ECE 8803: Online Decision Making in Machine Learning **Fall 2021**

Lecture 8: September 20

Lecturer: Vidya Muthukumar

8.1. Recap: Online linear optimization

Last lecture, we bridged prediction and decision-making, and introduced the online linear optimization (OLO) paradigm. In this setting, our decision at time step t is given by the vector $\boldsymbol{w}_t \in \mathcal{B}$ (where $\mathcal{B} \subset \mathbb{R}^d$ is the decision set), and we incur a loss given by $\langle \boldsymbol{w}_t, \boldsymbol{\ell}_t \rangle$, where $\boldsymbol{\ell}_t \in \mathbb{R}^d$ denotes the loss vector at time step t. We then define the algorithm's performance as $H_T := \sum_{t=1}^T \langle \boldsymbol{w}_t, \boldsymbol{\ell}_t \rangle$. Our objective is to minimize the regret with respect to the best fixed decision in hindsight, which is defined similarly as in the case of sequence prediction:

$$R_T := H_T - L_T^*, \text{ where} \tag{8.1}$$

$$L_T^* := \min_{\boldsymbol{w} \in \mathcal{B}} \sum_{t=1}^T \langle \boldsymbol{w}, \boldsymbol{\ell}_t \rangle. \tag{8.2}$$

We also define as shorthand the notation

$$\ell_t(\boldsymbol{w}) := \langle \boldsymbol{w}, \boldsymbol{\ell}_t \rangle \text{ and}$$

$$L_t(\boldsymbol{w}) := \sum_{s=1}^t \ell_s(\boldsymbol{w}).$$

This notation will come in handy later in the lecture.

We considered the example of a stock trading problem (with d types of stock), in which the decision set $\boldsymbol{w}_t$ represented how many units of each stock we wanted to buy/sell, and the loss vector $\boldsymbol{\ell}_t$ represented how much the value of each stock depreciated on day t. We considered the decision set $\mathcal{B}$ to be an ℓ_2-ball of the form $\mathcal{B} := \{\boldsymbol{w} : ||\boldsymbol{w}||_2 \leq D\}$. We also considered the loss vectors to be bounded, i.e. $||\boldsymbol{\ell}_t||_2 \leq G$.

We introduced two extreme algorithms:

- The Follow-the-Leader (FTL) algorithm, which simply selects $\boldsymbol{w}_t = \arg\min_{\boldsymbol{w} \in \mathcal{B}} L_{t-1}(\boldsymbol{w})$. This is the analog of FTL that was used in binary sequence prediction. We showed through examples that FTL would incur very low (in fact, constant) regret on a stock whose value drifts (in either direction) over time; but would incur linear regret if the stock's value fluctuated unpredictably. Thus, FTL effectively exploits meaningful information in the data but can be very unstable in the worst case, and constitutes a "high-risk" decision.

- The "zero" algorithm, which simply selects $\boldsymbol{w}_t = \mathbf{0}$ on all rounds. This is the analog of pure guessing in binary sequence prediction. We saw that the zero algorithm always obtains a loss $H_T = 0$ regardless of the evolution of the data. Thus, it would incur very low (in fact, zero) regret on a stock whose value fluctuates unpredictably, but linear regret if a stock's value drifts. This is because the zero algorithm is extremely stable to fluctuations in the data, but does not at all exploit a meaningful pattern in it, and it constitutes a "low-risk" decision.

Through these examples, we motivated the simple algorithm Follow-the-Regularized-Leader, which trades off high risk and low risk elements in the decision making. We recap its definition here.

Definition 1 *The Follow-the-Regularized-Leader (FTRL) algorithm chooses*

$$\boldsymbol{w}_t := \arg\min_{\boldsymbol{w}\in\mathcal{B}} \left[L_{t-1}(\boldsymbol{w}) + \frac{1}{\eta}||\boldsymbol{w}||_2^2 \right],$$

*where $\eta > 0$ is a **learning rate** parameter, analogous to the one that we picked in binary prediction.*

Notice that η naturally measures the amount of risk we take: a higher value of η leads to less regularization, and more weight placed on the past observations (therefore, higher risk), while a lower value of η leads to more regularization, and less weight placed on the past observations (therefore, lower risk). Two extreme cases are below:

- If $\eta \to \infty$, FTRL reduces to the highest-risk option, FTL.
- If $\eta \to 0$, FTRL reduces to the lowest-risk option of picking the most stable vector $\boldsymbol{w}_t = \mathbf{0}$.

From what we have seen in class so far, we should expect that we can trade off the high-risk and low-risk elements and achieve a sublinear regret guarantee using the FTRL algorithm. Next lecture, we will see this explicitly. Moreover, we will see that for the choice of decision set and constraint on loss functions defined here (both constrained ℓ_2-ball), the FTRL algorithm has a particularly nice closed-form expression that connects to fundamental algorithms used in convex optimization.

8.2. FTRL achieves low regret: Proof

We now prove that FTRL, with a learning rate set to $\eta = \Theta\left(\frac{1}{\sqrt{T}}\right)$, achieves the optimal regret guarantee of $\mathcal{O}(\sqrt{T})$ by effectively trading off stability and ability to exploit meaningful information in data. Recall that we define our decision set to be $\mathcal{B} := \{\boldsymbol{w} : ||\boldsymbol{w}||_2 \leq D\}$ and assume the loss vectors to be bounded, i.e. $||\boldsymbol{\ell}_t||_2 \leq G$.

Our proof is a two-part proof, provided below, and heavily mirrors the proof structure of FTPL (it is, in fact, much simpler and shorter). We denote $R(\boldsymbol{w}) := \frac{1}{\eta}||\boldsymbol{w}||_2^2$ as shorthand. We also denote $\widetilde{\boldsymbol{w}}^*$ as the best regularized decision-maker in hindsight, i.e. $\widetilde{\boldsymbol{w}}^* := \arg\min_{\boldsymbol{w}\in\mathcal{B}} [R(\boldsymbol{w}) + L_T(\boldsymbol{w})]$.

As in the proof of FTPL that we did a few lectures ago, we can now decompose the regret of FTRL as below:

$$R_T = \underbrace{H_T - L_T(\widetilde{\boldsymbol{w}}^*)}_{R_T^A} + \underbrace{L_T(\widetilde{\boldsymbol{w}}^*) - L_T^*}_{R_T^B}. \tag{8.3}$$

Above, R_T^B represents the *approximation-theoretic error*, i.e. the extent to which regularization makes us deviate from the true outcome. We should expect this term to *increase* as we add more regularization, i.e. decrease η. On the other hand, R_T^A represents the *estimation error*, i.e. regret with respect to the best regularized decision in hindsight. We should expect this term to *decrease* as we add more regularization, i.e. decrease η, as we are making the update more stable. We now characterize these terms below.

8.2.1 Bounding R_T^B

We will now show that $R_T^B \leq \frac{D^2}{\eta}$ in a manner very similar to how we bounded the corresponding term for FTPL. This scaling is intuitive for the following reason: the larger the value of η, the less we regularize, and the closer the overall loss of the best-regularized-decision in hindsight will be to that of the best decision-in-hindsight.

We denote the best-decision-in-hindsight by $\boldsymbol{w}^* := \arg\min_{\boldsymbol{w}\in\mathcal{B}} L_T(\boldsymbol{w})$, and recall that $\widetilde{\boldsymbol{w}}^*$ is the best decision in hindsight *under the regularized losses*. Thus, we have $R_T^B = L_T(\widetilde{\boldsymbol{w}}^*) - L_T(\boldsymbol{w}^*)$. We note that, by definition, we have $L_T(\widetilde{\boldsymbol{w}}^*) + R(\widetilde{\boldsymbol{w}}^*) \leq L_T(\boldsymbol{w}^*) + R(\boldsymbol{w}^*)$. This tells us that

$$\begin{aligned} L_T(\widetilde{\boldsymbol{w}}^*) - L_T(\boldsymbol{w}^*) &\leq R(\boldsymbol{w}^*) - R(\widetilde{\boldsymbol{w}}^*) \\ &\leq \max_{\boldsymbol{w}\in\mathcal{B}} R(\boldsymbol{w}) \leq \frac{D^2}{\eta}, \end{aligned}$$

where the last step uses the definition of the regularizer $R(\boldsymbol{w}) = \frac{||\boldsymbol{w}||^2}{\eta}$, and the fact that our decision set constrains $||\boldsymbol{w}||_2^2 \leq D^2$. This completes the proof.

8.2.2 Bounding R_T^A

Now, we show that $R_T^A \leq \eta G^2 T$, where we recall that we bound the loss vectors as $||\boldsymbol{\ell}_t||_2 \leq G$. Like in the case of FTL, the term R_T^A measures the extent of stability of the algorithm, and so will be small for a small value of η (as smaller η implies that we regularize more).

Our starting point is to note that FTRL is effectively FTL on the *regularized* loss sequence given by

$$\{R(\boldsymbol{w}), \ell_1(\boldsymbol{w}), \ell_2(\boldsymbol{w}), \ldots, \ell_T(\boldsymbol{w})\}. \tag{8.4}$$

Recall that, when we studied FTPL, we provided an upper bound on the regret of FTL in terms of the number of leader changes. This type of upper bound made sense in a discrete prediction setting; here, we are making decisions in a continuous set. Accordingly, we provide an upper bound on the regret of FTL that measures to what extent the leader changes in a more continuous manner.

Lemma 2 *The regret of FTL on a loss sequence* $\{\ell_t(\boldsymbol{w})\}_{t=1}^T$ *is upper bounded*

$$R_T \leq \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}_{t+1}).$$

The proof of Lemma 2 is provided in the appendix of this lecture note and is optional to read. It is worth noting the intuition for this lemma, however: if the decision barely changed at all from time t to time $t+1$, we would have $\boldsymbol{w}_{t+1} \approx \boldsymbol{w}_t$, and so $\ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}_{t+1}) \approx 0$. Thus, just like in the bound we provided for FTL in the case of binary prediction, this bound also measures the extent of leader change.

We apply this bound to the sequence given in Equation (8.4) to get:

$$R_T^A = H_T - L_T(\widetilde{\boldsymbol{w}}^*) \leq \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}_{t+1}), \tag{8.5}$$

where $\boldsymbol{w}_t$ is the decision that is taken by **FTRL** at every round.

Now, we are going to use the special structure of the FTRL update to make a statement about the quantity $\ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}_{t+1})$; in particular, we will show that it is small if η is small as a consequence of the decision vector not changing much. To do this, we will evaluate the FTRL update in closed-form. Recall that the FTRL update is given by

$$\begin{aligned} \boldsymbol{w}_t &:= \arg\min_{\boldsymbol{w} \in \mathcal{B}} F_t(\boldsymbol{w}) \text{ where} \\ F_t(\boldsymbol{w}) &:= \left[L_{t-1}(\boldsymbol{w}) + \frac{1}{\eta} ||\boldsymbol{w}||_2^2 \right]. \end{aligned}$$

Let us ignore the constraint $\boldsymbol{w} \in \mathcal{B}$ for a moment, and examine what the resulting decision vectors would look like. A calculus exercise gives us

$$\nabla_{\boldsymbol{w}} F_t(\boldsymbol{w}) := \left(\sum_{s=1}^{t-1} \boldsymbol{\ell}_s \right) + \frac{1}{\eta} \boldsymbol{w},$$

Also, note that the objective function $F_t(\boldsymbol{w})$ is clearly convex in $\boldsymbol{w}$, and so the *unconstrained* minimum of $F_t(\boldsymbol{w})$, which we denote by $\boldsymbol{z}_t$, is given by

$$\begin{aligned} \boldsymbol{z}_t &= -\eta \left(\sum_{s=1}^{t-1} \boldsymbol{\ell}_s \right) \\ \implies \boldsymbol{z}_{t+1} &= -\eta \left(\sum_{s=1}^{t} \boldsymbol{\ell}_s \right) \\ &= \boldsymbol{z}_t - \eta \boldsymbol{\ell}_t. \end{aligned}$$

Putting these together, we actually get

$$\boldsymbol{z}_{t+1} = \boldsymbol{z}_t - \eta \boldsymbol{\ell}_t, \tag{8.6}$$

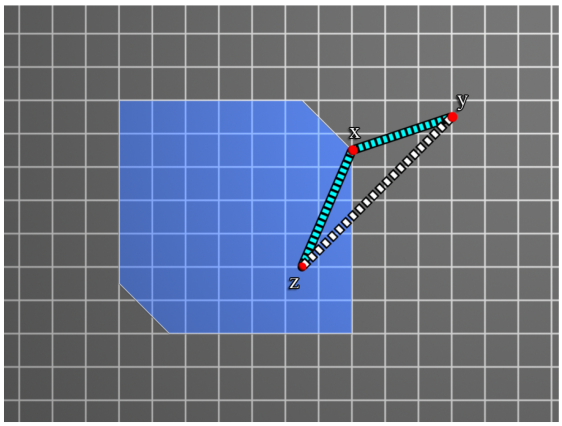


Figure 8.1: Geometric depiction of a projection of a vector $\boldsymbol{y}$ onto a convex set $\mathcal{B}$ (marked in blue). The projection is denoted by $\boldsymbol{x}$. The definition of projection implies that for any vector $\boldsymbol{z} \in \mathcal{B}$, we have $||\boldsymbol{x} - \boldsymbol{z}||_2 \leq ||\boldsymbol{x} - \boldsymbol{y}||_2$; projection decreases the distance. This fact is applied twice below, to show that $||\boldsymbol{w}_t - \boldsymbol{w}_{t+1}||_2 \leq ||\boldsymbol{z}_t - \boldsymbol{z}_{t+1}||_2$.

which heavily resembles a step of gradient descent in linear optimization! For this reason, the FTRL algorithm is synonymously described as the *online-gradient-descent*[1] (OGD) algorithm. We will discuss this connection more in the next lecture.

Of course, in general the unconstrained minimum $\boldsymbol{z}_t$ need not lie within the constrained to decision set $\mathcal{B}$. Thus, to get our actual minimum $\boldsymbol{w}_t$, we simply *project* the unconstrained optimum $\boldsymbol{z}_t$ onto the constrained decision set $\mathcal{B}$. (This follows because the decision set $\mathcal{B}$ is also convex.) See Figure 8.1 for a geometric depiction of the projection step.

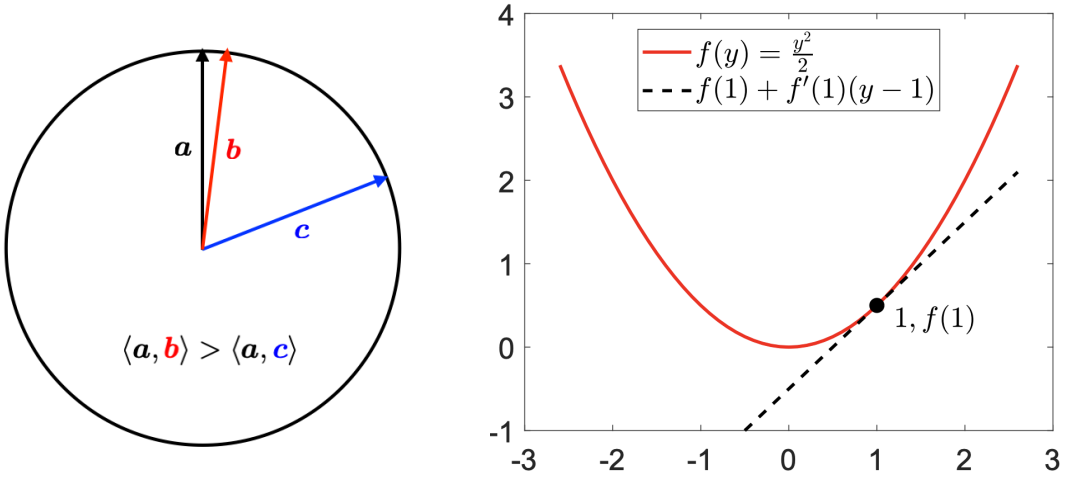
All in all, Equation (8.6) tells us exactly why a small η should lead us to a very small leader change: the decision vector barely moves! To see this concretely, notice that

$$\begin{aligned}
\ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}_{t+1}) &= \langle \boldsymbol{\ell}_t, \boldsymbol{w}_t - \boldsymbol{w}_{t+1} \rangle \\
&\leq ||\boldsymbol{\ell}_t||_2 ||\boldsymbol{w}_t - \boldsymbol{w}_{t+1}||_2 \\
&\leq ||\boldsymbol{\ell}_t||_2 ||\boldsymbol{z}_t - \boldsymbol{z}_{t+1}||_2 \\
&\leq \eta ||\boldsymbol{\ell}_t||_2^2 \\
&\leq \eta G^2.
\end{aligned}$$

Let us unpack these inequalities one by one:

- The first inequality uses the Cauchy-Schwarz inequality on vectors: $\langle \boldsymbol{a}, \boldsymbol{b} \rangle \leq ||\boldsymbol{a}||_2 ||\boldsymbol{b}||_2$ for any two vectors $\boldsymbol{a}, \boldsymbol{b}$. The Cauchy-Schwarz inequality simply says that the inner product between two vectors is maximized when they are exactly aligned; see Figure 8.2a for a pictorial depiction.

1. Strictly speaking, FTRL is a *lazy* variant of the online gradient descent algorithm. The more common form of OGD is given by $\boldsymbol{z}_{t+1} = \boldsymbol{w}_t - \eta \boldsymbol{\ell}_t$, and can also be analyzed via convex analysis techniques. We will not touch upon this in class.



(a) Cauchy-Schwarz inequality in 2 dimensions.

(b) First-order approximation to convex function in 1 dimension.

Figure 8.2: Geometric figures that depict the Cauchy-Schwarz inequality in 2 dimensions and the first-order approximation to a convex function in 1 dimension.

- The second inequality uses the fact that projecting into a constrained convex set only *decreases* the distance between two vectors. See Figure 8.1 for a pictorial depiction of this property and an explanation.
- The third inequality uses the nice form in Equation (8.6), which formally shows us that a smaller learning rate leads to a very small update in the decision due to large regularization.
- Finally, the fourth inequality uses the upper bound on $||\boldsymbol{\ell}_t||_2$ that we have assumed.

In summary, we have shown that $R_T^A \leq \eta G^2 T$. Putting these bounds together, we get

$$R_T \leq \eta G^2 T + \frac{D^2}{\eta} = \mathcal{O}(DG\sqrt{T})$$

where the last inequality is obtained as a consequence of setting $\eta := \frac{D}{G}\sqrt{T}$. This completes our proof.

8.3. Extension to convex loss functions

While the proof that we provided for FTRL was for the online linear optimization (OLO) setting, it turns out that this idea can easily be extended to the much more general *online convex optimization* (OCO) setting, where the loss functions $\ell_t(\boldsymbol{w})$ can be any convex function in $\boldsymbol{w}$. To see this, we define the following variant of FTRL to work with convex loss functions.

Definition 3 *We modify FTRL to work in the following sequence of steps for convex loss functions. For each $t = 1, \ldots, T$, we perform the following sequence of steps:*

1. *The algorithm selects $\boldsymbol{w}_t$, and incurs loss $\ell_t(\boldsymbol{w}_t)$.*
2. *We define the* first-order *linear approximation to the loss function around $\boldsymbol{w}_t$ as:*

$$\widehat{\ell}_t(\boldsymbol{w}) := \ell_t(\boldsymbol{w}_t) + \langle \nabla \ell_t(\boldsymbol{w}_t), \boldsymbol{w} - \boldsymbol{w}_t \rangle.$$

   See Figure 8.2b for a depiction of this first-order approximation.
3. *We select the decision at step $t+1$, denoted by $\boldsymbol{w}_{t+1}$, as FTRL on the linearized loss functions $\{\widehat{\ell}_s(\cdot)\}_{s=1}^t$.*

It turns out that we can show that we continue to have $R_T = \mathcal{O}(DG\sqrt{T})$, where G now represents a *norm* constraint on the gradient of the loss functions, i.e.

$$||\nabla \ell_t(\boldsymbol{w})||_2 \leq G \text{ for all } t \text{ and all } \boldsymbol{w}. \tag{8.7}$$

Note that Equation (8.7) intuitively means that the loss functions need to be *smooth*, i.e. they cannot arbitrarily spike with a change in the decision.

We will provide a proof sketch of this bound in the next lecture, and show where and why convexity is important. We will also see an intuitive connection to stochastic optimization algorithms, and show that online learning techniques provide a particularly clean understanding of what they are doing.

8.4. Additional references

- The proof structure that we have used for OLO is inspired in part by Lectures 3 and 4 of the following course: `https://courses.cs.washington.edu/courses/cse599s/14sp/scribes/lecture2/scribeNote.pdf`. A much more general version of this statement holds (for arbitrary regularizers and constraints on decisions), and is stated and proved in Theorem 5.2 of Hazan (2016). This is advanced reading (assumes convex analysis background), but worthwhile if you are interested.
- Online convex optimization (OCO) is a significantly more general framework than OLO, as it allows us to consider general convex losses. One of the early applications of OCO was to the portfolio management problem Hazan et al. (2007), that we discussed at a high level last lecture. More recently, the OCO paradigm has been applied to the problem of *model-predictive-control* with unpredictable disturbances in the environment; see Agarwal et al. (2019) for an example of such work.
- Multiplicative weights can actually be written in this FTRL framework! HW 2, Problem 1 explores this formally. Moreover, FTPL with various noise distributions can also be written as various instances of FTRL: see the book chapter `http://dept.stat.lsa.umich.edu/~tewaria/research/abernethy16perturbation.pdf` for more details on this connection.

Appendix A. Proof of Lemma 2

This proof is borrowed from Lecture 2 of the notes: `https://courses.cs.washington.edu/courses/cse599s/14sp/scribes/lecture2/scribeNote.pdf`. Recall that the regret

of FTL is given by

$$R_T := \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_t) - \min_{\boldsymbol{w}\in\mathcal{B}} \left[\sum_{t=1}^{T} \ell_t(\boldsymbol{w}) \right].$$

Thus, it suffices to upper bound

$$R_T(\boldsymbol{w}) := \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_t) - \sum_{t=1}^{T} \ell_t(\boldsymbol{w})$$

for any $\boldsymbol{w} \in \mathcal{B}$. We now use induction to prove that:

$$\sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{t+1}) \leq \sum_{t=1}^{T} \ell_t(\boldsymbol{w}) \tag{8.8}$$

for any $\boldsymbol{w} \in \mathcal{B}$. Clearly, this suffices to prove Lemma 2.

Base case: $T = 1$ Note that $\boldsymbol{w}_2$ minimizes $\ell_1(\cdot)$, and so we get $\ell_1(\boldsymbol{w}_2) \leq \ell_1(\boldsymbol{w})$ for any $\boldsymbol{w} \in \mathcal{B}$.

Inductive step: We assume that Equation (8.8) holds for $T-1$, i.e. we have

$$\sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}_{t+1}) \leq \sum_{t=1}^{T-1} \ell_t(\boldsymbol{w})$$

for any $\boldsymbol{w} \in \mathcal{B}$. We will now use this to show that Equation (8.8) holds. We show the following sequence of equivalent inequalities:

$$\begin{aligned}
&\sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}_{t+1}) \leq \sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}) \\
\iff &\sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}_{t+1}) + \ell_T(\boldsymbol{w}_{T+1}) \leq \sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}) + \ell_T(\boldsymbol{w}_{T+1}) \\
\implies &\sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{t+1}) \leq \sum_{t=1}^{T-1} \ell_t(\boldsymbol{w}) + \ell_T(\boldsymbol{w}_{T+1}).
\end{aligned}$$

Since the last inequality holds for all $\boldsymbol{w}$, we can take $\boldsymbol{w} := \boldsymbol{w}_{T+1}$, giving us

$$\sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{t+1}) \leq \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{T+1}).$$

But clearly, $\boldsymbol{w}_{T+1}$ minimizes $\sum_{t=1}^{T} \ell_t(\boldsymbol{w})$! Thus, we get

$$\sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{t+1}) \leq \sum_{t=1}^{T} \ell_t(\boldsymbol{w}_{T+1}) \leq \sum_{t=1}^{T} \ell_t(\boldsymbol{w}),$$

which completes our proof.

References

Naman Agarwal, Brian Bullins, Elad Hazan, Sham Kakade, and Karan Singh. Online control with adversarial disturbances. In *International Conference on Machine Learning*, pages 111–119. PMLR, 2019.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.

Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.