> You can try Problems 1 to 3 at home after the lecture; if you are stuck with one, ask during the tutorial for guidance, but attempt it on your own afterwards.
>
> Problem 4 is quite technical: good practice, but don't have time during the tutorial or afterwards, feel free to skip it and only read the solution.
>
> Problems 5 and 6 (Advanced) are good practice, a bit longer and less guided. Discuss them during the tutorial: attempt to fill in the gaps on your own or in groups if you don't have time to finish during the tutorial.

Warm-up

Problem 1. Try various parameters for Theorem 60: plot, for $\beta \in (0,1)$, the bound, when

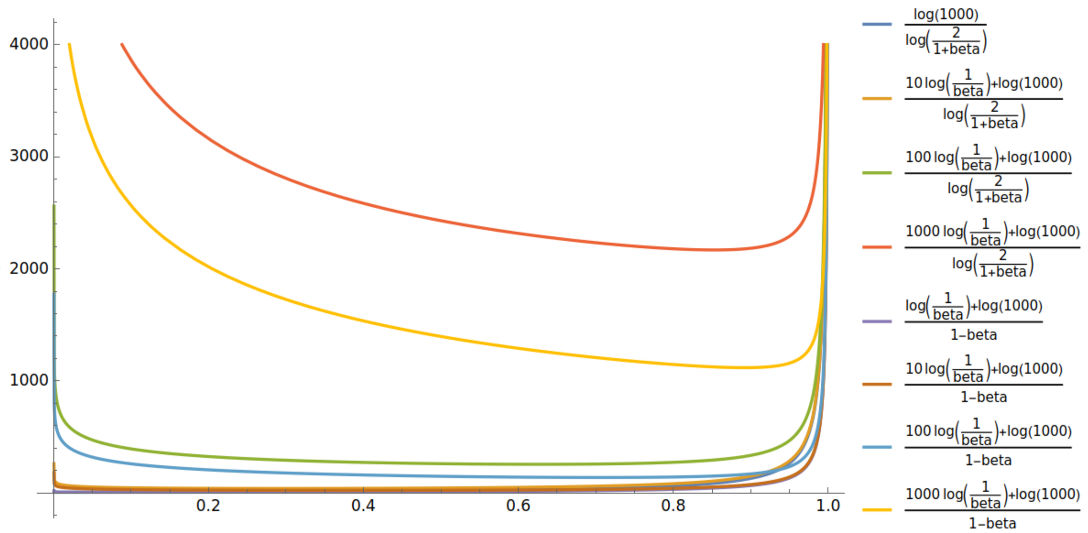
- $C^* = 0, n = 1000$
- $C^* = 10, n = 1000$
- $C^* = 100, n = 1000$
- $C^* = 1000, n = 1000$

Do the same for Theorem 61.

Solution 1. Try it at home! Here in Mathematica:

```
In[3]:= Plot[{(Log[1000])/Log[2/(1+beta)],
    (10*Log[1/beta]+Log[1000])/Log[2/(1+beta)],
    (100*Log[1/beta]+Log[1000])/Log[2/(1+beta)],
    (1000*Log[1/beta]+Log[1000])/Log[2/(1+beta)],
    (Log[1/beta]+Log[1000])/(1-beta),
    (10*Log[1/beta]+Log[1000])/(1-beta),
    (100*Log[1/beta]+Log[1000])/(1-beta),
    (1000*Log[1/beta]+Log[1000])/(1-beta)},
   {beta, 0, 1}, PlotLegends→"Expressions"]
```

Out[3]=



Problem 2. Assume you know both n and (an upper bound on) C^* in advance. How would you set β in the MWU? In the Randomised MWU?

Solution 2. Given explicit values: differentiate the expressions to find the minimum (or find the minimum numerically).

To find a reasonable approximation (up to a factor 2): choose β to balance the two terms in the numerator,

$$C^* \log(1/\beta) = \log n$$

This might not be th exact minimum, but will be within a constant factor, and is much simpler to derive.

Problem 3. Prove Fact 56.3: namely, consider $n = 2$ experts, one predicting always 0 and the other always 1, and consider all 2^T possible sequences $(u_1, \ldots, u_T)$. Show that for any deterministic algorithm A, there exists a sequence on which the algorithm makes T mistakes, and use this to conclude.

Solution 3. Same reasoning and construction as Fact 56.1, but need to also show that $C^*(T)$ is at most $T/2$ (best expert will make at most $T/2$ mistakes). Note that since one expert always outputs 0 and the other always 1, the best expert will make at most $T/2$ mistakes: denote $S_1 = \{t \in [T] : u_t = 0\}$ and $S_2 = [T] \setminus S_1$, expert 1 will be correct $|S_1|$ times and expert 2 correct for $|S_2|$ times –

$$\max\{|S_1|, |S_2|\} = \max\{|S_1|, T - |S_1|\} \geqslant T/2$$

Suppose given algorithm A (you can predict what it will output every step)

1) observes $\{0, 1\}$, predict A's output $\hat{u}_1$. Set $u_1 \leftarrow 1 - \hat{u}_1$.

2) observes $\{u_1, 0, 1\}$, predict A's output $\hat{u}_2$. Set $u_2 \leftarrow 1 - \hat{u}_2$.

3) observes $\{u_1, u_2, 0, 1\}$, predict A's output $\hat{u}_3$. Set $u_3 \leftarrow 1 - \hat{u}_3$.

etc. until T.

So your algorithm will make T mistakes while best expert will make at most $T/2$. The factor 2 is necessary for deterministic algorithms.

Problem solving

Problem 4. ($\star$) Suppose $C^* = C^*(T)$ is known in advance. We will show how to modify the MWU algorithm to achieve

$$C(T) \leq 2C^* + O(\sqrt{C^*(T) \log n} + \log n)$$

a) Argue that, if $C^* \leq \log n$, we are done.

b) Suppose $C^* > \log n$. Show how to achieve the desired bound by setting $\beta = 1 - \varepsilon$, for some suitable $\varepsilon = \varepsilon(C^*, n)$.

c) Conclude.

Solution 4.

a) If $C^* \leqslant 4\log n$, then we set $\beta = \frac{1}{2}$.

$$\frac{C^* \log \frac{1}{\beta} + \log n}{\log \frac{2}{1+\beta}} \leqslant 2.41(C^* + \log n) \leqslant O(\log n).$$

b) We need some approximation facts first. For $\varepsilon \in (0, 0.5)$, we have $\varepsilon \leqslant \log \frac{1}{1-\varepsilon} \leqslant 2\varepsilon$ and

$$\frac{\log \frac{1}{1-\varepsilon}}{\log \frac{1}{1-\varepsilon/2}} = 2 + \frac{\varepsilon}{2} + O(\varepsilon^2) \leqslant 2 + \varepsilon.$$

by series expansion. We restrict $1 - \beta = \varepsilon < \frac{1}{2}$. We proceed with the calculation:

$$\begin{aligned}
C^* \frac{\log \frac{1}{\beta}}{\log \frac{2}{1+\beta}} + \frac{\log n}{\log \frac{2}{1+\beta}} &= C^* \frac{\log \frac{1}{1-\varepsilon}}{\log \frac{1}{1-\varepsilon/2}} + \frac{\log n}{\log \frac{1}{1-\varepsilon/2}} \\
&\leqslant C^* \cdot (2+\varepsilon) + \frac{\log n}{\varepsilon} \\
&= 2C^* + C^*\varepsilon + \frac{\log n}{\varepsilon} \\
&\leqslant 2C^* + C^*\sqrt{\log n / C^*} + \frac{\log n}{\sqrt{\log n / C^*}} \\
&= 2C^* + 2\sqrt{C^* \log n}
\end{aligned}$$

Note that we need $\varepsilon = \sqrt{\log n / C^*} < \frac{1}{2}$, or, equivalently, $\frac{\log n}{C^*} < \frac{1}{4}$. (This motivates in hindsight question a)).

c) Combining the two, we have that

$$\frac{C^* \log \frac{1}{\beta} + \log n}{\log \frac{2}{1+\beta}} \leqslant 2C^* + O\left(\sqrt{C^* \log n} + \log n\right).$$

Problem 5. We again have n experts, each making a binary prediction at each time step. However, we would like to make sure we do well even if the best expert does badly overall, as long as for each "chunk" $I_{t_1,t_2} = \{t_1, t_1+1, \ldots, t_2-1, t_2\}$, we do well compared to the best expert *for this chunk*.

To try and get this, consider the variant of the MWU, where we only penalise an expert by multiplying its weight by $1/2$ *if its current weight is at least* $1/3$ *of the average weight of all experts*.

We want to show that for every $1 \leq t_1 \leq t_2 \leq T$, the maximum number of mistakes $C(t_1, t_2)$ that the algorithm makes over I_{t_1,t_2} is at most $O(C^*(t_1,t_2) + \log n)$, where $C^*(t_1, t_2)$ is the number of mistakes made by the best expert *in that chunk*. (Considering $\beta \in (0,1)$ to be a constant, e.g., $\beta = 1/2$.)

a) Write down the algorithm.

b) Consider any chunk $I = I_{t_1,t_2}$, and let $t \in I$ be a time step where a mistake is made. Let W_t be the total weight at the beginning of step t, and W_G, W_B, W_L be the total weight of (1) experts who made a mistake, (2) experts who did not, and (3) experts who made a mistake but have weight less than $\frac{1}{3} \cdot \frac{W}{n}$. Bound the weight W_{t+1} at the end of step t as a function of W_G, W_B, W_L, β.

c) Bound the weight W_{t+1} at the end of step t as a function of W_t, β: show that

$$W_{t+1} \leq \frac{5+\beta}{6} W_t$$

d) Give a lower bound on the weight w_{i,t_1} of *any* expert i at time t_1 (start of the chunk). Namely, show that

$$w_{i,t_1} \geq \frac{\beta W_{t_1}}{3n}, \qquad 1 \leq i \leq n$$

e) Letting W_{t_1} the total weight at the beginning of the chunk, and W_{t_2} at the end, show that

$$W_{t_2} \geq \beta^{C^*(t_1,t_2)} \cdot \frac{\beta W_{t_1}}{3n}$$

f) Conclude.

Solution 5.

b), c) If we make a mistake at time t, the algorithm will have $W_G > W_B$ and thus

$$W_G \geqslant \frac{1}{2}(W_G + W_B) = \frac{1}{2} W_t.$$

And we have $W_L \leqslant \frac{1}{3} W_t$,

$$\begin{aligned} W_{t+1} &= \beta W_G + W_B + (1-\beta) W_L \\ &= W_G + W_B + (1-\beta)(W_L - W_G) \\ &\leqslant W_t + (1-\beta)\left(\frac{1}{3}W_t - \frac{1}{2}W_t\right) \\ &= W_t - \frac{1-\beta}{6} = \frac{5+\beta}{6}. \end{aligned}$$

d) Denote by $\tilde{W}$ the total weight when expert i got penalised before t_1. Since we only decrease weights as time progress, we have $\tilde{W} \geqslant W_{t_1}$. If i gets penalised and denote its weight at that time $\tilde{w}_i$, it must at the time have

$$\tilde{w}_i \geqslant \frac{\tilde{W}}{3n}.$$

And $w_{i,t_1} = \beta\tilde{w}_i \geqslant \frac{\beta\tilde{W}}{3n} \geqslant \frac{\beta W_{t_1}}{3n}$. If expert i has not been penalised up until t_1, we know that $w_{t_1} = 1$ and therefore $w_{t_1} = 1 \geqslant \frac{\beta W_{t_1}}{3n}$.

e) The best expert makes at most $C^{*(t_1,t_2)}$ mistakes; let j be the index of that expert. Then,

$$w_{j,t_2} \geqslant \beta^{C^{*(t_1,t_2)}} w_{j,t_1} \geqslant \beta^{C^{*(t_1,t_2)}} \frac{\beta W_{t_1}}{3n}.$$

Finally,

$$W_{t_2} = \sum_{i=1}^{n} w_{i,t_2} \geqslant w_{j,t_2} \geqslant \beta^{C^{*(t_1,t_2)}} \frac{\beta W_{t_1}}{3n}$$

Advanced

Problem 6. In the setting of the MWU, we have n experts, each making a binary prediction at each time step. Now, assume that we know that, for every $1 \leq k \leq n$, the k-th expert makes at most k mistakes.

a) What bound can you show on $C(T)$ when running the MWU algorithm with parameter β?

b) What bound can you show on $\mathbb{E}[C(T)]$ when running the Randomised MWU algorithm with parameter β?

Solution 6.

a) We can use an analysis very similar to that given in class for the MWU algorithm. On the one hand, if the algorithm makes C mistakes then after these mistakes the total weight W of all experts will be at most $n\left(\frac{1+\beta}{2}\right)^C$. On the other hand, we now know that the k-th expert makes at most k mistakes, so the lower bound on the total weight we have is $W \geq \beta + \beta^2 + \cdots + \beta^n = \beta \cdot \frac{1-\beta^n}{1-\beta}$. Solving the inequality

$$\beta \cdot \frac{1-\beta^n}{1-\beta} \leq n\left(\frac{1+\beta}{2}\right)^C$$

we obtain

$$C(T) \leq \frac{\log_2 \frac{1-\beta}{\beta(1-\beta^n)} + \log_2 n}{\log_2 \frac{2}{1+\beta}} = \frac{\ln \frac{1-\beta}{\beta(1-\beta^n)} + \ln n}{\ln \frac{2}{1+\beta}}$$

as our bound on the number of mistakes.

b) The analysis is analogous to that of the Randomized MWU algorithm from the lecture. Let F_i denote the fraction of weight at the i-th trial on experts giving an incorrect advice, so that $C = \sum_{i=1}^{T} F_i$. On the one hand, we have that W (the final total weight of all experts) equals $n \prod_{i=1}^{T}(1-(1-\beta)F_i)$. On the other hand, we know that expert k has weight at least β^k, so here again $W \geq \beta + \beta^2 + \cdots + \beta^n = \beta \cdot \frac{1-\beta^n}{1-\beta}$. Putting these together as in the lecture,

$$\mathbb{E}[C(T)] = \sum_{i=1}^{T} F_i \leq \frac{\ln \frac{1-\beta}{\beta(1-\beta^n)} + \ln n}{1-\beta}.$$