IN4320 Machine Learning Assignment 3

Dilan Gecmen 4221168

March 27, 2018

Note: Worked together with Jasper Hemmes.

Exercise 1

Assume we are in the online learning setting with $d = 3$ experts e_1, e_2, and e_3. We evaluate using the mix loss $l_m(p_t, z_t) = -\log\left(\sum_{i=1}^{d} p_t^i e^{-z_t^i}\right)$. Consider the following adversary moves z_t for $t = 1, \ldots, 4$:

	z_1	z_2	z_3	z_4
e_1	0	0	1	0
e_2	0.1	0	0	0.9
e_3	0.2	0.1	0	0

Consider the following strategies:

A in round t choose $p_t = e_{b(t)}$, where the expert $b(t)$ is the expert with the smallest cumulative loss $L_{t-1}^i = \sum_{s=1}^{t-1} z_s^i$ up to time t. In other words $b(t) = \arg\min_i L_{t-1}^i$. For $t = 1$ choose $p_t = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$.

B the strategy of the Aggregating Algorithm (AA). Define cumulative loss of expert i up to time t: $L_t^i = \sum_{s=1}^{t} z_s^i$. AA strategy: $p_t^i = \frac{e^{L_{t-1}^i}}{C_{t-1}}$, where $C_{t-1} = \sum_{j=1}^{d} e^{-L_{t-1}^j}$ ensures $\sum_i p_t^i = 1$ (normalization).

(a) **Strategy A**

t=1 For $t = 1$ we choose $p_1 = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$

t=2 $L_1^1 = z_1^1 = 0$
$L_1^2 = z_1^2 = 0.1$
$L_1^3 = z_1^3 = 0.2$
expert e_1 has the smallest cumulative loss, so $p_2 = (1, 0, 0)$

t=3 $L_2^1 = \sum_{s=1}^{2} z_s^1 = 0 + 0 = 0$
$L_2^2 = \sum_{s=1}^{2} z_s^2 = 0.1 + 0 = 0.1$
$L_2^3 = \sum_{s=1}^{2} z_s^3 = 0.2 + 0.1 = 0.3$
expert e_1 has the smallest cumulative loss, so $p_3 = (1, 0, 0)$

t=4 $L_3^1 = \sum_{s=1}^{3} z_s^1 = 0 + 0 + 1 = 1$
$L_3^2 = \sum_{s=1}^{3} z_s^2 = 0.1 + 0 + 0 = 0.1$
$L_3^3 = \sum_{s=1}^{3} z_s^3 = 0.2 + 0.1 + 0 = 0.3$
expert e_2 has the smallest cumulative loss, so $p_4 = (0, 1, 0)$

Strategy B

t=1 In the first round $p_1 = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ since $L_0^i = 0 \; \forall$ experts e_i. Note that $C_0 = 3$.

t=2 $L_1^1 = z_1^1 = 0$
$L_1^2 = z_1^2 = 0.1$
$L_1^3 = z_1^3 = 0.2$
$C_1 = \sum j = 1^3 e^{-L_1^j} = e^{-L_1^1} + e^{-L_1^2} + e^{-L_1^3} \approx 2.723568171$
$p_2 = \left(\frac{e^{L_1^1}}{C_1}, \frac{e^{L_1^2}}{C_1}, \frac{e^{L_1^3}}{C_1}\right) \approx (0.3672, 0.3322, 0.3006)$

t=3 $L_2^1 = \sum_{s=1}^{2} z_s^1 = 0 + 0 = 0$
$L_2^2 = \sum_{s=1}^{2} z_s^2 = 0.1 + 0 = 0.1$
$L_2^3 = \sum_{s=1}^{2} z_s^3 = 0.2 + 0.1 = 0.3$
$C_2 = \sum j = 1^3 e^{-L_2^j} = e^{-L_2^1} + e^{-L_2^2} + e^{-L_2^3} \approx 2.645655639$
$p_3 = \left(\frac{e^{L_2^1}}{C_2}, \frac{e^{L_2^2}}{C_2}, \frac{e^{L_2^3}}{C_2}\right) \approx (0.3780, 0.3420, 0.2800)$

t=4 $L_3^1 = \sum_{s=1}^{3} z_s^1 = 0 + 0 + 1 = 1$
$L_3^2 = \sum_{s=1}^{3} z_s^2 = 0.1 + 0 + 0 = 0.1$
$L_3^3 = \sum_{s=1}^{3} z_s^3 = 0.2 + 0.1 + 0 = 0.3$
$C_3 = \sum j = 1^3 e^{-L_3^j} = e^{-L_3^1} + e^{-L_3^2} + e^{-L_3^3} \approx 2.0135308$
$p_4 = \left(\frac{e^{L_3^1}}{C_3}, \frac{e^{L_3^2}}{C_3}, \frac{e^{L_3^3}}{C_3}\right) \approx (0.1827, 0.4494, 0.3679)$

(b) The total mix loss after $n = 4$ rounds for both strategies is given by :

$$\sum_{t=1}^{4} l_m(p_t, z_t) = \sum_{t=1}^{4} -\log\left(\sum_{i=1}^{d} p_t^i e^{-z_t^i}\right)$$

For strategy A and B the expert regret after $n = 4$ rounds is given by:

$$\begin{aligned} R_4^E &= \sum_{t=1}^{4} l(p_t, z_t) - \min_i \sum_{t=1}^{4} l(e_i, z_t) \\ &= \sum_{t=1}^{4} l(p_t, z_t) - \min_i \sum_{t=1}^{4} z_t^i \end{aligned}$$

(Note that the expert regret simplifies for the mix loss)

We make use of Matlab to compute the total mix loss and the expert regret of Strategy A and B after $n = 4$ rounds, see Appendix for the matlab code.

Strategy A

Total mix loss: $\sum_{t=1}^{4} l_m(p_t, z_t) = 1.9967$
Expert regret: $R_4^E = 1.6967$

Strategy B

Total mix loss: $\sum_{t=1}^{4} l_m(p_t, z_t) = 0.7089$
Expert regret: $R_4^E = 0.4089$

(c) In this assignment we use the following theorem:

Theorem (AA, Mix Loss) *For any adversary , for any $n > 0$, if l is the mix loss l_m, we have for AA that*

$$R_n^E \leq \log(d)$$

.

We know that $R_n^E = \sum_{i=1}^{n} (l_m(p_t, z_t) - \min_i l(e_i, z_t))$ Let $n = 4$ and $d = 3$. Now the inequality in the previous theorem becomes

$$\sum_{t=1}^{4} l_m(p_t, z_t) - \min_i \sum_{t=1}^{4} l(e_i, z_t) \leq \log(d)$$
$$\Longrightarrow$$
$$\sum_{t=1}^{4} l_m(p_t, z_t) \leq \log(d) + \min_i \sum_{t=1}^{4} l(e_i, z_t) \approx 1.0986 + 0.300 = 1.3986 = C$$

An upperbound on the cumulative loss of AA for $n = 4$ is $\sum_{t=1}^{4} l_m(p_t, z_t) \leq C = 1.3986$

Strategy A Total mix loss: $\sum_{t=1}^{4} l_m(p_t, z_t) = 1.9967$
We see that the value of $C = 1.3986$ is lower than the total mix loss obtained by strategy A at $n = 4$ rounds.

Strategy B Total mix loss: $\sum_{t=1}^{4} l_m(p_t, z_t) = 0.7089$
We computed for $n = 4$ the upperbound C on the total mix loss at 4 rounds for strategy B. As we expect the value $C = 1.3986$ is higher than the total mix loss obtained by strategy B at $n = 4$ rounds.

(d) We need to show that the theoretical guarantee $R_n^E \leq \log(d)$ is tight. To show this, we need to show that for any strategy p_t and for any d we can find an adversary $z_t \in (-\infty, \infty]^d$ such that $R_n^E \geq \log(d)$

We know that for the mix loss the expert regret simplifies

$$\begin{aligned} R_n^E &= \sum_{t=1}^{n} l_m(p_t, z_t) - \min_i \sum_{t=1}^{n} l(e_i, z_t) \\ &= \sum_{t=1}^{n} l_m(p_t, z_t) - \min_i \sum_{t=1}^{n} z_t^i \\ &= \sum_{t=1}^{n} -\log(\sum_{i=1}^{d} p_t^i e^{-z_t^i}) - \min_i \sum_{t=1}^{n} z_t^i \end{aligned}$$

Now we need to find an adversary $z_t \in (-\infty, \infty]^d$ such that $R_n^E \geq \log(d)$

$$R_n^E \geq \log(d)$$

$$\Longleftrightarrow$$

$$R_n^E = \sum_{t=1}^{n} -\log(\sum_{i=1}^{d} p_t^i e^{-z_t^i}) - \min_i \sum_{t=1}^{n} z_t^i \geq \log(d)$$

Now set $j = \min_i \sum_{t=1}^{n}$, so let j be the best expert. As $z_t \in (-\infty, \infty]^d$ we can now set z_t^i as $z_t^i = \infty$ for $j \neq i$, and $z_t^i < \infty$ for $j = i$.

Notice $\min_i \sum_{t=1}^{n} z_t^i < \infty \Rightarrow -\min_i \sum_{t=1}^{n} z_t^i > \infty$. We also know that $\sum_{t=1}^{n} -\log(\sum_{i=1}^{d} p_t^i e^{-z_t^i})$ goes to infinity for $i \neq j$. Hence,

$$R_n^E = \sum_{t=1}^{n} -\log(\sum_{i=1}^{d} p_t^i e^{-z_t^i}) - \min_i \sum_{t=1}^{n} z_t^i \to \infty$$

.

We have showed that for any strategy p_t and for any d we can find an adversary $z_t \in (-\infty, \infty]^d$ such that $R_n^E \geq \log(d)$

Exercise 2

Now we consider the online learning with expert advice setting with d experts and the dot loss $l_d(p_t, z_t) = p_t^T z_t$. Consider the strategies:

A the Exp strategy with learning rate $\eta_A = \sqrt{4\log(d)}$
B the Exp strategy with learning rate $\eta_B = \sqrt{2\log(d)}$
Now define R_n^A as the expert regret of strategy A and R_n^B as the expert regret of strategy B. We do not consider any specific adversary this time.

(a) In this assignment we use the following theorem

Theorem (Exp Strategy, Dot Loss) *For any adversary, $n > 0$, if l is the dot loss l_d, we have for the Exp Strategy that*

$$R_n^E \leq n\frac{\eta}{8} + \frac{\log(d)}{\eta}$$

.

Let $n = 2$

1. First we are going to compute C_2^A for which $R_2^A \leq C_2^A$. Note that A is the Exp strategy with learning rate $\eta_A = \sqrt{4\log(d)}$. Using the previous theorem we have

$$\begin{aligned} R_2^A &\leq 2\frac{\sqrt{4\log(d)}}{8} + \frac{\log(d)}{\sqrt{4\log(d)}} \\ &= \frac{\sqrt{4\log(d)}}{4} + \frac{\log(d)}{\sqrt{4\log(d)}} \\ &= \frac{\sqrt{\log(d)}}{2} + \frac{\log(d)}{2\sqrt{\log(d)}} \\ &= \frac{4\log d}{4\sqrt{\log d}} \\ &= \frac{\log d}{\sqrt{\log d}} \\ &= \sqrt{\log d} = C_2^A \end{aligned}$$

2. Now we are going to compute C_2^B for which $R_2^B \leq C_2^B$. Note that B is the Exp strategy with learning rate $\eta_B = \sqrt{2\log(d)}$. Using the previous theorem we have

$$
\begin{aligned}
R_2^B &\leq 2\frac{\sqrt{2\log(d)}}{8} + \frac{\log(d)}{\sqrt{2\log(d)}} \\
&= \frac{\sqrt{2\log(d)}}{4} + \frac{\log(d)}{\sqrt{2\log(d)}} \\
&= \frac{2\log(d) + 4\log(d)}{4\sqrt{2\log(d)}} \\
&= \frac{6\log(d)}{4\sqrt{2\log(d)}} \\
&= \frac{3}{2\sqrt{2}}\sqrt{\log(d)} = C_2^B
\end{aligned}
$$

Now using 1. and 2. we see that $C_2^A < C_2^B$. Hence C_2^A is the tighter bound.

(b) Let $n = 4$

1. First we are going to compute C_4^A for which $R_4^A \leq C_4^A$. Note that A is the Exp strategy with learning rate $\eta_A = \sqrt{4\log(d)}$. Using the previous theorem we have

$$
\begin{aligned}
R_4^A &\leq 4\frac{\sqrt{4\log(d)}}{8} + \frac{\log(d)}{\sqrt{4\log(d)}} \\
&= \frac{\sqrt{4\log(d)}}{2} + \frac{\log(d)}{\sqrt{4\log(d)}} \\
&= \sqrt{\log(d)} + \frac{\log(d)}{2\sqrt{\log(d)}} \\
&= \frac{2\log(d) + \log(d)}{2\sqrt{\log d}} \\
&= \frac{3}{2}\sqrt{\log(d)} = C_4^A
\end{aligned}
$$

2. Now we are going to compute C_4^B for which $R_4^B < C_4^B$. Note that B is the Exp strategy with learning rate $\eta_B = \sqrt{2\log(d)}$. Using the previous theorem we have

$$
\begin{aligned}
R_4^B &\leq 4\frac{\sqrt{2\log(d)}}{8} + \frac{\log(d)}{\sqrt{2\log(d)}} \\
&= \frac{\sqrt{2\log(d)}}{2} + \frac{\log(d)}{\sqrt{2\log(d)}} \\
&= \frac{2\log(d) + 2\log(d)}{2\sqrt{2\log(d)}} \\
&= \sqrt{2}\sqrt{\log(d)} = C_4^B
\end{aligned}
$$

Now using 1. and 2. we see that $C_4^B < C_4^A$. Hence C_4^B is the tighter bound.

(c) Now assume that strategy B has a tighter bound for some n, meaning $C_n^B < C_n^A$. Does that necessarily mean that for any adversary moves z_t for $t = 1, \ldots, n$, that $R_n^B \leq R_n^A$. You may give an informal argument to explain your answer.

We make us of a counterexample for what was stated in the question. In the previous assignments we showed that for $n = 2$: $C_2^A < C_2^B$ and for $n = 4$: $C_4^B < C_4^A$. From this we can conclude that C_4^A is not a tight bound $\forall\ n$ as for $n = 2$ we showed that C_2^B is the tighter bound. Hence, this means we do not necessarily have that for any adversary moves z_t for $t = 1, \ldots, n$, that $R_n^B \leq R_n^A$.

(d) Generally, how does the strategy p_t change when we increase the learning rate? What do you expect to happen if we set the learning rate extremely high ($\eta = \infty$)?

The strategy p_t is given as

$$
p_t^i = \frac{e^{-\eta L_{t-1}^i}}{\sum_{j=1}^d e^{-\eta L_{t-1}^j}}
$$

When we increase the learning rate η we need to put our trust onto the expert with the smallest cumulative loss $L_{t=1}^i$

Now let $\eta \to \infty$

$$
\begin{aligned}
\lim_{\eta\to\infty} p_t^i &= \lim_{\eta\to\infty} \frac{e^{-\eta L_{t-1}^i}}{\sum_{j=1}^d e^{-\eta L_{t-1}^j}} \\
&= \lim_{\eta\to\infty} \frac{1}{\sum_{j=1}^d e^{-\eta L_{t-1}^j}} \frac{1}{\frac{1}{e^{-\eta L_{t-1}^i}}} \\
&= \lim_{\eta\to\infty} \frac{1}{\frac{\sum_{j=1}^d e^{-\eta L_{t-1}^j}}{e^{-\eta L_{t-1}^i}}} \\
&= \lim_{\eta\to\infty} \frac{1}{e^{\eta L_{t-1}^i} \sum_{j=1}^d e^{-\eta L_{t-1}^j}} \\
&= \lim_{\eta\to\infty} \frac{1}{1 + \sum_{1=j\neq i}^d e^{-\eta (L_{t-1}^j - L_{t-1}^i)}}
\end{aligned}
$$

$$
\lim_{\eta\to\infty} p_t^i = \begin{cases} \frac{1}{d} & \text{if } L_{t-1}^j = L_{t-1}^i \\ 1 & \text{if } L_{t-1}^j > L_{t-1}^i \\ 0 & \text{if } L_{t-1}^j < L_{t-1}^i \end{cases}
$$

The most likely case is that $L_{t-1}^j > L_{t-1}^i$. We put trust in the expert i with the smallest cumulative loss.

Exercise 3

(a) Derive that the optimal learning rate is given by $\eta_n = \frac{R}{G\sqrt{n}}$. This means that η_n minimizes the right hand side of the OCO regret bound for R_n from above. Show for this optimal choice that $R_n \leq RG\sqrt{n}$.

We want to find a global minimum $\eta_n \in (0, \infty)$ that minimizes the right hand side of the OCO regret bound for R_n. We take the first derivative of the right hand side of $R_n \leq \frac{R^2}{2\eta} + \frac{\eta G^2 n}{2}$

$$
\frac{\partial d}{\partial \eta}\left(\frac{R^2}{2\eta} + \frac{\eta G^2 n}{2}\right) = \left(-\frac{R^2}{2\eta^2} + \frac{G^2 n}{2}\right)
$$

Note that the function is increasing if $\left(-\frac{R^2}{2\eta^2} + \frac{G^2 n}{2}\right) > 0$. From the previous inequality it follows that $\eta > \frac{R}{G\sqrt{n}} = \eta_n$. Note that $\eta_n = \frac{R}{G\sqrt{n}}$ a global minimum as the function increases for $\eta > \eta_n$. For this optimal choice $\eta_n = \frac{R}{G\sqrt{n}}$ we have that $R_n \leq \frac{R^2}{2\eta_n} + \frac{\eta_n G^2 n}{2} = RG\sqrt{n}$.

(b) Show that if we set $\eta_t = \frac{R}{G\sqrt{t}}$ and $\frac{1}{\eta_0} = 0$ we can bound for all $n \geq 1$ the OCO regret by $R_n \leq \frac{3}{2}GR\sqrt{n}$. Explain every step you do as detailed as possible.

We follow the proof given in the lecture notes! Inequality of convex and differentiable functions is given as

$$l(a_t, z_t) - l(a, z_t) \leq \bigtriangledown_{a_t}^T l(a_t, z_t)(a_t - a)$$

Holds $\forall\ t$, so we can sum over $n+1$ rounds:

$$R_n^a := \sum_{t=0}^{n} l(a_t, z_t) - l(a, z_t) \leq \sum_{t=0}^{n} \bigtriangledown_{a_t}^T l(a_t, z_t)(a_t - a)$$

The left hand side side is the OCO regret. Note that we compare to a and not to the best a. So, we can continue now bounding the right hand side. First look at the following inequality.

$$||a_{t+1} - a||^2 \leq ||a_t - a||^2 - 2\eta \bigtriangledown_{a_t}^T l(a_t, z_t)(a_t - a) + ||\eta \bigtriangledown_{a_t} l(a_t, z_t)||^2$$

The first equality follows from the definition of the OGD. The inequality follows from the Pythagoras Theorem. The last equality is just expanding the squared norm. This can be rewritten as follows

$$2\eta \bigtriangledown_{a_t}^T l(a_t, z_t)(a_t - a) \leq ||a_t - a||^2 - ||a_{t+1} - a||^2 + ||\eta \bigtriangledown_{a_t} l(a_t, z_t)||^2$$

$$\Longrightarrow$$

$$\bigtriangledown_{a_t}^T l(a_t, z_t)(a_t - a) \leq \frac{1}{2\eta}(||a_t - a||^2 - ||a_{t+1} - a||^2) + \frac{\eta}{2}||\bigtriangledown_{a_t} l(a_t, z_t)||^2$$

$||\bigtriangledown_a l(a, z)|| \leq G$. Sum over $t = 0, \ldots, n$ gives us

$$\sum_{t=0}^{n} \bigtriangledown_{a_t}^T l(a_t, z_t)(a_t - a) \leq \sum_{t=0}^{n} \frac{1}{2\eta}(||a_t - a||^2 - ||a_{t+1} - a||^2) + \frac{\eta G^2}{2}$$

Exercise 4

a At time t you invest a part p_t of your wealth W_t in an asset with value x_t. The assets price changes with ratio r_t which is the ratio of the price at t and $t+1$. The wealth at $t+1$ is then

$$W_{t+1} = \sum_{i=1}^{d} r_t^i p_t^i W_t$$

b We need to show that with the adversary moves $z_t^i = -\log r_t^i$ that

$$-\log\left(\frac{W_{n+1}}{W_1}\right) = \sum_{t=1}^{n} -\log\left(\sum_{i=1}^{d} p_t^i e^{-z_t^i}\right)$$

We rewrite the equation

$$W_{t+1} = \sum_{i=1}^{d} r_t^i p_t^i W_t$$

we get

$$W_{n+1} = \sum_{i=1}^{d}\prod_{t=1}^{n} r_t^i p_t^i W_1$$
$$\Longrightarrow$$
$$-\log\left(\frac{W_{n+1}}{W_1}\right) = \sum_{t=1}^{n} -\log\left(\sum_{i=1}^{d} r_t^i p_t^i\right)$$

Now we make use of $z_t^i = -\log\left(r_t^i\right)$ and we get what was asked in this question

$$-\log\left(\frac{W_{n+1}}{W_1}\right) = \sum_{t=1}^{n} -\log\left(\sum_{i=1}^{d} p_t^i e^{-z_t^i}\right)$$

c1 I inspected the code!

c2 See appendix E. Adversary moves are z_t in appendix E. Now we have the following total losses of the experts
$L_{BCH} = -1.1530$
$L_{BTC} = -1.3649$
$L_{ETH} = -1.3249$
$L_{LTC} = -1.5774$
$L_{XRP} = -1.6543$

c3 See appendix E, final expert regret is 0.2232

c4 The expert losses are between -1.1530 and -1.6543, and can be seen in question c2. The total loss is -1.4311 which is in between the expert losses. The final expert regret of the AA algorithm is 0.2232, which is much lower than the guaranteed regret: $\log(5) = 1.6094$. Because the final expert regret is much lower than the theoretical bound we can conclude that the adversary is not generating difficult data.

c5 The strategy of AA and the value of the coins can be seen in figure 2. We see that when the value of a coin increases, the portion of wealth invested in that coin increases. Thus the strategy is to invest in coins that are performing well (value is increasing)

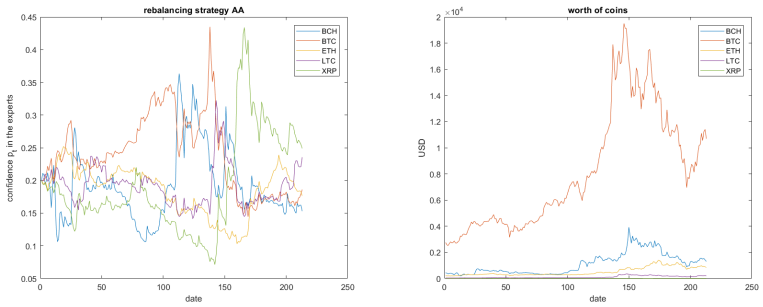

Figure 1: Strategy of AA and the worth of coins versus time

c6 Using the equation from Exercise 4a the total gain is calculated. After 213 days our wealth would be 4.1833 times the wealth invested at day 1.

d1 OCO regret is defined to consider all possible actions in action space A, expert regret only considers p_t from $\{e_1, e_2, e_3, e_4, e_5\}$.

d2

$$\begin{aligned}
\bigtriangledown_a l_m(a, z_t) &= \frac{\partial}{\partial a}\left(-\log(\sum_{i=1}^{d} a_t^i e^{z_t^i})\right) \\
&= \frac{\partial}{\partial a}\left(-\log(\sum_{i=1}^{d} a_t^i r_t^i)\right) \\
&= (-\frac{1}{\sum_{i=1}^{d} a_t^i r_t^i}) r_t \\
&= -\frac{1}{a_t^T r_t} r_t
\end{aligned}$$

d3 See appendix G

d4 See appendix G

d5 We have a good bound as for R: $||a||_2 = 1$ for all $a \in A$. Now for G the gradient is given as $\bigtriangledown_a l_m(a, z_t) = -\frac{r_t}{a^T r_t}$. We need to satisfy $|| \bigtriangledown_a l_m(a, z_t)|| \leq G$ should be satisfied. This is exactly what is done in the code.

d6 We can see the resulting graph in Figure 2. We see that when the value of a coin increases, the portion of wealth invested in that coin increases, very much the same as in the AA algorithm. However there is a large difference. In the AA algorithm the investment actions range from 0.0714 to 0.4349 while in the OGD this range is much smaller, namely 0.1944 to 0.2162. Using the OGD algorithm the invested wealth would be worth 6.1428 times more. Thus using this data the OGD algorithm outperforms the AA algorithm.

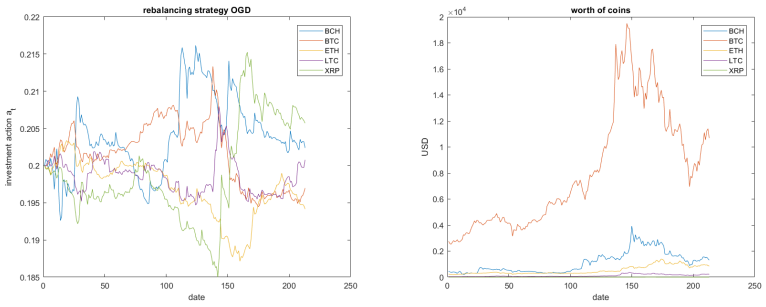

Figure 2: Strategy of AA and the worth of coins versus time

d7 See appendix G. To calculate the loss we used loss_fixed_action.m, to optimize for the best fixed action we use fmincon. The minimizer a is found to be; $a = [0.2611, 0.0, 0.0, 0.3075, 0.4314]$. This gives an OCO regret of 0.1756, notice that this is less than the 0.2232 regret of the AA algorithm.

d8 The loss of the best fixed action with a value of -1.9909 is slightly smaller than the loss of OGD with a value of -1.8153. The optimal bound of the OCO regret is $RG\sqrt{n} = 65.7803$. Because the OCO regret of the OGD algorithm is much smaller than the bound with a value of 0.1756 we conclude that the adversary is not generating difficult data.

d9 When $r_t^i = 0$ the algorithm would not yield an answer. This because then $z_t^i = -\log(r_t^i) = -\log(0)$ which is not defined, thus the algorithm would fail.

A

```
clear all
clc

z=[0 0 1 0; 0.1 0 0 0.9; 0.2 0.1 0 0];
p= zeros(3:4);
p(:,1)= [1/3; 1/3; 1/3];

cm = Cum_Loss(z)
ml = zeros(1,3);

for i=1:4
    ml(i) = mixLoss(p(:,i),z(:,i));
    [value, index] = min(cm(:,i));
    p(index,i+1)= 1
end

Total_Mix_Loss = sum(ml)


Expert_Regret = Total_Mix_Loss - min(cm(:,4))
```

B

```
clear all
clc

z=[0 0 1 0; 0.1 0 0 0.9; 0.2 0.1 0 0];
p= zeros(3:4);
p(:,1)= [1/3; 1/3; 1/3];

cm = Cum_Loss(z)
ml = zeros(1,3);

C=zeros(1,3)

for i=1:4
    C(:,i)=sum(exp(-cm(:,i)))
    p(:,i+1)= exp(-cm(:,i))/C(:,i)
    ml(i) = mixLoss(p(:,i),z(:,i));
end

Total_Mix_Loss=sum(ml)
Expert_Regret = Total_Mix_Loss - min(cm(:,4))
```

C

```
function closs = Cum_Loss(z)
closs = zeros(3,4);
closs(:,1)=z(:,1);
closs(:,2)=z(:,1)+z(:,2);
closs(:,3)=z(:,1)+z(:,2)+z(:,3);
closs(:,4)=z(:,1)+z(:,2)+z(:,3)+z(:,4);
end
```

D

```
function mloss = Mix_Loss(p,z)
%a.*b elementwise multiplication
%sum takes the sum of each column
mloss=-log(sum(p.*exp(-z)))
end
```

E

```
% Exercise: Aggregating Algorithm (AA)

clear all
load coin_data;

d = 5;
n = 213;

% compute adversary movez z_t
z_t = -log(r);

% compute strategy p_t (see slides)

p = nan(size(z_t));
z_t_c = cumsum(z_t);

for ii = 1:length(z_t)
    p(ii,:) = exp(-z_t_c(ii,:))/sum(exp(-z_t_c(ii,:)));
end

p = [0.2 0.2 0.2 0.2 0.2; p(1:end-1,:)];

% compute loss of strategy p_t
```

```
l_mix = nan(1,length(z_t));

for ii = 1:length(z_t)
    l_mix(ii) = -log(sum(p(ii,:).*exp(-z_t(ii,:))));
end
l_mix = l_mix';

% compute losses of experts

l_exp = z_t;

% compute regret

r_exp = cumsum(l_mix) - min(z_t_c, [], 2);

% compute total gain of investing with strategy p_t

gain = cumprod(dot(p,r,2));

%% plot of the strategy p and the coin data

% if you store the strategy in the matrix p (size n * d)
% this piece of code will visualize your strategy

figure
subplot(1,2,1);
plot(p)
legend(symbols_str)
title('rebalancing strategy AA')
xlabel('date')
ylabel('confidence p_t in the experts')

subplot(1,2,2);
plot(s)
legend(symbols_str)
title('worth of coins')
xlabel('date')
ylabel('USD')
```

F

```
function [l, g] = mix_loss(a, r)
%   [l, g] = MIX_LOSS(a, r)
%   Input:
%       a (column vector), the investment strategy (note a should
%       be normalized so that sum(a) = 1)
```

```
%       r (column vector), stock changes on day t (compared with t-1)
%   Output:
%       l (number), the mix loss
%       g (column vector), the gradient of the mix loss (with respect
%       to action a)

    l = -log(a'*r);
    g = -r/(a'*r);


end
```

G

```
% Exercise: Online Gradient Descent (OGD)

clear all;
load coin_data;

a_init = [0.2, 0.2, 0.2, 0.2, 0.2]'; % initial action

n = 213; % is the number of days
d = 5; % number of coins

% we provide you with values R and G.
alpha = sqrt(max(sum(r.^2,2)));
epsilon = min(min(r));
G = alpha/epsilon;
R = 1;

% set eta:
eta = R/G/sqrt(n);

a = a_init; % initialize action. a is always a column vector

L = nan(n,1); % keep track of all incurred losses
A = nan(d,n); % keep track of all our previous actions

for t = 1:n


    % we play action a
    [l,g] = mix_loss(a,r(t,:)'); % incur loss l, compute gradient g

    A(:,t) = a; % store played action
    L(t) = l; % store incurred loss
```

```
    % update our action, make sure it is a column vector
    a = a - eta*g;

    % after the update, the action might not be anymore in the action
    % set A (for example, we may not have sum(a) = 1 anymore).
    % therefore we should always project action back to the action set:
    a = project_to_simplex(a')'; % project back (a = \Pi_A(w) from lecture)

end

% compute total loss
total_loss = sum(L)

% compute total gain in wealth
gain = cumprod(dot(A',r,2));

% compute best fixed strategy (you may make use of
%loss_fixed_action.m and optimization toolbox if needed)

fun = @(a_init)loss_fixed_action(a_init);
[a_fixed, l_fixed, exitFlag] = fmincon(fun, a_init,
[], [], [] , [], [0 0 0 0 0], [1 1 1 1 1], @confuneq);

% [loss, grad] = loss_fixed_action(a_fix);
% compute regret
%%% your code here %%%
regret = total_loss - l_fixed;

%% plot of the strategy A and the coin data

% if you store the strategy in the matrix A (size d * n)
% this piece of code will visualize your strategy

figure
subplot(1,2,1);
plot(A')
legend(symbols_str)
title('rebalancing strategy OGD')
xlabel('date')
ylabel('investment action a_t')

subplot(1,2,2);
plot(s)
legend(symbols_str)
title('worth of coins')
```

```
xlabel('date')
ylabel('USD')clear all
```