1 Introduction

In this lecture we cover a tighter bound for Exp3 bandits, a primal dual proof of the tight upper bound for the EXP algorithm and introduce ($\gamma$-unfair) metrical task systems and illustrate their relations to the (shifting) expert problem.

2 $T^{1/2}$ regret bound for the Exp3 algorithm

In this section we show how the refined upper bound on the regret of the EXP algorithm proved using the potential function approach (KL divergence) also gives us a better bound for the expert game setup with bandit feedback.

Last lecture we showed how in the case of expert prediction with bandit feedback using the Exp3 algorithm, the regret is upper bounded by $T^{2/3}n^{1/3}$ using a rough upper bound on the regret for the EXP algorithm, a special instance of the multiplicative weight update. We also proved a tighter bound on the EXP regret as in the following theorem

**Theorem 1** For a fix $\epsilon$, suppose the gains $g^t$ satisfy $\|e^t\|_\infty \leq 1$. Using the EXP algorithm on the expert problem with gain vectors $g$ (the same holds for the loss case), horizon $T$ and $n$ arms, the regret is upper bounded by

$$\mathbb{E}[\text{Regret}] \leq \epsilon \sum_{t=1}^{T} p^t(g^t)^2 + \frac{\log n}{\epsilon} \quad (1)$$

where $(g^t)^2$ denotes the vector $(g^t)^2 := ((g^t_1)^2, \ldots, (g^t_n)^2)$.

Using this refinement, we can now prove a tighter bound on the Exp3 algorithm.

Let’s remind ourselves that the Exp3 algorithm updates

$$w^t_i = \frac{e^{\epsilon g^t_i}w^{t-1}_i}{Z} \quad (2)$$

with normalizer $Z = \sum_{i=1}^{n} w^t_i$ and then draws arm according to the probability distribution

$$q^t_i = (1 - \gamma)w^t_i + \frac{\gamma}{n} \quad (3)$$

**Theorem 2** For an expert game with bandit feedback with horizon $T$, the regret using the Exp3 algorithm is upper bounded by

$$\mathbb{E}[\text{Regret}] \leq c\sqrt{Tn \log n} \quad (4)$$

for some fixed constant $c$. 

Proof: We again use the notion of fake and real games introduced in the last lecture.

Let \( i_t \) be the index of the expert chosen at time \( i \). Then we can interpret the Exp3 algorithm as running an EXP algorithm on the fake game with complete information, where the expert reveals the entire gain vector \( \tilde{g}_t \) for \( i = i_t \) and samples according to probability vectors \( p^t = w^t \).

We first consider the fake game: we know from Theorem 1 that for the fake gain \( G_{\text{fake}} = \sum_t p^t \tilde{g}^t \) we have

\[
E[\text{Regret}_{\text{fake}}] = \max_{i=1,\ldots,n} \sum_t \tilde{g}_t^i - E G_{\text{fake}} \leq \epsilon E \sum_t p^t (\tilde{g}^t)^2 + \frac{\log n}{\epsilon},
\]

where the expectation is over the expert draws according to \( q \). We can bound the first term on the right hand side as

\[
E \sum_t p^t (\tilde{g}^t)^2 = \sum_t E \mathbb{1}_{(i=i_t)} p^t_i (g^t_i)^2 = \sum_{i=1}^n \sum_t p^t_i (g^t_i)^2 \leq Tn \frac{1}{1-\gamma}
\]

by definition (3) and since \( \|g^t\|_{\infty} \leq 1 \).

The real expected gain is lower bounded by definition of the real updates in (3) as

\[
E G_{\text{real}} = \sum_t q^t g^t \geq (1-\gamma) \sum_t p^t g^t.
\]

Also, since \( \tilde{g} \) is an unbiased estimator of \( g \) for all \( t \), it follows that

\[
E G_{\text{fake}} = E \sum_t p^t \tilde{g}^t = \sum_t p^t g^t.
\]

Putting things together, we finally have

\[
E[\text{Regret}_{\text{real}}] = \max_i \sum_t g_t^i - E G_{\text{real}} \leq \max_i \sum_t g_t^i - (1-\gamma) E G_{\text{fake}}
\leq \max_i \sum_t \tilde{g}_t^i - E G_{\text{fake}} + \gamma E G_{\text{fake}}
\leq \frac{cTn}{1-\gamma} + \frac{\log n}{\epsilon} + \gamma \sum_t p^t g^t \leq \frac{cTn}{1-\gamma} + \frac{\log n}{\epsilon} + \gamma T
\]

Using \( \epsilon = \frac{\gamma}{n} \) and small \( \gamma < 1/2 \), we have \( \frac{1}{1-\gamma} \leq 2 \) and thus

\[
E[\text{Regret}_{\text{real}}] \leq c\gamma T + \frac{n \log n}{\gamma}
\]

If we now explicitly choose \( \gamma = \sqrt{\frac{n \log n}{T}} \), the result (4) follows. \( \square \)
3 Proof of Theorem 1 using the Primal-Dual method

We now look at the online primal-dual method to prove the Theorem 1 where the online algorithm is doing the same update as the multiplicative weight. In particular, the original problem that the LP is supposed to solve is the one of a static expert which can also switch to one expert once in the very beginning. The result of the online primal-dual algorithm for this problem is then compared to the result of the offline algorithm which finally gives the bound in Theorem 1. The proof also neatly written up in [1]

Proof of Theorem 1. First observe that we can write the static problem (i.e. finding the best fixed expert that should have been chosen) as an LP in the following way

\[
\begin{align*}
\min_{y,z} \quad & \sum_{t=1}^{T} \sum_{i=1}^{n} c_{i,t}y_{i,t} + \sum_{t=2}^{T} \sum_{i=1}^{n} z_{i,t} \\
\text{s.t.} \quad & \sum_{i=1}^{n} y_{i,t} = 1 \quad \forall t \geq 0 \quad (\alpha_t) \\
& z_{i,t} \geq y_{i,t} - y_{i,t-1} \quad \forall t \geq 1, i \quad (\beta_{i,t}) \\
& z_{i,t}, y_{i,t} \geq 0 \quad \forall t
\end{align*}
\]

where \( c_{i,t} \) is the loss of expert \( i \) at a time step \( t \), \( y_{i,t} \) is the probability distribution over the experts at time \( t \) and \( z_{i,t} \) is the moving cost. The \( \alpha \) and \( \beta \) are the respective dual variables for each of the conditions. We assume that \( y_{i,0} = \frac{1}{n} \) for all \( i \).

The dual problem then reads

\[
\begin{align*}
\max_{\alpha, \beta} \quad & \sum_{t=0}^{T} \alpha_t \\
\text{s.t.} \quad & \alpha_0 - \beta_{i,1} \leq 0 \quad \forall i \\
& \alpha_t - \beta_{i,t} + \beta_{i,t+1} \leq c_{i,t} \quad \forall t \geq 1, i \\
& \beta_{i,t} \geq 0
\end{align*}
\]

The offline algorithm yields a vector with only one entry equal to one which corresponds exactly to the optimal loss of the static expert. By not enforcing it in our constraints however, the online algorithm can come up with \( y \) as proper probability distributions and the first term in the objective becomes our expected loss.

The online algorithm works as described in Algorithm 1. One can see directly that \( y_{i,t} \) is always primal feasible and also that \( y_{i,0} = \frac{1}{n} \) for all \( i \). In fact, the choice of \( \beta_{i,1} \) was made such that primal

\begin{algorithm}
1. Set \( \beta_{i,1} = \frac{\log n}{\epsilon} \) and \( \alpha_0 = -\frac{\log n}{\epsilon} \).
2. \( \beta_{i,t+1} \leftarrow c_{i,t} + \beta_{i,t} - \alpha_t \) where \( \alpha_t \) is raised until \( \sum_t y_{i,t} = 1 \).
3. Maintain \( y_{i,t} = e^{-\epsilon \beta_{i,t+1}} \)
\end{algorithm}
and dual feasibility is obtained while keeping \( \alpha_0 \) to a minimum. The main point now is that the update for \( y_{i,t} \) is exactly equivalent to the EXP update with parameter \( \epsilon \) since

\[
y_{i,t} = e^{\epsilon\alpha_t} e^{-\epsilon\beta_i t} e^{-\epsilon c_{i,t}} =: \frac{1}{Z} y_{i,t} e^{-\epsilon c_{i,t}}
\]

where raising the \( \alpha_t \) is equivalent to achieving normalization by having \( e^{-\epsilon\alpha_t} = \sum_{i=1}^{n} y_{i,t} e^{-\epsilon c_{i,t}} \).

We now turn to upper bound the effective cost of the online algorithm. By using simple inequalities for exponentials we obtain

\[
1 - \epsilon \alpha_t \leq e^{-\epsilon \alpha_t} \leq \sum_i y_{i,t-1}(1 - \epsilon c_{i,t} + \epsilon^2 c_{i,t}^2) \leq 1 - \epsilon \sum_i y_{i,t-1} c_{i,t} + \epsilon^2 \sum_i y_{i,t-1} c_{i,t}^2
\]

Using duality and hence \( L^* \geq \sum_{t=0}^{T} \alpha_t \), this directly implies

\[
\mathbb{E} L - L^* = \sum_{t=1}^{T} \sum_i y_{i,t-1} c_{i,t} - \sum_{t=1}^{T} \alpha_t \leq \epsilon \sum_{t=1}^{T} \sum_{i=1}^{n} y_{i,t-1} c_{i,t}^2 - \alpha_0
\]

which yields the theorem. \( \square \)

**Remark 1** The advantage of writing the expert game in the way of the LP with \( \infty \) penalty on movements is that shifting regret problems and the likes then becomes a straightforward generalization of this problem by replacing \( \infty \) by a finite number.

### 4 Metrical Task System

We now look at a different type of problem, with the help of which we can upper bound the shifting regret but is also a well-studied problem on its own. A metrical task system can be thought of as having tasks coming in at time \( t \) that need to be completed by one of the available machines \( i = 1, \ldots, n \) which all need different costs \( c_{i,t} \) to complete it. The system is revealed the cost vector, can either stay with a current machine \( i \) or move to a different machine \( j \) for a cost \( d(i,j) \) where \( d \) is some metric. The goal is to minimize the total cost of completing all tasks up to a horizon \( T \).

Apart from the obvious differences to the expert setting which include the 1-step look-ahead and the moving cost, in this case the regret won’t be comparing moving experts to a static one but rather both cases are allowed to move all the time. Moreover, the MTS allows unbounded cost vectors.

Tight bounds for the competitive ratio have been shown in [4] for both deterministic algorithms \((2n - 1)\) and randomized algorithms \( O(\log n) \). Notice that in this problem, the regret can be divided into a service or local cost which we denote by \( S \) and a move cost \( M \). This has lead to the consideration of the more general problem of \( r \)-unfair competitive ratios or \( r \)-unfair MTS (short UMTS) (see [3], [6]) with \( r \geq 1 \), where the cost of the offline algorithm at each time for index \( i \) is \( c_{i,t}/r \) compared to \( c_{i,t} \) for the online algorithm (i.e. the offline algorithm pays less).
For the r-unfair MTS problem with n servers, Bansal et al. [1] proved the following competitive ratio
\[ \mathbb{E}[\text{cost}_{\text{online}}] \leq [(1 + \epsilon)r + \log n(\frac{1}{\epsilon} + 1)]\text{cost}^* \]  
which is of order \( r + O(\log n) \) and where \( \text{cost}^* \) is the minimum cost incurred by the offline algorithm.

The MTS serves as a meta problem setup for many other problems, which makes it worthwhile to study and understand.

**Example 1 (Paging)** As a reminder, there are \( n \) pages and a cache of size \( k \). In the MTS setting, a state then corresponds to a combination of pages in the cache, which results in a total number of \( \binom{n}{k} \) states and the distance is the \( k \) minus the number of pages that are present in both states. The entries in the cost vectors are either 0 (if the current request is represented in the cache) or \( \infty \) (when otherwise). Note that paging is also equivalent to the \( k \)-server problem with uniform distance 1 and where the pages are the points in space. It has been shown by Koutsoupias and Papadimitriou [5] that the work function (deterministic) algorithm achieves a competitive ratio of 2\( k - 1 \). For randomized algorithms, people even conjecture that \( \log k \) should be achievable.

**Example 2 (Combining online algorithms online)** There are several algorithms \( A_1, \ldots, A_n \) that solve the paging problem all at different costs which the user might not know beforehand. In this case, one can consider the problem of shifting between different algorithms each time a request comes in. We can view this meta problem as an MTS where the states are the algorithms and the cost of moving once is uniformly upper bounded by \( k \), the size of the cache which you might need to clear completely. Using the result for randomized MTS in [4], we can directly obtain
\[ \mathbb{E}[\text{cost}_{\text{online}}] \leq \log \ell(\text{cost}_{A^*} + k) \]
by definition of the competitive ratio and where \( A^* := \arg\min_{i = 1, \ldots, \ell} \text{cost}_{A_i} \). Note that here, the offline “super algorithm” is also static in the sense that it could switch once in the beginning, induce a moving cost of \( k \) and henceforth stay with the best algorithm which yields \( \text{cost}_{A^*} \), i.e. \( \text{OPT} \leq \text{cost}_{A^*} + k \).

Blum et al. [2] managed to show a better upper bound which gets rid of the \( \log \ell \) factor in front of the \( \text{cost}_{A^*} \) and reads
\[ \mathbb{E}[\text{cost}_{\text{online}}] = O((1 + \epsilon)\text{cost}_{A^*} + \frac{k \log n}{\epsilon}). \]
See [2] Section 3.3 for the exact bound and a proof.

### 4.1 Using MTS for expert problems

Before we dive into details and proofs for the MTS in the next lecture, let us have a look how algorithms and guarantees for this problem can help to find regret bounds when allowing experts to shift in the offline problem. We assume here that the number of shifts is fixed by \( k \). One would imagine that this could potentially make the cost of the offline problem drop drastically and thus the regret rise, since the online algorithm stays the same.

The service cost for the MTS 1-step lookahead amounts to \( \sum_t p_t c_t \), whereas in the expert setting, \( c_t \) is not “hitting \( p_t \) but \( p_{t-1} \) since \( p_t \) was updated following knowledge of \( c_t \) which the expert does
not have. The scheme for the expert could therefore just be to follow the MTS algorithm which yields via triangle inequality and Cauchy Schwarz

$$\text{cost}_{\text{exp}} = \sum_{t=1}^{T} p_{t-1} c_t \leq \sum_{t} p_t c_t + \|c_t\|_{\infty} \sum_{t} \|p_t - p_{t-1}\|_1$$

Since in the expert setting the entries of the cost vector are always bounded by 1 and the $\ell_1$ norm is equivalent to the movement cost of the online MTS, we have

$$\text{cost}_{\text{exp}} \leq \text{cost}_{\text{MTS}} = S + M$$

We are now ready to show the following theorem for shifting regrets.

**Theorem 3** For the shifting regret problem, where the offline algorithm is allowed to move $k$ times, the regret reads

$$\mathbb{E}[\text{Regret}_{\text{shift}}] \leq \epsilon L_k + O\left(\frac{k \log n}{\epsilon}\right)$$

(6)

where $S_k$ is the

**Proof:** Takes $r = \frac{\log n}{\epsilon}$. For the online MTS, we have for service and moving costs $S, M$, that

$$S + M \leq (r + \log n)(S^* + M^*) = S^*(1 + \frac{\log n}{r}) + (r + \log n)M^*$$

by the r-UMTS result and where $S^*, M^*$ are the service costs of the offline r-UMTS algorithm. By choosing $r = \frac{\log n}{\epsilon}$ we achieve the best tradeoff between the terms and obtain

$$S + M \leq (1 + \epsilon)S^* + \log n(1 + \frac{1}{\epsilon})M^*$$

What is left to do is to compare the offline MTS cost with the offline $k$-shifting expert cost to obtain regret on the expert. For this purpose note that by optimality of $p^*_t$ we have $\sum_t p^*_t c_t + M^* \leq \frac{1}{\epsilon} p^*_t c_t + k$ where $p^*_t$ corresponds to the best “action” taken by an MTS that is restricted to move only $k$ times between elementary vectors. Therefore we can replace $S^*, M^*$ by $S_k := \sum_t p^*_t c_t$ and $k$ respectively. The loss of the best offline $k$-shifting expert is equal to $L_k := \sum_t p^*_t c_t$ for which we have by triangle inequality that $L_k \geq \sum_t p^*_t c_t - k$ and $L_k \leq S^* + k$. Putting things together we then obtain

$$\sum_t p_{t-1} c_t - p^*_t c_t \leq (1 + \epsilon)S^* + \log n(\frac{1}{\epsilon} + 1)k - S_k + k$$

$$\leq \epsilon S_k + \log n(\frac{1}{\epsilon} + 1)k + k \leq \epsilon L_k + \epsilon k + \log n(\frac{1}{\epsilon} + 2)k,$$

which yields the result (6). \qed

**References**


