UM400 Project Report

Multi Arm Bandit Learning in Adversarial Setting

Submitted in partial fulfillment of
the requirements for the award of the degree of

Bachelors of Science (Research)
in
Mathematics

Submitted by
Mukund Seethamraju
SR No: 11-01-00-10-91-12-1-09969

Under the guidance of
Prof. Aditya Gopalan (ECE Department)
Prof. Manjunath Krishnapur (Mathematics Department)

Department of Undergraduate Studies
Indian Institute of Science
Bangalore, India

Introduction

Multi Armed Bandit problem (MAB) is a fundamental decision making problem introduced by Robbins in 1952. This problem can be thought of as a player who is playing k-slot machines (or arms). Every time the player has to choose an arm from the k options available and gets reward for his decision. The game is played iteratively over T rounds and the aim of the player is to maximize his total reward over T rounds. It is practically impossible to find the sequence of decisions which gives maximum reward. Hence In MAB algorithms we try to minimize the regret which is defined as the difference between the reward received by the player for his sequence of decisions and the reward the player would have received if he had chosen only one best arm during all rounds during the play. This problem presents exploration vs exploitation dilemma where in each round, the player has to maintain the trade-off between exploiting the arm that has the highest expected reward and at the same time exploring remaining arms to learn more about the expected rewards of other arms.

Multi Armed Bandit learning in adversarial setting

Adversarial (non-stochastic) setting can be considered as the most general case in Multi Armed Bandit problem. In this setting, it is assumed that the arms generating rewards are non-stochastic. That is, no assumptions are made regarding existence of underlying probability distributions for the arms generating rewards.

Let us assume that the player plays the game with k arms for T rounds. Let K denote the set of arms $\{1, 2, 3, ..., k\}$ and let $t = 1, 2, 3..., T$ denote the t^{th} round. Let f_t be the vector containing rewards of all arms in t^{th} round i.e. $f_t(i)$ is the reward the player would receive if he chooses i^{th} arm in round t. Let i_t denote the arm the player chooses in round t. In adversarial setting, in round t, after the player has chosen arm i_t, he receives the reward $f_t(i_t)$ but he gets no information about the rewards he would have got if he had chosen other arm i.e. the player only sees $f_t(i_t)$ but not the whole vector f_t. Regret R is defined as:

$$R = \max_{i \in K} \sum_{t=1}^{T} f_t(i) - \sum_{t=1}^{T} f_t(i_t)$$

Related Studies

In 2001, Peter Auer presented EXP3 (Exponential-weight algorithm for Exploration and Exploitation) algorithm[1] which has an optimal regret $O(\sqrt{kT \log T})$ which is directly proportional to $\sqrt{T}$. EXP3 is a simple algorithm that assigns weights to all the arms initially and then picks an arm according to weighted probability distribution of the arms. After getting the reward, the weights are updated and the process goes on.

In real world scenarios, it is very unlikely that the arms are truly adversarial and hence regret bound in $\tilde{O}(\sqrt{T})$ is weak. In 2011, Hazan and Kale[2] proposed an algorithm for bandit linear optimisation problem which uses techniques such as reservoir sampling and self-concordant functions. They have shown tighter regret bound of $\tilde{O}(\sqrt{Q})$, which is in terms of Q which is defined as the quadratic variation in reward vectors.

$$Q = \sum_{t=1}^{T} ||f_t - \mu||^2 \quad \text{where} \quad \mu = \frac{1}{T} \sum_{t=1}^{T} f_t$$

Q gives a measure of variance in the reward vectors and hence, hardness in learning will be proportional to how much the reward vectors deviate from the mean rather than the total time of game play, T.

In 2015, Gergely Neu proposed EXP3-IX algorithm [4] which implements strategy called Implicit Exploration. Since, explicit exploration used in EXP3 has huge variance and can sometimes leads to large regrets, this strategy makes sure that the variance of the expected reward is under control. In adversarial setting, this method improves regret bound by constant factor with high probability.

Problem

Though Hazan and Kale in their paper[2] were able to provide an algorithm with regret bound in terms of variation, the algorithm is based on self-concordant barrier functions as regularizers and reservoir sampling. This makes the algorithm complicated and polynomial running time. So in 2011, they have put an open question[3]:

"The open question is to design a simple, linear-time algorithm for MAB which has a regret bound of $O(\sqrt{Q \log T})$, hence improving upon EXP3."

The aim of this project is to see how different algorithms like EXP3 and EXP3-IX behave upon providing information regarding quadratic variation to the algorithm. This can be done by creating reward vectors of different variation values. And then using these reward vectors, run the simulations over the EXP3 and other variants of the algorithm and see if there exists any useful relation between "Variation of the reward vectors" and "Regret". And, if any such relationship can be found, idea is to use this information to improve the existing EXP3 algorithm by bounding the regret in terms of quadratic variation.

Reward Vectors

Since for a particular quadratic variation value, multiple sets of arms with reward vectors can exist. Hence, it is important to generate reward vectors which are most likely to give high regrets (arms trying to fool the algorithm) so that the worst case performance of the algorithm can be tested. This can be done easily using markov chains. Example reward vectors: $f_t = (2, 0)$ for $t <= 300$ and $f_t = (0, 2)$ for $300 < t <= 1000$.

Algorithm Testing

During the whole project, each game (or algorithm for a particular set of arms) is simulated over number of simulations = 100 the number of arms is set $k = 2$ and each arm is allowed to output a reward from $\{0, 1, 2\}$. This helps in testing the algorithm in worst case environments. In each simulation the algorithm is played for $T = 1000$ rounds. Then every time, for particular set of arms, average regret over 100 simulations is calculated. Then 'average regret' vs 'quadratic variation' graph is plotted and the error bars in the graphs denote the standard deviation in regret values over 100 simulations.

EXP3 Algorithm

Though we generally don't know variation, Q in advance, assuming that we know value of Q for every game, EXP3 algorithm with γ proportional to $\frac{1}{\sqrt{Q}}$ has been tested on different sets of arms and average regret is calculated.

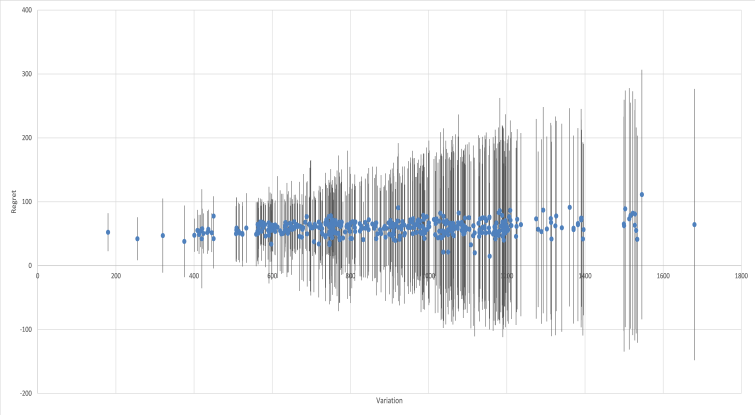


Figure 1: EXP3 with $\gamma \propto \frac{1}{\sqrt{Q}}$

It can be seen from the Figure 1 that the regret doesn't increase with variation as needed and also the high standard deviations show that the algorithm suffered to perform well in worst case environment. The regret is not proportional to $\sqrt{Q}$ after modifying learning parameter.

Reservoir Sampling

Reservoir sampling is a randomized sampling algorithm for choosing k items out of unknown number of items (usually very large compared to k). This is done by selecting the first k items with probability 1 and then for every t^{th} incoming element, it is selected with probability $\frac{k}{t}$. If the element is selected then, it is replaced randomly with one of the k elements.

EXP3-IX Algorithm

EXP3-IX is algorithm is tested with the assumption that value of Q is available for different sets of arms. The algorithm is given below:

Algorithm 1: EXP3-IX with $\gamma \propto \frac{1}{\sqrt{Q}}$ and $\eta \propto \frac{1}{\sqrt{T}}$

for *iter = 1,2,...., 100* **do**
  Parameters: $\eta = \sqrt{\frac{2 \log k}{kT}}, \gamma = \frac{1}{\sqrt{Q}}$;
  Initialization: $w_{1,i} = 1$;
  for t = *1,2,....T* **do**
    $p_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{k} w_{t,j}}$;
    Draw $i_t \quad p_t = (p_{t,1}, p_{t,2}, ...p_{t,k})$;
    Observe reward $r_t = f_t(i_t)$;
    $\tilde{r_t} \leftarrow \frac{r_t}{p_{t,i_t}+\gamma}$;
    $w_{t+1,i_t} = w_{t,i_t} e^{\eta \tilde{r_t}}$
  end
end

Theoretically, regret bound achieved by this algorithm in adversarial setting for $\eta = \sqrt{\frac{2 \log k}{kT}}$ and $\gamma = \frac{1}{\sqrt{Q}}$ as above is $\tilde{O}(\sqrt{Q} + \sqrt{kT})$.

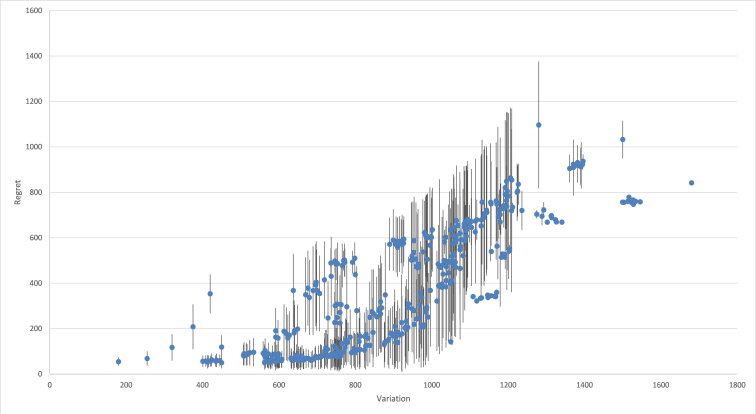


Figure 2: EXP3-IX with $\gamma \propto \frac{1}{\sqrt{Q}}$

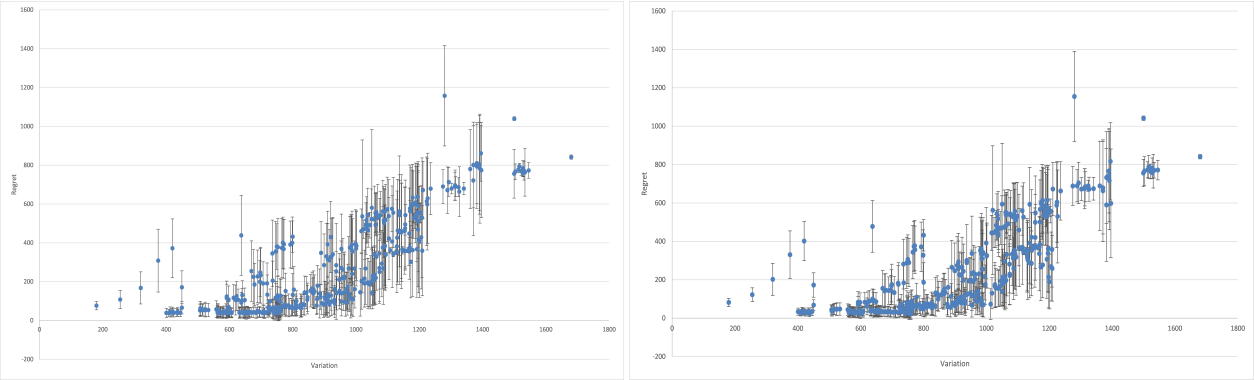
Here in this Figure 2, we can see that the average regret is increasing with quadratic variation in the reward vectors which is also proved in the theoretical bound.

But, in adversarial setting we don't get to know the value of Q beforehand. So, instead of knowing Q before, assuming that the player gets to see f_t (used only to calculate variation but not updating weights) at the end of round t, we can estimate Q in every round using reservoir sampling method. Let S be the reservoir and s be its size. The algorithm is given below:

Algorithm 2: EXP3-IX with $\gamma \propto \frac{1}{\sqrt{Q}}$ and $\eta \propto \frac{1}{\sqrt{T}}$ with Reservoir Sampling estimate of Q

for *iter = 1,2,...., 100* **do**
  Parameters: $\eta = \sqrt{\frac{2 \log k}{kT}}$;
  Initialization: $w_{1,i} = 1, \gamma = 0$, variation $= 0$;
  for t *= 1,2,....T* **do**
    if *variation = 0* **then**
      $\gamma = 0$
    else
      $\gamma = \frac{1}{\sqrt{variation}}$
    end
    $p_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{k} w_{t,j}}$;
    Draw $i_t \quad p_t = (p_{t,1}, p_{t,2}, ...p_{t,k})$;
    Observe reward $r_t = f_t(i_t)$;
    $\tilde{r_t} \leftarrow \frac{r_t}{p_{t,i_t}+\gamma}$;
    $w_{t+1,i_t} = w_{t,i_t} e^{\eta \tilde{r_t}}$;
    if $t = 1$ **then**
      set variation $= 0$;
      add f_t to S
    else if $2 \leq t \leq s$ **then**
      calculate quadratic variation(variation) of S;
      variation = variation*$\frac{T}{t-1}$;
      add f_t to S
    else
      calculate quadratic variation(variation) of S;
      variation = variation*$\frac{T}{s}$;
      choose f_t with probability $\frac{k}{t}$;
      if f_t *is chosen* **then**
        replace random item from S with f_t
    end
  end
end

This algorithm is tested for different reservoir sizes = 10, 20, 25, 40. It's been checked that, even though estimate of Q is used instead of Q itself, the algorithm has shown results almost similar to that of earlier one where it was assumed that Q is known.



(a) size $s = 20$ (b) size $s = 40$

Figure 3: EXP3-IX with $\gamma \propto \frac{1}{\sqrt{Q}}$ where Q is the estimate from reservoir of size s

References

[1] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, January 2003.

[2] Elad Hazan and Satyen Kale. Better algorithms for benign bandits. *J. Mach. Learn. Res.*, 12:1287–1311, July 2011.

[3] Elad Hazan and Satyen Kale. A simple multi-armed bandit algorithm with optimal variation-bounded regret. *J. Mach. Learn. Res.*, 12, July 2011.

[4] Gergely Neu. Explore no more: Improved high-probability regret bounds for non-stochastic bandits. pages 3168–3176, 2015.