Distributed Online Learning for Joint Regret with Communication Constraints

Dirk van der Hoeven DIRK@DIRKVANDERHOEVEN.COM
Dept. of Computer Science, Università degli Studi di Milano

Hédi Hadiji H.HADIJI@UVA.NL
Tim van Erven TIM@TIMVANERVEN.NL
Korteweg-de Vries Institute for Mathematics, University of Amsterdam

Editors: Sanjoy Dasgupta and Nika Haghtalab

Abstract

We consider distributed online learning for joint regret with communication constraints. In this setting, there are multiple agents that are connected in a graph. Each round, an adversary first activates one of the agents to issue a prediction and provides a corresponding gradient, and then the agents are allowed to send a b-bit message to their neighbors in the graph. All agents cooperate to control the joint regret, which is the sum of the losses of the activated agents minus the losses evaluated at the best fixed common comparator parameters $\boldsymbol{u}$. We observe that it is suboptimal for agents to wait for gradients that take too long to arrive. Instead, the graph should be partitioned into local clusters that communicate among themselves. Our main result is a new method that can adapt to the optimal graph partition for the adversarial activations and gradients, where the graph partition is selected from a set of candidate partitions. A crucial building block along the way is a new algorithm for online convex optimization with delayed gradient information that is comparator-adaptive, meaning that its joint regret scales with the norm of the comparator $\|\boldsymbol{u}\|$. We further provide near-optimal gradient compression schemes depending on the ratio of b and the dimension times the diameter of the graph.

Keywords: Online convex optimization, distributed algorithms

1. Introduction

We consider decentralized *online convex optimization* (OCO) with multiple agents that share information across a network to improve the prediction quality of the network as a whole. Our motivation comes from cases where local computation is cheap, but communication is relatively expensive. This is the case, for instance, in sensor networks, where the energy cost of wireless communication is typically the main bottleneck, and long-distance communication requires much more energy than communication between nearby sensors (Rabbat and Nowak, 2004). It also applies to cases where communication is relatively slow compared to the volume of prediction requests that each agent must serve. For instance, in climate informatics communication may be slow because agents are geographically spread out (McQuade and Monteleoni, 2012, 2017), and in finance or online advertising the rate of prediction requests may be so high that communication is slow by comparison. To model such scenarios, we limit communication in two ways: first, agents can only directly communicate to their neighbors in a graph $\mathcal{G}$ and, second, the messages that the agents can send are limited to contain at most b bits. We further assume that learning is fully decentralized, so there is no central coordinating

agent as in federated learning (Kairouz et al., 2019), and no single agent that dictates the predictions for all other agents as in distributed online optimization for consensus problems (Hosseini et al., 2013; Yan et al., 2013).

To fix the setting, assume there are N agents, which are cooperating to make sequential predictions over the course of T rounds. In every round t, first one of the agents I_t is activated by an adversary to select a prediction $\boldsymbol{w}_t$ from a closed and convex domain $\mathcal{W} \subseteq \mathbb{R}^d$. Then this agent receives feedback from the adversary in the form of the (sub)gradient $\boldsymbol{g}_t = \nabla \ell_t(\boldsymbol{w}_t)$ of a convex loss function ℓ_t over $\mathcal{W}$, with bounded Euclidean norm $\|\boldsymbol{g}_t\| \leqslant G$. Finally, all agents are allowed to communicate by sending a b-bit message to their neighbors in $\mathcal{G}$, and the round ends. The common goal of the agents is to control the *joint regret* with respect to comparator parameters $\boldsymbol{u} \in \mathcal{W}$:

$$\mathcal{R}_T(\boldsymbol{u}) = \sum_{t=1}^{T} \left(\ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{u})\right).$$

We refer to this setting, as *distributed online convex optimization for joint regret with communication constraints* (DOCO-JC). Apart from the communication limit b, the crucial distinction between DOCO-JC and standard OCO (Shalev-Shwartz, 2011; Hazan, 2016) is that information about the gradients $\boldsymbol{g}_t$ takes time to travel through the graph, so the agents suffer from delayed feedback (McMahan and Streeter, 2014; Joulani et al., 2016; Hsieh et al., 2020). This observation has prompted Hsieh et al. (2020) to consider a more abstract framework, in which there is no explicit graph, but only assumptions about the delays. For instance, if τ is the maximum delay before $\boldsymbol{g}_t$ is known by every agent, then Corollary 2 of Hsieh et al. implies a joint regret bound of

$$\mathcal{R}_T(\boldsymbol{u}) = \mathcal{O}\big(\max_{\boldsymbol{u}\in\mathcal{W}} \|\boldsymbol{u}\|\sqrt{\tau T}\big). \tag{1}$$

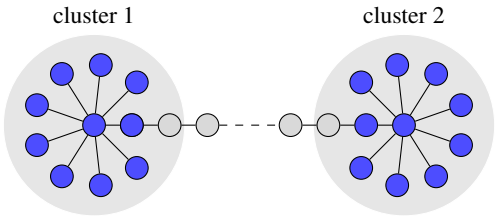
In our setting, τ corresponds to the maximum graph distance between any two agents that are ever active, i.e. the diameter $D(\mathcal{G})$ of $\mathcal{G}$ if all agents are activated at least once.

Although modeling only delays is an elegant abstraction, we argue that it is ultimately insufficient and that the graph structure should be explicitly taken into account. To see this, consider the graph $\mathcal{G}$ from Figure 1(*a*). In this graph, there are two clusters of agents that are very far apart. For simplicity, suppose that only agents from the two clusters are ever active, while the agents that connect the clusters only serve to pass on information. Then the maximum delay τ can be made arbitrarily large by extending the line that connects the two clusters. There exists a much better strategy, however, which is to have the two clusters operate independently with a maximum delay of $\tau_j = 2$ within each cluster $j = 1, 2$, leading to joint regret

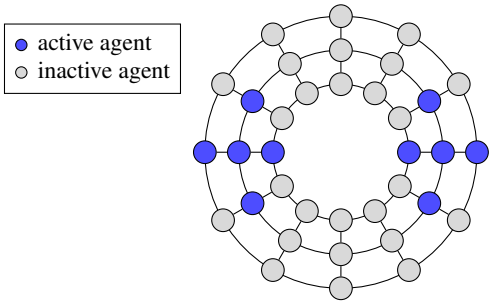
$$\mathcal{R}_T(\boldsymbol{u}) = \mathcal{O}\big(\max_{\boldsymbol{u}\in\mathcal{W}} \|\boldsymbol{u}\|\sqrt{\tau_1 T} + \max_{\boldsymbol{u}\in\mathcal{W}} \|\boldsymbol{u}\|\sqrt{\tau_2 T}\big) = \mathcal{O}\big(\max_{\boldsymbol{u}\in\mathcal{W}} \|\boldsymbol{u}\|\sqrt{T}\big). \tag{2}$$

Comparing (2) to (1) for arbitrarily large τ, we see that explicitly taking the graph structure into account can lead to an arbitrarily large improvement over modeling only delays. The takeaway from this example is that it is better for an agent to ignore information when it has to wait too long to receive it. The same conclusion still holds even if we replace τ by more refined measures of delay.

Unfortunately, partitioning $\mathcal{G}$ into subgraphs that exchange information is not always as easy as in Figure 1(*a*), because the clusters may be hidden in a larger graph (see Figure 1(*b*)) and the optimal



(*a*) Two clusters that can be made arbitrarily far apart by extending the line that connects them



(*b*) Two distant clusters of activated agents hidden in a larger graph

Figure 1: Two clusters far apart

partition may depend on the adversarial activations I_t, and also on the gradients $\boldsymbol{g}_t$ and the number of bits b that are allowed for communication. We therefore introduce a method that can learn the optimal partition from a set of candidate partitions. Formally, let $\mathcal{Q}$ be a collection of subgraphs of $\mathcal{G}$, which will be the building blocks for the candidate partitions. Then by a *$\mathcal{Q}$-partition* of the active agents we mean a disjoint collection $\{\mathcal{F}_1, \dots, \mathcal{F}_r\}$ of elements $\mathcal{F}_j \in \mathcal{Q}$ such that every node in $\mathcal{G}$ that is ever activated during any of the T rounds, is contained in one of the $\mathcal{F}_j$. The size r may vary between $\mathcal{Q}$-partitions. We show, in Theorem 9, that we can adapt to the best partition of the active agents at a cost that scales logarithmically with the size of $\mathcal{Q}$:

$$\sum_{j=1}^{r} \mathcal{R}_{\mathcal{F}_j}(\boldsymbol{u}_j) = \mathcal{O}\Big(\sum_{j=1}^{r} \|\boldsymbol{u}_j\| \Big(\sqrt{D(\mathcal{F}_j) T_j \ln\left(1 + |\mathcal{Q}| D(\mathcal{F}_j) \|\boldsymbol{u}_j\| T_j\right)}\Big) + \text{communication cost}\Big)$$

$$\text{for any } \mathcal{Q}\text{-partition } \{\mathcal{F}_1, \dots, \mathcal{F}_r\} \text{ and any } \boldsymbol{u}_1, \dots, \boldsymbol{u}_r \in \mathcal{W}, \quad (3)$$

where $\mathcal{R}_{\mathcal{F}_j}(\boldsymbol{u}_j) = \sum_{t: I_t \in \mathcal{F}_j} (\ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{u}_j)$ is the joint regret measured for the T_j rounds in which the active agent I_t is a node in $\mathcal{F}_j$. Up to the logarithmic factor, this bound implies (2) with the maximum delays τ_j replaced by the diameters $D(\mathcal{F}_j)$ of the partition cells $\mathcal{F}_j$, as is natural in our setting. In fact, there are two further improvements: first, the comparators $\boldsymbol{u}_j$ may differ between subgraphs $\mathcal{F}_j$, which makes the procedure more robust against heterogeneous environments and sensor malfunctions. And, second, we do not place any restriction on the size of the domain $\mathcal{W}$, but instead we automatically adapt to the unknown comparator norms $\|\boldsymbol{u}_j\|$. In fact, comparator-adaptivity is crucial to our approach: it allows aggregating over different delays in receiving the gradients using the iterate addition trick by Cutkosky (2019b). This would not be possible using existing aggregation methods for prediction with expert advice with delayed gradients, which would all incur an overhead growing with the largest possible gradient delay. Regarding the logarithmic factors, it is known that a factor $\ln(1 + \|\boldsymbol{u}_j\| T_j)$ is unavoidable for comparator-adaptive algorithms (Orabona, 2013). We discuss the logarithmic dependence on $|\mathcal{Q}|$ further below.

Our approach is based on having the agents communicate compressed approximations $\widehat{\boldsymbol{g}}_t$ of the gradients $\boldsymbol{g}_t$, which are forwarded through the network for at most $D_{\mathcal{Q}} = \max_{\mathcal{F} \in \mathcal{Q}} D(\mathcal{F}) \leqslant D(\mathcal{G})$ rounds. This means that nodes may need to forward up to $D_{\mathcal{Q}}$ compressed gradients at the same time, leaving $\lfloor b / D_{\mathcal{Q}} \rfloor$ bits per gradient, and thus the communication cost grows with $D_{\mathcal{Q}}$. Approximations

$\widehat{\boldsymbol{g}}_t$ may either be deterministic or stochastic, depending on whether the encoder that produces them is allowed to randomize. The method that achieves (3) uses a deterministic encoding scheme, for which

$$\text{communication cost} = \mathcal{O}\Big(2^{-b/(dD_{\mathcal{Q}})} \sum_{j=1}^{r} \|\boldsymbol{u}_j\| T_j\Big). \tag{4}$$

We see that we need roughly $b = \Omega(dD_{\mathcal{Q}} \ln T)$ bits to be sure that the communication cost is under control. In contrast, if we allow for stochastic encodings, then the expected communication cost can be reduced further. As shown in Theorem 20, it is possible to obtain the following bound, provided that $b \geqslant D_{\mathcal{Q}}\,(3\lceil \log_2(d)\rceil + 2)$:

$$\mathbb{E}\Big[\sum_{j=1}^{r} \mathcal{R}_{\mathcal{F}_j}(\boldsymbol{u}_j)\Big] = \mathcal{O}\Big(\sum_{j=1}^{r} \|\boldsymbol{u}_j\| G \Big(\sqrt{\Big(1 + \frac{dD_{\mathcal{Q}}}{b}\Big) D(\mathcal{F}_j) T_j \ln\big(1 + |\mathcal{Q}| D(\mathcal{F}_j) \|\boldsymbol{u}_j\| T_j G\big)}\Big)\Big)$$
$$\text{for any } \mathcal{Q}\text{-partition } \{\mathcal{F}_1, \dots, \mathcal{F}_r\} \text{ and any } \boldsymbol{u}_1, \dots, \boldsymbol{u}_r \in \mathcal{W}. \tag{5}$$

Comparing (5) to (3)+(4), we now obtain the same rate as soon as $b = \Theta(dD_{\mathcal{Q}})$, gaining an $\ln T$ factor. And, more importantly, whereas the deterministic communication cost in (4) can be linear in T for $b = o(dD_{\mathcal{Q}} \ln T)$, the stochastic encoding result in (5) allows for a number of bits b that is sublinear in $dD_{\mathcal{Q}}$, at the cost of (only) a worse constant factor in the bound. This makes it possible to choose a trade-off between communication cost and joint regret performance.

Approach and Organization of the Paper As mentioned, our approach aggregates multiple comparator-adaptive subalgorithms that each incur their own maximum delay. Since existing comparator-adaptive algorithms are not suited for compressed or delayed gradients, we introduce a new comparator-adaptive algorithm for the DOCO-JC setting in Section 2. As discussed below Lemma 3, the key to its development is a novel inequality that generalizes the so-called prod bound (Cesa-Bianchi and Lugosi, 2006, Lemma 2.4). Since the prod bound is at the core of many adaptive algorithms in the literature, for example the algorithms in (Koolen and Van Erven, 2015; Van Erven and Koolen, 2017; Cutkosky and Orabona, 2018; Wang et al., 2019; Van Erven et al., 2021), our new inequality may also be useful to develop other adaptive algorithms for settings with delayed gradients. For compressed gradients such that $\|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\| \leqslant \varepsilon$, the new comparator-adaptive algorithm satisfies the following regret bound:

$$\mathcal{R}_T(\|\boldsymbol{u}\|) = \mathcal{O}\left(\|\boldsymbol{u}\| \sqrt{\Lambda_T \ln(\|\boldsymbol{u}\|\Lambda_T + 1)} + \|\boldsymbol{u}\| T\varepsilon\right),$$

where $\Lambda_T = \sum_{t=1}^{T} \left(\|\boldsymbol{g}_t\|^2 + 2\|\boldsymbol{g}_t\| \sum_{i\in\gamma(t)} \|\boldsymbol{g}_i\|\right)$ is a standard measure of the effect of gradient delays called the *lag* (Hsieh et al., 2020; Joulani et al., 2016; McMahan and Streeter, 2014), with $\gamma(t) \subseteq \{1, \dots, t-1\}$ denoting the set of indices of past gradients that are unavailable to the active agent I_t. As the maximum delay in $\mathcal{G}$ is $D(\mathcal{G})$, there can be at most $|\gamma(t)| \leqslant D(\mathcal{G})$ gradients that are unavailable at any time, and consequently the lag satisfies $\Lambda_T \leqslant G^2(1 + 2D(\mathcal{G}))T$. In Section 3 we combine the algorithm from Section 2 with both deterministic and stochastic encodings for the gradients. Our encodings are based on simple combinations of standard covering arguments, but we prove matching lower bounds showing that they yield guarantees that are worst-case optimal (up to log factors), and they have the appeal of being straightforward to analyse and implement. Finally, in Section 4, we obtain (3) and (5) by aggregating multiple instances of the methods from Section 3,

instantiated for different maximum delays. Even though the algorithms from Section 3 are worst-case optimal, it is not clear whether the logarithmic dependence on $|\mathcal{Q}|$ in (3) and (5) that results when combining them, is also optimal. We leave this as an open question for future work.

1.1. Related Work

There has been much work on distributed architectures. However, the majority of the literature is about federated or parallel computation (see Kairouz et al. (2019) for an extensive review of the federated setting), where multiple workers are under the supervision of a central coordinator. In contrast, we study a decentralized setting, in which no central authority coordinates the learning. We also study the impact of delays and communication limits. We therefore focus our literature review on decentralized learning and on other works with communication limits.

Decentralized Online Convex Optimization Most directly related to our setting are decentralized OCO settings, in which a set of agents in a network collectively try to optimize an objective that is revealed sequentially. This includes the work of Hsieh et al. (2020) on delay-tolerant algorithms. The main technical difficulty they encounter is to tune the learning rates for a dual-averaging/follow-the-regularized-leader type approach, which is especially challenging because of the requirement of maintaining a non-decreasing learning rate. Cesa-Bianchi et al. (2020) consider a setting where multiple nodes can be active per round. In each round all active nodes make a prediction and suffer the same loss. The most important difference with our setting is that information is not forwarded through the network, so agents only hear about the gradients of their direct neighbors in $\mathcal{G}$. The authors show that it is possible to obtain $\mathcal{R}_T(\boldsymbol{u}) = \mathcal{O}(\sqrt{A(\mathcal{G})T})$ and $\mathcal{R}_T(\boldsymbol{u}) = \mathcal{O}(\sqrt{Q(\mathcal{G})T})$ for stochastic and adversarial activations respectively, where $A(\mathcal{G})$ is the independence number of $\mathcal{G}$ and $Q(\mathcal{G})$ is the clique covering number. Finally, in (Cao and Başar, 2021), a setting with event-triggered communication is introduced.

Distributed Online Optimization Distributed Online Optimization is inspired by (offline) distributed optimization (Duchi et al., 2010; Scaman et al., 2018) and developed in (Hosseini et al., 2013; Yan et al., 2013). The difference with the setting we consider is the notion of regret. In Distributed Online Optimization, the *collective regret* is analysed, in which the global loss per round is a sum of local losses per agent, but this global loss is always evaluated at the prediction of one of the agents. This collective regret is closer to the distributed optimization objective used, e.g., for wireless sensor networks (Rabbat and Nowak, 2004). There exist extensions for time-varying networks with a specific structure, (Mateos-Núñez and Cortés, 2014; Akbari et al., 2015), and there is a version of collective regret where the comparator changes between rounds (Shahrampour and Jadbabaie, 2018; Zhang et al., 2019). Hsieh et al. (2020) provide an extensive review of Distributed Online Optimization and a reduction from collective regret to joint regret.

Communication-Limited Settings Communication can be a performance bottleneck in distributed systems (see, e.g., a discussion of performance in the context of parallel training of deep neural networks in (Seide et al., 2014)). This has generated much interest in diverse fields for communication-constrained distributed tasks, including in optimization (Alistarh et al., 2017), for mean-estimation (Suresh et al., 2017), for hypothesis testing (Szabo et al., 2020), and for inference (Acharya et al., 2020). Two lines of research are closest to our work. The first, in (Tang et al., 2018; Koloskova et al., 2019; Vogels et al., 2020) and references therein, studies variants of Stochastic Gradient Descent (SGD) used in decentralized optimization under bandwidth-limited gossip communication, often

with the aim of training deep neural networks. Another line of work is devoted to online learning with communication constraints, with lower bounds for online learning problems with communication constraints (Shamir, 2014), and online learning in a serial multi-agent framework (Acharya et al., 2019). Most of these works focus on cases where the number of bits per message is at least linear in d with the exceptions of (Acharya et al., 2019; Mayekar and Tyagi, 2020b). To our knowledge, we are the first to incorporate communication constraints into a decentralized online learning framework.

Comparator-Adaptive Algorithms Recently a series of work has developed comparator-adaptive algorithms for various settings. For example, for standard OCO (McMahan and Orabona, 2014; Orabona and Pál, 2016; Foster et al., 2017; Cutkosky and Boahen, 2017; Cutkosky and Orabona, 2018), scale-free comparator-adaptive algorithms (Kotłowski, 2017; Kempka et al., 2019), comparator-adaptive algorithms with unbounded stochastic gradients (Jun and Orabona, 2019; Van der Hoeven, 2019), for convex bandits (Van der Hoeven et al., 2020), for dynamic and strongly adaptive OCO (Cutkosky, 2020), or with an unknown bound on the gradients (Cutkosky, 2019a; Mhammedi and Koolen, 2020).

1.2. Further Assumptions and Notation

A network is an (undirected) graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, consisting of a set of nodes $\mathcal{N}$ and edges $\mathcal{E}$ between them. Throughout the paper norms are always the Euclidean norm. We assume that the (sub)gradients are bounded by $\|\boldsymbol{g}_t\| \leqslant G$, and that G and the time horizon T are known to the agents in advance. We do not need to assume an oblivious adversary, because the agents only randomize when choosing $\widehat{\boldsymbol{g}}_t$, which happens after the adversary has already revealed $\boldsymbol{g}_t$.

Encoding the Gradients Since a b-bit message may contain at most $D(\mathcal{G})$ gradients, we reserve $k = \lfloor b/D(\mathcal{G}) \rfloor$ bits per gradient. After the active node I_t observes the gradient $\boldsymbol{g}_t$, it builds a k-bit compressed gradient $C(\boldsymbol{g}_t) \in \{0,1\}^k$ and sends it to other nodes. These then decode to $\widehat{\boldsymbol{g}}_t = F(C(\boldsymbol{g}_t)) \in \mathbb{R}^d$, which is used as an approximation of the true gradient. We assume that it is common knowledge among the agents at which time t each compressed gradient $\widehat{\boldsymbol{g}}_t$ was produced, and that agents also do not need to explicitly encode how many gradients they are forwarding at any given time. These assumptions can always be satisfied by adding a few extra bits of meta-information.

2. Comparator-Adaptive Algorithm for DOCO-JC

As announced in Section 1, here we introduce the main building block for our approach: a comparator-adaptive algorithm that can handle both missing and approximate gradients. With some minor modifications the algorithms in this section can also be used in the OCO with delays setting, where one only needs to track which gradients are available for prediction, and not the node that made the prediction. Without loss of generality, we only consider $\mathcal{W} = \mathbb{R}^d$, because it is straightforward to reduce constrained domains to this case using a reduction by Cutkosky and Orabona (2018, Theorem 3). As observed by Cutkosky and Orabona (2018) comparator-adaptive algorithms can be constructed by separately learning the direction $\boldsymbol{u}/\|\boldsymbol{u}\|$ and scale $\|\boldsymbol{u}\|$, where learning the direction is a standard constrained learning task on the unit ball and most of the difficulty lies in solving the unconstrained 1-dimensional scale problem while being adaptive to $\|\boldsymbol{u}\|$. Suppose agent I_t predicts $\boldsymbol{z}_t$ for the direction, satisfying $\|\boldsymbol{z}_t\| \leqslant 1$, according to an algorithm $\mathcal{A}_{\mathcal{Z}}$, and it predicts $v_t \in \mathbb{R}_+$ for the scale following an algorithm $\mathcal{A}_{\mathcal{V}}$. Then its joint prediction is $\boldsymbol{w}_t = v_t \boldsymbol{z}_t$. The corresponding notions of joint regret are $\widetilde{\mathcal{R}}_T^{\mathcal{Z}}\left(\frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}\right) = \sum_{t=1}^T \langle \boldsymbol{z}_t - \frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}, \boldsymbol{g}_t \rangle$ for the direction, and

$\widetilde{\mathcal{R}}_T^{\mathcal{V}}(\|\boldsymbol{u}\|) = \sum_{t=1}^T (v_t - \|\boldsymbol{u}\|)\langle \boldsymbol{z}_t, \boldsymbol{g}_t\rangle$ for the scale. Then, by the black-box reduction in Algorithm 2 in Appendix B.1, due to Cutkosky and Orabona (2018), the total joint regret of the algorithm is bounded by

$$\mathcal{R}_T(\boldsymbol{u}) \leqslant \|\boldsymbol{u}\| \widetilde{\mathcal{R}}_T^{\mathcal{Z}}\left(\frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}\right) + \widetilde{\mathcal{R}}_T^{\mathcal{V}}(\|\boldsymbol{u}\|).$$

It follows that, as long as $\mathcal{A}_{\mathcal{V}}$ is comparator-adaptive, our entire algorithm is comparator-adaptive.

Controlling $\widetilde{\mathcal{R}}_T^{\mathcal{Z}}$ is an online linear optimization (OLO) task. For $\mathcal{A}_{\mathcal{Z}}$, it suffices to use any OLO algorithm on the unit ball that is *delay-tolerant*, by which we mean that it satisfies $\mathcal{R}_T(\boldsymbol{u}) = \mathcal{O}\left(\sqrt{\Lambda_T}\right)$. If such an algorithm is used with approximate gradients such that $\|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\| \leqslant \varepsilon$, then it enjoys the bound $\mathcal{R}_T(\boldsymbol{u}) = \mathcal{O}\left(\sqrt{\Lambda_T} + T\varepsilon\right)$. In the remainder of this section, we will present a one-dimensional algorithm for learning the range, such that, when combined with a delay-tolerant algorithm for learning the direction, we obtain the following comparator-adaptive guarantee. In Appendix B.1, we state and prove Theorem 18, a version of the result with an explicit finite-time bound.

Theorem 1 *Suppose $\mathcal{A}_{\mathcal{Z}}$ is a delay-tolerant algorithm, and $\mathcal{A}_{\mathcal{V}}$ is Algorithm 1, defined below and tuned with any $\nu > 0$ and error parameter $\varepsilon \geqslant \|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\|$. Then the combination of $\mathcal{A}_{\mathcal{Z}}$ and $\mathcal{A}_{\mathcal{V}}$ by the black-box reduction described above (i.e. Algorithm 2 in Appendix B.1) satisfies*

$$\mathcal{R}_T(\boldsymbol{u}) \leqslant \nu + \|\boldsymbol{u}\|\,\mathcal{B}(T)\,, \text{ where } \mathcal{B}(T) = \mathcal{O}\left(\varepsilon T + \sqrt{\Lambda_T \ln\left(1 + \frac{\|\boldsymbol{u}\|}{\nu} TGD(\mathcal{G})\right)}\right).$$

As shown by Proposition 15 in the Appendix, the Ada-Delay-Dist algorithm of Hsieh et al. (2020) is delay-tolerant and can therefore be used for $\mathcal{A}_{\mathcal{Z}}$. It remains to adapt to scale, which requires designing a suitable one-dimensional algorithm $\mathcal{A}_{\mathcal{V}}$.

Learning the scale is actually a special case of the general problem of designing a comparator-adaptive algorithm, in which the gradients are projected down to one dimension via $h_t = \langle \boldsymbol{z}_t, \boldsymbol{g}_t\rangle$ and approximate gradients correspond to $\widehat{h}_t = \langle \boldsymbol{z}_t, \widehat{\boldsymbol{g}}_t\rangle$. If $\|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\| \leqslant \varepsilon$, then we also have $|\widehat{h}_t - h_t| \leqslant \|\boldsymbol{z}_t\|\|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\| \leqslant \varepsilon$. It therefore inherits all the difficulties of dealing with approximate and delayed gradients.

Let us first sketch the main difficulties. The obvious approach to handling approximate gradients, which would work well if we did not aim for comparator-adaptivity, is to observe that

$$\sum_{t=1}^T \langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{g}_t\rangle = \sum_{t=1}^T \langle \boldsymbol{w}_t - \boldsymbol{u}, \widehat{\boldsymbol{g}}_t\rangle + \sum_{t=1}^T \langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{g}_t - \widehat{\boldsymbol{g}}_t\rangle. \tag{6}$$

Then use a standard algorithm to control $\sum_{t=1}^T \langle \boldsymbol{w}_t - \boldsymbol{u}, \widehat{\boldsymbol{g}}_t\rangle$, and attempt to bound $\langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{g}_t - \widehat{\boldsymbol{g}}_t\rangle$. While this is possible in expectation for stochastic encodings by making $\widehat{\boldsymbol{g}}_t$ an unbiased approximation of $\boldsymbol{g}_t$, for deterministic encodings it is not clear how $\langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{g}_t - \widehat{\boldsymbol{g}}_t\rangle$ can be bounded by a term that scales with $\|\boldsymbol{u}\|$. Scaling with $\|\boldsymbol{u}\|$ is crucial, as we will exploit the comparator-adaptive property of our algorithms to learn a $\mathcal{Q}$-partition. We therefore cannot use this approach.

The second difficulty is due to the fact that gradients may be unavailable at prediction time due to the delayed feedback. Considerable work has been done in the delayed feedback setting to tune

Algorithm 1 Comparator-Adaptive Algorithm on a Graph for $d = 1$

Input: $\nu > 0$, upper bound G on $\max_t \|\boldsymbol{g}_t\|$, error parameter $\varepsilon > 0$

Initialize: $\mathcal{S}_n(1) = \varnothing$ and $\gamma_n(t) = \varnothing$ for all time-steps t and all nodes $n \in \mathcal{N}$, set distribution $\mathrm{d}\rho(\eta) = \exp(-\eta^2)/Z\mathrm{d}\eta$ over $\eta \in [0, a]$, where $a = ((G+\varepsilon)20(1+2D(\mathcal{G})))^{-1}$ and define $Z = \int_0^a \exp(-\eta^2)\mathrm{d}\eta$.

for $t = 1 \dots T$ **do**

Play $v_t = \mathbb{E}_{\eta\sim\rho}\left[\nu \exp\left(-\sum_{s\in\mathcal{S}_{I_t}(t)}\left(\eta(\widehat{h}_s+\varepsilon) + \eta^2(\widehat{h}_s+\varepsilon)^2 + 2\eta^2\widehat{\zeta}_{I_t}(s)\right)\right)\eta\right]$, where $\widehat{\zeta}_{I_t}(s) = |\widehat{h}_s+\varepsilon|\sum_{i\in\gamma_{I_t}(s)}|\widehat{h}_i+\varepsilon|)$

For all $n \in \mathcal{N}$: send messages, receive messages, update $S_n(t+1)$, and update $\gamma_n(s)$ for all $s \in S_n(t+1)$.

end

the learning rate of standard OCO algorithms to deal with missing gradients (Joulani et al., 2016; Hsieh et al., 2020). Unfortunately, these existing approaches do not work for comparator-adaptive algorithms. For a more in-depth discussion of the difficulties faced in designing a comparator-adaptive algorithm which can handle missing gradients we refer the reader to Appendix A. To resolve the aforementioned difficulties simultaneously, we provide a new one-dimensional comparator-adaptive algorithm given in Algorithm 1, which can be used to select v_t. It turns out that these predictions v_t can be computed in linear time (see (12) in Appendix B). In the algorithm and the discussion below, $\mathcal{S}_{I_t}(t) \subset \{1, \dots, t-1\}$ denotes the set of indices of gradients that are available at node I_t in round t. Similarly, $\gamma(s) = \{1, \dots, s-1\} \setminus \mathcal{S}_{I_s}(s)$ is the set of indices of gradients that were missing in round s at node I_s, and $\gamma_{I_t}(s) = \gamma(s) \cap \mathcal{S}_{I_t}(t)$ is the set indices of gradients that were missing at node I_s in round s, but are available at node I_t in round t. The regret of Algorithm 1 is bounded by the following result, whose proof can be found in Appendix B.

Theorem 2 *Let* $\Lambda_T^h = \sum_{t=1}^T \left(h_t^2 + 2|h_t|\sum_{i\in\gamma(t)}|h_i|\right)$. *Algorithm 1, tuned with any* $\nu > 0$ *and* $\varepsilon > 0$ *such that* $|\widehat{h}_t - h_t| \leqslant \varepsilon$, *satisfies for any* $u \geqslant 0$,

$$\sum_{t=1}^T (v_t - u)h_t \leqslant \nu + u\,\mathcal{B}^h(T) \text{ where } \mathcal{B}^h(T) = \mathcal{O}\left(\varepsilon T + \sqrt{\Lambda_T^h \ln\left(1 + \frac{u}{\nu}TGD(\mathcal{G})\right)}\right).$$

The fact that the regret of Algorithm 1 scales with u is a crucial property that we will use to a $\mathcal{Q}$-partition, in particular the property that for $u = 0$ the regret is ν will be repeatedly used.

We proceed to discuss the main ideas behind Algorithm 1 and Theorem 2. One of the essential parts in deriving any comparator-adaptive algorithm is designing a potential function Φ_T. To see how the potential function is used, suppose that we could get a sequence of predictions $v_1, \dots, v_T$ that satisfy

$$\nu - \sum_{t=1}^T v_t h_t \geqslant \Phi_T\Big(-\sum_{t=1}^T(\widehat{h}_t + \varepsilon)\Big) \qquad \text{for some } \nu > 0. \tag{7}$$

Then these predictions would satisfy the regret bound $\sum_{t=1}^T (v_t - u)h_t \leqslant \nu + \Phi_T^\star(u) + 2u\varepsilon T$, where $\Phi_T^\star$ is the convex conjugate of Φ_T. To see this, recall Fenchel's inequality $\Phi_T(x) + \Phi_T^\star(u) \geqslant xu$,

which implies $\Phi_T\Big(-\sum_{t=1}^T(\widehat{h}_t+\varepsilon)\Big) \geqslant -\Phi_T^\star(u) - u\sum_{t=1}^T(\widehat{h}_t+\varepsilon) \geqslant -\Phi_T^\star(u) - 2u\varepsilon T - u\sum_{t=1}^T h_t$, and combine with (7) to obtain the bound on the regret. We therefore require a potential Φ_T for which we can satisfy (7) and for which $\Phi_T^*(u)$ is small enough. Now, suppose that we could bound the increase in potential per round by

$$\Phi_t\bigg(-\sum_{s=1}^{t}(\widehat{h}_s+\varepsilon)\bigg) \leqslant \Phi_{t-1}\bigg(-\sum_{s=1}^{t-1}(\widehat{h}_s+\varepsilon)\bigg) - v_t h_t. \tag{8}$$

Then summing over t would lead to the desired inequality (7) with $\nu = \Phi_0(0)$. But herein lies exactly the technical challenge caused by the missing gradients. To guarantee (8), existing comparator-adaptive algorithms base their prediction for round t on knowledge of Φ_{t-1}, but the missing gradients prevent us from doing the same. Instead, we have to use whatever gradients are available at prediction time.

To account for the missing gradients, our predictions include a correction term $\widehat{\zeta}_{I_t}(s) = |\widehat{h}_s+\varepsilon|\sum_{i\in\gamma_{I_t}(s)}|\widehat{h}_i+\varepsilon|)$, which we incorporate in the predictions v_t defined in Algorithm 1. Similar to the correction for approximate gradients, the correction for missing gradients decreases the effective learning rate of our algorithm. The corrections play a crucial role in our potential function:

$$\Phi_t\bigg(-\sum_{s=1}^{t}\big(\widehat{h}_s+\varepsilon\big)\bigg) = \mathbb{E}_{\eta\sim\rho}\left[\nu\exp\left(-\sum_{s=1}^{t}\Big(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2\widehat{\zeta}(s)\Big)\right)\right], \tag{9}$$

where $\widehat{\zeta}(s) = |\widehat{h}_s+\varepsilon|\sum_{i\in\gamma(s)}|\widehat{h}_i+\varepsilon|$. The potential function includes a similar correction term as our predictions, with the difference that the potential corrects for all missing gradients, not just the ones available at the active node. Together, these corrections allow us to establish (8):

Lemma 3 *Suppose $\|\boldsymbol{z}_t\| \leqslant 1$, $\|\boldsymbol{g}_t\| \leqslant G$, and $\|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\| \leqslant \varepsilon$ for all t. Then the predictions v_t defined in Algorithm 1 satisfy* (8).

The proof of Lemma 3 (see Appendix B) involves carefully tracking which gradients are missing. Whereas the analysis of standard comparator-adaptive algorithms relies on an inequality called the prod bound (Cesa-Bianchi and Lugosi, 2006, Lemma 2.4) to obtain an analogue of (8), the standard prod bound fails in the presence of missing gradients. The key to our proof is therefore a novel inequality given in Lemma 11 in Appendix B, which substantially generalizes the prod bound. Finally, it remains to show that $\Phi_T^\star(u)$ is small enough, which we do in Lemma 13 in Appendix B. Together, the above provides a comparator adaptive algorithm which can handle approximate and missing gradients.

3. Limited Communication and Optimality

We proceed to construct both deterministic and stochastic communication strategies for the gradients, which can be used to apply Algorithm 1 in the DOCO-JC setting. We will restrict attention to communication strategies in which nodes send and receive messages containing approximate gradients. We say an algorithm uses the *standard forwarding strategy* if every node, upon receiving a gradient that it has not seen yet, immediately forwards the gradient to all its neighbors. To enable this strategy, we assume that the messages containing the gradients include meta-data with a unique identifier, e.g.,

the time-step at which they were first sent. We do not account for this meta-data in the discussion below, because it may already be naturally present in the network protocol or otherwise it can be encoded at a minor overhead of $\mathcal{O}(\log T)$ additional bits.

Under the standard forwarding strategy, a single node sends at most $D(\mathcal{G})$ distinct messages at a time. Conversely, there exists an activation sequence under which a node will forward $D(\mathcal{G}) - 1$ messages at the same time. Indeed, on a path of length $D(\mathcal{G})$ in the graph, consider an activation sequence selecting adjacent nodes, going from one end of the path to the other. Using the standard forwarding strategy, the penultimate node forwards the $D(\mathcal{G}) - 1$ previous messages at the same time. Accordingly, under a total b-bit constraint on the bandwidth, we assume that the b bits are divided into $D(\mathcal{G})$ slots of $k = \lfloor b/D(\mathcal{G}) \rfloor$ bits, each slot corresponding to a message.

Deterministic encodings We first provide upper (Theorem 4) and lower (Theorem 5) bounds on the regret for deterministic encodings. A possible encoding is to fix a cover of the set of possible gradients, and communicate the element of the cover to which the gradient belongs; see Appendix D.1 for more details. The approximate gradients $\widehat{\boldsymbol{g}}_t$ obtained from this encoding are then given as inputs to the black-box reduction, with AdaDelay-dist (from Hsieh et al. (2020)]) as $\mathcal{A}_{\mathcal{Z}}$ and Algorithm 1 as $\mathcal{A}_{\mathcal{V}}$; we tune Algorithm 1 with $\varepsilon = 3 \cdot 2^{-k/\lfloor b/D(\mathcal{G}) \rfloor} G$ and the upper bound $4G$ on $\|\widehat{\boldsymbol{g}}_t\|$. As detailed in Appendix D.1, these values are valid upper bounds on the error and the norm of the encodings, and allow us to apply Theorem 1, and obtain the following guarantee:

Theorem 4 (Regret Bound with Deterministic Coding) *Using $k = \lfloor b/D(\mathcal{G}) \rfloor$ bits per gradient, the algorithm described above satisfies*

$$\mathcal{R}_T(\boldsymbol{u}) \leqslant \nu + \|\boldsymbol{u}\| \, \mathcal{B}(T), \quad \textit{where} \quad \mathcal{B}(T) = \widetilde{\mathcal{O}}\left(\sqrt{\Lambda_T} + T 2^{-b/(dD(\mathcal{G}))} G \right).$$

In Section D.1, we also propose a simpler per-coordinate encoding. This more practical encoding comes at the cost of an extra $\sqrt{d}$ factor in the second term of the regret bound, which is acceptable when b is very large. We further provide the following matching lower bound for a natural class of algorithms we call gradient-oblivious; see Appendix E for a detailed discussion.

Theorem 5 (Lower Bound I: Deterministic Encoding) *There exists an activation sequence such that for any gradient-oblivious algorithm using a deterministic encoding with $\lfloor b/D(\mathcal{G}) \rfloor$ bits per gradient, with $\mathcal{R}_T(\mathbf{0}) \leqslant \nu$, and for any comparator norm $U \geqslant 0$, for T large enough, there exists a comparator $\boldsymbol{u} \in \mathbb{R}^d$ such that $\|\boldsymbol{u}\| = U$ and*

$$\sup_{\textit{G-Lipschitz losses}} \mathcal{R}_T(\boldsymbol{u}) \geqslant \max\left(0.15 \, \|\boldsymbol{u}\| G \sqrt{D(\mathcal{G}) \, T \ln\left(\frac{\|\boldsymbol{u}\|^2 T}{72 \nu^2 D(\mathcal{G})} \right)}, \, T 2^{-b/(dD(\mathcal{G}))} \, \|\boldsymbol{u}\| G \right).$$

Since $\Lambda_T \leqslant 3G^2 D(\mathcal{G}) T$, the upper bound in Theorem 4 matches this lower bound up to multiplicative constants and lower order terms. A notable feature of both the upper and lower bounds is the term containing $T 2^{-b/(dD(\mathcal{G}))}$, which shows that we need roughly $b = \Theta(dD(\mathcal{G}) \log_2(T))$ bits to get non-trivial regret. This means that, for large dimensions d, the number of bits b must also be large. The proof of Theorem 5 is in Appendix E.2.1. It consists of two parts: we obtain the first term in the maximum by modifying a lower bound for norm-adaptive OCO from Orabona (2013) to incorporate the effect of the graph structure, which adds a $\sqrt{D(\mathcal{G})}$ multiplicative factor compared to the original

lower bound. The second term in the maximum, which is linear in T, is new and arises from the communication limit k on the number of bits that can be transmitted per gradient.

Stochastic Encodings As discussed above, deterministic encodings require a large number of bits, which grows at least linearly with $dD(\mathcal{G})$. In the regime where $b \leqslant dD(\mathcal{G})$, a better solution is to inject randomness into the encodings, which bypasses the lower bound from Theorem 5. It turns out that near-optimal guarantees can be obtained along with a straightforward analysis and implementation by combining two known techniques. The first technique may be called *sparsification* and consists of sampling (uniformly at random) a single coordinate of the gradient vector to be communicated. Encoding the index of this coordinate requires $\lceil \log_2 d \rceil$ bits. The second technique may be called *p-level stochastic quantization*. It consists of truncating the gradient coordinate to be transmitted to the first p digits in its binary expansion. We use this with $p = \lceil \log_2(d) \rceil$. Finally, to reduce the variance, we repeat this construction $m = \Theta(k/\log_2 d)$ times, sending less than k bits per vector in total. We refer to the joint construction as *sparsified quantization* with precision p and number of repetitions m. See Appendix D.2 for a detailed account of the scheme. Very similar constructions have previously been used by Mayekar and Tyagi (2020a) and Acharya et al. (2019) in related contexts; Appendix D.2 contains a detailed comparison. In spite of the simplicity of the construction, we show that sparsified quantization gives near-optimal theoretical guarantees:

Theorem 6 *For any vector $\boldsymbol{x} \in \mathcal{B}_2(G)$, sparsified quantization with precision $p = \lceil \log_2(d) \rceil$ and $m = \lfloor k/(3\lceil \log_2(d) \rceil + 2) \rfloor$ repetitions produces a (randomized) approximation $\widehat{\boldsymbol{x}}$ that satisfies $\|\widehat{\boldsymbol{x}}\| \leqslant 2dG$ and $\mathbb{E}\big[\|\widehat{\boldsymbol{x}} - \boldsymbol{x}\|^2\big] \leqslant (2d/m)\|\boldsymbol{x}\|^2 + G^2/m = O\big((\log d)d/k\big)$, provided that the number of bits per vector is at least $k \geqslant 3\lceil \log_2(d) \rceil + 2$.*

The approximate gradients $\widehat{\boldsymbol{g}}_t$ obtained from sparsified quantization (Theorem 6) are then used as inputs to the black-box reduction, with AdaDelay-dist (from Hsieh et al. (2020)) as $\mathcal{A}_{\mathcal{Z}}$ and Algorithm 1 as $\mathcal{A}_{\mathcal{Y}}$; we tune Algorithm 1 with $\varepsilon = 0$ and the upper bound $2dG$ on $\|\widehat{\boldsymbol{g}}_t\|$. Using this algorithm, we obtain the following result (see Appendix B.2 for a full proof).

Theorem 7 (Regret bound with Stochastic Encoding) *Using $k = \lfloor b/D(\mathcal{G}) \rfloor$ bits per gradient, the algorithm described above satisfies*

$$\mathbb{E}[\mathcal{R}_T(\boldsymbol{u})] \leqslant \nu + \|\boldsymbol{u}\|\, \mathcal{B}(T) \quad \textit{where} \quad \mathcal{B}(T) = \widetilde{\mathcal{O}}\left(G\sqrt{\left(1 + \frac{dD(\mathcal{G})}{b}\right) D(\mathcal{G})T} \right).$$

The following theorem is a matching lower bound, up to log factors, for the natural class of gradient-oblivious algorithms; Appendix E.2.2 contains a definition, as well as a proof of the theorem.

Theorem 8 (Lower bound II: Stochastic Encodings) *For any gradient-oblivious algorithm using $\lfloor b/D(\mathcal{G}) \rfloor$ bits per gradient, there exists an activation sequence such that, for any $U > 0$ there exists a sequence of losses and a comparator $\boldsymbol{u} \in \mathbb{R}^d$ such that $\|\boldsymbol{u}\| = U$ and*

$$\mathbb{E}\big[\mathcal{R}_T(\boldsymbol{u})\big] \geqslant c\,\|\boldsymbol{u}\| G\sqrt{\left(1 + \frac{dD(\mathcal{G})}{b}\right) D(\mathcal{G})T}\,.$$

To summarize the proof, the lower bound for OCO is $\|\boldsymbol{u}\|G\sqrt{T}$, the encodings add a factor of $\sqrt{1 + dD(\mathcal{G})/b}$, and the delays add another $\sqrt{D(\mathcal{G})}$ factor on top. In Appendix E.2.2, we analyze in detail how each characteristic of the setting (graph and encoding) affects the hardness.

4. Learning a $\mathcal{Q}$-Partition

As discussed in the introduction, it can be highly suboptimal for agents to wait for gradients that take too long to arrive. Instead, the graph should be partitioned according to a $\mathcal{Q}$-partition. In this section we show how to exploit the comparator-adaptive property of our algorithms to learn a $\mathcal{Q}$-partition.

For a fixed subgraph $\mathcal{F} \subseteq \mathcal{G}$, consider the DOCO-JC problem restricted to that subgraph, that is, discarding all gradients and communications coming from nodes outside $\mathcal{F}$. Given a general algorithm for the DOCO-JC setting, we denote by $\boldsymbol{w}_t^{\mathcal{F}}$ the iterate generated by the algorithm restricted to $\mathcal{F}$. We define $\Lambda(\mathcal{F}) = \sum_{t:I_t\in\mathcal{F}} \left(\|\boldsymbol{g}_t\|^2 + 2\|\boldsymbol{g}_t\| \sum_{i\in\gamma(t,\mathcal{F})} \|\boldsymbol{g}_i\|\right)$, with $\gamma_{\mathcal{F}}(t) = \left[t\big(\mathcal{F}\big) - 1\right] \setminus \mathcal{S}_{I_t}(t,\mathcal{F})$, and $t(\mathcal{F}) = \{s : I_s \in \mathcal{F}\}$, and where $\mathcal{S}_{I_t}(t,\mathcal{F})$ is the set of indices of gradients that have been observed by I_t before round t. Using the typical bounds we obtain in this article, e.g., in Theorem 4, we upper bound its regret as $\mathcal{R}_{\mathcal{F}}(\boldsymbol{u}) = \sum_{t:I_t\in\mathcal{F}} \left(\ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{u}^{\mathcal{F}})\right) = \widetilde{\mathcal{O}}(\|\boldsymbol{u}\|\sqrt{\Lambda(\mathcal{F})})$. (We neglect the encoding costs here for the sake of simplicity.) Then, using this approach, we could fix an oracle partition $\mathcal{P} = \{\mathcal{F}_1, \dots, \mathcal{F}_r\}$ of the graph into disjoint subgraphs and apply this strategy on each subgraph. This splits the DOCO-JC task into r independent subtasks, and the total joint regret is simply the sum of joint regrets of the subtasks:

$$R_T(\boldsymbol{u}) = \sum_{\mathcal{F}\in\mathcal{P}} R_{\mathcal{F}}(\boldsymbol{u}) = \sum_{\mathcal{F}\in\mathcal{P}} \widetilde{\mathcal{O}}\left(\|\boldsymbol{u}\|\sqrt{\Lambda(\mathcal{F})}\right).$$

(There is even some extra flexibility, which is that each subtask $\mathcal{F}$ could have different comparator parameters $\boldsymbol{u}^{\mathcal{F}}$.) An apparent drawback of this strategy is that each node gets access to less information. Whether partitioning is worth it depends on the activation sequence. This raises an issue of adaptation, as the activation sequence is not known in advance.

Iterate Addition To adapt to the activation sequence we exploit the following special property of comparator-adaptive algorithms, observed by Cutkosky (2019b). For an algorithm $\mathcal{A}$, denote by $\boldsymbol{w}_t^{\mathcal{A}}$ its predictions and by $\widetilde{\mathcal{R}}_T^{\mathcal{A}}(\boldsymbol{u}) = \sum_{t=1}^T \langle \boldsymbol{w}_t^{\mathcal{A}} - \boldsymbol{u}, \boldsymbol{g}_t\rangle$ its linearised regret. Then consider two algorithms $\mathcal{A}$ and $\mathcal{B}$ that both have constant regret at most ν against the null comparator: $\widetilde{\mathcal{R}}_T^{\mathcal{A}}(\boldsymbol{0}) \leqslant \nu$ and $\widetilde{\mathcal{R}}_T^{\mathcal{B}}(\boldsymbol{0}) \leqslant \nu$. Then simply playing $\boldsymbol{w}_t = \boldsymbol{w}_t^{\mathcal{A}} + \boldsymbol{w}_t^{\mathcal{B}}$ ensures that

$$\mathcal{R}_T(\boldsymbol{u}) \leqslant \sum_{t=1}^T \langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{g}_t\rangle = \min_{\substack{\boldsymbol{x},\boldsymbol{y}\\ \boldsymbol{x}+\boldsymbol{y}=\boldsymbol{u}}} \widetilde{\mathcal{R}}_T^{\mathcal{A}}(\boldsymbol{x}) + \widetilde{\mathcal{R}}_T^{\mathcal{B}}(\boldsymbol{y}) \leqslant \nu + \min_{\mathcal{K}\in\{\mathcal{A},\mathcal{B}\}} \widetilde{\mathcal{R}}_T^{\mathcal{K}}(\boldsymbol{u}),$$

where the second inequality comes from minimizing over $(\boldsymbol{x} = \boldsymbol{0}, \boldsymbol{y} = \boldsymbol{u})$ and $(\boldsymbol{x} = \boldsymbol{u}, \boldsymbol{y} = \boldsymbol{0})$. We can choose an arbitrary collection of subgraphs $\mathcal{Q}$, play

$$w_t = \sum_{\mathcal{F}\in\mathcal{Q}} \boldsymbol{w}_t^{\mathcal{F}} \mathbb{1}\{I_t \in \mathcal{F}\} \tag{10}$$

and exploit the property that $\widetilde{\mathcal{R}}_{\mathcal{F}}(\boldsymbol{0}) \leqslant \nu$ to learn how to partition the graph to minimize the regret. The aforementioned predictions combined with ignoring messages older than $D_{\mathcal{Q}}$ rounds, using $k = \lfloor b/D_{\mathcal{Q}} \rfloor$ bits per gradient, and our new comparator-adaptive algorithm with deterministic encoding yields the following result, proved in Appendix C:

Theorem 9 (Learning a $\mathcal{Q}$-Partition, deterministic encoding) *Let $\mathcal{Q}$ be a collection of subgraphs of $\mathcal{G}$. Suppose the learner uses the algorithm of Theorem 4 with deterministic encodings for each*

subgraph $\mathcal{F} \in \mathcal{Q}$, discarding any message coming from outside of $\mathcal{F}$. Then, setting $\nu = 1/|\mathcal{Q}|$ and $k = \lfloor b/D_{\mathcal{Q}} \rfloor$, and playing $\boldsymbol{w}_t$ as specified in equation (10) guarantees that

$$\sum_{j=1}^{r} \mathcal{R}_{\mathcal{F}_j}(\boldsymbol{u}_j) = \mathcal{O}\left(\sum_{j=1}^{r} \|\boldsymbol{u}_j\| \left(\sqrt{\Lambda(\mathcal{F}_j) \ln\left(1 + |\mathcal{Q}| D(\mathcal{F}_j) \|\boldsymbol{u}_j\| T_j G\right)} + 2^{-b/(dD_{\mathcal{Q}})} T_j G \right) \right)$$

for any $\mathcal{Q}$-partition $\{\mathcal{F}_1, \dots, \mathcal{F}_r\}$ and any $\boldsymbol{u}_1 \dots, \boldsymbol{u}_r \in \mathbb{R}^d$.

An analogous result holds in expectation for stochastic encodings by using the algorithm from Theorem 7 for each subgraph (see Theorem 20 in Appendix C). An alternative for tuning $\nu = 1/|\mathcal{Q}|$ is to set $\nu = 1$, in which case we obtain a bound of order $\widetilde{O}\left(|\mathcal{Q}| + \sum_{j=1}^{r} \|\boldsymbol{u}_j\| \Lambda(\mathcal{F}_j)\right)$. In Appendix C.1 we provide an example collection of subgraphs $\mathcal{Q}$ with which the learner can adapt to the activation sequences from Figures 1(*a*) and 1(*b*). In case the full graph is included in $\mathcal{Q}$-partition, i.e. $\mathcal{G} \in \mathcal{Q}$, we have $D_{\mathcal{Q}} = D(\mathcal{G})$. However, notice that the learner may choose not to include $\mathcal{G}$ in $\mathcal{Q}$ to increase the number of bits available to encode each gradient. This in turn allows the learner to improve the regret in some cases, as using more bits per gradient improves the regret bound for the subgraphs.

5. Conclusion

We provided a comparator-adaptive algorithm for the DOCO-JC setting. We provided upper and lower bounds for deterministic and stochastic encoded gradients and we demonstrated how to exploit the comparator-adaptive property of our algorithm to learn the best partition in a subset of partitions of the graph. An interesting direction to improve the communication strategy would be to send information chosen more efficiently, instead of systematically forwarding every gradient. This might be achieved by implementing other protocols, or error-feedback approaches; see the recent work by Cao and Başar (2021); Wen et al. (2020). Another limitation is the assumption in the protocol that all agents communicate in each round, which may be slow because it requires synchronization. However, as long as the maximum delay before each gradient is observed is bounded, this issue can be overcome by simply changing the $D(\mathcal{G})$ in the parameter settings of Algorithm 1 to the maximum delay. If no such bound is known beforehand, the problem becomes more complex, but a related problem for comparator-adaptive algorithms is learning G, rather than providing G up front. This problem has been studied in (Cutkosky, 2019a; Mhammedi and Koolen, 2020) and perhaps their techniques transfer to learning the maximum delay.

Acknowledgments

Van der Hoeven gratefully acknowledges support by the MIUR PRIN grant Algorithms, Games, and Digital Markets (ALGADIMAR). Hadiji and Van Erven were supported by the Netherlands Organization for Scientific Research (NWO) under grant number VI.Vidi.192.095.

References

Jayadev Acharya, Chris De Sa, Dylan J. Foster, and Karthik Sridharan. Distributed learning with sublinear communication. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 40–50, 2019.

Jayadev Acharya, Clément Canonne, and Himanshu Tyagi. Inference under information constraints II: Communication constraints and shared randomness. *IEEE Transactions on Information Theory*, 66(12):7856–7877, 2020. doi: 10.1109/TIT.2020.3028439.

Mohammad Akbari, Bahman Gharesifard, and Tamás Linder. Distributed online convex optimization on time-varying directed graphs. *IEEE Transactions on Control of Network Systems*, 4(3):417–428, 2015.

Alyazeed Albasyoni, Mher Safaryan, Laurent Condat, and Peter Richtárik. Optimal gradient compression for distributed and federated learning. *arXiv preprint arXiv:2010.03246*, 2020.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, pages 1709–1720, 2017.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

Xuanyo Cao and Tamer Başar. Decentralized online convex optimization with event-triggered communications. *IEEE Transactions on Signal Processing*, 69:284–299, 2021. doi: 10.1109/TSP.2020.3044843.

Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.

Nicolò Cesa-Bianchi, Tommaso Cesari, and Claire Monteleoni. Cooperative online learning: Keeping your neighbors updated. In *Proceedings of the 31th International Conference on Algorithmic Learning Theory (ALT)*, pages 234–250, 2020.

Ashok Cutkosky. Artificial constraints and hints for unbounded online learning. In *Proceedings of the 32nd Annual Conference on Learning Theory (COLT)*, pages 874–894, 2019a.

Ashok Cutkosky. Combining online learning guarantees. In *Proceedings of the 32th Annual Conference on Learning Theory (COLT)*, pages 895–913, 2019b.

Ashok Cutkosky. Parameter-free, dynamic, and strongly-adaptive online learning. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 2250–2259, 2020.

Ashok Cutkosky and Kwabena Boahen. Online learning without prior information. In *Proceedings of the 30th Annual Conference on Learning Theory (COLT)*, pages 643–677, 2017.

Ashok Cutkosky and Francesco Orabona. Black-box reductions for parameter-free online learning in banach spaces. In *Proceedings of the 31th Annual Conference on Learning Theory (COLT)*, pages 1493–1529, 2018.

John Duchi, Alekh Agarwal, and Martin Wainwright. Distributed dual averaging in networks. In *Advances in Neural Information Processing Systems 23 (NeurIPS)*, pages 550–558, 2010.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Tim van Erven and Wouter M. Koolen. Metagrad: Multiple learning rates in online learning. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, pages 3666–3674. Neural Information Processing Systems Foundation, 2017.

Tim van Erven, Wouter M. Koolen, and Dirk van der Hoeven. Metagrad: Adaptation using multiple learning rates in online learning. *Journal of Machine Learning Research*, 22(161):1–61, 2021.

Fartash Faghri, Iman Tabrizian, Ilia Markov, Dan Alistarh, Daniel M Roy, and Ali Ramezani-Kebrya. Adaptive gradient quantization for data-parallel sgd. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, pages 3174–3185, 2020.

Dylan J. Foster, Satyen Kale, Mehryar Mohri, and Karthik Sridharan. Parameter-free online learning via model selection. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, pages 6020–6030, 2017.

Subhashis Ghosal and Aad van der Vaart. *Fundamentals of Nonparametric Bayesian Inference*. Cambridge University Press, 2017.

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

Jean-Baptiste Hiriart-Urruty. A note on the Legendre-Fenchel transform of convex composite functions. In *Nonsmooth Mechanics and Analysis*, pages 35–46. Springer, 2006.

Dirk van der Hoeven. User-specified local differential privacy in unconstrained adaptive online learning. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pages 14103–14112, 2019.

Dirk van der Hoeven, Ashok Cutkosky, and Haipeng Luo. Comparator-adaptive convex bandits. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, 2020.

Saghar Hosseini, Airlie Chapman, and Mehran Mesbahi. Online distributed optimization via dual averaging. In *52nd IEEE Conference on Decision and Control*, pages 1484–1489, 2013.

Yu-Guan Hsieh, Franck Iutzeler, Jérôme Malick, and Panayotis Mertikopoulos. Multi-agent online optimization with delays: Asynchronicity, adaptivity, and optimism. *arXiv preprint arXiv:2012.11579*, 2020.

Pooria Joulani, Andras Gyorgy, and Csaba Szepesvári. Delay-tolerant online convex optimization: Unified analysis and adaptive-gradient algorithms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Kwang-Sung Jun and Francesco Orabona. Parameter-free online convex optimization with sub-exponential noise. *Proceedings of the 32nd Annual Conference on Learning Theory (COLT)*, pages 1802–1823, 2019.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan

Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Michal Kempka, Wojciech Kotłowski, and Manfred K. Warmuth. Adaptive scale-invariant online algorithms for learning linear models. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 3321–3330, 2019.

Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 3478–3487, Long Beach, California, USA, 09–15 Jun 2019.

Wouter M. Koolen and Tim van Erven. Second-order quantile methods for experts and combinatorial games. In *Proceedings of the 28th Annual Conference on Learning Theory (COLT)*, pages 1155–1175, 2015.

Wojciech Kotłowski. Scale-invariant unconstrained online learning. In *Proceedings of the 28th International Conference on Algorithmic Learning Theory (ALT)*, pages 412–433, 2017.

David Mateos-Núñez and Jorge Cortés. Distributed online convex optimization over jointly connected digraphs. *IEEE Transactions on Network Science and Engineering*, 1(1):23–37, 2014. doi: 10.1109/TNSE.2014.2363554.

Prathamesh Mayekar and Himanshu Tyagi. Limits on gradient compression for stochastic optimization. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 2658–2663. IEEE, 2020a.

Prathamesh Mayekar and Himanshu Tyagi. RATQ: A universal fixed-length quantizer for stochastic optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1399–1409, 2020b.

Brendan McMahan and Francesco Orabona. Unconstrained online linear learning in Hilbert spaces: Minimax algorithms and normal approximations. In *Proceedings of the 27th Annual Conference on Learning Theory (COLT)*, pages 1020–1039, 2014.

Brendan McMahan and Matthew Streeter. Delay-tolerant algorithms for asynchronous distributed online learning. In *Advances in Neural Information Processing Systems 27 (NeurIPS)*, pages 2915–2923, 2014.

Scott McQuade and Claire Monteleoni. Global climate model tracking using geospatial neighborhoods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, 2012.

Scott McQuade and Claire Monteleoni. Spatiotemporal global climate model tracking. In *Large-Scale Machine Learning in the Earth Sciences*, pages 33–54. Chapman and Hall/CRC, 2017.

Zakaria Mhammedi and Wouter M Koolen. Lipschitz and comparator-norm adaptivity in online learning. In *Proceedings of the 33rd Annual Conference on Learning Theory (COLT)*, pages 2858–2887, 2020.

Francesco Orabona. Dimension-free exponentiated gradient. In *Advances in Neural Information Processing Systems 26 (NeurIPS)*, pages 1806–1814, 2013.

Francesco Orabona and Dávid Pál. Coin betting and parameter-free online learning. In *Advances in Neural Information Processing Systems 29 (NeurIPS)*, pages 577–585, 2016.

Michael Rabbat and Robert Nowak. Distributed optimization in sensor networks. In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, pages 20–27, 2004.

Kevin Scaman, Francis Bach, Sebastien Bubeck, Laurent Massoulié, and Yin Tat Lee. Optimal algorithms for non-smooth distributed optimization in networks. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, volume 31, pages 2740–2749, 2018.

Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 1-bit stochastic gradient descent and application to data-parallel distributed training of speech DNNs. In *Interspeech 2014*, September 2014.

Shahin Shahrampour and Ali Jadbabaie. Distributed online optimization in dynamic environments using mirror descent. *IEEE Transactions on Automatic Control*, 63(3):714–725, 2018. doi: 10.1109/TAC.2017.2743462.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and trends in Machine Learning*, 4(2):107–194, 2011.

Ohad Shamir. Fundamental limits of online and distributed algorithms for statistical learning and estimation. In *Advances in Neural Information Processing Systems 27 (NeurIPS)*, volume 27, pages 163–171, 2014.

Shaohuai Shi, Qiang Wang, Kaiyong Zhao, Zhenheng Tang, Yuxin Wang, Xiang Huang, and Xiaowen Chu. A distributed synchronous sgd algorithm with global top-k sparsification for low bandwidth networks. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 2238–2247. IEEE, 2019.

Sebastian U. Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified SGD with memory. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, pages 4447–4458, 2018.

Ananda Theertha Suresh, X Yu Felix, Sanjiv Kumar, and H Brendan McMahan. Distributed mean estimation with limited communication. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 3329–3337, 2017.

Botond Szabo, Lasse Vuursteen, and Harry van Zanten. Optimal distributed testing in high-dimensional gaussian models. *arXiv preprint arXiv:2012.04957*, 2020.

Hanlin Tang, Shaoduo Gan, Ce Zhang, Tong Zhang, and Ji Liu. Communication compression for decentralized training. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, pages 7652–7662, 2018.

Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi. Powergossip: Practical low-rank communication compression in decentralized deep learning. *arXiv preprint arXiv:2008.01425*, 2020.

Guanghui Wang, Shiyin Lu, and Lijun Zhang. Adaptivity and optimality: A universal algorithm for online convex optimization. In Amir Globerson and Ricardo Silva, editors, *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 659–668, 2019.

Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, pages 1509–1519, 2017.

Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. CSER: Communication-efficient SGD with error reset. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, 2020.

Feng Yan, Shreyas Sundaram, SVN Vishwanathan, and Yuan Qi. Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties. *IEEE Transactions on Knowledge and Data Engineering*, 25(11):2483–2493, 2013.

Yan Zhang, Robert J. Ravier, Michael M. Zavlanos, and Vahid Tarokh. A distributed online convex optimization algorithm with improved dynamic regret. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 2449–2454. IEEE, 2019.

Martin Zinkevich, John Langford, and Alex J. Smola. Slow learners are fast. In *Advances in Neural Information Processing Systems 22 (NeurIPS)*, pages 2331–2339, 2009.

Appendix A. Analysis of Unprojected Online Gradient Descent in One Dimension in the DOCO-JC Setting

The role of this section is to provide context for the one-dimensional algorithm in Section 2. To do so, we analyse an unprojected version of Online Gradient Descent (OGD) in a simplified setting: in one dimension with a constant learning rate η on each node, and with no communication limits. We denote the gradient (a real number) at time t by h_t, and use the notation defined in the introduction.

As mentioned in Section 2, one of the principal technical challenges that Algorithm 1 overcomes is that at prediction time, some gradients are not available to the agents. Let us see how missing gradients influence the regret of OGD. Denote by $H_t(I_t) = \sum_{s \in \mathcal{S}_{I_t}} h_s$ the sum of the gradients available at prediction time on the active node I_t in round t, and denote by $H_t = \sum_{s<t} h_s$ the sum of *all* gradients before round t.

For comparison, define w_t to be the sequence of predictions that OGD would output if all gradients were immediately available, that is, $w_t = -\eta H_t$. These predictions enjoy the standard regret bound

$$\sum_{t=1}^{T} (w_t - u) h_t \leqslant \frac{|u|^2}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} |h_t|^2 \tag{11}$$

against any comparator $u \in \mathbb{R}$. To bound the regret of OGD with only the available gradients, $w_t(I_t) = -\eta H_t(I_t)$, observe that

$$\begin{aligned}\sum_{t=1}^{T} \big(w_t(I_t) - u\big)h_t &= \sum_{t=1}^{T}(w_t - u)h_t + \sum_{t=1}^{T}(w_t(I_t) - w_t)h_t \\ &\leqslant \frac{|u|^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^{T}|h_t|^2 + \sum_{t=1}^{T}\big(w_t(I_t) - w_t\big)h_t,\end{aligned}$$

where we used (11). Recall that $\gamma(t)$ is the set of indices of the missing gradients at the active agent in round t. We have that $(w_t(I_t) - w_t)g_t = \eta h_t \sum_{s\in\gamma(t)} h_s \leqslant \eta|h_t| \sum_{s\in\gamma(t)} |h_s|$, which is roughly the term $\widehat{\zeta}(s)$ that appears in the definition of our potential function in (9). The regret can then be bounded by

$$\sum_{t=1}^{T}(w_t(I_t) - u)h_t \leqslant \frac{|u|^2}{2\eta} + \eta\sum_{t=1}^{T}\left(\frac{1}{2}h_t^2 + |h_t|\sum_{s\in\gamma(t)}|h_s|\right).$$

An ideal tuning of the learning rate would be to optimize the expression above over η and set

$$\eta = |u|\left(\sum_{t=1}^{T}\left(h_t^2 + 2|h_t|\sum_{s\in\gamma(t)}|h_s|\right)\right)^{-2},$$

to obtain

$$\sum_{t=1}^{T}(w_t(I_t) - u)h_t \leqslant |u|\sqrt{2\sum_{t=1}^{T}\left(|h_t|^2 + 2|h_t|\sum_{s\in\gamma(t)}|h_s|\right)}.$$

This is exactly the type of regret bound that would be suitable to learn a $\mathcal{Q}$-partition. However, this ideal tuning is not possible for two reasons: we know neither $|u|$ nor $\sum_{t=1}^{T}\left(|h_t|^2 + 2|h_t|\sum_{s\in\gamma(t)}|h_s|\right)$.

Let us define $\lambda_t = |h_t|^2 + 2|h_t|\sum_{s\in\gamma(t)}|h_s|$ in the following discussion to reduce clutter. Note that the active node I_t at round t may not be able to compute λ_s for some $s < t$. Indeed, gradients missing at past rounds might not have reached I_t yet, making it impossible to compute some gradient in $\gamma(s)$. Even with knowledge of $|u|$, this rules out natural ideas using learning rate schemes such as $\eta_t = \sqrt{|u|^2/\sum_{s<t}\lambda_s}$.

As discussed in the introduction, a considerable amount of effort has been made in the literature to obtain approximations of the optimal learning rate in the delayed OCO setting. However, to our knowledge, none of these existing approaches apply to comparator-adaptive algorithms. Similarly, although adapting to $|u|$ is relatively well understood in the standard setting, there are no straightforward extensions to deal with missing gradients.

Algorithm 1 solves these challenges simultaneously. As illustrated above, a crucial part of the difficulty lies in trying to adapt the unknown quantity $|h_t|\sum_{s\in\gamma(t)}|h_s|$, which appears in the ideal learning rate. We accomplish this by showing that only knowing an approximation of this quantity is sufficient: our prediction v_t in Algorithm 1 contains the approximation $\widehat{\zeta}_{I_t}$, which plays a similar role as $|h_t|\sum_{s\in\gamma(t)}|h_s|$, in the sense that it reduces the learning rate of the algorithm to account for

the extra uncertainty due to the missing gradients. Our analysis reveals that this approximation does not hurt the regret bound of the algorithm significantly, and that we can still recover the optimal comparator-adaptive regret bound.

Appendix B. Details of Section 2

The predictions v_t in Algorithm 1 can be computed as follows. Let $L_{I_t} = \sum_{s\in\mathcal{S}_{I_t}(t)}(\widehat{h}_s+\varepsilon)$ and $V_{I_t} = 1 + \sum_{s\in\mathcal{S}_{I_t}(t)}\left((\widehat{h}_s+\varepsilon)^2 + 2\eta^2\widehat{\zeta}_{I_t}(s)\right)$. Then

$$v_t = \sqrt{\pi}\exp\left(\frac{L_{I_t}^2}{4V_{I_t}}\right)\left(\operatorname{erf}\left(\frac{2aV_{I_t}+L_{I_t}}{2\sqrt{V_{I_t}}}\right) - \operatorname{erf}\left(\frac{L_{I_t}}{2\sqrt{V_{I_t}}}\right)\right)\left(2Z\sqrt{V_{I_t}}\right)^{-1}. \tag{12}$$

We refer the reader to Appendix B of Koolen and Van Erven (2015) for numerically stable evaluation.

Lemma 3 *Suppose $\|\boldsymbol{z}_t\| \leqslant 1$, $\|\boldsymbol{g}_t\| \leqslant G$, and $\|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\| \leqslant \varepsilon$ for all t. Then the predictions v_t defined in Algorithm 1 satisfy (8).*

Proof As a first step, observe that by the Cauchy-Schwarz inequality and the condition on $\boldsymbol{z}_t$ we have that

$$|\widehat{h}_t - h_t| = |\langle \boldsymbol{z}_t, \widehat{g}_t\rangle - \langle \boldsymbol{z}_t, \boldsymbol{g}_t\rangle| \leqslant \|\boldsymbol{z}_t\|\|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\| \leqslant \varepsilon.$$

Next, we replace w_t with its definition,

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\eta\sim\rho}\left[\nu\exp\left(-\sum_{s=1}^{t-1}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2\sum_{i\in\gamma(s)}|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)|\right)\right)\right] - v_t h_t \\
&= \mathop{\mathbb{E}}_{\eta\sim\rho}\left[\nu\exp\left(-\sum_{s=1}^{t-1}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2\sum_{i\in\gamma(s)}|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)|\right)\right)\right] - \\
&\mathop{\mathbb{E}}_{\eta\sim\rho}\left[\nu\exp\left(-\sum_{s\in\mathcal{S}_{I_t}(t)}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2\sum_{i\in\gamma_{I_t}(s)}|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)|\right)\right)\eta\right] h_t\,.
\end{aligned}$$

Denote by $\widehat{\Omega}_T$ the sum, which is the central part of the update w_t,

$$\widehat{\Omega}_t = \sum_{s\in\mathcal{S}_{I_t}(t)}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2\sum_{i\in\gamma_{I_t}(s)}|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)|\right).$$

We factor this $\exp(-\widehat{\Omega}_t)$ term in the expression above, so that it is equal to

$$\begin{aligned}
&\mathbb{E}_{\eta\sim\rho}\Bigg[\nu\exp\left(-\widehat{\Omega}_t\right) \\
&\times\left(\exp\left(-\sum_{s\in[t-1]\setminus\mathcal{S}_{I_t}(t)}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2\sum_{i\in\gamma(s)}|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)|\right)\right)\right. \\
&\left.\times\exp\left(-\sum_{s\in\mathcal{S}_{I_t}(t)}2\eta^2\sum_{i\in\gamma(s)\setminus\gamma_{I_t}(s)}|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)|\right)-\eta h_t\right)\Bigg].
\end{aligned}$$

Consider the double sum in this last term, right above,

$$\sum_{s\in\mathcal{S}_{I_t}(t)}\sum_{i\in\gamma(s)\setminus\gamma_{I_t}(s)}\left|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)\right|.$$

Let us switch the order of summation here. To do so, note that if a gradient $\widehat{h}_i$ is available to node I_t at time t, and if it was unavailable when $\widehat{g}_s$ appeared, then it would be used in $\gamma_{I_t}(s)$. In other words, if $i\in\gamma(s)\setminus\gamma_{I_t}(s)$ then, $i\notin\mathcal{S}_{I_t}(t)$. Therefore, when s varies in $S_{I_t}(t)$, the set of values taken by $i\in\gamma(s)\setminus\gamma_{I_t}(s)$ is in fact $[t-1]\setminus S_{I_t}(t)$. When switching the sums, we may thus restrict the values taken by i to $[t-1]\setminus S_{I_t}(t)$ and write

$$\sum_{s\in\mathcal{S}_{I_t}(t)}\sum_{i\in\gamma(s)\setminus\gamma_{I_t}(s)}\left|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)\right| = \sum_{i\in[t-1]\setminus S_{I_t}}\sum_{\substack{s\in\mathcal{S}_{I_t}(t)\text{ s.t.}\\ i\in\gamma(s)\setminus\gamma_{I_t}(s)}}\left|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)\right|. \tag{13}$$

Finally, we also switch the notation for i and s in the indices, factor out the common $\widehat{h}_s+\varepsilon$ term, and incorporate this sum with the preceding term, obtaining

$$\sum_{s\in[t-1]\setminus\mathcal{S}_{I_t}(t)}\Bigg(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2|\widehat{h}_s+\varepsilon|\underbrace{\Big(\sum_{i\in\gamma(s)}|(\widehat{h}_i+\varepsilon)|+\sum_{\substack{i\in\mathcal{S}_{I_t}(t)\text{ s.t.}\\ s\in\gamma(i)\setminus\gamma_{I_t}(i)}}|\widehat{h}_i+\varepsilon|\Big)}_{:=\Theta_{I_t}(s)}.\Bigg)$$

Furthermore, since a new gradient takes less than $D(\mathcal{G})$ rounds to reach all nodes, there can be only at most $D(\mathcal{G})$ new gradients for which $\gamma_{I_t}(s)\neq\gamma(s)$. Thus the last sum has at most $D(\mathcal{G})$ terms, and the freshly defined $\Theta_{I_t}(s)$ involves at most $2D(\mathcal{G})$ terms.

We may now apply Lemma 11, with $x=\eta g_t$, to see that for any η,

$$\begin{aligned}
&\exp\Bigg(-\sum_{s\in[t-1]\setminus\mathcal{S}_{I_t}(t)}\Big(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2|\widehat{h}_s+\varepsilon|\Theta_{I_t}(s)\Big)\Bigg)-\eta h_t\\
&\geqslant\exp\Bigg(-\sum_{s\in[t-1]\setminus\mathcal{S}_{I_t}(j)}\Big(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2|\widehat{h}_s+\varepsilon|\Theta_{I_t}(s)\Big)\Bigg)\\
&\times\exp\Bigg(-\Big(\eta h_t+\eta^2h_t^2+2\eta^2\sum_{i\in[t-1]\setminus\mathcal{S}_{I_t}(t)}|h_t(\widehat{h}_i+\varepsilon)|\Big)\Bigg).
\end{aligned}$$

Thus, upon multiplying by $\exp(-\widehat{\Omega}_t)$, and after integrating over $\eta \sim \rho$, we obtain

$$\begin{aligned}
&\mathbb{E}_{\eta\sim\rho}\left[\nu\exp\left(-\sum_{s=1}^{t-1}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2\sum_{i\in\gamma(s)}|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)|\right)\right)\right]-v_th_t\\
&\geqslant \mathbb{E}_{\eta\sim\rho}\left[\nu\exp\left(-\sum_{s\in\mathcal{S}_{I_t}(t)}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2\right)+2\eta^2\sum_{i\in\gamma_{I_t}(s)}|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)|\right)\right.\\
&\quad\times\exp\left(-\sum_{s\in[t-1]\setminus\mathcal{S}_{I_t}(j)}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2|\widehat{h}_s+\varepsilon|\,\Theta_{I_t}(s)\right)\right)\\
&\quad\left.\times\exp\left(-\left(\eta h_t+\eta^2h_t^2+2\eta^2\sum_{i\in[t-1]\setminus\mathcal{S}_{I_t}(t)}|h_t(\widehat{h}_i+\varepsilon)|\right)\right)\right].
\end{aligned}$$

Define $f(x)=-x-x^2-2|x|y$, where $y=\sum_{i\in\gamma(t)}|\eta(\widehat{h}_i+\varepsilon)|>0$. We have that $f'(x)\leqslant 0$ for $x\geqslant-\frac{1}{4}$ and $y\in[0,\frac{1}{4}]$. Since $\sum_{i\in\gamma(t)}|\eta(\widehat{h}_i+\varepsilon)|\leqslant 1/4$ by the restriction on η, the function $f(x)$ is non-increasing for $x>-1/4$. Since $-\frac{1}{4}\leqslant\eta h_t\leqslant\eta(\widehat{h}_t+\varepsilon)$ we have that $f(\eta h_t)\geqslant f\big(\eta(\widehat{h}_t+\varepsilon)\big)$, which gives us

$$\begin{aligned}
&\mathbb{E}_{\eta\sim\rho}\left[\nu\exp\left(-\sum_{s=1}^{t-1}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2\sum_{i\in\gamma(s)}|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)|\right)\right)\right]-v_th_t\\
&\geqslant \mathbb{E}_{\eta\sim\rho}\left[\nu\exp\left(-\sum_{s\in\mathcal{S}_{I_t}(t)}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2\right)+2\eta^2\sum_{i\in\gamma_{I_t}(s)}|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)|\right)\right.\\
&\quad\times\exp\left(-\sum_{s\in[t-1]\setminus\mathcal{S}_{i_t}(j)}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2|\widehat{h}_s+\varepsilon|\Theta_{I_t}(s)\right)\right)\\
&\quad\left.\times\exp\left(-\left(\eta(\widehat{h}_t+\varepsilon)+\eta^2(\widehat{h}_t+\varepsilon)^2+2\eta^2\sum_{i\in\gamma(t)}|(\widehat{h}_t+\varepsilon)(\widehat{h}_i+\varepsilon)|\right)\right)\right]\\
&=\mathbb{E}_{\eta\sim\rho}\left[\nu\exp\left(-\sum_{s=1}^{t}\left(\eta(\widehat{h}_s+\varepsilon)+\eta^2(\widehat{h}_s+\varepsilon)^2+2\eta^2\sum_{i\in\gamma(s)}|(\widehat{h}_s+\varepsilon)(\widehat{h}_i+\varepsilon)|\right)\right)\right],
\end{aligned}$$

which completes the proof. ■

Theorem 10 *Suppose that $\|\boldsymbol{g}_t\|\leqslant G$ and $\|\widehat{\boldsymbol{g}}_t-\boldsymbol{g}_t\|\leqslant\varepsilon$ for all t. For all $u\in\mathbb{R}_+$ and $\nu>0$, Algorithm 1 satisfies the following regret bound:*

$$\begin{aligned}
&\sum_{t=1}^{T}v_th_t-\sum_{t=1}^{T}uh_t\leqslant\nu+2uT\varepsilon+\\
&|u|\max\left\{264\,G\tau\ln_+\Big(\frac{312\,|u|G\,\tau}{\nu}\Big),\sqrt{8\Big(\Lambda_t^h+24TG\varepsilon+1\Big)\ln_+\Big(\frac{2036\,u^2T\tau G^2}{\nu^2}\Big)}\right\}.
\end{aligned}$$

where $\ln_+(x) = \ln(\max(e, x))$ *and*

$$\Lambda_T^h = \sum_{t=1}^{T} \left(h_t^2 + 2|h_t| \sum_{s \in \gamma(t)} |h_s| \right).$$

Proof First, by repeatedly applying Lemma 3 we find that

$$\nu - \sum_{t=1}^{T} v_t h_t \geqslant \mathbb{E}_{\eta \sim \rho} \left[\nu \exp\left(- \sum_{t=1}^{T} \left(\eta(\widehat{h}_t + \varepsilon) + \eta^2 (\widehat{h}_t + \varepsilon)^2 + 2\eta^2 \zeta(t) \right) \right) \right]. \tag{14}$$

Now, consider the case in which $-\sum_{t=1}^{T}(\widehat{h}_t + \varepsilon) \leqslant \sqrt{2\sum_{t=1}^{T}\left((\widehat{h}_t + \varepsilon)^2 + 2\zeta(t)\right)}$. In this case we have:

$$\begin{aligned}
\sum_{t=1}^{T} v_t h_t - \sum_{t=1}^{T} u h_t \leqslant & - \Phi\left(- \sum_{t=1}^{T} (\widehat{h}_t + \varepsilon) \right) + \nu - \sum_{t=1}^{T} u h_t \\
\leqslant & \nu - \sum_{t=1}^{T} u h_t \\
\leqslant & \nu + uT\varepsilon - \sum_{t=1}^{T} u \widehat{h}_t && (u \geqslant 0) \\
= & \nu + 2uT\varepsilon - \sum_{t=1}^{T} u(\widehat{h}_t + \varepsilon) \\
\leqslant & \nu + 2uT\varepsilon + u\sqrt{2\sum_{t=1}^{T}\left((\widehat{h}_t + \varepsilon)^2 + 2\zeta(t)\right)},
\end{aligned}$$

which implies the result.

Next, we consider the case in which $-\sum_{t=1}^{T}(\widehat{h}_t + \varepsilon) > \sqrt{2\sum_{t=1}^{T}\left((\widehat{h}_t + \varepsilon)^2 + 2\zeta(t)\right)}$. By Fenchel's inequality we have

$$\begin{aligned}
-\sum_{t=1}^{T} v_t h_t \geqslant & \Phi_T\left(- \sum_{t=1}^{T} (\widehat{h}_t + \varepsilon) \right) - \nu \\
\geqslant & - u \sum_{t=1}^{T} (\widehat{h}_t + \varepsilon) - \Phi_T^\star(u) - \nu \\
\geqslant & - u \sum_{t=1}^{T} h_t - 2uT\varepsilon - \Phi_T^\star(u) - \nu \,,
\end{aligned}$$

where $\Phi_T^\star$ is the convex conjugate of Φ_T. Using the upper bound on $\Phi^\star(u)$ from Lemma 13 we get,

$$\sum_{t=1}^{T} v_t h_t - \sum_{t=1}^{T} u h_t \leqslant \nu + 2uT\varepsilon + \max\left\{ u\,44(G+\varepsilon)(2D(\mathcal{G})+1)\left(\ln\left(|u|44(G+\varepsilon)(D(\mathcal{G})+1)\right) - 1 + \ln\frac{1}{2}\left(\frac{\pi}{\nu^2}\right)\right), \sqrt{8u^2\left(\sum_{t=1}^{T}\left(\left(\widehat{h}_t+\varepsilon\right)^2 + 2\widehat{\zeta}(t)\right)+1\right)\ln\left(24u^2\left(\sum_{t=1}^{T}\left(\left(\widehat{h}_t+\varepsilon\right)^2 + 2\widehat{\zeta}(t)\right)+1\right)\frac{\pi}{\nu^2}+1\right)}\right\},$$

Simplifying the bound Since $|\widehat{h}_t - h_t| \leqslant \varepsilon$, we have, using $\varepsilon \leqslant G$ and $1 \leqslant \tau$,

$$\begin{aligned}(\widehat{h}_t+\varepsilon)^2 + 2\widehat{\zeta}(t) \leqslant & (h_t+2\varepsilon)^2 + 2\sum_{s\in\gamma(t)} |(h_t+2\varepsilon)(h_s+2\varepsilon)| \\ \leqslant & h_t^2 + 4G\varepsilon + 4\varepsilon^2 + 8\tau(G\varepsilon+\varepsilon^2) + 2|h_t|\sum_{s\in\gamma(t)}|h_s| \,. \\ \leqslant & h_t^2 + 2|h_t|\sum_{s\in\gamma(t)}|h_s| + 24G\tau\varepsilon\,.\end{aligned} \tag{15}$$

Therefore, by summing over $t \in [T]$:

$$\sum_{t=1}^{T}\left(\left(\widehat{h}_t+\varepsilon\right)^2 + 2\widehat{\zeta}(t)\right) \leqslant \Lambda_t^h + 24\varepsilon GT \leqslant 27\,G^2\tau T\,.$$

We shall use the first inequality to bound the main term inside the square root, and the second cruder bound to bound the term inside the logarithm.

Crudely bounding the other terms with $\varepsilon \leqslant G$ and $1 \leqslant \tau$ we get that:

$$\sum_{t=1}^{T} v_t h_t - \sum_{t=1}^{T} u h_t \leqslant \nu + 2uT\varepsilon + |u| \max\left\{ 264\,G\tau\ln_+\left(\frac{312\,|u|G\,\tau}{\nu}\right), \sqrt{8\left(\Lambda_t^h + 24TG\varepsilon + 1\right)\ln_+\left(\frac{2036\,u^2\tau TG^2}{\nu^2}\right)}\right\}.$$

■

Lemma 11 *For $x, y_1, \ldots, y_\tau \in [-1/20(1+\tau), 1/20(1+\tau)]$ and $a_i \in [0, 1/20]$, we have*

$$\exp\left(\sum_{i=1}^{\tau}\left(-y_i - y_i^2 - 2a_i|y_i| - 2|xy_i|\right) - x - x^2\right) \leqslant \exp\left(\sum_{i=1}^{\tau}\left(-y_i - y_i^2 - 2a_i|y_i|\right)\right) - x\,.$$

Proof In the case where $x = 0$ the inequality holds trivially. Define the function $f(x) = x + x^2 + \sum_{i=1}^{\tau} y_i + y_i^2 + 2|x||y_i|$. We have that

$$\begin{aligned}
&\exp\left(\sum_{i=1}^{\tau}\left(-y_i - y_i^2 - 2a_i|y_i| - 2|xy_i|\right) - x - x^2\right) \\
&= \exp\left(\sum_{i=1}^{\tau}\left(-2a_i|y_i|\right) - f(x)\right) \\
&\leqslant \exp\left(\Big(\sum_{i=1}^{\tau} -2a_i|y_i|\Big) - f(0)\right) - \exp\left(\Big(\sum_{i=1}^{\tau} -2a_i|y_i|\Big) - f(0)\right)(\partial f(0))x \\
&= \exp\left(\Big(\sum_{i=1}^{\tau} -2a_i|y_i|\Big) - f(0)\right) - \exp\left(\sum_{i=1}^{\tau}\left(-y_i - y_i^2 - 2a_i|y_i|\right)\right)(\partial f(0))\,x,
\end{aligned}$$

where $\partial f(x)$ denotes a subdifferential of f evaluated at x and we used that $\exp(-f(x)) \leqslant \exp(-f(0)) - \exp(-f(0))(\partial f(0))(x - 0)$ since $\exp(-f)$ is concave by Lemma 12. If $x > 0$ then we set $\partial f(0) = \sum_{i=1}^{\tau} 2|y_i| + 1$ and

$$\begin{aligned}
&- \exp\left(\sum_{i=1}^{\tau}\left(-y_i - y_i^2 - 2a_i|y_i|\right)\right)\left(\sum_{i=1}^{\tau} 2|y_i| + 1\right)x \\
&\leqslant - \exp\left(\sum_{i=1}^{\tau}\left(2|y_i| - y_i - y_i^2 - 2a_i|y_i|\right) - 4\Big(\sum_{i=1}^{\tau}|y_i|\Big)^2\right)x && \text{(prod bound)} \\
&\leqslant - \exp\left(\sum_{i=1}^{\tau}\left(|y_i| - y_i^2 - 2a_i|y_i|\right) - 4\Big(\sum_{i=1}^{\tau}|y_i|\Big)^2\right)x \\
&\leqslant - \exp\left(\sum_{i=1}^{\tau}\left(|y_i| - 5y_i^2 - 2\Big(a_i + \frac{4}{20}\Big)|y_i|\right)\right)x && (|y_i| \leqslant \tfrac{1}{20(\tau+1)}) \\
&\leqslant - \exp\left(\sum_{i=1}^{\tau}\left(\frac{1}{2}|y_i| - 5y_i^2\right)\right)x && (|a_i| \leqslant \tfrac{1}{20}) \\
&\leqslant -x
\end{aligned}$$

where the prod bound is $1 + v \geqslant \exp(v - v^2)$ for $v > -1/2$ (see Lemma 2.4 by Cesa-Bianchi and Lugosi (2006)) and the last inequality follows because $\frac{1}{2}|v| - 5v^2$ is non-negative for when

$|v| \leqslant 1/10$. If $x < 0$ then we set $\partial f(0) = 1 - \sum_{i=1}^{\tau} 2|y_i|$ and

$$
\begin{aligned}
& - \exp\left(\sum_{i=1}^{\tau}\left(-y_i - y_i^2 - 2a|y_i|\right)\right)\left(1 - \sum_{i=1}^{\tau} 2|y_i|\right) x \\
& \leqslant - \exp\left(\sum_{i=1}^{\tau}\left(-y_i - y_i^2\right)\right)\left(1 - \sum_{i=1}^{\tau} 2|y_i|\right) x \\
& \leqslant - \left(\prod_{i=1}^{\tau}(1 - y_i)\right)\left(1 - \sum_{i=1}^{\tau} 2|y_i|\right) x && \text{(repeated prod bound)} \\
& \leqslant - \left(\prod_{i=1}^{\tau}(1 + |y_i|)\right)\left(1 - \sum_{i=1}^{\tau} 2|y_i|\right) x \\
& \leqslant - \left(\prod_{i=1}^{\tau}(1 + |y_i|)\right)\left(\prod_{i=1}^{\tau}(1 - 2|y_i|)\right) x && \text{(Weierstrass inequality)} \\
& = - \left(\prod_{i=1}^{\tau}(1 + |y_i|)(1 - 2|y_i|)\right) x \\
& \leqslant -x, && ((1 + |y|)(1 - 2|y|) \leqslant 1 \text{ and } x < 0)
\end{aligned}
$$

which completes the proof. ■

Lemma 12 *Define* $f : x \mapsto x + x^2 + \sum_{i=1}^{\tau}\left(y_i + y_i^2 + 2|x||y_i|\right)$ *, the function* $g = \exp(-f)$ *is concave on the interval* $[-1/10,\, 1/10]$*, as soon as* $y_1, \ldots, y_\tau$ *satisfy* $|y_i| \leqslant 1/(10\tau)$.

Proof The function $g : x \mapsto \exp(-f(x))$ is continuous on $\mathbb{R}$, and twice differentiable on $(-\infty, 0)$ and on $(0, +\infty)$ with,

$$
g'(x) = \begin{cases} \left(-1 - 2x + 2\sum_{i=1}^{\tau}|y_i|\right)\exp(-f(x)) & \text{if} \quad x < 0 \\ \left(-1 - 2x - 2\sum_{i=1}^{\tau}|y_i|\right)\exp(-f(x)) & \text{if} \quad x > 0\,. \end{cases}
$$

and

$$
g''(x) = \begin{cases} \left(\left(-1 - 2x + 2\sum_{i=1}^{\tau}|y_i|\right)^2 - 2\right)\exp(-f(x)) & \text{if} \quad x < 0 \\ \left(\left(-1 - 2x - 2\sum_{i=1}^{\tau}|y_i|\right)^2 - 2\right)\exp(-f(x)) & \text{if} \quad x > 0\,. \end{cases}
$$

Note that $g''(x) < 0$ for all x, as

$$
1 + 2x + 2\sum_{i=1}^{\tau}|y_i| \leqslant 1 + \frac{2}{10} + \frac{2}{10} \leqslant \sqrt{2}\,.
$$

Finally, $\lim_{x\to 0^-} g'(x) \geqslant \lim_{x\to 0^+} g'(x)$, concluding the proof of the concavity. ■

Lemma 13 *Suppose* $L > \sqrt{2(V+1)}$. *Let* $\Phi(L) = \nu\,\mathbb{E}_{\eta\sim\rho}[\exp(\eta L - \eta^2 V)]$ *with* $\mathrm{d}\rho(\eta) \propto \exp(-\eta^2)\mathrm{d}\eta$, *where* $\eta \in [0, \frac{1}{20(G+\varepsilon)(2\tau+1)}]$. *If* $L \leqslant \frac{1}{20(G+\varepsilon)(\tau+1)}(V+1)$ *then for* $u \geqslant 0$

$$\Phi^\star(u) \leqslant \sqrt{8u^2(V+1)\ln\left(24u^2(V+1)\frac{\pi}{\nu^2}+1\right)}.$$

If $L \geqslant \frac{1}{20(G+\varepsilon)(2\tau+1)}(V+1)$ *then*

$$\Phi^\star(u) \leqslant 44u(G+\varepsilon)(\tau+1)\left(\ln\left(44u(G+\varepsilon)(\tau+1)\right) - 1 + \frac{1}{2}\ln\left(\frac{\pi}{\nu^2}\right)\right).$$

Proof The proof of this lemma is a slight variation of the proof of Lemma 8 of Van der Hoeven (2019) and a similar result has been obtained by Jun and Orabona (2019). The initial part of the analysis is parallel to the analysis of Theorem 3 by Koolen and Van Erven (2015). Denote by $B = V+1$ and by $Z = \int_0^{\frac{1}{20(G+\varepsilon)(2\tau+1)}} \exp(-\eta^2)$. For $\eta \leqslant \widehat{\eta} = \frac{L}{2B}$, the function $\eta \mapsto \eta L - \eta^2 B$ is non-decreasing. Therefore, for $[v,\mu] \subseteq [0, \left(20(G+\varepsilon)(2\tau+1)\right)^{-1}]$ such that $\mu \leqslant \widehat{\eta}$:

$$\Phi(L) = \nu\frac{1}{Z}\int_0^{\frac{1}{20(G+\varepsilon)(2\tau+1)}} \exp(\eta L - \eta^2 B)d\eta \geqslant \frac{1}{Z}\exp(vL - v^2 B).$$

First suppose that $\widehat{\eta} \leqslant \frac{1}{20(G+\varepsilon)(2\tau+1)}$. Take $v = \widehat{\eta} - \frac{1}{\sqrt{2B}}$, which yields

$$\Phi(L) \geqslant \frac{\nu}{Z}\exp\left(\frac{L^2}{4B} - \frac{1}{2}\right) = g(m(L))$$

where $g(x) = \exp\left(x - 1/2 - \ln\left(Z/\nu\right)\right)$ and $m(x) = x^2/4B$. By Hiriart-Urruty (2006, Theorem 2) we have

$$\begin{aligned}\Phi^\star(u) \leqslant g^\star \circ m^\star(\boldsymbol{u}) &= \inf_{\gamma\geqslant 0} g^\star(\gamma) + \gamma m^\star\left(\frac{u}{\gamma}\right) \\ &= \inf_{\gamma\geqslant 0} \gamma\ln(\gamma) + \gamma\left(\ln\left(\frac{Z}{\nu}\right) - \frac{1}{2}\right) + \frac{1}{\gamma}4u^2B.\end{aligned} \tag{16}$$

Denote by $S = \ln(Z/\nu) - \frac{1}{2}$ and $H = 4u^2B$. Setting the derivative to 0, we find that the value $\widehat{\gamma} = \sqrt{\frac{2H}{W(2H\exp(2S+2))}}$ minimizes (16), where W is the Lambert function. Plugging $\widehat{\gamma}$ in (16) and simplifying by using $\ln(W(x)) = \ln(x) - W(x)$ for $x > 0$ gives

$$\Phi^\star(u) \leqslant \sqrt{2HW(2H\exp(2S+2))} - \sqrt{\frac{2H}{W(2H\exp(2S+2))}} \leqslant \sqrt{2HW(2H\exp(2S+2))}.$$

Using $W(x) \leqslant \ln(x+1)$ (Orabona and Pál, 2016, Lemma 17) we obtain

$$\Phi^\star(u) \leqslant \sqrt{2H\ln(2H\exp(2S+2)+1)} \leqslant \sqrt{8u^2B\ln\left(24u^2B\frac{\pi}{\nu^2}+1\right)},$$

where we used that $Z \leqslant \sqrt{\pi}$.

Now suppose that $\widehat{\eta} > \frac{1}{20(G+\varepsilon)(2\tau+1)}$, which is equivalent to $10(G+\varepsilon)(2\tau+1)L > B$. Then

$$\Phi(L) \geqslant \frac{\nu}{Z} \exp\big(\big(v - v^2 10(G+\varepsilon)(2\tau+1)\big)\, L\big).$$

The convex conjugate of $f(L) = c_1 \exp(c_2 L)$, where $c_1, c_2 > 0$, can be computed using standard properties of convex conjugates and is given by $f^\star(y) = \frac{y}{c_2} \ln\left(\frac{y}{c_1 c_2}\right) - \frac{y}{c_2}$ for $y \geqslant 0$ (see for example Boyd and Vandenberghe (2004, section 3.3)), which can be use to upper bound $\Phi^\star(u)$ for $u \geqslant 0$ by the order reversing property of convex conjugates:

$$\Phi^\star(u) \leqslant \frac{u}{v - v^2 10(G+\varepsilon)(2\tau+1)} \left(\ln\left(\frac{u}{v - v^2 10(G+\varepsilon)(2\tau+1)}\right) - 1 + \frac{1}{2}\ln\left(\frac{\pi}{\nu^2}\right)\right),$$

where we used that $Z \leqslant \sqrt{\pi}$. Picking $v = \frac{5-\sqrt{5}}{200(G+\varepsilon)(2\tau+1)}$ gives us:

$$\Phi^\star(u) \leqslant u44\big(G+\varepsilon)(2\tau+1)\left(\ln\big(u44(G+\varepsilon)(2\tau+1)\big) - 1 + \frac{1}{2}\ln\left(\frac{\pi}{\nu^2}\right)\right),$$

which completes the proof. ■

B.1. Black-Box Reduction

Algorithm 2 Black-Box Reduction

Input: "Direction" algorithm $\mathcal{A}_{\mathcal{Z}}$ and "scaling" algorithm $\mathcal{A}_{\mathcal{V}}$
for $t = 1 \dots T$ **do**
  Get $\boldsymbol{z}_t \in \mathcal{Z}$ from $\mathcal{A}_{\mathcal{Z}}$
  Get $v_t \in \mathbb{R}$ from algorithm $\mathcal{A}_{\mathcal{V}}$
  Play $\boldsymbol{w}_t = v_t \boldsymbol{z}_t$ and receive $\widehat{\boldsymbol{g}}_t$
  Send $\widehat{\boldsymbol{g}}_t$ to algorithm $\mathcal{A}_{\mathcal{Z}}$
  Send $\langle \boldsymbol{z}_t, \widehat{\boldsymbol{g}}_t \rangle$ to algorithm $\mathcal{A}_{\mathcal{V}}$
end

To derive a d-dimensional comparator-adaptive algorithm for a graph we will use the black-box reduction in Algorithm 2. As an alternative to the black-box reduction one could also run a variation of Algorithm 1 for $u \in \mathbb{R}$ in each dimension, which would result in a regret bound similar to that of AdaGrad (Duchi et al., 2011). One could also generalize Algorithm 1 to a higher dimensional version by replacing the scalar η with a vector and adjusting the feedback accordingly, but then no closed-form solution of the integral defining the predictions exists. The algorithm in this section is the black-box reduction presented in Cutkosky and Orabona (2018). The guarantee of Algorithm 2 can be found in Lemma 14 below, whose short proof was originally given by Cutkosky and Orabona (2018) and is repeated below for completeness.

Lemma 14 *Let $\widetilde{\mathcal{R}}_T^{\mathcal{V}}(\|\boldsymbol{u}\|) = \sum_{t=1}^T (v_t - \|\boldsymbol{u}\|)\langle \boldsymbol{z}_t, \boldsymbol{g}_t\rangle$ be the regret for learning $\|\boldsymbol{u}\|$ by Algorithm $\mathcal{A}_{\mathcal{V}}$ and let $\widetilde{\mathcal{R}}_T^{\mathcal{Z}}\left(\frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}\right) = \sum_{t=1}^T \langle \boldsymbol{z}_t - \frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}, \boldsymbol{g}_t\rangle$ be the regret for learning $\frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}$ by $\mathcal{A}_{\mathcal{Z}}$. Then Algorithm2 satisfies*

$$\mathcal{R}_T(\boldsymbol{u}) \leqslant \widetilde{\mathcal{R}}_T^{\mathcal{V}}(\|\boldsymbol{u}\|) + \|\boldsymbol{u}\| \widetilde{\mathcal{R}}_T^{\mathcal{Z}}\left(\frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}\right).$$

Proof of Lemma 14 By definition we have

$$\mathcal{R}_T(\boldsymbol{u}) \leqslant \sum_{t=1}^{T} \langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{g}_t \rangle = \sum_{t=1}^{T} \langle \boldsymbol{z}_t, \boldsymbol{g}_t \rangle \big(v_t - \|\boldsymbol{u}\|\big) + \|\boldsymbol{u}\| \sum_{t=1}^{T} \Big\langle \boldsymbol{z}_t - \frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}, \boldsymbol{g}_t \Big\rangle$$
$$= \widetilde{\mathcal{R}}_T^{\mathcal{V}}\big(\|\boldsymbol{u}\|\big) + \|\boldsymbol{u}\| \widetilde{\mathcal{R}}_T^{\mathcal{Z}}\Big(\frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}\Big).$$

■

Note that in the regret bound of Algorithm 2 the regret of $\mathcal{A}_{\mathcal{Z}}$ scales with $\|\boldsymbol{u}\|$. This means that we can use (6) and upper bound $\langle \boldsymbol{z}_t - \frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}, \boldsymbol{g}_t - \widehat{\boldsymbol{g}}_t \rangle \leqslant 2\varepsilon$ by using Hölder's inequality. In turn this allows us to use any multiagent algorithm for $\mathcal{A}_{\mathcal{Z}}$ with suitable guarantees.

For example, in Theorem 18, we detail the regret bound obtained when using the multiagent algorithm of Hsieh et al. (2020), which is delay-tolerant. The following results lead up to the proof of Theorem 18.

Proposition 15 (Proposition 9 from Hsieh et al. (2020)) *The Ada-Delay-Dist algorithm for Online Learning with Delays bounded by $D(\mathcal{G})$ on the unit sphere satisfies*

$$\mathcal{R}_T \leqslant 4\sqrt{\Lambda_T} + 6GD(\mathcal{G})\,.$$

We account for the inexact gradients in an elementary way, by just adding up the error. This coarse treatment of the error is sufficient for our purpose.

Corollary 16 (AdaDelay-Dist with approximate gradients) *Under the assumptions of Proposition 15, when given approximate gradients as input $\widehat{\boldsymbol{g}}_t$ such that $\|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\| \leqslant \varepsilon$, AdaDelay-Dist enjoys the bound*

$$\mathcal{R}_T \leqslant 4\sqrt{\Lambda_T + 9\,\varepsilon G D(\mathcal{G}) T} + 2\varepsilon T + 12GD(\mathcal{G})\,.$$

Corollary 16 is proved directly thanks to the following lemma that bounds on the lag on the approximate gradients, and using the fact that approximate gradients are bounded by $G + \varepsilon \leqslant 2G$.

Lemma 17 *If approximate gradients are such that $\|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\| \leqslant \varepsilon$, then*

$$\widehat{\Lambda}_T \leqslant \Lambda_T + 9\varepsilon G D(\mathcal{G}) T\,.$$

Proof For any t and s in $[T]$,

$$\|\widehat{\boldsymbol{g}}_t\|\|\widehat{\boldsymbol{g}}_s\| \leqslant \big(\|\boldsymbol{g}_t\| + \varepsilon\big)\big(\|\boldsymbol{g}_s\| + \varepsilon\big) \leqslant \|\boldsymbol{g}_t\|\|\boldsymbol{g}_s\| + 2G\varepsilon + \varepsilon^2 \leqslant \|\boldsymbol{g}_t\|\|\boldsymbol{g}_s\| + 3G\varepsilon\,.$$

Thus, as there are less than $D(\mathcal{G})$ terms in the sum,

$$\|\widehat{\boldsymbol{g}}_t\|^2 + 2\|\widehat{\boldsymbol{g}}_t\| \sum_{s \in \gamma(t)} \|\widehat{\boldsymbol{g}}_s\| \leqslant \|\boldsymbol{g}_t\|^2 + 2\|\boldsymbol{g}_t\| \sum_{s \in \gamma(t)} \|\boldsymbol{g}_s\| + 3G\varepsilon + 6D(\mathcal{G})G\varepsilon$$

We get the claimed result by upper bounding $3G\varepsilon + 6D(\mathcal{G})G\varepsilon \leqslant 9GD(\mathcal{G})\varepsilon$ and summing over $t \in [T]$. ■

Theorem 18 *Suppose that $\|\boldsymbol{g}_t\| \leqslant G$ and that $\|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\| \leqslant \varepsilon$, and $\mathcal{A}_{\mathcal{Z}}$ is AdaDelay-dist. Using Algorithm 1 as $\mathcal{A}_{\mathcal{V}}$ guarantees that Algorithm 2 satisfies*

$$\mathcal{R}_T(\boldsymbol{u}) \leqslant \nu + \|\boldsymbol{u}\|\mathcal{B}(T)\,,$$

where

$$\mathcal{B}(T) = 4\varepsilon T + \sqrt{8\Big(\Lambda_T + 24\varepsilon G D(\mathcal{G})T + 1\Big)\ln_+\Big(\frac{2036\,\|\boldsymbol{u}\|^2 D(\mathcal{G})G^2T}{\nu^2}\Big)} + 4\sqrt{\Lambda_T + 9\,\varepsilon G D(\mathcal{G})T} + 276\,GD(\mathcal{G})\ln_+\Big(\frac{312\,\|\boldsymbol{u}\|G\,D(\mathcal{G})}{\nu}\Big)\,.$$

The proof is merely a combination of the decomposition of regret (Lemma 14), and algorithm guarantees. More generally, similar bounds can be derived as long as the algorithm $\mathcal{A}_{\mathcal{Z}}$ for learning the direction is delay-tolerant, by adapting Proposition 15 with different numerical constants.

Proof of Theorem 18 By Lemma 14 and the guarantee of $\mathcal{A}_{\mathcal{Z}}$ from Corollary 16, we have

$$\begin{aligned}\mathcal{R}_T(\boldsymbol{u}) &\leqslant \widetilde{\mathcal{R}}_T^{\mathcal{V}}(\|\boldsymbol{u}\|) + \|\boldsymbol{u}\|\widetilde{\mathcal{R}}_T^{\mathcal{Z}}\Big(\frac{\boldsymbol{u}}{\|\boldsymbol{u}\|}\Big)\\ &\leqslant \widetilde{\mathcal{R}}_T^{\mathcal{V}}(\|\boldsymbol{u}\|) + \|\boldsymbol{u}\|\Big(4\sqrt{\Lambda_T + 9\,\varepsilon G D(\mathcal{G})T} + 2\varepsilon T + 12GD(\mathcal{G})\Big).\end{aligned}$$

Importing the bound for the scale learning (Theorem 10), we get

$$\widetilde{\mathcal{R}}_T^{\mathcal{V}}(\|\boldsymbol{u}\|) \leqslant \nu + 2\|\boldsymbol{u}\|T\varepsilon + \|\boldsymbol{u}\|\max\Bigg\{264\,GD(\mathcal{G})\ln_+\Big(\frac{312\,\|\boldsymbol{u}\|G\,D(\mathcal{G})}{\nu}\Big)\,, \sqrt{8\Big(\Lambda_t^h + 24\varepsilon G D(\mathcal{G})T + 1\Big)\ln_+\Big(\frac{2036\,\|\boldsymbol{u}\|^2 D(\mathcal{G})G^2T}{\nu^2}\Big)}\Bigg\}.$$

Note finally that $|h_t| = |\langle z_t, \boldsymbol{g}_t\rangle| \leqslant \|\boldsymbol{g}_t\|$ so that $\Lambda_T^h \leqslant \Lambda_T$. The claimed result follows through standard boundings. ∎

B.2. Proof of Theorem 7

Our bound is actually stronger than stated in Theorem 7. As can be seen from the proof, the bound could also be stated in terms of $\widehat{\Lambda}_T$, which is Λ_T with $\widehat{\boldsymbol{g}}_t$ instead of $\boldsymbol{g}_t$. This means that the bound scales with the effective gradient range $\max_t \|\boldsymbol{g}_t\|$ rather than the worst-case bound on the gradients. Similarly, we have presented the bound as if the learner suffers the maximum delay $D(\mathcal{G})$ each round, but the bound could also be stated in terms of the effective delay $|\gamma(t)|$. For simplicity, we only present the worst-case regret bound.

We recall that the algorithm used is the black-box reduction applied to the gradients encoded with sparsified quantization, with AdaDelay-dist (from Hsieh et al. (2020)) as $\mathcal{A}_{\mathcal{Z}}$ and Algorithm 1 as $\mathcal{A}_{\mathcal{V}}$. Algorithm 1 is tuned with $\varepsilon = 0$ and the upper bound $2dG$ on $\|\widehat{\boldsymbol{g}}_t\|$.

Theorem 7 (Regret bound with Stochastic Encoding) *Using $k = \lfloor b/D(\mathcal{G}) \rfloor$ bits per gradient, the algorithm described above satisfies*

$$\mathbb{E}[\mathcal{R}_T(\boldsymbol{u})] \leqslant \nu + \|\boldsymbol{u}\|\, \mathcal{B}(T) \quad \textit{where} \quad \mathcal{B}(T) = \widetilde{\mathcal{O}}\Big(G\sqrt{\Big(1 + \frac{dD(\mathcal{G})}{b}\Big) D(\mathcal{G}) T}\, \Big)\,.$$

Proof First, by convexity of ℓ_t and using that $\mathbb{E}_t[\widehat{\boldsymbol{g}}_t] = \boldsymbol{g}_t$ we have that

$$\mathbb{E}[\mathcal{R}_T(\boldsymbol{u})] \leqslant \mathbb{E}\left[\sum_{t=1}^{T} \langle \boldsymbol{w}_t - \boldsymbol{u}, \widehat{\boldsymbol{g}}_t \rangle\right].$$

We have that $\max_t \|\widehat{\boldsymbol{g}}_t\| \leqslant 2dG$ by Theorem 6. Let us now call to the detailed regret bound for our algorithm, that is, Theorem 18 used with the estimated losses $\{\widehat{\boldsymbol{g}}_t\}$ and with $\varepsilon = 0$. Note that in this case, Theorem 18 is applied *directly* to the approximate gradients, which makes the choice $\varepsilon = 0$ valid. We find

$$\sum_{t=1}^{T} \langle \boldsymbol{w}_t - \boldsymbol{u}, \widehat{\boldsymbol{g}}_t \rangle \leqslant \nu + \|\boldsymbol{u}\|\, \widehat{\mathcal{B}(T)}\,,$$

where

$$\begin{aligned}
\widehat{\mathcal{B}(T)} :=& \sqrt{8\Big(\widehat{\Lambda}_T + 1\Big) \ln_+\Big(\frac{2036\, \|\boldsymbol{u}\|^2 T (2dG)^2}{\nu^2}\Big)} + 4\sqrt{\widehat{\Lambda}_T} \\
&+ 276\, (2dG) D(\mathcal{G}) \ln_+\Big(\frac{312\, \|\boldsymbol{u}\| 2dG\, D(\mathcal{G})}{\nu}\Big) \\
\leqslant& \sqrt{47\Big(\widehat{\Lambda}_T + 1\Big) \ln_+\Big(\frac{8144\, \|\boldsymbol{u}\|^2 T d^2\, G^2}{\nu^2}\Big)} \\
&+ 552\, dG D(\mathcal{G}) \ln_+\Big(\frac{624\, \|\boldsymbol{u}\| dG\, D(\mathcal{G})}{\nu}\Big)\,.
\end{aligned}$$

Now let us upper bound $\mathbb{E}\big[\widehat{\Lambda}_T\big]$. Define the short-hand notation

$$\alpha = \frac{2d}{m} \quad \text{and} \quad \beta = \frac{1}{m} G^2 \quad \text{with} \quad m = \lfloor k/(3\lceil \log_2(d) \rceil + 2) \rfloor.$$

Then

$$\mathbb{E}\left[\|\widehat{\boldsymbol{g}}_t\|^2\right] = \mathbb{E}\left[\|\widehat{\boldsymbol{g}}_t - \boldsymbol{g}_t\|^2\right] + \|\boldsymbol{g}_t\|^2 \leqslant (\alpha + 1)\|\boldsymbol{g}_t\|^2 + \beta\,.$$

Using Jensen's inequality and the guarantees of Theorem 6 we find

$$\begin{aligned}
\mathbb{E}\left[\|\widehat{\boldsymbol{g}}_t\| \|\widehat{\boldsymbol{g}}_s\|\right] \leqslant& \sqrt{\mathbb{E}\big[\|\widehat{\boldsymbol{g}}_t\|^2\big]\, \mathbb{E}\big[\|\widehat{\boldsymbol{g}}_s\|^2\big]} \\
\leqslant& \sqrt{(\alpha+1)\|\boldsymbol{g}_t\|^2 + \beta}\, \sqrt{(\alpha+1)\|\boldsymbol{g}_s\|^2 + \beta} \\
\leqslant& (\alpha+1)G^2 + \beta.
\end{aligned}$$

Now, replacing α and β by their values we see that

$$\mathbb{E}\left[\|\widehat{\boldsymbol{g}}_t\| \|\widehat{\boldsymbol{g}}_s\|\right] \leqslant G^2\Big(1 + \frac{d+1}{m}\Big).$$

We sum these inequalities in the expression of $\widehat{\Lambda}_T$ and use that $|\gamma(t)| \leqslant D(\mathcal{G})$ to get

$$\mathbb{E}\left[\widehat{\Lambda}_T\right] \leqslant G^2 T(1+D(\mathcal{G}))\left(1+\frac{d+1}{m}\right) \leqslant G^2 T(1+D(\mathcal{G}))\left(1+\frac{(d+1)(3\log_2(d)+3)}{k}\right).$$

which, after replacing $\widehat{\Lambda}_T$ in $\widehat{\mathcal{B}(T)}$ and setting $k = D(\mathcal{G})/b$ completes the proof. ■

Appendix C. Details of Section 4

We denote by $\widetilde{\mathcal{R}}_{\mathcal{F}}(\boldsymbol{u}) = \sum_{t:I_t\in\mathcal{F}}\langle \boldsymbol{w}_t^{\mathcal{F}} - \boldsymbol{u}, \boldsymbol{g}_t\rangle$ the linearised regret in graph $\mathcal{F}$.

Lemma 19 *Let $\mathcal{Q}$ be a collection of subgraphs of $\mathcal{G}$ and suppose for each subgraph $\mathcal{F} \in \mathcal{Q}$ $\widetilde{\mathcal{R}}_{\mathcal{F}_j}(\mathbf{0}) \leqslant \nu$, discarding any message older than $D_{\mathcal{Q}} = \max_{\mathcal{F}\in\mathcal{Q}} D(\mathcal{F})$ rounds. Then, playing $\boldsymbol{w}_t = \sum_{\mathcal{F}\in\mathcal{Q}} \boldsymbol{w}_t^{\mathcal{F}} \mathbb{1}\{I_t \in \mathcal{F}\}$ simultaneously guarantees for any $\mathcal{Q}$-partition $\{\mathcal{F}_1,\dots,\mathcal{F}_r\}$ and for any $\boldsymbol{u}_1\dots,\boldsymbol{u}_r \in \mathbb{R}^d$ that*

$$\sum_{t=1}^{T}\ell_t(\boldsymbol{w}_t) - \sum_{j=1}^{r}\sum_{t:I_t\in\mathcal{F}_j}\ell_t\left(\boldsymbol{u}_j\right) \leqslant |\mathcal{Q}|\nu + \sum_{j=1}^{r}\widetilde{\mathcal{R}}_{\mathcal{F}_j}(\boldsymbol{u}_j)\,.$$

Proof We start by upper bounding the regret by its linearized version:

$$\sum_{t=1}^{T}\ell_t(\boldsymbol{w}_t) - \sum_{j=1}^{r}\sum_{t:I_t\in\mathcal{F}_j}\ell_t(\boldsymbol{u}_j) = \sum_{j=1}^{r}\sum_{t:I_t\in\mathcal{F}_j}\left(\ell_t(\boldsymbol{w}_t) - \ell_t\left(\boldsymbol{u}_j\right)\right) \leqslant \sum_{j=1}^{r}\sum_{t:I_t\in\mathcal{F}_j}\langle \boldsymbol{w}_t - \boldsymbol{u}_j, \boldsymbol{g}_t\rangle\,.$$

Denote by $\mathcal{P} = \{\mathcal{F}_1,\dots,\mathcal{F}_r\}$ an arbitrary $\mathcal{Q}$-partition. We rewrite the bound above, replacing $\boldsymbol{w}_t$ by its value, and doing some manipulations on the indices,

$$\begin{aligned}
\sum_{t=1}^{T}\langle \boldsymbol{w}_t, \boldsymbol{g}_t\rangle - \sum_{j=1}^{r}\sum_{t:I_t\in\mathcal{F}_j}\langle \boldsymbol{u}_j, \boldsymbol{g}_t\rangle &= \sum_{t=1}^{T}\sum_{\mathcal{F}:I_t\in\mathcal{F}}\langle \boldsymbol{w}_t^{\mathcal{F}}, \boldsymbol{g}_t\rangle - \sum_{j=1}^{r}\sum_{t:I_t\in\mathcal{F}_j}\langle \boldsymbol{u}_j, \boldsymbol{g}_t\rangle \\
&= \sum_{\mathcal{F}\in\mathcal{Q}\setminus\mathcal{P}}\widetilde{\mathcal{R}}_{\mathcal{F}}(\mathbf{0}) + \sum_{j=1}^{r}\widetilde{\mathcal{R}}_{\mathcal{F}_j}(\boldsymbol{u}_j) \\
&\leqslant |\mathcal{Q}|\nu + \sum_{j=1}^{r}\widetilde{\mathcal{R}}_{\mathcal{F}_j}(\boldsymbol{u}_j)\,,
\end{aligned}$$

where we used that $\widetilde{\mathcal{R}}_{\mathcal{F}}(\mathbf{0}) \leqslant \nu$ for all $\mathcal{F}$. ■

Theorem 9 (Learning a $\mathcal{Q}$-Partition, deterministic encoding) *Let $\mathcal{Q}$ be a collection of subgraphs of $\mathcal{G}$. Suppose the learner uses the algorithm of Theorem 4 with deterministic encodings for each subgraph $\mathcal{F} \in \mathcal{Q}$, discarding any message coming from outside of $\mathcal{F}$. Then, setting $\nu = 1/|\mathcal{Q}|$ and $k = \lfloor b/D_{\mathcal{Q}}\rfloor$, and playing $\boldsymbol{w}_t$ as specified in equation (10) guarantees that*

$$\sum_{j=1}^{r}\mathcal{R}_{\mathcal{F}_j}(\boldsymbol{u}_j) = \mathcal{O}\left(\sum_{j=1}^{r}\|\boldsymbol{u}_j\|\left(\sqrt{\Lambda(\mathcal{F}_j)\ln\left(1+|\mathcal{Q}|D(\mathcal{F}_j)\|\boldsymbol{u}_j\|T_jG\right)} + 2^{-b/(dD_{\mathcal{Q}})}T_jG\right)\right)$$

for any $\mathcal{Q}$-partition $\{\mathcal{F}_1,\dots,\mathcal{F}_r\}$ and any $\boldsymbol{u}_1\dots,\boldsymbol{u}_r \in \mathbb{R}^d$.

Theorem 20 *Suppose that* $b \geqslant D_{\mathcal{Q}}\left(3\lceil \log_2(d)\rceil + 2\right)$. *Let* $\mathcal{Q}$ *be a collection of subgraphs of* $\mathcal{G}$. *Suppose the learner uses the algorithm of Theorem 18 with* $\varepsilon = 0$, *the upper bound* $2dG$ *on* $\|\widehat{\boldsymbol{g}}_t\|$, *and the stochastic encoding of Theorem 6, for each subgraph* $\mathcal{F} \in \mathcal{Q}$, *discarding any message older than* $D_{\mathcal{Q}} = \max_{\mathcal{F}\in\mathcal{Q}} D(\mathcal{F})$ *rounds. Then, setting* $\nu = 1/|\mathcal{Q}|$, $k = \lfloor b/D_{\mathcal{Q}} \rfloor$, *and playing* $\boldsymbol{w}_t = \sum_{\mathcal{F}\in\mathcal{Q}} \boldsymbol{w}_t^{\mathcal{F}} \mathbb{1}\{I_t \in \mathcal{F}\}$ *guarantees that*

$$\mathbb{E}\left[\sum_{j=1}^{r} \mathcal{R}_{\mathcal{F}_j}(\boldsymbol{u}_j)\right] = \mathcal{O}\Big(\sum_{j=1}^{r} \|\boldsymbol{u}_j\| G \sqrt{\left(1 + \frac{dD_{\mathcal{Q}}}{b}\right) D(\mathcal{F}_j) T_j \ln\left(1 + |\mathcal{Q}| D(\mathcal{F}_j) \|\boldsymbol{u}_j\| T_j G\right)}\ \Big),$$

for any $\mathcal{Q}$*-partition* $\{\mathcal{F}_1, \dots, \mathcal{F}_r\}$ *and for any* $\boldsymbol{u}_1 \dots, \boldsymbol{u}_r \in \mathbb{R}^d$.

The proofs of Theorems 9 and 20 follow from applying Lemma 19, and a slight modification of the proofs of Theorems 4 and 7, in which we can use that $k = \lfloor b/D_{\mathcal{Q}} \rfloor$.

C.1. Example Collections of Subgraphs

In this section we provide an example collection of subgraphs. For each node n, take all nodes that are with in distance $\alpha(n) = 0, 1, \dots, E(n)$, where $E(n)$ is the eccentricity of node n, and form them into a subgraph. The collection of this set of subgraphs is has cardinality $|\mathcal{Q}| = |\mathcal{N}| + \sum_{n\in\mathcal{N}} E(n) \leqslant |\mathcal{N}|(1 + |\mathcal{N}|)$. For the line graph, this collection of subgraphs includes all possible partitions of the graph, meaning we compete with the best possible partition of the graph.

Alternatively, one may take $\alpha(n) = 2^{\omega(n)}$ for $\omega(n) = 1, \dots, \lfloor \log_2(E(n)) \rfloor$ to approach the regret bounds of the previously defined $\mathcal{Q}$ within a factor 2. Both collections can be used to adapt to the scenarios described in Figures 1(*a*) and 1(*b*) since the clusters may will be contained in one of the subgraphs.

Appendix D. Details on the Encoding Schemes

In this section, we provide the detailed presentation and analysis of the encoding schemes referred to in Section 3, eventually proving Theorem 6. We also discuss briefly some relevant literature.

D.1. (Two) Determinisitic encoding Schemes

We discuss two possible encoding schemes. The first encoding scheme is a non-constructive encoding using covering numbers. Given well-known results on the covering number of the ball, there exists a covering of the ball $\mathcal{B}_2(G)$ with 2^k balls of radius $3 \cdot 2^{-k/d} G$. Therefore, for any vector $\boldsymbol{x} \in \mathcal{B}_2(G)$, one can transmit $\widehat{\boldsymbol{x}}$ the center of a ball of the covering it belongs to, using k bits. Doing so, we have an encoding which ensures $\|\boldsymbol{x} - \widehat{\boldsymbol{x}}\| \leqslant 3 \cdot 2^{-k/d} G$. Note that this implies in particular that $\|\widehat{\boldsymbol{x}}\| \leqslant G(1 + 3 \cdot 2^{-k/d}) \leqslant 4G$.

The drawback of this first encoding scheme is that it requires an explicit optimal covering of the ball. The second encoding scheme we provide is slightly worse in terms of error, but is more practical due to its simplicity.

In the second encoding scheme, the approach is to send each coordinate separately. Although this encoding scheme is suboptimal in terms of error, it has the advantage of being straightforward

to implement. Using $\lfloor k/d \rfloor$ bits per coordinate (we assume here that $k \geqslant d$), and sending each coordinate separately yields for all coordinates i,

$$|x_i - \widehat{x}_i| \leqslant 2^{-k/d+2} G, \quad \text{thus} \quad \|\boldsymbol{x} - \widehat{\boldsymbol{x}}\| \leqslant \sqrt{d}\, 2^{-k/d+2} G\,.$$

Compared to the theoretical bound from covering numbers, we lose a $\sqrt{d}$ factor in the error bound. However, the practicality gain can make this choice worthwhile when k/d is sufficiently large.

D.2. Stochastic Encoding

To stochastically encode an arbitrary vector $\boldsymbol{x} \in \mathcal{B}_2(G) \subset \mathbb{R}^d$, we will combine p-level quantization with sparsification, which are defined as follows:

p-Level Quantization of a Single Coordinate Given a precision parameter $p \in \mathbb{N}$, the p-level quantization of any coordinate x_i of $\boldsymbol{x}$ is

$$\widetilde{x}_i = \operatorname{sign}(x_i) 2^{-p} G \Big\lfloor \frac{2^p x_i}{G} \Big\rfloor + 2^{-p} G\, b_i.$$

The first term in this expression is deterministic and essentially corresponds to the truncation of x_i/G to the first p digits in its binary expansion. The second term is random, with $b_i \in \{0, 1\}$ a Bernoulli variable with $\Pr(b_i = 1) = (2^p/G)(x_i - \operatorname{sign}(x_i) 2^{-p} G \lfloor 2^p x_i / G \rfloor)$ chosen to make $\widetilde{x}_i$ an unbiased estimate of x_i. The essential properties of the $\widetilde{x}_i$ are that

$$\mathbb{E}[\widetilde{x}_i] = x_i \quad \text{and} \quad (x_i - \widetilde{x}_i)^2 \leqslant 2^{-2p} G\,.$$

All in all, this randomized encoding requires 1 bit to encode the sign of x_i, and p bits for the deterministic part, and 1 bit for b_i, so $p + 2$ bits in total.

Sparsification Instead of encoding all coordinates of $\boldsymbol{x}$, we sample a single coordinate i_S uniformly at random from $\{1, \dots, d\}$ and transmit only $\widetilde{x}_{i_S}$. At the decoder, this results in a sparse vector

$$\widehat{\boldsymbol{x}} = d \widetilde{x}_{i_S} \mathbf{e}_{i_S},$$

where the factor d is an importance weight that ensures that $\mathbb{E}[\widehat{\boldsymbol{x}}] = \boldsymbol{x}$. Encoding i_S requires $\lceil \log_2(d) \rceil$ bits, so p-level quantization and sparsification together require $\lceil \log_2(d) \rceil + p + 2$ bits.

Sparsified Quantization To reduce the variance, we repeat the construction above m times and average the resulting vectors $\widehat{\boldsymbol{x}}_1, \dots, \widehat{\boldsymbol{x}}_m$ to obtain our final estimate

$$\widehat{\boldsymbol{x}} = \frac{1}{m} \sum_{j=1}^{m} \widehat{\boldsymbol{x}}_j\,.$$

We call $\widehat{\boldsymbol{x}}$ the sparsified quantization of $\boldsymbol{x}$ with precision p and number of repetitions m. It can be communicated with $m(\lceil \log_2(d) \rceil + p + 2)$ bits, and satisfies the following property:

Proposition 21 *For any $\boldsymbol{x} \in \mathcal{B}_2(G)$, the sparsified quantization $\widehat{\boldsymbol{x}}$ of $\boldsymbol{x}$ with $m = 1$ repetition satisfies*

$$\mathbb{E}\big[\widehat{\boldsymbol{x}}\big] = \boldsymbol{x}, \qquad \mathbb{E}\left[\|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|^2\right] \leqslant 2d\|\boldsymbol{x}\|^2 + d^2\, 2^{-2p} G^2$$

and

$$\|\widehat{\boldsymbol{x}}\| \leqslant d(\|\boldsymbol{x}\|_\infty + 2^{-p}G) \leqslant 2dG.$$

In particular, for $p = \lceil \log_2 d \rceil$, the expected squared error is less than $2d\|\boldsymbol{x}\|^2 + G^2$ and the number of bits communicated is less than $3\lceil \log_2 d \rceil + 2$.

Proof The unbiasedness is straightforward from the independence between $\widetilde{x}_i$ and i_S:

$$\mathbb{E}\left[\widehat{\boldsymbol{x}}\right] = d\sum_{i=1}^{d} \mathbb{P}[i = i_S]\,\mathbb{E}[\,\widetilde{x}_i \mathbf{e}_i] = d\sum_{i=1}^{d} \frac{1}{d}\, x_i\, \mathbf{e}_i = \boldsymbol{x}\,.$$

Let us compute the square-error for a fixed coordinate $i \in \{1, \dots, d\}$,

$$\begin{aligned}\mathbb{E}\left[(x_i - \widehat{x}_i)^2\right] &= \mathbb{E}\left[(x_i - \widehat{x}_i)^2 \mathbb{1}\{i = i_S\}\right] + x_i^2\,\mathbb{P}[i \neq i_S] \\ &= \mathbb{E}\left[(x_i - d\widetilde{x}_i)^2\right]\mathbb{P}[i = i_S] + x_i^2\,\mathbb{P}[i \neq i_S] = \mathbb{E}\left[(x_i - d\widetilde{x}_i)^2\right]\frac{1}{d} + x_i^2\Big(1 - \frac{1}{d}\Big)\end{aligned}$$

where we used the fact that $\widetilde{x}_i$ is independent from i_S. Moreover, using the unbiasedness of $\widetilde{x}_i$,

$$\mathbb{E}\big[(x_i - d\,\widetilde{x}_i)^2\big] = \mathbb{E}\left[\big(x_i - d\,x_i + d(x_i - \widetilde{x}_i)\big)^2\right] = (d-1)^2 x_i^2 + d^2\,\mathbb{E}\left[(x_i - \widetilde{x}_i)^2\right] \leqslant d^2(x_i^2 + 2^{-2p}G^2)\,.$$

Finally, by summing over the coordinates,

$$\mathbb{E}\big[\|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|^2\big] \leqslant d^2\frac{1}{d}\sum_{i=1}^{d}\left(x_i^2 + 2^{-2p}G^2\right) + \left(1 - \frac{1}{d}\right)\sum_{i=1}^{d} x_i^2 \leqslant 2d\|\boldsymbol{x}\|^2 + d^2\,2^{-2p}G^2\,,$$

concluding the proof. ■

Adding repetitions reduces the variance of any unbiased stochastic encoding:

Proposition 22 (Combining Encodings) *If $\widehat{\boldsymbol{x}}_1, \dots, \widehat{\boldsymbol{x}}_m$ are independent unbiased encodings of a vector $\boldsymbol{x} \in \mathcal{B}_2(G)$, then the error of the average encoding scheme is*

$$\mathbb{E}\left[\left\|\boldsymbol{x} - \frac{1}{m}\sum_{i=1}^{m}\widehat{\boldsymbol{x}}_i\right\|^2\right] = \frac{1}{m^2}\sum_{i=1}^{m}\mathbb{E}\left[\|\boldsymbol{x} - \widehat{\boldsymbol{x}}_i\|^2\right].$$

This is proved by developing the squared-norm.

In particular, sparsified quantization with precision $p = \lceil \log_2 d \rceil$ and $m = \lfloor k/(3\lceil \log_2(d) \rceil + 2) \rfloor$ repetitions satisfies the claim of Theorem 6.

Commonly used Stochastic Encodings. The popular schemes of Alistarh et al. (2017) and Wen et al. (2017) use varying-length coding, and have results depending on the number of bits used to represent a floating-point number. Moreover, these results essentially assume that the number of bits per gradient is at least equal to the dimension $b = \Omega(d)$; the same restrictions apply to Albasyoni et al. (2020), Faghri et al. (2020). With a different formalism, Stich et al. (2018), Shi et al. (2019) use sparsification with no quantization and do not impose a constraint on the number of bits sent. Acharya et al. (2019) propose a sparsification scheme specific to encoding probability vectors, and

Mayekar and Tyagi (2020a) generalize it to other norm balls. A number of schemes assume shared randomness between the agents, e.g., Mayekar and Tyagi (2020b), Suresh et al. (2017). This is not appropriate for our setting, because shared randomness can only be achieved without communication by using pseudo-randomness based on a shared seed. But then the adversary might guess the seed or otherwise exploit the lack of real randomness to predict the randomness in the stochastic encodings.

Appendix E. Lower Bounds

We start by mapping out how each specific characteristic of our setting influences the regret, and how we can import known lower bounds from other settings.

E.1. Overview

As discussed at the start of Section 3, we restrict attention to algorithms that send messages containing approximate gradients, with a limit of $k = \lfloor b/D(\mathcal{G}) \rfloor$ bits per message.

Nodes only have access to approximations $\widehat{\boldsymbol{g}}_t$ of the gradients $\boldsymbol{g}_t$ that are computed at other nodes, but they have full access to $\boldsymbol{g}_t$ for the subset of rounds in which they are active themselves. To simplify the treatment of lower bounds, we restrict attention to *gradient-oblivious* methods, which only use $\widehat{\boldsymbol{g}}_t$ at all the nodes, even if a node could have used $\boldsymbol{g}_t$ instead. Although this assumption restricts the class of algorithms, we believe it is a minor restriction. For instance, all algorithms considered in the rest of the paper satisfy it, and for large networks, in which the same node is activated only a small number of times, the difference seems minor.

Comparison with Memory-Limited OCO There are tight relations to the $(k, 1, 1)$ distributed online protocol defined by Shamir (2014). Under this protocol, an agent only has access to a collection of k-bit messages stored in its memory, each message encoding one loss function it has received. (The two ones in $(k, 1, 1)$ correspond to other parameters in the reference.)

In fact, for gradient oblivious algorithms that use k bits per gradient, in the special case where the same node is selected at every time step, our setting simplifies to the $(k, 1, 1)$ setting. Therefore, for any activation sequence, the game is at least as hard as $(k, 1, 1)$, as the active agent only has less information available. In particular, lower bounds for the $(k, 1, 1)$ setting automatically hold in our setting.

Deterministic vs. Stochastic Encodings We distinguish between deterministic and stochastic encodings. Deterministic encodings only give non-trivial guarantees when k is at least of the same order as the dimension d (Theorem 5). Stochastic encodings can still work in the regime that $k \ll d$, as long as $k \gg d/T$ (Theorem 8). We will prove the lower bound for stochastic encodings by reduction from a previous lower bound by Mayekar and Tyagi (2020b).

The Network Causes Delays Because gradients take time to be transmitted through the network, there is a direction connection to the lower bound for Online Learning with Delays Zinkevich et al. (2009). This effect is combined with the memory limits described above. Informally, we can say that if $B(k, T)$ is a lower bound for the memory-limited setting, then, under the activation sequence that maximizes delays, a lower bound of $D(\mathcal{G})B\big(k, T/D(\mathcal{G})\big)$ holds for our setting. This statement is made precise in Lemma 27.

E.2. Proofs of Lower Bounds and Intermediate Results

In the remainder of this section, we prove Theorems 5 (deterministic encodings) and 8 (stochastic encodings). We first establish lower bounds for the $(k, 1, 1)$ protocol separately for the deterministic and stochastic case, based on the connection to the memory-limited setting. We then exploit the connection to the online learning with delays setting to turn these lower bounds for the $(k, 1, 1)$ protocol into the lower bounds of the theorems.

E.2.1. LIMITED MEMORY I: DETERMINISTIC ENCODINGS

As discussed earlier, the decentralized online learning setting for gradient-oblivious algorithms with k bits per gradient is harder than the memory-limited $(k, 1, 1)$ setting. In this section, we provide lower bounds that are based on this connection.

The limiting factor with deterministic encodings is that there is finite resolution, making some vectors indistinguishable from each other once encoded. Formally, let $C : B_2(G) \to \{0, 1\}^k$ be any deterministic encoding scheme. Then we define the *resolution* ε of C to be the maximum distance between two vectors encoded to the same message, i.e., $\varepsilon = \sup\big\{\|\boldsymbol{g} - \mathbf{h}\| : C(\boldsymbol{g}) = C(\mathbf{h})\big\}$. We can lower bound the resolution of C using the covering numbers of $\mathcal{B}_2(G)$ as follows:

Lemma 23 *Any deterministic k-bit encoding scheme on $\mathcal{B}_2(G)$ has resolution at least $\varepsilon \geqslant 2G\, 2^{-k/d}$.*

Proof The family of sets $C^{-1}(\mathbf{m})$ for $\mathbf{m} \in \{0, 1\}^k$ forms a partition of $\mathcal{B}_2(G)$, and each of these sets is contained in a Euclidean ball of diameter ε by definition of the resolution. Thus the balls form an $(\varepsilon/2)$-cover of $\mathcal{B}_2(G)$. Since covering $\mathcal{B}_2(G)$ with balls of radius $\varepsilon/2$ requires at least $(2G/\varepsilon)^d$ balls (Ghosal and van der Vaart, 2017, Appendix C)[1], it follows that $2^k = |\{0, 1\}^k| \geqslant (2G/\varepsilon)^d$, from which the result follows. ■

Knowing that two gradient values that are ε apart are encoded the same way, an adversary can design a hard loss sequence against which a player would suffer linear regret in the $(k, 1, 1)$ setting.

Lemma 24 (Deterministic Encoding with Limited Memory) *In the $(k, 1, 1)$ protocol, for any algorithm using a deterministic encoding with resolution $\varepsilon > 0$, for any $U > 0$*

$$\sup_{G-\textit{Lipschitz losses}} \sup_{\boldsymbol{u}:\, \|\boldsymbol{u}\|=U} \mathcal{R}_T(\boldsymbol{u}) \geqslant \frac{1}{4}\varepsilon\, T\, \|\boldsymbol{u}\|\,.$$

Proof Let $\boldsymbol{g}$ and $\mathbf{h}$ be two vectors with same the encoding such that $\|\boldsymbol{g} - \mathbf{h}\| \geqslant \varepsilon$. In the following, we consider exclusively linear losses $\ell_t : \boldsymbol{u} \mapsto \langle \boldsymbol{g}_t, \boldsymbol{u}\rangle$, which are fully specified by their gradient $\boldsymbol{g}_t$. Consider an i.i.d. sequence of losses where at every time step $\boldsymbol{g}_t = \boldsymbol{g}$ with probability $1/2$ and $\boldsymbol{g}_t = -(\boldsymbol{g} + \mathbf{h})/2$ with probability $1/2$.

Then at all times t, since the loss $\boldsymbol{g}_t$ is generated independently from $\boldsymbol{w}_t$,

$$\mathbb{E}[\boldsymbol{g}_t] = \frac{1}{2}\,\boldsymbol{g} + \frac{1}{2}\Big(-\frac{\boldsymbol{g} + \mathbf{h}}{2}\Big) = \frac{\boldsymbol{g} - \mathbf{h}}{4} \quad \text{and} \quad \mathbb{E}\left[\langle \boldsymbol{g}_t, \boldsymbol{w}_t\rangle \,|\, \boldsymbol{w}_t\right] = \left\langle \frac{\boldsymbol{g} - \mathbf{h}}{4}, \boldsymbol{w}_t \right\rangle.$$

1. This can be shown by noting that the sum of the volumes of the balls in the cover must exceed the volume of $\mathcal{B}_2(G)$.

Therefore, for any comparator $\boldsymbol{u} \in \mathcal{W}$, the expected value of the regret against $\boldsymbol{g}_{1:T}$ is

$$\mathbb{E}\big[\mathcal{R}_T(\boldsymbol{u}; \boldsymbol{g}_{1:T})\big] = \mathbb{E}\left[\left\langle \frac{\boldsymbol{g}-\mathbf{h}}{4}, \sum_{t=1}^{T} (\boldsymbol{w}_t - \boldsymbol{u})\right\rangle\right] = \left\langle \frac{\boldsymbol{g}-\mathbf{h}}{4}, \mathbb{E}\left[\sum_{t=1}^{T} \boldsymbol{w}_t\right] - T\boldsymbol{u}\right\rangle,$$

where we explicitely wrote the dependence on $\boldsymbol{g}_{1:T}$ for the sake of clarity.

Now given a sequence $\boldsymbol{g}_{1:T}$ generated as above, define another sequence $\widetilde{\boldsymbol{g}}_{1:T}$ swapping $\boldsymbol{g}$ and $\mathbf{h}$; that is, where $\widetilde{\boldsymbol{g}}_t = \mathbf{h}$ if $\boldsymbol{g}_t = \boldsymbol{g}$, and $\widetilde{\boldsymbol{g}}_t = \boldsymbol{g}_t$ otherwise. As above, we also have the identity

$$\mathbb{E}\big[\mathcal{R}_T(\boldsymbol{u}; \widetilde{\boldsymbol{g}}_{1:T})\big] = \left\langle \frac{\mathbf{h}-\boldsymbol{g}}{4}, \mathbb{E}\left[\sum_{t=1}^{T} \boldsymbol{w}_t\right] - T\boldsymbol{u}\right\rangle.$$

Since the algorithm is gradient oblivious, and since the two loss sequences are encoded the same way, the algorithm cannot distinguish between a sequence $\boldsymbol{g}_{1:T}$ and its switched version $\widetilde{\boldsymbol{g}}_{1:T}$. Therefore the distribution of $\boldsymbol{w}_{1:T}$ is the same in both cases. In particular the expected value of the sum of the predictions $\mathbb{E}[\sum_{t=1}^{T} \boldsymbol{w}_t]$ is the same under both loss distributions.

Therefore, by distinguishing cases on whether $\langle \mathbb{E}[\sum_{t=1}^{T} \boldsymbol{w}_t], \mathbf{h} - \boldsymbol{g}\rangle$ is non-negative or not, we observe that at least one of the following statement holds: either

$$\mathbb{E}\big[\mathcal{R}_T(\boldsymbol{u}; \boldsymbol{g}_{1:T})\big] \geqslant -\frac{T}{4}\langle \boldsymbol{g}-\mathbf{h}, \boldsymbol{u}\rangle \quad \text{for all } \boldsymbol{u} \in \mathcal{W},$$

or

$$\mathbb{E}\big[\mathcal{R}_T(\boldsymbol{u}; \widetilde{\boldsymbol{g}}_{1:T})\big] \geqslant \frac{T}{4}\langle \boldsymbol{g}-\mathbf{h}, \boldsymbol{u}\rangle \quad \text{for all } \boldsymbol{u} \in \mathcal{W}.$$

Let us conclude, for example, in the first case; the second case is completely symmetric. Picking $\boldsymbol{u} = \lambda(\mathbf{h}-\boldsymbol{g})/\|\mathbf{h}-\boldsymbol{g}\|$ for any $\lambda > 0$, we get

$$\mathbb{E}\big[\mathcal{R}_T(\boldsymbol{u}; \boldsymbol{g}_{1:T})\big] \geqslant \frac{T}{4}\|\boldsymbol{g}-\mathbf{h}\|\|\boldsymbol{u}\| \geqslant \frac{T}{4}\varepsilon\|\boldsymbol{u}\|.$$

Since this holds in expectation for our distribution of losses, there exists a sequence $\boldsymbol{g}_{1:T}$ satisfying the claimed inequality. ∎

We also recall the comparator-adaptive lower bound from Orabona (2013). This lower bound holds in the standard OCO setting, and thus all the more so in the memory-limited setting. Together with Lemma 24, these two results yield Theorem 5, up to the impact of the delays.

Theorem 25 (Theorem 2 in Orabona (2013)) *In the standard OCO setting, fix an algorithm such that $\mathcal{R}_T(\mathbf{0}) \leqslant B$ for some number $B > 0$. For any $\boldsymbol{u} \in \mathbb{R}^d$, for T large enough,*

$$\sup_{G-\text{Lipschitz losses}} \mathcal{R}_T(\boldsymbol{u}) \geqslant 0.3\,\|\boldsymbol{u}\|G\sqrt{T\ln\left(\frac{\|\boldsymbol{u}\|\sqrt{T}}{6B}\right)}.$$

Note that at first sight our statement seems stronger than that of the reference, as we transformed the "there exists" quantifier into "for any". In fact, looking at the proof in the reference, the comparator is fixed to be $(U, 0, \dots, 0)$ and the sequence of (linear) losses takes values only in $\{\pm(G, 0, \dots, 0)\}$. This same construction can be made in the direction of any vector $\boldsymbol{u} \in \mathbb{R}^d$.

E.2.2. LIMITED MEMORY II: STOCHASTIC ENCODINGS

The main downside of the deterministic encoding, namely the finite resolution, can be avoided using randomness, see Section 3. However, when there are less than d bits available, even stochastic procedures have strong limitations. Indeed, Mayekar and Tyagi (2020b) show, via an optimization problem, that in the $(k, 1, 1)$ setting the limit on the number of bits appears as a constant in front of the minimax regret.

Theorem 26 (Corollary of Theorem 2 in Mayekar and Tyagi (2020b)) *In the $(k, 1, 1)$ protocol, for any algorithm, for any $U > 0$*

$$\sup_{G-\textit{Lipschitz losses}} \sup_{\boldsymbol{u}:\, \|\boldsymbol{u}\|=U} \mathbb{E}\left[\mathcal{R}_T(\boldsymbol{u})\right] \geqslant cUG\sqrt{\max\left(\frac{d}{k}, 1\right)T} \quad ,$$

for some numerical constant c, and where the expectation is taken with respect to the randomness in the algorithm and the encoding.

In the mentioned reference, the setting refered to is Stochastic Optimization with access to a k-bit estimate of the gradient through an oracle. This is in fact equivalent to the $(k, 1, 1)$ setting, with only the objective differing. This lower bound follows directly from their result via a standard online-to-batch conversion; see e.g. Hazan (2016).

E.2.3. DELAYS

Let us now incorporate the impact of the delays induced by the transmission of information through the network. It consists in implementing an instance of online learning with delays in our decentralized setting.

Note also that the randomization cannot improve on the worst-case regret, as a randomized algorithm can always be converted into a deterministic algorithm incurring the same regret against linear losses. For example, such a conversion is obtained by considering the virtual algorithm playing the expected prediction.

Lemma 27 (Reduction to Learning with Delays) *For any network with diameter $D = D(\mathcal{G})$, there exists an activation sequence such that decentralized OCO for gradient-oblivious algorithms with k bits per message is equivalent to online learning with delays of length $\lfloor D/2 \rfloor$ and with k-bit memory.*

In particular, for any $U > 0$, if $B(k, t, U)$ is a lower bound on the minimax regret for the $(k, 1, 1)$ protocol in OCO against comparators $\boldsymbol{u} \in \mathcal{B}_2(U)$ for all $t \in \mathbb{N}$, then

$$\sup_{G-\textit{Lipschitz losses}} \sup_{\boldsymbol{u}:\, \|\boldsymbol{u}\|\leqslant U} \mathbb{E}\left[\mathcal{R}_T(\boldsymbol{u})\right] \geqslant \left\lfloor \frac{D}{2} \right\rfloor B\left(k, \left\lfloor \frac{2T}{D} \right\rfloor, U\right) ,$$

where the expectation is taken with respect to the randomness in the algorithm and the encoding.

Proof Consider a maximal path through the graph of length D. Without loss of generality, we index the sequence of nodes by $\{1, \ldots, D\}$, assuming that node n is connected to node m if and only if $|n - m| = 1$.

Then consider the activation sequence that sequentially selects the nodes $1, 3, \ldots, 2\lfloor D/2 \rfloor - 1$ and starts over to 1. The information available to the active node I_t is always $\lfloor D/2 \rfloor$ rounds old; we denote this number by τ to reduce clutter.

Under this worst-delay activation sequence, solving the decentralized online learning problem is equivalent to solving an OCO instance with the same losses and delay τ. Furthermore, the k-bit memory constraint affects identically the OCO instance. Note also that the definition of regret coincides under both settings.

The final claim is from Zinkevich et al. (2009, Lemma 3); see also Hsieh et al. (2020, Proposition 16) for a more complete argument that accounts for non-deterministic algorithms. ■

All the ingredients to conclude are now available. Plugging in the lower bounds from Lemma 24 with Theorem 25, and Theorem 26 into Lemma 27, we obtain the results of Theorems 5 and Theorem 8, respectively.