Efficient Online Multiclass Prediction on Graphs via Surrogate Losses

Alexander Rakhlin
University of Pennsylvania

Karthik Sridharan
Cornell University

Abstract

We develop computationally efficient algorithms for online multi-class prediction. Our construction is based on carefully-chosen data-dependent surrogate loss functions, and the new methods enjoy strong mistake bound guarantees.

To illustrate the technique, we study the combinatorial problem of node classification and develop a prediction strategy that is linear-time per round. In contrast, the offline benchmark is NP-hard to compute in general. We demonstrate the empirical performance of the method on several datasets.

1 Introduction

As a motivating example, consider the problem of classifying nodes in a network in an online manner. On each round, we make a multi-class prediction, observe the outcome, and move to the next node. The aim is to incur a small number of mistakes by taking advantage of the known network structure. Properties of the graph that may lead to better prediction accuracy in this natural prediction problem have been investigated in [9, 10, 4, 17, 12], among others.

How does one model such a multi-class prediction problem, and what are the good prediction methods? The first two sections of this paper are devoted to these general questions, and the later sections address the particular problem of node classification.

Let $y_1, \ldots, y_n$ be a sequence with values in $\{1, \ldots, k\} \triangleq [k]$ that we wish to predict in an online manner. If we denote by $\widehat{y}_t = \widehat{y}_t(y_1, \ldots, y_{t-1}) \in [k]$ the (possibly randomized) prediction made by an algorithm $\mathcal{A}$ on round t, the expected average number of mistakes incurred on a sequence $\mathbf{y} = (y_1, \ldots, y_n)$ is

$$\mu_{\mathcal{A}}(\mathbf{y}) \triangleq \mathbb{E}\left[\frac{1}{n}\sum_{t=1}^{n} \mathbf{1}\left\{\widehat{y}_t \neq y_t\right\}\right], \tag{1}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function. For *any* algorithm $\mathcal{A}$, the function $\mu_{\mathcal{A}}$ can be shown to satisfy

$$\frac{1}{k^n}\sum_{\mathbf{y}} \mu_{\mathcal{A}}(\mathbf{y}) = 1 - \frac{1}{k}. \tag{2}$$

In words, the performance of *any* algorithm, when averaged over all possible sequences, cannot be better than random guessing. This version of a No-Free-Lunch Theorem should come as no surprise, since we have not made any assumptions or specifications that distinguish one sequence from another.

Prior knowledge about the prediction problem at hand may be formalized by positing distributional assumptions on the sequence, a prevalent approach in time-series prediction. In the context of node classification, this might involve an assumption on the process that generated the graph, such as the Stochastic Block Model with latent class memberships.

An alternative approach is to directly model the solutions to the prediction problem, bypassing the step of modeling and estimating the stochastic process generating the data. This paper is devoted to this second approach, which appears to be a convenient alternative for complex online problems such as node classification in a social network. It has also been shown that this latter approach, aimed directly at the problem of prediction rather than the problem of estimation, may circumvent computational hardness of finding the best model given the data [13]. Let us now describe this approach.

From (2), an algorithm may be better on some sequences (in terms of the expected average number of mistakes) only at the expense of being worse on some other sequences. This immediately suggests a modeling approach: pick an algorithm $\mathcal{A}$ such that $\mu_{\mathcal{A}}$ is small on the sequences we expect to see in practice. For instance, in node classification, we might hope to develop an algorithm with small $\mu_{\mathcal{A}}$ on sequences of node labels that do not have too many disagreements on the edges of the graph. At the outset, it is not entirely clear that this task is possible in general. More

Appearing in Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Flordia, USA. JMLR: W&CP volume 54. Copyright 2017 by the authors.

precisely, suppose we pick a function $\phi : [k]^n \to \mathbb{R}$ that should control the expected number of mistakes on each sequence. Is there an algorithm that guarantees $\mu_{\mathcal{A}} = \phi$?

For $k = 2$, this question was asked and answered by Cover [7]. More precisely, Cover's result states that under the stability condition

$$|\phi(\ldots, 1, \ldots) - \phi(\ldots, 2, \ldots)| \le 1/n, \tag{3}$$

(for any coordinate, keeping the rest fixed), there is an algorithm satisfying $\mu_{\mathcal{A}} = \phi$ *if and only if* $2^{-n} \sum_{\mathbf{y}} \phi(\mathbf{y}) = 1/2$. The proof of this fact relies crucially on the stability condition (3), which ensures that a certain optimization problem has a solution within the probability simplex.

For $k > 2$, the analogous result may be stated as:

Lemma 1. *Suppose* $\phi : [k]^n \to \mathbb{R}$ *has the stability property*

$$\max_{r \in [k]} \phi(\ldots, r, \ldots) - \frac{1}{k} \sum_{i=1}^{k} \phi(\ldots, i, \ldots) \le \frac{1}{nk} \tag{4}$$

for any coordinate. Then there exists an algorithm $\mathcal{A}$ *with the expected average number of mistakes function* $\mu_{\mathcal{A}} = \phi$ *if and only if*

$$\mathbb{E}\phi(u_1, \ldots, u_n) = 1 - \frac{1}{k}, \tag{5}$$

where u_i*'s are independent uniform on* $[k]$*. Moreover, the randomized algorithm* $\mathcal{A}$ *has an explicit form: on round t, given* $y_1, \ldots, y_{t-1}$*, the probability distribution* q_t *for predicting the class label is given by*

$$q_t(i) = \frac{1}{k} + \psi_i - \frac{1}{k} \sum_{j=1}^{k} \psi_j, \tag{6}$$

where the scores ψ_i *are computed as*

$$\psi_i = -n\mathbb{E}\phi(y_1, \ldots, y_{t-1}, i, u_{t+1}, \ldots, u_n). \tag{7}$$

The proof is postponed to the Appendix.

Lemma 1 should be viewed as a tool for modeling prior knowledge without stochastic assumptions on the sequence. As soon as (5) is verified for the chosen function ϕ, there is an explicit algorithm with a guaranteed performance $\mu_{\mathcal{A}}(\mathbf{y}) = \phi(\mathbf{y})$ on all sequences.

Occasionally, it is easier to check that the inequality $\mathbb{E}\phi(u_1, \ldots, u_n) \ge 1 - \frac{1}{k}$ (rather than the equality) holds. In this case, the algorithm is guaranteed a mistake bound $\mu_{\mathcal{A}}(\mathbf{y}) \le \phi(\mathbf{y})$ for all $\mathbf{y} \in [k]^n$.

A standard approach to constructing ϕ is to take $F \subseteq [k]^n$ and define

$$\phi(\mathbf{y}) = \mathrm{d_H}(\mathbf{y}, F) + C_n(F) \tag{8}$$

for the normalized Hamming distance

$$\mathrm{d_H}(\mathbf{y}, F) \triangleq \min_{\mathbf{w} \in F} \frac{1}{n} \sum_{t=1}^{n} \mathbf{1}\{w_t \ne y_t\} \tag{9}$$

and $C_n(F)$ a constant that measures "complexity" of F. From (5), the smallest allowed value of $C_n(F)$ is

$$C_n(F) = \left(1 - \frac{1}{k}\right) - \mathbb{E}\mathrm{d_H}(\boldsymbol{u}, F) \tag{10}$$

$$= \max_{\mathbf{w} \in F} \frac{1}{n} \sum_{t=1}^{n} e_{w_t}{}^{\mathsf{T}} e_{u_t} - \frac{1}{k} \tag{11}$$

where e_j is the jth standard unit vector and $\boldsymbol{u} = (u_1, \ldots, u_n)$. For $k = 2$, this value of $C_n(F)$ can be written as half of the Rademacher averages of the set F, a result that has appeared many times in the literature (e.g. in [5], where elements of F are termed static experts). Indeed, the reader familiar with the online learning literature will recognize (8) as an upper bound in the regret inequality and $C_n(F)$ as the regret with respect to the set F of "experts."

Lemma 1 together with the standard definition (8) can be employed to model many interesting prediction problems. Consider, once again, the problem of online node classification on a network. How can the graph structure help us in making good predictions? As already mentioned, one basic property we may expect is label similarity for neighboring nodes. To capture this "smoothness" of the labeling with respect to the graph in the ϕ function, one may define a set of κ-smooth labelings as the set

$$F_\kappa = \{\mathbf{w} \in [k]^n : \mathbf{w}^{\mathsf{T}} L \mathbf{w} \le \kappa\} \tag{12}$$

in terms of the graph Laplacian L and a parameter κ, and then define ϕ as in (8) (recall that $L = D - A$ where D is the diagonal matrix of node degrees and A is the adjacency matrix). This approach was investigated in [13].

To compute the prediction when ϕ is defined as in (8), one needs to be able to evaluate ϕ, which is an integer program (maximization of a linear function over the discrete set F). While this computation may be NP-hard—e.g. for the case when F_κ in (12) is defined with respect to a weighted graph Laplacian with negative weights—the authors of [13, 12] proposed polynomial-time methods that nearly attain the mistake bound of $\phi(\mathbf{y})$. The idea is to relax the combinatorial subset F, used in the definition of ϕ, to a larger subset of the hypercube with a polynomial number of constraints, so that the optimization problem can be solved in polynomial-time. As shown in [13, 12], the integrality gap for relaxing the integer program only appears in the lower-order term $C_n(F)$, which is good news from the point of view of prediction accuracy. However, while the approach leads to polynomial-time methods, the algorithms are not easily implementable and not computationally efficient in practice on large-scale problems. The obstacle is, in fact, the stability condition (4), as it forces the optimization problem to be some approximation to the combinatorial problem restricted to the hypercube.

In summary, the applicability of Lemma 1 is limited by the stability condition (4), and the methods developed so far (for combinatorial prediction problems described in [13, 12]) are polynomial-time, but may not be efficient in practice. This served as a motivation for the present paper.

When computational efficiency is a concern, a natural approach is to employ a convex surrogate loss function $\boldsymbol{\ell}$ that serves as an upper bound on the zero-one loss. As a first attempt, we may define

$$\phi(\mathbf{y}) = \min_{\mathbf{w}\in F} \frac{1}{n}\sum_{t=1}^{n} \boldsymbol{\ell}(w_t, y_t) + C'_n(F). \tag{13}$$

However, the stability condition required by the lemma fails for interesting loss functions, such as the hinge loss, and Lemma 1 no longer applies. A different approach is needed.

The use of convex surrogates, such as the hinge loss, has a long history within both online and statistical learning, with roots in the analysis of Perceptron and large-margin classifiers, and dating back to the early days of Machine Learning. Within the context of online supervised learning, the surrogate-loss approach leads to mistake bounds for linear predictors (see [11, 8, 15, 6] and the references therein) in terms of the surrogate loss of the best classifier. In contrast, Lemma 1 with ϕ defined in (8) yields a prediction method that enjoys a mistake bound in terms of the *zero-one loss of the comparator* (rather than a surrogate loss), and the method *can* be polynomial-time if F can be relaxed as in [13, 12]. Therefore, it is natural to ask whether the use of a surrogate loss function can further speed up computation and still yield a mistake bound in terms of the zero-one loss of the comparator. In the next section we indeed show that this is possible, with an interesting transition between binary classification and $k > 2$.

The main contribution of this paper is a new approach to multi-class classification that employs a carefully-constructed data-dependent surrogate loss. Section 2 is devoted to this general analysis. Section 3 is devoted to the node classification problem, while Section 4 contains experiments.

We close this section with an informal statement about the prediction performance and running time of the developed method for node classification.

Theorem 2 (Informal, see Theorem 6)**.** *Consider the set F_κ defined in* (12) *and let $M = \left(\frac{1}{2\kappa}L + \frac{1}{2n}I_n\right)^{-1}$, with I_n identity and L the graph Laplacian. There is an $O(n)$ per round algorithm $\mathcal{A}$ with expected average number of mistakes $\mu_{\mathcal{A}}$ at most*

$$d_H(\mathbf{y}, F_\kappa) + \frac{1}{n}\sqrt{2\cdot \mathrm{trace}(M)},$$

for $k = 2$. For $k > 2$, the first term gains a multiplicative factor $2\left(1 - \frac{1}{k}\right)$.

2 Surrogate Loss

Why is stability (4) required for Lemma 1? For brevity, let us drop the time index t and consider a single time step. Denoting by $\bar{\psi}$ the average of all ψ_i, the condition $q(i) \geq 0$ for all i is equivalent to

$$\bar{\psi} - \min_i \psi_i \leq 1/k, \tag{14}$$

which follows from the stability condition (4) and the definition of ψ_i. Hence, stability of ϕ is a sufficient condition for ensuring that q is a proper probability distribution.

A general approach to designing a surrogate loss function is to first compute scores $\psi = (\psi_1, \ldots, \psi_k)$ that correspond to "likelihood" of each class for the given example. We will compute these scores in a manner different from (7), and, in particular, we will not be able to guarantee (14). Yet, we will design a prediction strategy that will coincide with (6) if it happens that the scores do satisfy (14). Hence, this paper provides a strict generalization of Lemma 1.

In the setting of supervised learning with side-information x and multi-class label y, scores are typically computed as functions of the x-variable (see e.g. [16, Chapter 17]). For the online multi-class prediction problem, we also follow the route of calculating the scores, but the analysis will be more intricate since the scores depend on the global structure of F. The scores will be computed as solutions to a certain optimization problem, and, for the time being, suppose that ψ_i's are given to us.

A natural approach to extending Lemma 1 beyond stability is to construct q greedily. Let us find the level τ such that

$$\sum_{i=1}^{k} [\psi_i - \tau]_+ = 1, \tag{15}$$

where $[a]_+ = \max\{a, 0\}$, and define the support set

$$\mathsf{S}(\psi) = \{j \in [k] : \psi_j > \tau\}$$

of size $s \triangleq |\mathsf{S}(\psi)|$ and the distribution $q = q(\psi)$ via

$$q_j = [\psi_j - \tau]_+ = (\psi_j - \tau)\cdot \mathbf{1}\left\{j \in \mathsf{S}(\psi)\right\}.$$

Just as in Lemma 1, the value q_j can be written as

$$q_j = \left(\frac{1}{s} + \psi_j - \frac{1}{s}\sum_{i\in \mathsf{S}(\psi)} \psi_i\right)\cdot \mathbf{1}\left\{j \in \mathsf{S}(\psi)\right\}. \tag{16}$$

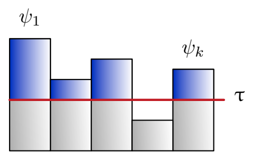


Figure 1: Blue region represents the distribution q.

Henceforth, we shall encode the class label by the standard basis vector $y \in \{e_1, \dots, e_k\}$.

For $g \in \mathbb{R}^k$ and $y \in \{e_1, \dots, e_k\}$ we define a ψ-dependent surrogate loss as

$$\ell_\psi(g, y) = \begin{cases} \xi(g, y) & \text{if } y \notin \mathsf{S}(\psi) \\ 1 - g^\top y + \frac{\sum_{j \in \mathsf{S}(\psi)} g_j - 1}{|\mathsf{S}(\psi)|} & \text{if } y \in \mathsf{S}(\psi) \end{cases}$$

for any surrogate loss function $\xi : \mathbb{R}^k \times \{e_1, \dots, e_k\} \to \mathbb{R}$ with the property that

$$\xi(\psi, y) \geq 1 \quad \text{whenever} \quad y \notin \mathsf{S}(\psi).$$

We now claim that ℓ_ψ is a surrogate loss function in the following sense:

Lemma 3. *For any $\psi \in \mathbb{R}^k$ and any $j \in [k]$,*

$$\begin{aligned} &\mathbb{E}_{\widehat{y} \sim q(\psi)} \left[\mathbf{1}\{\widehat{y} \neq y\} \right] \\ &\quad - \left[\xi(e_j, y) \mathbf{1}\{y \notin \mathsf{S}(\psi)\} + \mathbf{1}\{y \neq e_j\} \mathbf{1}\{y \in \mathsf{S}(\psi)\} \right] \\ &\leq \ell_\psi(\psi, y) - \ell_\psi(e_j, y). \end{aligned} \tag{17}$$

Before proving the Lemma, let us interpret the result. We think of the statement as an interpolation between the "nice" case when stability holds and the case when it does not hold. If ψ satisfies stability (14), $\mathsf{S}(\psi) = [k]$ and the left-hand side of the inequality (17) is

$$\mathbb{E}_{\widehat{y} \sim q(\psi)} \left[\mathbf{1}\{\widehat{y} \neq y\} \right] - \mathbf{1}\{y \neq e_j\},$$

which corresponds to the situation in Lemma 1 when ϕ is defined as in (8). That is, the expected error is compared to the zero-one loss. If the support of the distribution q is smaller than $[k]$ and y is outside this support, the zero-one loss of the prediction strategy is compared instead to the surrogate loss of the comparator.

Proof of Lemma 3. First, for any $j \in [k]$ and any $y \notin \mathsf{S}(\psi)$,

$$\ell_\psi(e_j, y) = \xi(e_j, y).$$

On the other hand if $y \in \mathsf{S}(\psi)$, then for any $j \in [k]$

$$\begin{aligned} \ell_\psi(e_j, y) &= 1 - e_j^\top y + \frac{\sum_{i \in \mathsf{S}(\psi)} e_j(i) - 1}{|\mathsf{S}(\psi)|} \\ &= \mathbf{1}\{e_j \neq y\} + \frac{\sum_{i \in \mathsf{S}(\psi)} e_j(i) - 1}{|\mathsf{S}(\psi)|} \\ &\leq \mathbf{1}\{e_j \neq y\}. \end{aligned}$$

The expected classification loss when the prediction is drawn from q is simply

$$\mathbb{E}_{\widehat{y} \sim q} \mathbf{1}\{\widehat{y} \neq y\} = 1 - q^\top y.$$

From the form of q in (16) (which we now write as $q(\psi)$ to emphasize the dependence on the given score vector), if $y \notin \mathsf{S}(\psi)$,

$$1 - q(\psi)^\top y = 1 \leq \xi(\psi, y).$$

On the other hand, when $y \in \mathsf{S}(\psi)$,

$$1 - q(\psi)^\top y = 1 - \psi_y + \frac{1}{|\mathsf{S}(\psi)|} \left(\sum_{i \in \mathsf{S}(\psi)} \psi_i - 1 \right).$$

Hence, we have that

$$\mathbb{E}_{\widehat{y} \sim q(\psi)} \left[\mathbf{1}\{\widehat{y} \neq y\} \right] = 1 - q(\psi)^\top y \leq \ell_\psi(\psi, y)$$

and for any $j \in [k]$,

$$\begin{aligned} &\xi(e_j, y) \mathbf{1}\{y \notin \mathsf{S}(\psi)\} + \mathbf{1}\{y \neq e_j\} \mathbf{1}\{y \in \mathsf{S}(\psi)\} \\ &\quad \geq \ell_\psi(e_j, y). \end{aligned}$$

□

We now bring back the time index t. Elements of $F \subseteq [k]^n$ will now be denoted by f, and, abusing the notation, f_t shall stand for the basis vector associated to the class f_t (to avoid writing double subscripts in e_{f_t}). We proved

$$\begin{aligned} &\sum_{t=1}^n \mathbb{E}_{\widehat{y}_t \sim q(\psi_t)} \mathbf{1}\{\widehat{y}_t \neq y_t\} \\ &\quad - \inf_{f \in F} \left\{ \sum_{t=1}^n \Big(\xi(f_t, y_t) \mathbf{1}\{y_t \notin \mathsf{S}(\psi_t)\} \right. \\ &\qquad\qquad \left. + \mathbf{1}\{y_t \neq f_t\} \mathbf{1}\{y_t \in \mathsf{S}(\psi_t)\} \Big) \right\} \\ &\leq \sum_{t=1}^n \ell_{\psi_t}(\psi_t, y_t) - \inf_{f \in F} \sum_{t=1}^n \ell_{\psi_t}(f_t, y_t), \end{aligned}$$

Using convexity of the loss for any ψ and y, we have that,

$$\ell_\psi(g, y) - \ell_\psi(h, y) \leq \nabla_g \ell_\psi(g, y)^\top (g - h)$$

for any $g, h \in \mathbb{R}^k$. Thus

$$\sum_{t=1}^n \mathbb{E}_{\widehat{y}_t \sim q(\psi_t)} \mathbf{1}\{\widehat{y}_t \neq y_t\} \tag{18}$$

$$\begin{aligned} &- \inf_{f \in F} \left\{ \sum_{t=1}^n \Big(\xi(f_t, y_t) \mathbf{1}\{y_t \notin \mathsf{S}(\psi_t)\} \right. \\ &\qquad\qquad \left. + \mathbf{1}\{y_t \neq f_t\} \mathbf{1}\{y_t \in \mathsf{S}(\psi_t)\} \Big) \right\} \end{aligned}$$

$$\leq \sum_{t=1}^n \nabla_t^\top (\psi_t - f_t) \tag{19}$$

where ∇_t is defined as

$$\nabla_a \ell_{\psi_t}(a, y_t)|_{a=\psi_t} = \begin{cases} \nabla\xi(\psi_t, y_t) & \text{if } y_t \notin \mathsf{S}(\psi_t) \\ \frac{1}{|\mathsf{S}(\psi_t)|}\mathbf{1}_{\mathsf{S}(\psi_t)} - y_t & \text{otherwise} \end{cases} \tag{20}$$

with $\mathbf{1}_\mathsf{S}$ denoting the indicator vector of coordinates in S and $\nabla\xi(\psi_t, y_t)$ being the gradient with respect to the first coordinate.

In Section 3.1 we show how one can use a relaxation-based approach to produce ψ_t's that yield a good upper bound for the right-hand side of (18). For now, let us just denote this bound by B_n.

What could be a good convex surrogate loss $\xi(g, y)$? First, observe that the surrogate itself can depend on ψ. Of course, we require that $\xi_\psi(\psi, y) \geq 1$ when $y \notin \mathsf{S}(\psi)$. An additional property that would make the bound interpretable is that for any $h \in \{e_1, \ldots, e_k\}$, $\xi_\psi(h, y) \leq \alpha\mathbf{1}\{h \neq y\}$ for some α. Such an inequality would allow us to express the mistake bound purely in terms of the indicator loss of the comparator. As we now show, the hinge loss

$$\boldsymbol{\ell}_{\text{hinge}}(g, y) \triangleq 1 + \max_{r: e_r \neq y} g^\top e_r - g^\top y. \tag{21}$$

possesses the desired properties after an appropriate scaling.

Lemma 4. *Suppose there is a method for choosing ψ_t such that* (18) *is upper bounded by B_n. Take*

$$\xi_\psi(g, y) = \frac{\boldsymbol{\ell}_{\text{hinge}}(g, y)}{1 + 1/|\mathsf{S}(\psi)|}. \tag{22}$$

Then

$$\sum_{t=1}^n \mathop{\mathbb{E}}_{\widehat{y}_t \sim q(\psi_t)} \mathbf{1}\{\widehat{y}_t \neq y_t\} \leq \inf_{f \in F}\left\{2\left(1 - \frac{1}{k}\right)\sum_{t: y_t \notin \mathsf{S}(\psi_t)} \mathbf{1}\{f_t \neq y_t\} + \sum_{t: y_t \in \mathsf{S}(\psi_t)} \mathbf{1}\{y_t \neq f_t\}\right\} + B_n$$

In particular, for binary classification ($k = 2$), we obtain the regret bound

$$\sum_{t=1}^n \mathop{\mathbb{E}}_{\widehat{y}_t \sim q(\psi_t)} \mathbf{1}\{\widehat{y}_t \neq y_t\} \leq \inf_{f \in F} \sum_{t=1}^n \mathbf{1}\{f_t \neq y_t\} + B_n.$$

Proof of Lemma 4. First, when $y_t \notin \mathsf{S}(\psi_t)$, it holds that $\mathop{\mathbb{E}}_{\widehat{y}_t \sim q(\psi_t)} \mathbf{1}\{\widehat{y}_t \neq y_t\} = 1$. Hence, from (18),

$$\sum_{t=1}^n \mathop{\mathbb{E}}_{\widehat{y}_t \sim q(\psi_t)} \mathbf{1}\{\widehat{y}_t \neq y_t\} \leq \inf_{f \in F}\left\{\sum_{t: y_t \notin \mathsf{S}(\psi_t)} \xi(f_t, y_t) \mathop{\mathbb{E}}_{\widehat{y}_t \sim q(\psi_t)} \mathbf{1}\{\widehat{y}_t \neq y_t\} + \sum_{t: y_t \in \mathsf{S}(\psi_t)} \mathbf{1}\{y_t \neq f_t\}\right\} + B_n.$$

Notice, that for any $j \in [k]$,

$$\boldsymbol{\ell}_{\text{hinge}}(e_j, y) = 2 \cdot \mathbf{1}\{e_j \neq y\}.$$

On the other hand, $\boldsymbol{\ell}_{\text{hinge}}(\psi, y) \geq 1 + 1/|\mathsf{S}(\psi)|$ whenever $y \notin \mathsf{S}(\psi)$. Indeed, the value of τ in (15) is $\frac{1}{|\mathsf{S}(\psi)|}\sum_{i \in \mathsf{S}(\psi)} \psi_i - 1/|\mathsf{S}(\psi)|$, and the maximum value of ψ is at least its average on $\mathsf{S}(\psi)$, implying that the difference in (21) is at least $1/|\mathsf{S}(\psi)|$.

Hence, $\xi_\psi(\psi, y) \geq 1$ when $y \notin \mathsf{S}(\psi)$ and, further, for any $j \in [k]$, we have that $\xi_\psi(e_j, y) \leq \frac{2}{1 + \frac{1}{\mathsf{S}(\psi)}}\mathbf{1}\{e_j \neq y\}$. When $y \notin \mathsf{S}(\psi)$, we have that $|\mathsf{S}(\psi)| \leq k - 1$, implying $\xi_\psi(e_j, y) \leq 2\left(1 - \frac{1}{k}\right)\mathbf{1}\{e_j \neq y\}$. ☐

Surprisingly, for binary classification we are able to upper bound regret with respect to the *zero-one loss* of the comparator class F with an efficient method based on surrogate loss, as long as the linearized problem in (18) can be solved efficiently. We contrast our result with the typical mistake bounds in terms of the surrogate loss of the comparator, a price paid for computational efficiency.

While the online linear optimization problem with respect to F may be, once again, computationally intractable, we can now take a superset $F' \supset F$ for which this optimization is efficient and B_n is nontrivial. What distinguishes this approach from the one taken in [13, 12] is that F' no longer needs to be a subset of the hypercube. In Section 3 we shall take an ellipse that roughly approximates F, making the computation of the scores ψ essentially $O(n)$ per step after initial pre-processing.

3 Node Classification in Networks

The techniques developed in this paper will be illustrated on the problem of node classification in networks. In its simplest form, the problem is to predict a multi-class label at each time step, with the true outcome revealed after each step. We suppose that the order of node presentation is fixed ahead of time, although this restriction is easily removed since the proposed method will not depend on the order of the unseen nodes.

Several properties of the graph have been investigated in connection to the number of mistakes incurred by a prediction method. These include the cluster structure [9], the cut-size of the graph [10], cuts of random spanning trees [4], and random spanning trees on weighted graphs [17].

The approach of this paper is more aligned with the notions of supervised online or statistical learning, whereby the quality of prediction is compared to a benchmark set. This benchmark set captures the prior knowledge of the practitioner as the set of good solutions, as described in Section 1. In particular, we are interested in problems where the graph structure induces homogeneity (or, a like-dislike structure in case of signed graphs) of the observed sequence of outcomes.

To be more concrete, let the graph $G = (V, E)$, $|V| = n$, be fixed and known to us. At each time step $t = 1, \ldots, n$, we are required to predict a label of each node, without repetition. Our prediction is $\widehat{y}_t \in \{e_1, \ldots, e_k\}$, and the observed outcome is $y_t \in \{e_1, \ldots, e_k\}$. To model the problem, we choose, as before, a set $F \subseteq [k]^n$.

To be consistent with the vector notation used in the previous sections, we represent F as a subset of

$$\mathcal{W} = \left\{ W \in \{0,1\}^{k\times n} : \sum_i W_{i,t} = 1 \; \forall t \in [n] \right\}. \quad (23)$$

A vector $g \in [k]^n$ is identified with a matrix W such that $W_{i,t} = \mathbf{1}\{g(t) = i\}$. Let W^i denote the ith column of $W^\intercal$.

For the rest of the paper, we shall focus on the following generalization of the set F defined in (12), written in the present notation as

$$F_\kappa = \left\{ W \in \mathcal{W} : \sum_{i=1}^{k} (W^i)^\intercal L W^i \leq \kappa \right\} \quad (24)$$

where L is a positive semidefinite matrix. Examples of L are:

1. graph Laplacian: $L = D - A$, where D is the diagonal matrix of node degrees and A is the adjacency matrix
2. weighted/signed Laplacian: $L = D - W$, where D is diagonal with $D_{i,i} = \sum_j |W_{i,j}|$, W is the matrix of edge weights
3. normalized graph Laplacian: $L = I - D^{-1/2} A D^{-1/2}$

For the first example, the quadratic form in (24) is the value of the cut induced by the labeling W. The other two examples have similar interpretations as measuring "homogeneity" of the labeling W with respect to the graph. As mentioned before, optimization over these sets is, in general, an NP-hard problem.

Let I_n be the $n \times n$ identity matrix. For a positive semidefinite matrix L, the set F_κ can be relaxed as follows:

$$\begin{aligned} F_\kappa \subseteq & \left\{ W \in \mathbb{R}^{k\times n} : \sum_{i=1}^{k} (W^i)^\intercal \left(\frac{1}{2\kappa} L \right) W^i \leq \frac{1}{2} \right\} \\ & \cap \left\{ W \in \mathbb{R}^{k\times n} : \sum_{i=1}^{k} (W^i)^\intercal \left(\frac{1}{2n} I_n \right) W^i \leq \frac{1}{2} \right\} \\ & \subseteq \left\{ W \in \mathbb{R}^{k\times n} : \sum_{i=1}^{k} (W^i)^\intercal \left(\frac{1}{2\kappa} L + \frac{1}{2n} I_n \right) W^i \leq 1 \right\}. \end{aligned}$$

Denote the last set by $\bar{F}_\kappa$ and observe that

$$\sup_{W \in \bar{F}_\kappa} W \bullet Y = \sqrt{\sum_{i=1}^{k} (Y^i)^\intercal M Y^i} \quad (25)$$

for $M \triangleq \left(\frac{1}{2\kappa} L + \frac{1}{2n} I_n \right)^{-1}$ and any $Y \in \mathbb{R}^{k\times n}$.

The sets F_κ defined in (24) can be employed to model smoothness of prediction with respect to the graph. While other models are also possible, we only focus on these for the lack of space, and because the solutions are particularly simple.

3.1 Computing the Scores via Relaxations

We now introduce one last ingredient before deriving the algorithm. Consider the following online protocol. On each round $t = 1, \ldots, n$, we predict $\psi_t \in \mathbb{R}^k$ and observe $\nabla_t \in \sqrt{k} B_2$ in the Euclidean ball of $\mathbb{R}^k$ of radius $\sqrt{k}$. The goal is to attain small regret

$$\sum_{t=1}^{n} \nabla_t^\intercal (\psi_t - f_t)$$

for any $f \in F \subset \mathcal{W}$, as defined in (23). The vector f_t is now the standard basis vector corresponding to the tth column of f, matching the convention used in the previous sections.

It is tempting to view the problem through the lens of Online Convex Optimization. However, the dimensionality of the set F is n and a naive application of gradient or mirror descent yields a trivial bound. With a bit of work, however, a closed-form algorithm can be developed. The analysis below is based on the idea of relaxations.

A real-valued function on $\cup_{t=1}^{n} (B_2(D))^t$ such that

$$\mathbf{Rel}_n(\nabla_1, \ldots, \nabla_n) \geq - \inf_{f \in F} \sum_{t=1}^{n} \nabla_t^\top f_t \quad (26)$$

and

$$\inf_{\psi_t \in \mathbb{R}^k} \sup_{\|\nabla_t\| \leq D} \left\{ \nabla_t^\intercal \psi_t + \mathbf{Rel}_n(\nabla_1, \ldots, \nabla_t) \right\} \leq \mathbf{Rel}_n(\nabla_1, \ldots, \nabla_{t-1}) \quad (27)$$

is called admissible. In this case, any algorithm for choosing ψ_t such that (27) holds guarantees that

$$\sum_{t=1}^{n} \nabla_t^\top \psi_t - \inf_{f \in F} \sum_{t=1}^{n} \nabla_t^\top f_t \le \mathbf{Rel}_n(\emptyset). \tag{28}$$

Next, we will write down a relaxation function for the case of F_κ defined in (24), prove its admissibility, and derive an algorithm for choosing ψ_t's. At time t, define the matrix $Y_t = [\nabla_1, \dots, \nabla_t, \mathbf{0}, \dots, \mathbf{0}] \in \mathbb{R}^{k \times n}$ by stacking the observed vectors. Let Y_t^i be ith row (whose jth coordinate is $\nabla_j[i] \cdot \mathbf{1}\{j \le t\}$). Since $\bar{F}_\kappa \supseteq F_\kappa$,

$$\begin{aligned} -\inf_{f \in F_\kappa} \sum_{t=1}^{n} \nabla_t^\top f_t &\le -\inf_{f \in \bar{F}_\kappa} \sum_{t=1}^{n} \nabla_t^\top f_t \\ &= \sqrt{\sum_{i=1}^{k} (Y_n^i)^\intercal M Y_n^i} \\ &\triangleq \mathbf{Rel}_n(\nabla_1, \dots, \nabla_n) \end{aligned}$$

by (25), where recall that $M = \left(\frac{1}{2\kappa} L + \frac{1}{2n} I_n\right)^{-1} \in \mathbb{R}^{n \times n}$. For $t \le n$, the proof of admissibility suggests the definition

$$\mathbf{Rel}_n(\nabla_1, \dots, \nabla_t) \triangleq \sqrt{\sum_{i=1}^{k} (Y_t^i)^\top M Y_t^i + D^2 \sum_{j=t+1}^{n} M[j,j]}. \tag{29}$$

Lemma 5. *The relaxation in* (29) *is admissible, the algorithm has closed form*

$$\psi_t = -\frac{M[t,:] Y_{t-1}}{\sqrt{\sum_{i=1}^{k} (Y_{t-1}^i)^\top M Y_{t-1}^i + D^2 \sum_{j=t}^{n} M[j,j]}} \tag{30}$$

and

$$\begin{aligned} \mathbf{Rel}_n(\emptyset) &= D\sqrt{\mathrm{trace}(M)} \\ &= D\sqrt{\sum_{i=1}^{n} \lambda_i^{-1}(L/2\kappa + \mathbf{1}_{n \times n}/2n)} \end{aligned}$$

where $D \ge \max_t \|\nabla_t\|$ and $\lambda_i(A)$ denotes the ith eigenvalue of A.

The vector of scores ψ_t can be computed in $O(t \times K)$ time, given that M is initially pre-computed.

3.2 The Algorithm

We are now ready to put all the pieces together and write down the complete algorithm. For concreteness, take ξ_ψ to be the surrogate loss defined in (22) of Lemma 4, and observe that $D^2 \le 2$.

Algorithm 1 Prediction with Surrogate Loss

```
input  Matrix $L$, parameter $\kappa$
 1: Pre-compute $M = (L/2\kappa + I_n/2n)^{-1}$
 2: Set $T = \mathrm{trace}(M)$, $A = 0$, $G = []$
 3: for  $t = 1, \dots, n$ do
 4:     Compute $v = M_{t,\cdot} \times G$, the numerator in (30)
 5:     Compute scores $\psi_t = -\frac{v}{\sqrt{A + k \cdot T}}$
 6:     Compute the mixed strategy $q_t(\psi_t)$ as in (16), predict $\widehat{y}_t \sim q_t$, and observe $y_t$
 7:     Compute gradient $\nabla_t$ as in (20) and set $G = [G; \nabla_t]$
 8:     Update $A = A + 2\nabla_t^\top v + M_{t,t} \cdot \|\nabla_t\|_2^2$
 9:     $T = T - M_{t,t}$
10: end for
output
```

Theorem 6. *The average expected number of mistakes $\mu_{\mathcal{A}}$ for the above algorithm is at most*

$$\inf_{f \in F_\kappa} \frac{1}{n} \sum_{t=1}^{n} \mathbf{1}\{f_t \ne y_t\} + \frac{1}{n}\sqrt{2 \cdot \mathrm{trace}(M)}.$$

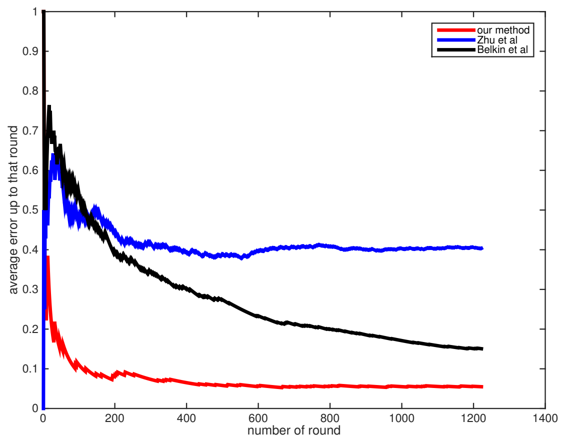
for the binary prediction case. For $k > 2$, the bound is

$$2\left(1 - \frac{1}{k}\right) \inf_{f \in F_\kappa} \frac{1}{n} \sum_{t=1}^{n} \mathbf{1}\{f_t \ne y_t\} + \frac{1}{n}\sqrt{2 \cdot \mathrm{trace}(M)}.$$

4 Experiments

To evaluate our algorithm on real world data, we used the following graph datasets: political blogs dataset (1222 nodes and binary labels), citeseer dataset (3312 nodes and 6 classes), and cora dataset (2708 nodes and 6 classes)[14, 1]. We also used the MNIST with random background noise dataset [2] and created a graph with 12000 nodes (one node per image) with points being connected if the Euclidean distances between the images were smaller than a given threshold. This dataset has 10 classes, one for every digit from '0' to '9'.

Since our method is built for the online classification scenario, we first run experiments where we track the average accuracy so far over a single pass over the n nodes as the labels of each one are revealed in an online fashion. The following figure shows the plot of average accuracies with number of rounds for (a) our method, (b) Zhu et al [18] (we added extra regularization for better accuracy) and (c) Belkin et al [3]. The first plot shows the average error up to a given round while predicting online, that is, $\sum_{j=1}^{t} \mathbf{1}\{\hat{y}_t \ne y_t\}/t$ against t. We run the other two methods in an online fashion re-evaluating predictions based on most recently observed labels. The first plot shows the results for the political blogs dataset. If this is contrasted to the table, which has results in the batch setting, it is clear that while on the Political blogs dataset the Belkin et al method and our method perform roughly the same, in the online setting our method far outperforms the other two.



Since the Zhu et al [18] and Belkin et al [3] results are developed for the statistical semi-supervised setting we also use our method in this setting by updating the online algorithm first over the labeled points and then using it over unlabeled nodes to make predictions. Specifically, we used a 90-10 split of points into labeled and unlabeled (randomly chosen split each time). For the Zhu et al [18] and Belkin et al [3], we used a validation set to pick parameters (10% of training data). We found that in general our method was very robust against changes in parameters. Hence, we simply picked a fixed κ for each experiment that only depended on n the number of data and not the data itself. The following table summarizes the accuracies on the various datasets.

	MNIST	Citeseer	Cora	Political
Ours	**85.9± 0.5**	**74.0 ± 1.0**	**81.7 ±1.6**	**95.2 ± 0.9**
[3]	82.0 ± 1.7	71.2 ± 1.6	78.3 ± 1.4	**94.9 ± 1.7**
[18]	19.3 ± 3.0	72.9 ± 1.4	67.0 ± 2.2	54.1 ± 2.9

We see that our method performs favorably in most cases, the bold font represents best method to within statistical significance. In fact, that a graph based approach (where graph simply picked based on tresholding distances between images) should even work this well for the MNIST with random background noise is surprising. Nearest neighbor and soft nearest neighbor approaches are known to perform badly for this dataset as random noise blows up distances between images even within the same class.

5 Summary and Future Directions

This paper developed new efficient algorithms that combine several ideas: surrogate loss minimization and relaxations. This approach falls outside the scope of Lemma 1 since the required stability condition fails. Yet, we presented a mistake bound in terms of the zero-one loss for binary classification, and a mix of zero-one and surrogate losses more generally. For the case of prediction on graphs, where the ϕ function is defined in terms of quadratic forms, the proposed methods run in time linear in n per iteration. The proposed method — while theoretically derived — exhibits good predictive performance in our experiments.

Future work includes developing efficient methods for other interesting models, beyond the Laplacian-based set F_κ. Among other interesting directions is the setting with side information at the nodes, and the case of a time-evolving graph.

Acknowledgments

Research supported in part by the NSF under grants CDS&E-MSS 1521529 and 1521544.

References

[1] Linqs, statistical relational learning group at UMD: Datasets. `http://linqs.umiacs.umd.edu/projects/projects/lbc/`.

[2] MNIST with random background dataset. `https://sites.google.com/a/lisa.iro.umontreal.ca/public_static_twiki/variations-on-the-mnist-digits`.

[3] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7(Nov):2399–2434, 2006.

[4] N. Cesa-Bianchi, C. Gentile, F. Vitale, and G. Zappella. Random spanning trees and the prediction of weighted graphs. *The Journal of Machine Learning Research*, 14(1):1251–1284, 2013.

[5] N. Cesa-Bianchi and G. Lugosi. On prediction of individual sequences. *Annals of Statistics*, pages 1865–1895, 1999.

[6] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[7] T. M. Cover. Behaviour of sequential predictors of binary sequences. In *Proc. 4th Prague Conf. Inform. Theory, Statistical Decision Functions, Random Processes*, 1965.

[8] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7(Mar):551–585, 2006.

[9] M. Herbster. Exploiting cluster-structure to predict the labeling of a graph. In *Algorithmic Learning Theory*, pages 54–69. Springer, 2008.

[10] M. Herbster and M. Pontil. Prediction on a graph with a Perceptron. In *Advances in neural information processing systems*, pages 577–584, 2006.

[11] J. Kivinen and M. K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, 1997.

[12] A. Rakhlin and K. Sridharan. Hierarchies of relaxations for online prediction problems with evolving constraints. In *COLT*, 2015.

[13] A. Rakhlin and K. Sridharan. A tutorial on online supervised learning with applications to node classification in social networks. *CoRR*, abs/1608.09014, 2016.

[14] P. Sen, G. M. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

[15] S. Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.

[16] S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

[17] F. Vitale, N. Cesa-Bianchi, C. Gentile, and G. Zappella. See the tree through the lines: The shazoo algorithm. In *Advances in Neural Information Processing Systems*, pages 1584–1592, 2011.

[18] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, volume 3, pages 912–919, 2003.

A Proof of Admissibility

Proof of Lemma 1. Define functions $\mathbf{Rel}_n : \cup_t [k]^t \to \mathbb{R}$ as

$$\mathbf{Rel}_n(y_1, \dots, y_n) = -\phi(y_1, \dots, y_n)$$

and

$$\mathbf{Rel}_n(y_1, \dots, y_{t-1}) = \mathbb{E}_{y_t \sim \text{Unif}[k]} \mathbf{Rel}_n(y_1, \dots, y_t) + \frac{1}{n}\left(1 - \frac{1}{k}\right), \tag{31}$$

with $\mathbf{Rel}_n(\emptyset)$ being a constant. We desire to prove that there is an algorithm such that

$$\forall \mathbf{y} \in [k]^n, \quad \mathbb{E}\left[\frac{1}{n}\sum_{t=1}^{n} \mathbf{1}\left\{\widehat{y}_t \neq y_t\right\}\right] - \phi(y_1, \dots, y_n) = 0.$$

Consider the last time step n and write the above expression as

$$\mathbb{E}\left[\frac{1}{n}\sum_{t=1}^{n-1} \mathbf{1}\left\{\widehat{y}_t \neq y_t\right\} + \frac{1}{n}\mathbf{1}\left\{\widehat{y}_n \neq y_n\right\} + \mathbf{Rel}_n(y_1, \dots, y_n)\right]. \tag{32}$$

Let $\mathbb{E}_{n-1}$ denote the conditional expectation given $\widehat{y}_1, \dots, \widehat{y}_{n-1}$. We shall prove that there exists a randomized strategy for the last step such that for any $y_n \in [k]$,

$$\mathbb{E}_{n-1}\left[\frac{1}{n}\mathbf{1}\left\{\widehat{y}_n \neq y_n\right\}\right] + \mathbf{Rel}_n(y_1, \dots, y_n) = \mathbf{Rel}_n(y_1, \dots, y_{n-1}). \tag{33}$$

This last statement is translated as

$$\min_{q_n \in \Delta_k} \max_{y_n \in [k]} \left\{\mathbb{E}_{n-1}\left[\frac{1}{n}\mathbf{1}\left\{\widehat{y}_n \neq y_n\right\}\right] + \mathbf{Rel}_n(y_1, \dots, y_n)\right\} = \mathbf{Rel}_n(y_1, \dots, y_{n-1}). \tag{34}$$

Writing $\mathbf{1}\left\{\widehat{y}_n \neq y_n\right\} = 1 - \boldsymbol{e}_{\widehat{y}_n}^{\top} \boldsymbol{e}_{y_n}$, the left-hand side of (34) is

$$\frac{1}{n} \min_{q_n \in \Delta_k} \max_{y_n \in [k]} \left\{1 - q_n^{\top} \boldsymbol{e}_{y_n} + n\mathbf{Rel}_n(y_1, \dots, y_n)\right\}. \tag{35}$$

The stability condition means that we can choose q_n to *equalize* the choices of y_n. Let $\psi(1), \dots, \psi(k)$ be the *sorted* values of

$$n\mathbf{Rel}_n(y_1, \dots, y_{n-1}, 1), \dots, n\mathbf{Rel}_n(y_1, \dots, y_{n-1}, k),$$

in non-increasing order. In view of the stability condition,

$$\sum_{i=1}^{k} (\psi(i) - \psi(k)) \leq 1.$$

Hence, q_n can be chosen so that all $\psi(i) - q_n(i)$ have the same value. One can check that this is the minimizing choice for q_n. Let q_n^* denote this optimal choice. The common value of $\psi(i) - q_n^*(i)$ can then be written as

$$\psi(k) - \frac{1}{k}\left(1 - \sum_{i=1}^{k} (\psi(i) - \psi(k))\right) = \frac{1}{k}\sum_{i=1}^{k} \psi(i) - \frac{1}{k}$$

and hence (35) is equal to

$$\frac{1}{n}\left(1 - \frac{1}{k}\right) + \frac{1}{k}\sum_{i=1}^{k} \mathbf{Rel}_n(y_1, \dots, y_{n-1}, i). \tag{36}$$

This value is precisely $\mathbf{Rel}_n(y_1, \dots, y_{n-1})$, as per Eq. (31), thus verifying (34). Repeating the argument for $t = n - 1$ until $t = 0$, we find that

$$\mathbf{Rel}_n(\emptyset) = -\mathbb{E}\phi + \left(1 - \frac{1}{k}\right) = 0,$$

thus ensuring existence of an algorithm with (32) equal to zero. The other direction of the statement is proved by taking sequences $\mathbf{y}$ uniformly at random from $[k]^n$, concluding the proof. $\square$

Proof of Lemma 5. Recall that

$$Y_t = [\nabla_1, \dots, \nabla_t, \mathbf{0}, \dots, \mathbf{0}]^\intercal.$$

We can write

$$\begin{aligned}
\mathbf{Rel}_n(\nabla_1, \dots, \nabla_t) &= \sqrt{\sum_{i=1}^k (Y_t^i)^\intercal M Y_t^i + D^2 \sum_{j=t+1}^n M[j,j]} \\
&= \sqrt{\sum_{i=1}^k (Y_{t-1}^i)^\intercal M Y_{t-1}^i + 2\nabla_t[i] M[t,:] Y_{t-1}^i + \nabla_t^2[i] M[t,t] + D^2 \sum_{j=t+1}^n M[j,j]} \\
&\le \sqrt{\sum_{i=1}^k \left((Y_{t-1}^i)^\intercal M Y_{t-1}^i + 2\nabla_t[i] M[t,:] Y_{t-1}^i\right) + D^2 M[t,t] + D^2 \sum_{j=t+1}^n M[j,j]}
\end{aligned}$$

since $\|\nabla_t\|_2^2 \le D^2$. Hence,

$$\begin{aligned}
&\inf_{\psi_t \in \mathbb{R}^k} \sup_{\|\nabla_t\| \le D} \left\{\nabla_t^\intercal \psi_t + \mathbf{Rel}_n(\nabla_1, \dots, \nabla_t)\right\} \\
&\qquad \le \inf_{\psi_t \in \mathbb{R}^k} \sup_{\|\nabla_t\| \le D} \left\{\nabla_t^\intercal \psi_t + \sqrt{\sum_{i=1}^k \left((Y_{t-1}^i)^\intercal M Y_{t-1}^i + 2\nabla_t[i] M[t,:] Y_{t-1}^i\right) + D^2 \sum_{j=t}^n M[j,j]}\right\}
\end{aligned}$$

Now the claim is that

$$\psi_t = -\frac{M[t,:] Y_{t-1}}{\sqrt{\sum_{i=1}^k (Y_{t-1}^i)^\intercal M Y_{t-1}^i + D^2 \sum_{j=t}^n M[j,j]}}$$

is the solution to the above minimization problem. To see this, note that for the given ψ_t, the gradient with respect to ∇_t is 0 and hence this ∇_t is the maximizer. Plugging in this solution we get an upper bound on the value

$$\begin{aligned}
&\sup_{\|\nabla_t\| \le D} \left\{\nabla_t^\intercal \psi_t + \sqrt{\sum_{i=1}^k \left((Y_{t-1}^i)^\intercal M Y_{t-1}^i + 2\nabla_t[i] M[t,:] Y_{t-1}^i\right) + D^2 \sum_{j=t}^n M[j,j]}\right\} \\
&\le \sqrt{\sum_{i=1}^k ((Y_{t-1}^i)^\intercal M Y_{t-1}^i) + D^2 \sum_{j=t}^n M[j,j]} \\
&= \mathbf{Rel}_n(\nabla_1, \dots, \nabla_{t-1}).
\end{aligned}$$

The bound at the end is given by

$$\mathbf{Rel}_n(\emptyset) = D\sqrt{\mathrm{trace}(M)}.$$

Now once the matrix M is pre-computed, the time complexity per round is $O(t)$. □