \sum_{t=1}^{T} l_i^t + \rho \cdot \beta$
3. The Multiplicative Weights algorithm for experts with rewards in $[0, \rho] \;\; (\rho > 1)$ achieves regret $\mathbf{Regret^T} = \mathbf{2}\rho\sqrt{\mathbf{T \ln k}}$ (This is just the regret of slide 20 scaled up by ρ).

Multiplicative Weights to Bandits: Attempt 3

Note:

- Good news 1: Expert algorithms generalize to any upper bounded loss functions.

Multiplicative Weights to Bandits: Attempt 3

Note:

- Good news 1: Expert algorithms generalize to any upper bounded loss functions.
- Good news 2: For bandits, the fact that the (unbiased) estimate $\hat{l}_i^t$ can be larger than 1 (see slide 24), is **not** a problem!

Multiplicative Weights to Bandits: Attempt 3

Note:

- Good news 1: Expert algorithms generalize to any upper bounded loss functions.
- Good news 2: For bandits, the fact that the (unbiased) estimate $\hat{l}_i^t$ can be larger than 1 (see slide 24), is **not** a problem!
- Bad news: The fact that it is **not upper bounded** (i.e., $\rho = \infty$) **IS** a major problem.

Multiplicative Weights to Bandits: Attempt 3

Note:

- Good news 1: Expert algorithms generalize to any upper bounded loss functions.
- Good news 2: For bandits, the fact that the (unbiased) estimate $\hat{l}_i^t$ can be larger than 1 (see slide 24), is **not** a problem!
- Bad news: The fact that it is **not upper bounded** (i.e., $\rho = \infty$) **IS** a major problem.

MW algorithm adapted to (adversarial) bandits

- Solution: add some **exploration**:

Multiplicative Weights to Bandits: Attempt 3

Note:

- Good news 1: Expert algorithms generalize to any upper bounded loss functions.
- Good news 2: For bandits, the fact that the (unbiased) estimate $\hat{l}_i^t$ can be larger than 1 (see slide 24), is **not** a problem!
- Bad news: The fact that it is **not upper bounded** (i.e., $\rho = \infty$) **IS** a major problem.

MW algorithm adapted to (adversarial) bandits

- Solution: add some **exploration**: Define $q_i^t = (1-\epsilon)p_i^t + \frac{\epsilon}{k}$
- Pick arm based on q_i^t

Multiplicative Weights to Bandits: Attempt 3

Note:

- Good news 1: Expert algorithms generalize to any upper bounded loss functions.
- Good news 2: For bandits, the fact that the (unbiased) estimate $\hat{l}_i^t$ can be larger than 1 (see slide 24), is **not** a problem!
- Bad news: The fact that it is **not upper bounded** (i.e., $\rho = \infty$) **IS** a major problem.

MW algorithm adapted to (adversarial) bandits

- Solution: add some **exploration**: Define $q_i^t = (1-\epsilon)p_i^t + \frac{\epsilon}{k}$
- Pick arm based on q_i^t (rather than p_i^t)

Multiplicative Weights to Bandits: Attempt 3

Note:

- Good news 1: Expert algorithms generalize to any upper bounded loss functions.
- Good news 2: For bandits, the fact that the (unbiased) estimate $\hat{l}_i^t$ can be larger than 1 (see slide 24), is **not** a problem!
- Bad news: The fact that it is **not upper bounded** (i.e., $\rho = \infty$) **IS** a major problem.

MW algorithm adapted to (adversarial) bandits

- Solution: add some **exploration**: Define $q_i^t = (1-\epsilon)p_i^t + \frac{\epsilon}{k}$
- Pick arm based on q_i^t (rather than p_i^t)
- Feed to algorithm $l_i^t \hat{=} \frac{l_i^t}{q_i^t}$, for picked arm i (0 for all other arms)

Multiplicative Weights to Bandits: Attempt 3

Note:

- Good news 1: Expert algorithms generalize to any upper bounded loss functions.
- Good news 2: For bandits, the fact that the (unbiased) estimate $\hat{l}_i^t$ can be larger than 1 (see slide 24), is **not** a problem!
- Bad news: The fact that it is **not upper bounded** (i.e., $\rho = \infty$) **IS** a major problem.

MW algorithm adapted to (adversarial) bandits

- Solution: add some **exploration**: Define $q_i^t = (1-\epsilon)p_i^t + \frac{\epsilon}{k}$
- Pick arm based on q_i^t (rather than p_i^t)
- Feed to algorithm $l_i^t \hat{=} \frac{l_i^t}{q_i^t}$, for picked arm i (0 for all other arms)
- $\rho = \frac{k}{\epsilon}$ in this case $\Rightarrow$ we can apply the previous "black box" method (and proof).

Performance of MW for adversarial Bandits

Theorem

- The total loss of the Multiplicative Weights algorithm for the adversarial bandits setting is $\mathbf{L_{ALG}} \leq (\mathbf{1}+\eta) \cdot \mathbf{min_i} \sum_{\mathbf{t=1}}^{\mathbf{T}} \mathbf{l_i^t} + \frac{\mathbf{k \ln k}}{\epsilon\eta} + \epsilon\mathbf{T}$

Performance of MW for adversarial Bandits

Theorem

- The total loss of the Multiplicative Weights algorithm for the adversarial bandits setting is
$\mathbf{L_{ALG}} \leq (\mathbf{1} + \eta) \cdot \mathrm{min}_{\mathbf{i}} \sum_{\mathbf{t=1}}^{\mathbf{T}} \mathbf{l_i^t} + \frac{\mathbf{k \ln k}}{\epsilon \eta} + \epsilon \mathbf{T}$
- By properly picking η (the discount parameter) and ϵ (the exploration probability)

Performance of MW for adversarial Bandits

Theorem

- The total loss of the Multiplicative Weights algorithm for the adversarial bandits setting is
  $\mathbf{L_{ALG}} \leq (\mathbf{1}+\eta)\cdot \min_{\mathbf{i}} \sum_{\mathbf{t=1}}^{\mathbf{T}} \mathbf{l_i^t} + \frac{\mathbf{k}\ln\mathbf{k}}{\epsilon\eta} + \epsilon\mathbf{T}$
- By properly picking η (the discount parameter) and ϵ (the exploration probability) :$\mathbf{Regret^T} = \mathbf{O}(\sqrt{\mathbf{Tk}\ln\mathbf{k}})$.

Performance of MW for adversarial Bandits

Theorem

- The total loss of the Multiplicative Weights algorithm for the adversarial bandits setting is $\mathbf{L_{ALG}} \leq (\mathbf{1} + \eta) \cdot \mathrm{min}_\mathbf{i} \sum_{\mathbf{t=1}}^{\mathbf{T}} \mathbf{l_i^t} + \frac{\mathbf{k \ln k}}{\epsilon\eta} + \epsilon \mathbf{T}$
- By properly picking η (the discount parameter) and ϵ (the exploration probability) :$\mathbf{Regret^T} = \mathbf{O}(\sqrt{\mathbf{Tk \ln k}})$.

Remarks:

- It is (almost) amazing that we can achieve the same $O(\sqrt{T})$ regret in this *very hostile* setting

Performance of MW for adversarial Bandits

Theorem

- The total loss of the Multiplicative Weights algorithm for the adversarial bandits setting is $\mathbf{L_{ALG}} \leq (\mathbf{1}+\eta)\cdot \min_{\mathbf{i}} \sum_{\mathbf{t=1}}^{\mathbf{T}} \mathbf{l_i^t} + \frac{\mathbf{k \ln k}}{\epsilon\eta} + \epsilon\mathbf{T}$
- By properly picking η (the discount parameter) and ϵ (the exploration probability) :$\mathbf{Regret^T} = \mathbf{O}(\sqrt{\mathbf{Tk \ln k}})$.

Remarks:

- It is (almost) amazing that we can achieve the same $O(\sqrt{T})$ regret in this *very hostile* setting (**adversary and partial observability**)

Performance of MW for adversarial Bandits

Theorem

- The total loss of the Multiplicative Weights algorithm for the adversarial bandits setting is $\mathbf{L_{ALG}} \leq (\mathbf{1} + \eta) \cdot \min_{\mathbf{i}} \sum_{\mathbf{t=1}}^{\mathbf{T}} \mathbf{l_i^t} + \frac{\mathbf{k \ln k}}{\epsilon \eta} + \epsilon \mathbf{T}$
- By properly picking η (the discount parameter) and ϵ (the exploration probability) :$\mathbf{Regret^T} = \mathbf{O}(\sqrt{\mathbf{Tk \ln k}})$.

Remarks:

- It is (almost) amazing that we can achieve the same $O(\sqrt{T})$ regret in this *very hostile* setting (**adversary and partial observability**)
- Observe the extra factor K in the regret.

Performance of MW for adversarial Bandits

Theorem

- The total loss of the Multiplicative Weights algorithm for the adversarial bandits setting is
$\mathbf{L_{ALG} \leq (1+\eta) \cdot \min_i \sum_{t=1}^{T} l_i^t + \frac{k \ln k}{\epsilon\eta} + \epsilon T}$
- By properly picking η (the discount parameter) and ϵ (the exploration probability) :$\mathbf{Regret^T = O(\sqrt{Tk \ln k})}$.

Remarks:

- It is (almost) amazing that we can achieve the same $O(\sqrt{T})$ regret in this *very hostile* setting (**adversary and partial observability**)
- Observe the extra factor K in the regret. This is the penalty of partial observability!

Performance of MW for adversarial Bandits

Theorem

- The total loss of the Multiplicative Weights algorithm for the adversarial bandits setting is
  $\mathbf{L_{ALG}} \leq (\mathbf{1} + \eta) \cdot \mathrm{min_i} \sum_{\mathbf{t=1}}^{\mathbf{T}} \mathbf{l_i^t} + \frac{\mathbf{k \ln k}}{\epsilon \eta} + \epsilon \mathbf{T}$
- By properly picking η (the discount parameter) and ϵ (the exploration probability) :$\mathbf{Regret^T} = \mathbf{O}(\sqrt{\mathbf{Tk} \ln \mathbf{k}})$.

Remarks:

- It is (almost) amazing that we can achieve the same $O(\sqrt{T})$ regret in this *very hostile* setting (**adversary and partial observability**)
- Observe the extra factor K in the regret. This is the penalty of partial observability! ($\rightarrow$ We need k rounds to get the same feedback that the experts get in 1 round)