Regret Bounds for Prediction Problems

Geoffrey J. Gordon*
CMU Computer Science Department
5000 Forbes Avenue
Pittsburgh, PA 15213
ggordon@cs.cmu.edu

Abstract

We present a unified framework for reasoning about worst-case regret bounds for learning algorithms. This framework is based on the theory of duality of convex functions. It brings together results from computational learning theory and Bayesian statistics, allowing us to derive new proofs of known theorems, new theorems about known algorithms, and new algorithms.

1 The inference problem

We are interested in the following kind of inference problem: on each time step $t = 1 \ldots T$ we must choose a prediction vector w_t from a set of allowable predictions $\mathcal{W}$. The interpretation of w_t depends on the details of the problem, but for example w_t might be our guess at the mean of a sequence of numbers or the coefficients of a linear regression. Then the loss function $l_t(w)$ is revealed to us, and we are penalized $l_t(w_t)$. These penalties are additive, so our overall goal is to minimize $\sum_{t=1}^{T} l_t(w_t)$. Our choice of w_t may depend on $l_1 \ldots l_{t-1}$, and possibly on some additional prior information, but it may not depend on $l_t \ldots l_T$.

Many well-known inference problems, such as linear regression and estimation of mixture coefficients, are special cases of this one. To express one of these specific problems as an instance of our general inference problem, we will usually interpret the loss function l_t as encoding both a training example and a criterion to be minimized: the location of the set of minima of l_t encodes the training example, while the shape of l_t encodes the cost of deviations in each direction. This double role for l_t means that the loss function will usually change from step to step, even if we are always trying to minimize the same kind of errors. For example, if we wanted to estimate the mean of a population of numbers from a sample $z_1, z_2, \ldots$, then $l_t(w)$ might be $(w - z_t)^2$. This choice of l_t encodes both the current training point z_t and the fact that we are minimizing squared error. (See Figure 1 for more detail.) Or, if we were interested in a linear regression of y_t on x_t, $l_t(w)$ might be $(y_t - w \cdot x_t)^2$. This choice encodes both the current example (x_t, y_t) and the fact that we want to minimize the squared prediction error. Or, if we were trying to solve a mixture estimation problem, $l_t(w)$ might be $-\ln(w \cdot p_t)$, where w is the vector of mixture proportions and $p_{t,i}$ is the probability of the current training point under the ith model. (Here and below, the notation $p_{t,i}$ stands for the ith component of the vector p_t.) This choice of loss function encodes properties of the current example as well as the fact that we want to maximize log-likelihood.

We want to develop an algorithm for choosing a sequence of w_ts so as to minimize our total loss $\sum_{t=1}^{T} l_t(w_t)$, even if the sequence of loss functions l_t is chosen by an adversary. Unfortunately this problem is impossible without further assumptions: for example, the adversary could choose loss functions with corners or discontinuities and make the losses of two predictions v_t and w_t arbitrarily different even if v_t and w_t were close together. So, we will make two basic simplifications. The first is that we will place restrictions on the form of the functions l_t that the adversary may choose. The chief restrictions will be that l_t is convex and that a measure of the amount of information contained in l_t does not increase too quickly from trial to trial.

The second simplification is that we will seek a relative loss bound rather than an absolute one. That is, we will define a comparison class $\mathcal{U}$ of predictions, and we will seek to minimize our regret $\sum_{t=1}^{T}(l_t(w_t) - l_t(u))$ versus the best predictor $u \in \mathcal{U}$. (Often we will take $\mathcal{U} = \mathcal{W}$, so that we are comparing our predictions to the best constant prediction. Sometimes, though, we will need to take $\mathcal{U} \subset \mathcal{W}$ in order to prove a bound.) Since u can be chosen post hoc, with knowledge of the loss functions l_t, such a regret bound is a strong statement.

The focus on regret instead of just loss is the chief place where our results differ from traditional statistical estimation theory. It is what allows us to handle sequences of loss functions that are too difficult to predict: our theorems will still hold, but since there will be no comparison u that has small loss, the theorems will not tell us much about our total loss $\sum_{t=1}^{T} l_t(w_t)$.

*This research was sponsored in part by the DARPA HPKB program under contract F30602-97-1-0215.

Trial t	0	1	2	3	4
Prediction w_t	—	0	2	3	3
Training example z_t	—	4	5	3	8
Error type	—	Squared	Squared	Squared	Squared
Loss function $l_t(w)$	w^2	$(w-4)^2$	$(w-5)^2$	$(w-3)^2$	$(w-8)^2$
Loss of w_t	—	16	9	0	25
Ttl loss of $w_1 \ldots w_t$	0	16	25	25	50
Best constant u	4				
Loss of u	16	0	1	1	16
Ttl loss of u	16	16	17	18	34
Ttl regret	-16	0	8	7	16

Figure 1: An example of the MAP algorithm in action, trying to minimize sum of squared errors. The prediction at trial t is the mean of all examples up to trial $t-1$, while the comparison vector is the mean of all examples.

Surprisingly, with only weak restrictions on l_t and u, we will be able to prove bounds that are similar to the best possible average-case bounds (that is, bounds where l_t is chosen by some fixed probability law). Our theorems will unify results from classical statistics (inference in exponential families and generalized linear models) with those from computational learning theory (weighted majority, aggregating algorithm, exponentiated gradient).

This regret bound framework has been studied before in [LW92, KW97, KW96, Vov90, CBFH$^+$95] among others. Also, some of our results are similar to results from classical statistics such as the Cramer-Rao variance bound [SO91]. Our theorems are more general than each of these previous results in at least one of the following ways. First, they apply to more general classes of convex loss functions, including non-differentiable ones. Second, they apply to both online (*i.e.*, bounded computation per example) and offline (unbounded computation) algorithms. Third, they apply to all sequences of loss functions, not just on average. Finally, they apply at all time steps, not just asymptotically. Our theorems are also less general than traditional statistical results in some ways. For example, while the Cramer-Rao bound requires differentiability of the loss functions, it does not require global convexity, just local convexity.

All of our theorems will concern variations on the following simple and intuitively appealing algorithm, which takes as input the loss functions $l_1 \ldots l_{t-1}$ observed on previous trials plus one additional loss function l_0 which encodes our prior knowledge before the first trial.

MAP ALGORITHM: Predict any

$$w_t \in \arg\min_w \sum_{i=0}^{t-1} l_i(w)$$

The notation $\arg\min_w f(w)$ means the set of ws that minimize f. We assume that the minimum is always achieved so that a legal prediction always exists. Conditions which ensure the existence are described below. The algorithm is called "MAP" or "maximum a posteriori" because of its Bayesian roots: if we want to apply the MAP algorithm to the problem of estimating some population parameters w from an i.i.d. sample $z_1, z_2, \ldots$, then a good choice of loss function is the negative of the log likelihood $l_t(w) = -\ln p(z_t|w)$. With this setting for l_t the MAP algorithm always chooses the prediction with maximal posterior probability given the available information. Of course, we can still use the MAP algorithm when we do not have i.i.d. samples; in this case l_t will be unrelated to any likelihood, and so "maximum a posteriori" may be a misnomer.

As the MAP algorithm is stated above it is not operational, since we may not know how to perform the required minimization. A striking feature of the MAP algorithm is that, despite the complicated machinery required to prove its theoretical properties, it often has a simple and efficient implementation. In fact, as we will see below, many well-known inference algorithms are MAP algorithms.

One example of a specific implementation of the MAP algorithm is shown in Figure 1. In this example, the learner is trying to minimize the sum of squared distances between its predictions w_t and a sequence of training examples $z_1 \ldots z_4$. For this problem the MAP algorithm will always predict w_t equal to the mean of all examples from trials $0 \ldots t-1$. (By convention we set $z_0 = 0$.) As shown in the figure, the best possible constant prediction is $u = 4$, since that is the mean of $z_0 \ldots z_4$. The total loss of $u = 4$ is 34, so the regret of the MAP algorithm is the difference between the loss $\sum_{t=1}^{4} l_t(w_t)$ and 34.

The rest of the paper is organized as follows. In Section 2 we will review some basic facts about convex analysis that we will need later on. In section 3 we will outline our main results and the strategy that we will use to prove them. In the remaining sections we will prove specific results.

2 Convex duality

For the proofs below we will need some definitions and basic results about convex functions. A convex function is any function f from a vector space $\mathcal{X}$ to $\mathbb{R} \cup \{+\infty, -\infty\}$ which satisfies

$$\lambda f(x) + (1-\lambda) f(y) \geq f(\lambda x + (1-\lambda) y)$$

for all $x, y \in \mathcal{X}$ and $\lambda \in [0,1]$. A strictly convex function is one for which we can replace $\geq$ by $>$ in the above inequality. A proper convex function is one which is always greater than $-\infty$ and not uniformly $+\infty$. The domain of f, $\mathrm{dom}\, f$, is the set of points where f is finite. Convex functions are continuous on $\mathrm{int}\,\mathrm{dom}\, f$, and differentiable on $\mathrm{int}\,\mathrm{dom}\, f$ except

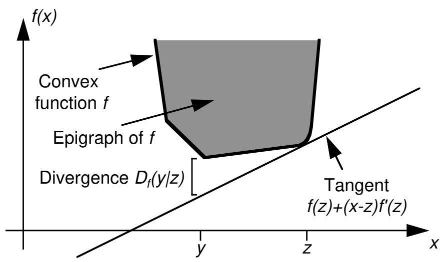

Figure 2: Definitions for convex functions.

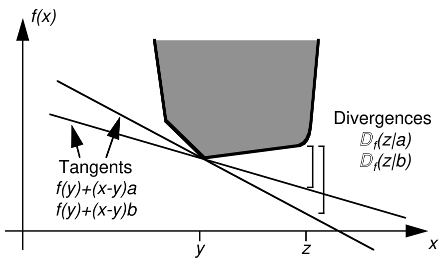

Figure 3: Generalized Bregman divergences.

for a set of measure zero. (The notation $\operatorname{int} C$ refers to the interior of a set C, that is, the points of C which can be surrounded by an open set contained within C.)

Some special cases of convex functions are the linear functions, $f(x) = a \cdot x + b$ for a vector a and scalar b, and the indicator functions

$$\delta(x|C) = \begin{cases} 0 & x \in C \\ \infty & x \notin C \end{cases}$$

for a convex set C. (We will sometimes write a predicate instead of a set C, as in $\delta(x|\sum x_i = 1)$. There should be no danger of confusion.)

A convex function f is closed if its epigraph $\{(x,y)|y \geq f(x)\}$ is closed. The closure of f, $\operatorname{cl} f$, is the function whose epigraph is the closure of f's epigraph. For proper convex functions, closedness is the same as lower semicontinuity.

The convex hull of a function f, $\operatorname{conv} f$, is the pointwise supremum of all of the convex functions which are everywhere less than f. In other words, $\operatorname{conv} f$ is the function whose epigraph is the convex hull of f's epigraph. The convex hull always exists and is convex, although it may be the constant function $-\infty$.

The subgradient of a convex function at some point, written $\partial f(x)$, is the set of vectors a such that $f(y) \geq f(x) + (y - x) \cdot a$ for all y. In other words, the subgradient of f at x is the set of slopes of all tangent planes to f at x. We will write $\operatorname{dom} \partial f$ for the set of x such that $\partial f(x)$ is nonempty. We have

$$\operatorname{int} \operatorname{dom} f \subseteq \operatorname{dom} \partial f \subseteq \operatorname{dom} f$$

The subgradient of a smooth convex function f is single-valued on $\operatorname{int} \operatorname{dom} f$, and $\partial f(x) = \{f'(x)\}$ where $f'(x)$ stands for the usual derivative $\frac{df}{dx}$. By a slight abuse of notation we will write f' even when the subgradient is not single-valued; in this case f' will mean any (fixed) function such that $f'(x) \in \partial f(x)$. The rules for working with subgradients are similar to the rules for working with derivatives; in particular, $\partial(\lambda f)(x) = \lambda \partial f(x)$ and $\partial(f+g)(x) \supseteq \partial f(x) + \partial g(x)$. We may replace containment by equality in the latter formula under mild conditions, for example if $\operatorname{relint} \operatorname{dom} f$ and $\operatorname{relint} \operatorname{dom} g$ have a point in common.

For every function f we can define a new function f^*, called the dual of f, by the formula

$$f^*(a) = \sup_x (a \cdot x - f(x))$$

The notation $\sup$ denotes the supremum or least upper bound of an expression. The dual tells us how the optimal value of a maximization problem changes if we add a linear function to the objective. The dual is always closed and convex, and $f^{**} = \operatorname{cl} \operatorname{conv} f$. If $f \geq g$ pointwise then $f^* \leq g^*$.

For example, the dual of $\exp(x)$ is $x \ln x - x$. The dual of $-\ln x$ is $-1 - \ln(-x)$. The quadratic function $x^2/2$ is self-dual. The dual of $|x|$ is $\delta(x|[-1,1])$.

The dual of $kf(x)$ is $kf^*(\frac{x}{k})$. The dual of a linear function $a \cdot x + b$ is $\delta(x|\{a\}) - b$. The dual of $f + g$ is $f^* \Box g^*$, where the infimal convolution $u \Box v$ is defined as

$$(u \Box v)(x) = \inf_y (u(x - y) + v(y))$$

A special case is that, if $g = a \cdot x + b$, then $(f + g)^*(x) = f^*(x - a) - b$. Another special case happens when we can partition $\mathcal{X}$ into two subspaces $\mathcal{X}_f$ and $\mathcal{X}_g$ so that $f(x)$ depends only on the component of x in $\mathcal{X}_f$ and $g(x)$ depends only on the component of x in $\mathcal{X}_g$. For example, if we write $f(x, y) = g(x) + h(y)$, then $f^*(x, y) = g^*(x) + h^*(y)$; so the dual of $|x| + |y|$ is $\delta(x|[-1,1]) + \delta(y|[-1,1])$.

The subgradients of f and f^* are (almost) inverses of each other. If f is strictly convex, then $(f^*)'(f'(x)) = x$ for all x where f' is defined. More generally, for any closed convex function f, $a \in \partial f(x)$ is equivalent to $x \in \partial f^*(a)$.

Let f be closed and convex. From the subgradient inequality, we know that

$$D_f(x|y) \stackrel{\text{def}}{=} f(x) - f(y) - (x - y) \cdot f'(y) \geq 0$$

whenever $f'(y)$ is defined. The function D_f is called a Bregman divergence. Some examples of Bregman divergences include squared Euclidean distance (which is $D_{x \cdot x}$) and information divergence (which is $D_{\sum x_i \ln x_i}$).

Bregman divergences can be symmetric like squared Euclidean distance, or asymmetric like information divergence. If f is strictly convex, then $D_f(x|y) = 0$ is equivalent to $x = y$. If g is linear, then $D_{f+g} = D_f$.

The Bregman distances given by f and f^* are strongly related: if f is strictly convex, then

$$D_f(x|y) = D_{f^*}(f'(y)|f'(x))$$

If f is not strictly convex, this equality may not hold: if x is at a corner or flat spot of f, then x and $f'(x)$ do not uniquely specify each other.

Figure 3 illustrates this difficulty: if a point $(y, f(y))$ is at a corner of f, then there are infinitely many possible tangent planes to f at y. So, there are infinitely many possible Bregman divergences all represented by $D_f(z|y)$.

One solution is to pick a divergence arbitrarily and fix D_f to mean just that divergence. This solution is the one we have been using implicitly so far, since we have defined $f'(y)$ to be an arbitrary but fixed element of $\partial f(y)$. A better solution is to generalize the definition of Bregman divergence.

We can motivate our generalization by noticing that, while a point y does not define a unique tangent plane to f, a slope a does. There is always at most one plane with slope a tangent to f, and if it exists it is given by the equation

$$f((f^*)'(a)) + (x - (f^*)'(a))a$$

There is the same ambiguity in computing $(f^*)'$ that there was in computing f', but it doesn't matter: if ∂f^* is multivalued, then each value refers to a different point along a linear segment of f, and the tangent plane at any of these points is the same.

So, we define the generalized Bregman divergence, which measures the dissimilarity between a point x and a slope a, to be

$$\mathbb{D}_f(x|a) \stackrel{\text{def}}{=} f(x) + f^*(a) - x \cdot a$$

This definition is a generalization of the original Bregman divergence since, if $a = f'(y)$, then $D_f(x|y) = \mathbb{D}_f(x|a)$. All of the properties of Bregman divergences given above carry over straightforwardly to $\mathbb{D}_f$.

Generalized Bregman divergences satisfy a simple symmetry property: our assumption that f is closed implies that

$$\mathbb{D}_f(x|a) = \mathbb{D}_{f^*}(a|x)$$

Another advantage of the new definition is that $\mathbb{D}_f(x|a)$ is defined for any x and a (although it may be infinite) and convex separately in x and in a (although it may not be convex jointly in x and a). By contrast, $D_f(x|y)$ is undefined if $\partial f(y)$ is empty, and it may not be convex in y.

A function is called positively homogeneous if $f(\lambda x) = \lambda f(x)$ for all $\lambda \geq 0$. A nonnegative, positively homogeneous, closed, convex function is called a gauge. Gauges are a generalization of norms: a norm is a gauge that is symmetric ($f(x) = f(-x)$) and strictly positive except at the origin ($f(x) = 0 \Leftrightarrow x = 0$). The dual of a gauge is an indicator function for a convex set containing the origin, and vice versa.

Two gauges g and g° are called polar to each other if

$$g^\circ(y) = \inf\{\lambda \geq 0 | (\forall x)\; x \cdot y \leq \lambda g(x)\}$$

For example, the L_p norm on $\mathbb{R}^n$ is defined to be

$$||x||_p \stackrel{\text{def}}{=} \left(\sum_{i=1}^{n} x_i^p\right)^{\frac{1}{p}}$$

and $||\cdot||_p$ and $||\cdot||_q$ are polar to each other when $\frac{1}{p} + \frac{1}{q} = 1$. Polar gauges satisfy a generalization of Hölder's inequality:

$$x \cdot y \leq g(x) g^\circ(y)$$

for all x, y, with equality iff $\lambda y \in \partial g(x)$ for some $\lambda \geq 0$. Polarity between gauges is related to duality between convex functions: if $f(x) = \frac{1}{2}g(x)^2$, then $f^*(x) = \frac{1}{2}g^\circ(x)^2$.

For more background on convex duality, see [Roc70] or [OR70].

3 Proof strategy

Our main result is a bound on the total regret of the MAP algorithm. It is stated below as Theorem 1, and an important specialization is given as Theorem 2. There are three basic steps in its proof and application.

Our proof is by an amortized analysis [CLR90]. So, the first step is to define a potential function for the MAP algorithm. This potential function will decrease on trials where the algorithm suffers a large regret, and increase on trials where it suffers a small or negative regret. That way, our analysis will be able to handle trials with large regret by averaging them out against other trials with smaller regret. This kind of amortized analysis is a generalization of an idea which was introduced in [LW92] and also used in many other regret-bound proofs.

The second step is to sum the regret over all trials. In order to perform this step, we will introduce some constants that, roughly speaking, summarize the amount of information available to the algorithm at the beginning of each trial. These constants depend on the type of loss function we are interested in, so we will leave them unspecified.

The third and final step is to calculate the values of the constants for the specific algorithms we wish to analyze. We will leave this step for subsequent sections.

3.1 Existence

Before we prove any regret bounds, we will look at when the MAP algorithm is well-defined, that is, when the minimum of $L_t = \sum_{i=0}^{t-1} l_i$ is guaranteed to be attained. While it is difficult to derive necessary and sufficient conditions for attainment of the minimum, there are some sufficient conditions which are easy to check. Throughout this section (and the rest of the paper) we will assume that each l_t is closed and convex. Because it will avoid extra notation, we will adopt the convention that any prediction is legal if L_t is the constant function $+\infty$.

The simplest condition to check is whether $\operatorname{dom} l_0^*$ is all of $\mathcal{W}$, since this condition does not depend on l_t for $t \geq 1$. Often this condition is the only one we can check. Examples of functions that satisfy this condition are $l_0(w) = w^2$ and $l_0(w) = w \ln w$. An example of a function that does not satisfy this condition is $l_0(w) = |w|$. Loosely speaking, this condition captures functions such that the norm of $l_0'(w)$ keeps increasing without bound as w approaches the border of $\operatorname{dom} l_0$.

Another simple condition to check is whether l_t attains its minimum for each t. Examples of this kind of function include $l_t(w) = (w - z)^2$ and $l_t(p) = D_{KL}(p|q)$. Linear functions (such as the loss functions used in generalized gradient descent, described below) do not usually satisfy this condition.

If l_t is linear for $t \geq 1$, say $l_t(w) = w \cdot x_t$, then L_t will attain its minimum exactly when

$$X_t \stackrel{\text{def}}{=} \sum_{i=1}^{t-1} x_t \in -\operatorname{dom} \partial l_0^*$$

since this condition is true iff there is some w so that

$$0 \quad \in \quad \partial L_t(w)$$

$$-X_t \in \partial l_0(w)$$
$$w \in \partial l_0^*(-X_t)$$

We can combine and generalize these conditions into the following lemma.

Lemma 1 *Suppose that the functions $l_0, l_1, \ldots$ are convex and closed. Let $m_1, m_2, \ldots$ be closed convex functions, each of which attains its minimum, such that $l_t - m_t$ is convex. Suppose there is a point ω_t for each t so that*

$$-\sum_{i=1}^{t-1}(l_i - m_i)'(\omega_t) \in \operatorname{dom} \partial l_0^*$$

Then the MAP algorithm applied to the losses $l_0, l_1, \ldots$ produces a legal prediction at each trial t.

PROOF: Fix a trial t and write $x_i = (l_i - m_i)'(\omega_t)$ for $1 \le i < t$. The function $M(w) = l_0(w) + w \cdot \sum_{i=1}^{t-1} x_i$ achieves its minimum, since it is closed and convex and since the condition $0 \in \partial l_0(w) + \sum_{i=1}^{t-1} x_i$ is equivalent to $w \in \partial l_0^*(-\sum_{i=1}^{t-1} x_i)$.

The functions $l_i(w) - m_i(w) - w \cdot x_i$ also achieve their minima, since $0 \in \partial(l_i - m_i)(\omega_t) - x_i$. But L_t is the sum of $M(w)$, $l_i(w) - m_i(w) - w \cdot x_i$, and $m_i(w)$ for $i = 1 \ldots t-1$. So, since the sum of closed convex minimum-achieving functions is also a closed convex minimum-achieving function, L_t achieves its minimum. □

3.2 One-step regret

Our potential function will be a generalized Bregman divergence involving the comparison vector u, the loss functions l_t, and the MAP algorithm's current prediction w_t. The reason we use a divergence involving u and w_t is that we want to prove that, on trials where the MAP algorithm suffers a large regret compared to u, it will move its next prediction closer to u. That way, we can conclude that if it sees the same loss function again, it will incur a smaller regret.

Let $L_t = \sum_{i=0}^{t-1} l_i$, so that the w_t chosen by the MAP algorithm will be $\arg\min_w L_t(w)$. We define our potential function to be $\mathbb{D}_{L_t}(u|0)$. The potential change on each time step is given by the following lemma.

Lemma 2 *On trial t, the change in potential is*

$$\mathbb{D}_{L_{t+1}}(u|0) - \mathbb{D}_{L_t}(u|0) = l_t(u) - L_t^*(0) + L_{t+1}^*(0)$$

PROOF: The potential on step t is

$$\mathbb{D}_{L_t}(u|0) = L_t(u) + L_t^*(0)$$

So, the difference in potential between trials t and $t+1$ is

$$(L_{t+1} - L_t)(u) - L_t^*(0) + L_{t+1}^*(0)$$

But $L_{t+1} - L_t$ is just l_t, so the result follows. □

The function L_t^* is important, since it encodes both the best possible loss so far and the MAP algorithm's next prediction: Theorem 27.1 in [Roc70] states

$$L_t^*(0) = -L_t(w_t)$$
$$w_t \in \partial L_t^*(0)$$

Most of the work in applying Theorem 1 to specific problems will come in analyzing L_t^*. For example, in the Weighted Majority proof below, $L_t^*(0)$ will be the log of the sum of the unnormalized weights, and the main part of the proof will be to connect the change in this quantity to the algorithm's loss.

3.3 Amortized analysis

In order to complete the proof of our bound, we need to relate the quantity $L_t^*(0) - L_{t+1}^*(0)$ to the loss of the MAP algorithm. Since the relationship depends on the type of loss function we are using, for now we will just assume that there are constants $c_1 \ge c_2 \ge \ldots > 0$ so that

$$c_t(L_t^*(0) - L_{t+1}^*(0)) \ge \tilde{l}_t(w_t) \quad (1)$$

Here $\tilde{l}_t$ is some function related to l_t. Often we will just use $\tilde{l}_t = l_t$, but we will sometimes need the extra generality. The smaller we take c_t, the better our bounds will be.

We can think of $1/c_t$ as a lower bound on how much information is available to the algorithm at the beginning of trial t. The best allowable value of c_t will depend on how convex L_t is when compared to l_t. For example, if every l_t is quadratic with the same second derivative, we will show below that we can take $1/c_t$ proportional to the sample size t.

With the assumption (1), Lemma 2 becomes

$$\mathbb{D}_{L_t}(u|0) - \mathbb{D}_{L_{t+1}}(u|0) \ge \frac{1}{c_t}\tilde{l}_t(w_t) - l_t(u)$$

or

$$\tilde{l}_t(w_t) \le c_t l_t(u) + c_t \mathbb{D}_{L_t}(u|0) - c_t \mathbb{D}_{L_{t+1}}(u|0) \quad (2)$$

If we now apply lemma 2 to trial $t+1$, we get

$$\tilde{l}_{t+1}(w_{t+1}) \le c_{t+1} l_{t+1}(u) + c_{t+1}\mathbb{D}_{L_{t+1}}(u|0) - c_{t+1}\mathbb{D}_{L_{t+2}}(u|0) \quad (3)$$

Notice that $\mathbb{D}_{L_{t+1}}(u|0)$ appears in both Equation 2 and 3, once with coefficient $-c_t$ and once with coefficient c_{t+1}. Since $c_{t+1} \le c_t$ and since Bregman divergences are nonnegative, the two terms together are less than or equal to zero; so, we can drop them from our bound on total regret.

But now we have proven

Theorem 1 *Let $l_0, l_1, \ldots$ satisfy the assumptions of Lemma 1, so that the MAP algorithm produces a prediction w_t at trial t. Define $L_t = \sum_{i=1}^{t-1} l_t$. Let the constants c_t and the functions $\tilde{l}_t$ be such that $c_t(L_t^*(0) - L_{t+1}^*(0)) \ge \tilde{l}_t(w_t)$. Then the for all u total regret of the MAP algorithm is bounded by*

$$\sum_{t=1}^{T} \tilde{l}_t(w_t) \le \sum_{t=1}^{T} c_t l_t(u) + c_1 \mathbb{D}_{l_0}(u|0)$$

PROOF: Sum lemma 2 over all trials, then cancel terms as described above. Finally, ignore the term containing the ending potential $\mathbb{D}_{L_{T+1}}(u|0)$. □

3.4 Specific bounds

All that remains is to evaluate the constants c_t for specific types of loss functions. In the following sections we will do just that. The next section analyzes the Weighted Majority algorithm. Theorem 2, proved in Section 6, covers cases in which the one-step losses can be represented as Bregman divergences. In particular, Sections 5 and 7 cover generalized gradient descent algorithms, and Sections 8 and 9 cover generalized linear regression algorithms including linear regression and exponentiated gradient.

Another interesting problem is inference of the natural parameter in an exponential family. We will not have space to cover this problem here, but see [Gor99] for more detail.

4 Weighted Majority

One of the simplest MAP algorithms is Weighted Majority, described in [LW92]. For a more detailed analysis of WM in our framework, see [Gor99]; here we will only give a brief review.

The legal predictions for WM are the vectors in the unit simplex $P = \{w | w \geq 0 \wedge \sum w = 1\}$. The ith component of w_t is interpreted as the weight given to the ith expert on step t. The loss functions for $t \geq 1$ are $l_t(w) = x_t \cdot w$, where $x_{t,i}$ is the prediction error of the ith expert at step t.

Since we already know l_t for $t \geq 1$, the only decision we need to make to interpret WM as a MAP algorithm is to choose the initial loss function l_0. To derive WM, we will pick

$$l_0 = \frac{1}{-\ln \beta} H(w)$$

where $\beta \in [0, 1)$ is a parameter of the algorithm, and H is defined as

$$H(w) \stackrel{\text{def}}{=} \delta(w|P) + \sum_i w_i \ln w_i$$

where $\delta(\cdot|P)$ is the function which is ∞ outside of P and 0 inside of P. It is easy to verify that

$$H^*(x) = \ln \sum_i \exp(x_i)$$

$$\frac{d}{dx_i} H^*(x) = \exp(x_i) / \sum_j \exp(x_j)$$

This choice of l_0 means that our prediction on step t will be $w_t = (H^*)'(X_t \ln \beta)$, or

$$w_{t,i} = \beta^{X_{t,i}} / \sum_j \beta^{X_{t,j}}$$

where $X_t = \sum_{i=1}^{t-1} x_t$. This update is identical to the one given in [LW92]. In fact, the analysis of that paper shows that we can take $c_t = -\ln \beta / (1 - \beta)$ for all t, and applying Theorem 1 with this choice of c_t yields their Corollary 6.1. A similar application of Theorem 1 bounds the regret of a WM-like algorithm which uses the log loss $-\ln(w \cdot x_t)$ instead of a linear loss.

The fact that l_0 is infinite outside of P ensures that our prediction w_t is always in P. A look at the update equations shows how WM translates this requirement into practice: first it computes v_t, which is what we would have predicted if l_0 did not have the indicator term $\delta(w|P)$, and then it scales each element of v_t proportionally to find the perspective projection of v_t onto P. Since scaling v_t is equivalent to adding a constant to each element of X_t, the projection step is equivalent to finding a constant k so that $\sum_i \beta^{X_{t,i}+k} = 1$. This behavior does not depend on the entropy term in l_0. In fact, if we replace l_0 with any function of the form $\delta(w|P) + \sum_i l(w_i)$, our prediction will be $w_{t,i} = (l^*)'(X_{t,i} + k)$, with k chosen to ensure that $w_t \in P$.

5 Generalized gradient descent

In the previous section we analyzed a simple MAP algorithm in which all of the loss functions except the prior were linear. Because of the linearity of the loss functions, it was easy to compute the prediction w_t on each time step: the update rule for WM is of the form $w_t = f(-\eta X_t)$, where X_t is the sum of the gradients of the previous loss functions and f is a function that we can compute efficiently. The variant of WM for log loss has a similar update rule; in fact, we can derive it by replacing the log loss function by a linear approximation.

We would like to be able to play the same trick for an arbitrary convex loss function l_t. That is, we would like to replace l_t by a linear function m_t, then apply the MAP algorithm to the functions m_t instead of l_t so that it will run more efficiently. Of course, the predictions will be different if we use m_t in place of l_t, and so the regret may be larger. But, we may have to do significantly less work per trial, and we will still be able to bound the regret.

The key inequalities which will allow us to replace l_t by m_t are

$$\begin{aligned} l_t(w_t) &\leq m_t(w_t) \\ l_t(u) &\geq m_t(u) \qquad \forall u \in \mathcal{U} \end{aligned} \tag{4}$$

If $\mathcal{U} = \mathcal{W}$ then these inequalities force m_t to be tangent to l_t at w_t; if $\mathcal{U} \subset \mathcal{W}$ then m_t may be a secant to l_t that passes above $(w_t, l_t(w_t))$. Subtracting the second inequality from the first gives $l_t(w_t) - l_t(u) \leq m_t(w_t) - m_t(u)$ for all $u \in \mathcal{U}$, so that when we apply Theorem 1 to bound the difference $m_t(w_t) - m_t(u)$ we also get a bound on the regret $l_t(w_t) - l_t(u)$.

If we set m_t to be a tangent to l_t at w_t, $m_t(w) = l_t(w_t) + (w - w_t) \cdot l_t'(w_t)$, and then feed the sequence of loss functions $l_0, m_1, m_2, \ldots$ to the MAP algorithm, the result is an algorithm called generalized gradient descent or GGD. It is "generalized" because, when l_0 is quadratic, the update rule reduces to ordinary gradient descent. We can write the GGD update rule as follows:

GGD ALGORITHM: Predict any

$$w_t \in \arg\min_w \left[l_0(w) + w \cdot \sum_{i=1}^{t-1} l_i'(w_i) \right]$$

The GGD algorithm is often written in an additive form that looks different from its statement above. If we write $X_t = \sum_{i=1}^{t-1} l_i'(w_i)$ then the additive form of the GGD prediction rule is $w_t = f(-\eta X_t)$. Here η is a learning rate and f is a function from $\mathbb{R}^n$ to $\mathbb{R}^n$ satisfying appropriate conditions. For example, choosing f to be the identity yields ordinary gradient descent. The advantage of this form of the prediction rule comes from the fact it may be difficult to compute l_0 from f, while it is often easier to compute f from l_0; so, if we are given f, we can use the additive form of the GGD rule without needing to compute l_0.

We can prove that the two forms of the GGD algorithm are equivalent: if $\eta = 1$, then we can set $f = (l_0^*)'$. For different learning rates we can just multiply l_0 by a constant, since $(\frac{1}{\eta} l_0)^*(x) = \frac{1}{\eta} l_0^*(\eta x)$ and so $((\frac{1}{\eta} l_0)^*)'(x) = (l_0^*)'(\eta x)$.

The function $f(x)$ (or equivalently $(l_0^*)'(\frac{x}{\eta})$) is called a link function. Figure 4 shows some useful link functions and their corresponding loss functions. The one-dimensional link functions in Figure 4 can easily be generalized to multiple dimensions by applying them separately to each coordinate.

Name	Link $(l_0^*)'$	Loss l_0
Identity	a	$\frac{1}{2}w^2$
Logistic	$\frac{1}{1+\exp(-a)}$	$w \ln w + (1-w)\ln(1-w)$
Inverse logistic	$\ln \frac{a}{1-a}$	$\ln(1+\exp(w))$
Exponential	$\exp a$	$w \ln w - w$
Logarithmic	$\ln a$	$\exp w$
Normalized exponential	$\frac{\exp a_i}{\sum_i \exp a_i}$	$\sum_i w_i \ln w_i - 1 + \delta(w \mid \sum_i w_i = 1)$

Figure 4: Some examples of link functions.

Some examples of GGD algorithms are ordinary gradient descent, the perceptron learning rule, and the Exponentiated Gradient algorithm of [KW97]. We will examine some of these algorithms in more detail below. But first, we will prove regret bounds for a class of algorithms that includes GGD.

6 General regret bounds

6.1 Preliminaries

In many common MAP algorithms, each individual loss function can be written as a Bregman divergence. For example, in linear regression, the loss functions are of the form $(y_t - w_t \cdot x_t)^2$, which we may think of as a scaled Euclidean distance between w_t and any of the infinitely many perfect predictions w satisfying $y_t = w \cdot x_t$. (The scaling is such that all directions perpendicular to x_t have weight zero.) For a more general example, in GGD, if we adopt the convention that $\inf_w l_t(w) = 0$, then the loss $m_t(w_t)$ is $l_t(w_t) = \mathbb{D}_{l_t}(w_t|0)$. In this section we will derive regret bounds that hold when the loss functions are divergences.

To that end, assume that we are running the MAP algorithm with loss functions $l_0, m_1, m_2, \ldots$, and that $m_t(w_t) = \mathbb{D}_{l_t}(w_t|a_t)$. Also assume $m_t(w) \leq \mathbb{D}_{l_t}(w|a_t)$ for all w. (These inequalities are a tangency condition similar to (4).) Write $L_t = l_0 + \sum_{i=1}^{t-1} m_t$. This notation is similar to the notation from the section on GGD, but in this section we are not assuming that the functions m_t are linear. In particular, we may take $m_t = l_t$.

In order to bound the loss of the MAP algorithm, we have to make sure that the prior loss L_t before each trial t is sufficiently convex. To see why, consider what would happen if we took $l_0 = L_1$ to be $\frac{1}{\eta}\delta(w|[0,1])$. With this choice of prior loss, our predicted w can change discontinuously from 0 to 1 even when the one-step loss has only a small gradient. So, for example, if we see $m_1 = w/2$ and then $m_2, m_3, \ldots = (1-w), w, (1-w), w, \ldots$, our predictions will alternate between 0 and 1 no matter how small η is. In fact, we will always choose the worst possible w, and so our loss will be twice that of the comparison vector $u = .5$.

We also have to make sure that the one-step divergence functions l_t for $t \geq 1$ are not too convex. If they are, we can cause the MAP algorithm to suffer an arbitrarily large regret per trial: the more convex l_t is as compared to L_t, the more of an advantage it is to pick the comparison vector after knowing m_t. For example, if $l_0(w) = w^2$ (so that $w_1 = 0$), then the loss function $m_1(w) = 10^6(w-1)^2$ will cause the MAP algorithm a loss of 10^6, while the optimal comparison vector $\frac{1}{1+10^{-6}}$ will suffer a loss of approximately 10^{-6} even though its l_0-divergence from w_1 is less than 1.

So, to ensure that L_t is sufficiently convex, we will pick a gauge g and constants $\eta_g \in (0,1)$ and require that

$$\eta_t \mathbb{D}_{L_t}(v|a) \geq \frac{1}{2}(g(v-w))^2$$

for all v and w and $a \in \partial L_t(w)$. And, to ensure that l_t is not too convex, we will require that

$$\mathbb{D}_{l_t^*}(a_t|w) \geq \frac{1}{2}(g^\circ(a_t - a))^2$$

for all w and $a \in \partial l_t(w)$.

A consequence of the first assumption is that

$$L_t(w) - L_t(w_t) \geq \frac{1}{2}(g(w-w_t))^2$$

since the LHS is equal to $\mathbb{D}_{L_t}(w|0)$ and $0 \in \partial L_t(w_t)$. A consequence of the second assumption is that

$$m_t(w_t) \geq \frac{1}{2}(g^\circ(-m_t'(w_t)))^2$$

as long as $\partial m_t(w_t)$ is nonempty, since

$$\begin{aligned} m_t(w_t) &= l_t(w_t) \\ &= \mathbb{D}_{l_t}(w_t|a_t) \\ &\geq \frac{1}{2}(g^\circ(a_t - a))^2 \end{aligned}$$

for any $a \in \partial l_t(w_t)$, and since $\partial l_t(w_t) - a_t \supseteq \partial m_t(w_t)$.

Scaling the gauge g will scale η_t inversely. So, in order to make the constant η_t as small as possible in the first assumption, it is important to take g to be as shallow as possible while still satisfying the second assumption.

6.2 Examples

To interpret our assumptions, it will help to compute the best gauge g and learning rate η for some examples. First suppose that L_t and l_t are both quadratic, say $L_t(w) = \frac{k}{2}w^{\mathrm{T}}Mw$ and $l_t(w) = \frac{1}{2}w^{\mathrm{T}}Mw$ for some symmetric positive definite matrix M. (This choice of l_t means that $m_t(w_t) = \frac{1}{2}(w_t - z_t)^{\mathrm{T}}M(w_t - z_t)$, where $z_t = M^{-1}a_t$.) Then we can choose $\eta_t = \frac{1}{k}$ and $g(w) = \sqrt{w^{\mathrm{T}}Mw}$, since

$$\frac{1}{k}\mathbb{D}_{L_t}(v|a) = \frac{1}{2}(v-w)^{\mathrm{T}}M(v-w) = \frac{1}{2}g(v-w)^2$$

where $w = kM^{-1}a$, and

$$\mathbb{D}_{l_t^*}(a_t|w) = \frac{1}{2}a_t^{\mathrm{T}}M^{-1}a_t + \frac{1}{2}w^{\mathrm{T}}Mw - a_t \cdot w = \frac{1}{2}g^\circ(a_t - a)^2$$

where $a = Mw$.

Or suppose that l_t is quadratic but L_t is proportional to the entropy function. In particular, let

$$l_t(w) = \frac{1}{2}||w||_2^2$$

$$L_t(w) = kH(w)$$

$$H(w) \stackrel{\text{def}}{=} \sum_{i=1}^{n} w_i \ln w_i + k \ln n$$

It is well known that $D_H(v|w) \geq 2||v - w||_2^2$ for any v, w. So, $\frac{1}{4k}D_{L_t}(v|w) \geq \frac{1}{2}||v - w||_2^2$. And just as in the previous example $\mathbb{D}_{l_t^*}(a_t|w) \geq \frac{1}{2}||a_t - w||_2^2$. So, we can choose g to be Euclidean distance and let $\eta_t = \frac{1}{4k}$.

These two examples show that g and η together provide a global analog to the Fisher information matrix. When the Fisher information $L_t''(w)$ is constant over all possible parameter values w, as it is in the first example, the local and global information measures are the same. On the other hand, when the Fisher information varies, as it does in the second example, the global measure may be much more conservative. This conservatism is necessary: in the average case we can count on having our estimates stay near the optimal value, while in the worst case our opponent can cause our estimates to wander into a region with lower information.

Finally, suppose that $l_t(w) = \frac{1}{2}(y_t - w \cdot x_t)^2$, and let $a_t = 0$ so that $m_t(w_t) = l_t(w_t)$. This choice of loss function is appropriate for linear regression problems. It depends on w only through $w \cdot x_t$, so any change in w perpendicular to x_t leaves l_t constant. That means that we can represent l_t^* as the sum of two components, one of which depends only on $w \cdot x_t$ and the other of which depends only on $w \backslash x_t \stackrel{\text{def}}{=} w - \frac{w \cdot x_t}{x_t \cdot x_t} x_t$. A little algebra shows

$$l_t^*(x) = \delta(x | x \setminus x_t = 0) + \frac{x \cdot x_t}{x_t \cdot x_t} y + \frac{1}{2}\left(\frac{x \cdot x_t}{x_t \cdot x_t}\right)^2$$

In other words, l_t^* is infinite everywhere except along the line through x_t, and along that line it is quadratic. The quadratic term (the last term in the expression above) is scaled so that it is equal to $\frac{1}{2}$ at x_t and $-x_t$. So, to bound l_t^*, we will need to make some assumption about x_t.

If we suppose that the gauge g is symmetric and scaled so that $g^\circ(x_t) \leq 1$, then it is not hard to see that

$$\mathbb{D}_{l_t^*}(0|w) = l_t(w) \geq \frac{1}{2}(g^\circ(x))^2$$

since the latter expression is also quadratic along the line through x_t and scaled so that it is no larger than $\frac{1}{2}$ at $\pm x_t$. So, for example, if $||x_t||_\infty \leq X$, we can take $g(w)$ to be $X||w||_1$.

Now, since $||w||_1 \leq ||w||_2$, we have $D_H(v|w) \geq 2||v - w||_1^2$. So, if $L_t = kH$, we can take $\eta_t = \frac{X^2}{4k}$.

6.3 The bound

We will now prove our regret bound.

Theorem 2 *Suppose that the loss functions $l_0, m_1, m_2, \ldots$ satisfy the conditions of Lemma 1, so that the MAP algorithm applied to these loss functions always produces a prediction w_t at each trial. Suppose that for all t, $\partial m_t(w_t)$ is nonempty, $m_t(w_t) = \mathbb{D}_{l_t}(w_t|a_t)$, and $m_t(w) \leq \mathbb{D}_{l_t}(w|a_t)$ for all w. Write $L_t = l_0 + \sum_{i=1}^{t-1} m_t$. Suppose that there exists a gauge g and constants $1 > \eta_1 \geq \eta_2 \geq \ldots > 0$ so that for all t we have*

$$\eta_t \mathbb{D}_{L_t}(v|a) \geq \frac{1}{2}(g(v - w))^2$$

for all v and w and $a \in \partial L_t(w)$ and

$$\mathbb{D}_{l_t^*}(a_t|w) \geq \frac{1}{2}(g^\circ(a_t - a))^2$$

for all w and $a \in \partial l_t(w)$. Then the loss of the MAP algorithm is bounded by

$$\sum_{t=1}^{T} m_t(w_t) \leq \sum_{t=1}^{T} \frac{1}{1 - \eta_t} m_t(u) + \frac{1}{1 - \eta_1}\mathbb{D}_{l_0}(u|0)$$

PROOF: We have

$$\begin{aligned}
L_t(w) &\geq \frac{1}{2\eta_t}(g(w - w_t))^2 + L_t(w_t) \\
m_t(w) &\geq m_t(w_t) + (w - w_t) \cdot m_t'(w_t) \\
L_{t+1}(w) &\geq \frac{1}{2\eta_t}(g(w - w_t))^2 + L_t(w_t) + \\
&\quad m_t(w_t) + (w - w_t) \cdot m_t'(w_t) \\
L_{t+1}^*(x) &\leq \frac{1}{2\eta_t}(g^\circ(\eta_t(x - m_t'(w_t))))^2 + \\
&\quad x \cdot w_t - L_t(w_t) - m_t(w_t) \\
L_{t+1}^*(0) - L_t^*(0) &\leq \frac{1}{2\eta_t}(g^\circ(-\eta_t m_t'(w_t)))^2 - m_t(w_t) \\
&\leq (\eta_t - 1)m_t(w_t)
\end{aligned}$$

The fourth line above is true because the dual of $af(w - c) + b \cdot (w - c)$ is $af^*((x - b)/a) + x \cdot c$. The fifth is true because $L_t^*(0) = -L_t(w_t)$. The last line is true because $g^\circ(-\eta_t m_t'(w_t))^2 = \eta_t^2 g^\circ(-m_t'(w_t))^2$.

The desired result now follows by applying Theorem 1 to the loss functions $l_0, m_1, m_2, \ldots$, taking $\tilde{l}_t = m_t$ and $c_t = \frac{1}{1-\eta_t}$. □

The way it is stated, this theorem bounds the loss in terms of the functions m_t; it is just as easy to give a bound in terms of l_t by substituting $m_t(w_t) = \mathbb{D}_{l_t}(w_t|a_t)$ and $m_t(u) \leq \mathbb{D}_{l_t}(u|a_t)$.

7 GGD examples

Perhaps the simplest use of GGD is to approximate the mean of a population of vectors by looking at a sample $z_1, z_2, \ldots$. This application of GGD corresponds to the prior loss $l_0(w) = k||w||^2$ and the one-step losses $l_t(w) = ||w - z_t||^2$. With these loss functions, GGD will predict $w_{t+1} = w_t + \frac{1}{k}(z_t -$

w_t). We saw above that we can take g to be Euclidean distance and $\eta = \frac{1}{k}$; so Theorem 2 tells us that our loss is bounded by

$$\sum_{t=1}^{T} ||w_t - z_t||^2 \leq \frac{1}{1-\frac{1}{k}} \sum_{t=1}^{T} ||z_t - \bar{z}||^2 + \frac{k}{1-\frac{1}{k}} ||\bar{z}||^2$$

where $\bar{z} = \frac{1}{T+k} \sum_{t=1}^{T} z_t$ is the optimal constant prediction.

The first term on the right-hand side of the above inequality depends on the training examples z_t only through their variance; the second depends on the examples only through their mean. So, the inequality tells us that even if the training examples are chosen by an adversary, as long as they have bounded mean and variance, we can still achieve bounded average regret per trial. More specifically, suppose that as $T \to \infty$ the mean of $z_1 \dots z_T$ approaches μ and the covariance approaches $\sigma^2 I$. Then for large enough T the second term becomes negligible, and our average loss per trial will approach $\frac{\sigma^2}{1-\eta}$. So, our average regret per trial will approach $\frac{\sigma^2 \eta}{1-\eta}$.

By way of comparison, we can compute the asymptotic average case regret per trial for this variant of GGD: suppose that the training examples z_t are independent indentically distributed random variables that follow a normal distribution with mean μ and covariance $\sigma^2 I$. Then the optimal prediction will approach μ for sufficiently large T, and its expected loss on each trial will approach σ^2. On the other hand, by solving the recurrences $Ew_{t+1} = (1-\eta)Ew_t + \eta E z_t$ and $\text{Var}\, w_{t+1} = (1-\eta)^2 \text{Var}\, w_t + \eta^2 \text{Var}\, z_t$, we can see that $Ew_t \to \mu$ and $\text{Var}\, w_t \to \frac{\eta}{2-\eta}\sigma^2 I$. So, the expected loss per trial of the GGD algorithm approaches

$$(Ew_t - \mu)^{\mathrm{T}}(Ew_t - \mu) + E((w_t - \mu)^{\mathrm{T}}(w_t - \mu))$$

$$\to \sigma^2 \left(1 + \frac{\eta}{2-\eta}\right) = \frac{\sigma^2}{1 - \frac{\eta}{2}}$$

and the average regret per trial approaches $\frac{\sigma^2 \eta}{2-\eta}$. That means that as $\eta \to 0$ there is a difference of approximately a factor of two between the worst-case and average-case regret for this algorithm. This gap appears to be necessary: at least for small learning rates, the sequence of z_ts $1, -1, 1, -1, \dots$ forces nearly as much regret as our bound.

For another example, take l_0 to be a multiple of the entropy function on the unit simplex. That is, suppose $l_0 = kH$ with

$$H(w) = \sum_{i=1}^{N} w_i \ln w_i + \ln N + \delta(w | \sum w = 1)$$

The resulting update is

$$w_{t+1,i} = \frac{w_{t,i} \exp(-x_{t,i}/k)}{\sum_{i=1}^{N} w_{t,i} \exp(-x_{t,i}/k)}$$

where $x_t = l_t'(w_t)$. This is the Exponentiated Gradient algorithm of [KW97]. (If the loss functions l_t for $t \geq 1$ are linear, it is also the same as the WMC algorithm.)

If now $l_t(w) = \frac{1}{2}(w - z_t)^2$ for $t \geq 1$, we saw above that we can take $\eta = \frac{1}{4k}$. So Theorem 2 tells us that our loss is bounded by

$$\sum_{t=1}^{T} ||w_t - z_t||^2 \leq \frac{1}{1-\frac{1}{4k}} \sum_{t=1}^{T} ||u - z_t||^2 + \frac{k}{1-\frac{1}{4k}} \mathbb{D}_H(u|0)$$

for any u. This bound is not the same as any bound in [KW97] or [KW96], since those papers consider only regression problems; so, we defer a comparison until Section 9 below.

8 Regression problems

A common type of prediction problem is generalized linear regression [MN83, LW92, KW97], which includes linear regression, logistic regression, other generalized linear models, perceptron learning, and many other problems. In generalized linear regression, on each time step t we must predict a vector of regression coefficients w_t. We are then given an input vector x_t, from which we form a prediction $\hat{y}_t = f(x_t \cdot w_t)$. The monotone function f is called the prediction link function, since it provides a link between the coefficients w_t and the prediction $\hat{y}_t$. Finally, we find out the desired output y_t and receive a loss $l_t(w) = l(\hat{y}_t, y_t)$. Regression problems are a special case of our general prediction problem, since they differ only in that we have specified a particular form for the loss function l_t: for example, the loss functions for linear regression are of the form $(y_t - w \cdot x_t)^2$.

We should not confuse the prediction link function, which is a mapping from $\mathbb{R}$ to $\mathbb{R}$ that connects $w \cdot x_t$ with the prediction $\hat{y}$, with the link function described earlier, which is a function from $\mathcal{W}$ to $\mathcal{W}$. In designing an algorithm, we can choose the two kinds of link functions separately. When there is a danger of confusion, we will call the latter the parameter link function. All of the one-dimensional parameter link functions in Figure 4 are also possible choices for the prediction link function.

8.1 Matching loss functions

In order to apply our theory, we need the one-step losses $l_t(w)$ to be convex. This is a condition on the relationship between the prediction link function f and the loss function $l(\hat{y}, y)$. It turns out that, given a monotone link function, we can always define a matching loss function so that $l_t(w)$ is convex. If f is invertible, we follow [AHW96] and define its matching loss function to be

$$l(\hat{y}, y) = D_{F^*}(y|\hat{y})$$

where F is any convex function with $f = F'$.

If f is not invertible (that is, if F has a linear segment, so that F^* has a corner) then the above definition no longer works. Intuitively, the problem is that our predictions get "stuck" as they cross the corner in F^*: there is a whole range of p with the same $f(p)$ and therefore the same loss, producing an extraneous flat spot in l_t.

We can fix the problem by allowing $l_t(w)$ to depend on $p_t = x_t \cdot w$ directly, rather than just on $f(p_t)$. More specifically, we generalize the definition of [AHW96] and set

$$m(p, y) = \mathbb{D}_{F^*}(y|p) = \mathbb{D}_F(p|y) = F(p) + F^*(y) - y \cdot p$$

With this definition, it is easy to see that $m(p, y)$ is convex as a function of p, so $l_t(w) = m(x_t \cdot w, y_t)$ is convex in w.

Intuitively, what we have done is allow ourselves to specify not just which y will give us zero loss, but also what the derivative of F^* is at that point. When F^* is smooth, there is only one possible choice of derivative for each prediction, so we have not changed anything; but when our prediction is at a corner of F^* we can choose from a range of possible derivatives.

We will use the derivative of the loss function below. It turns out that the prediction error is a derivative of m with respect to p:

$$f(p) - y \in \partial F(p) - y = \partial_p m(p, y)$$

So, a derivative of $l_t(w)$ is $(f(x_t \cdot w) - y_t)x_t$.

8.2 Regret bounds

In order to bound the regret of the MAP algorithm for regression problems, we need to find a gauge g so that $l_t(w) \geq \frac{1}{2}(g^\circ(-l_t'(w)))^2$. We have already done so for the special case of the identity link with squared loss: in section 6.2, we showed that the allowable choices for g are the symmetric gauges such that $g^\circ(x_t) \leq 1$ for all t. (Symmetric gauges are also called seminorms.)

The situation is similar for general link functions and their matching loss functions. In this case, though, we must make one additional assumption: we must bound how quickly the prediction $\hat{y}_t$ changes when we change the raw prediction p_t.

So, we will assume that $\mathbb{D}_F(p|y) \geq \frac{\lambda^2}{2}(y - f(p))^2$ for some $\lambda > 0$. (This is essentially a Lipschitz condition on f.) With this assumption, we can write

$$\begin{aligned} l_t(w) &= \mathbb{D}_F(x_t \cdot w | y_t) \\ &\geq \frac{\lambda^2}{2}(y_t - f(x_t \cdot w))^2 \end{aligned}$$

But we saw above that $l_t'(w) = (f(x_t \cdot w) - y_t)x_t$. So, if g is a symmetric gauge such that $\lambda g^\circ(x_t) \leq 1$, then

$$\begin{aligned} (y_t - f(x_t \cdot w))^2 &\geq \frac{1}{\lambda^2} g^\circ(l_t'(w))^2 \\ l_t(w) &\geq \frac{1}{2} g^\circ(l_t'(w))^2 \end{aligned}$$

But now we have proven

Theorem 3 *Let F be a closed convex function with*

$$\mathbb{D}_F(p|y) \geq \frac{\lambda^2}{2}(y - f(p))^2$$

Suppose that the functions $l_1, l_2, \ldots$ are of the form $l_t(w) = \mathbb{D}_F(y_t | w \cdot x_t)$ for given vectors x_t and scalars y_t. Pick a prior loss l_0 and functions $m_1, m_2, \ldots$, and suppose that $l_0, m_1, m_2, \ldots$ satisfy the conditions of Lemma 1, so that the MAP algorithm applied to these loss functions always produces a prediction w_t at each trial. Suppose that for all t, $\partial m_t(w_t)$ is nonempty, $m_t(w_t) = l_t(w_t)$, and $m_t(w) \leq l_t(w)$ for all w. Write $L_t = l_0 + \sum_{i=1}^{t-1} m_t$. Let the symmetric gauge g be so that $\lambda g^\circ(x_t) \leq 1$ for all t. Finally, let the constants $1 \geq \eta_1 \geq \eta_2 \geq \ldots > 0$ be such that

$$\eta_t \mathbb{D}_{L_t}(v|a) \geq \frac{1}{2}(g(v - w))^2$$

for all v and w and $a \in \partial L_t(w)$. Then

$$\sum_{t=1}^{T} l_t(w_t) \leq \sum_{t=1}^{T} \frac{1}{1 - \eta_t} l_t(u) + \frac{1}{1 - \eta_1} \mathbb{D}_{l_0}(u|0)$$

for all u.

PROOF: Apply Theorem 2 to the functions $l_0, l_1, \ldots$ and $m_1, m_2, \ldots$, with $a_t = 0$, using the gauge g and the learning rates η_t. □

While the size of the input vectors x_t doesn't appear explicitly in this bound, it affects the choice of g and therefore the allowed values of η_t. For example, depending on the size of the input vectors, we might need to set $g(w)$ to either $||w||_1$ or $10||w||_1$. At the cost of introducing an extra parameter, we could have written the theorem to allow us to set $g(w) = ||w||_1$ no matter the scale of the input vectors. For examples of the application of this theorem, see Section 9.

8.3 Multidimensional outputs

So far in our regression problems we have assumed that the target y_t is one-dimensional. Our proofs work equally well, though, if y_t is selected from an arbitrary vector space $\mathcal{Y}$. In that case, the parameter matrix w_t will be a linear mapping that takes x_t to $p_t \in \mathcal{Y}$. The prediction link function f will be the derivative of some convex function F on $\mathcal{Y}$, and the matching loss will be $\mathbb{D}_F(p_t|y_t)$, so the derivative of $l_t(w)$ will be $(f(wx_t) - y_t)x_t^{\mathrm{T}}$.

The only part of the proof that requires some modification is the definition of the gauge g. Since w is a matrix and x_t and y_t are vectors of possibly different lengths, we need different gauges to measure the size of each one. (Previously we had used g for w, g° for x_t, and $|\cdot|$ for y_t.) So, we will assume that we have symmetric gauges r and s so that $r^\circ(x_t) \leq 1$ and $\mathbb{D}_F(p|y) \geq \frac{1}{2}s(y - f(p))^2$. Then we will define g by the relation

$$g(w) = \sup_{\{u | r(u) \leq 1\}} s(wu)$$

This g is called the matrix gauge for r and s. Since r and s are symmetric, so is g. Also, if x and y are vectors with $r^\circ(x) = 1$ and $s(y) = 1$, then $g(yx^{\mathrm{T}}) = 1$, since $s(yx^{\mathrm{T}}u) = s(y)x^{\mathrm{T}}u \leq s(y)r^\circ(x)r(u)$, with equality for an appropriately chosen u.

With this choice of g, we have

$$\begin{aligned} \frac{1}{2}g(l_t'(w))^2 &= \frac{1}{2}g((f(wx_t) - y_t)x_t^{\mathrm{T}})^2 \\ &= r^\circ(x_t)^2 \frac{1}{2}s(f(wx_t) - y_t)^2 \\ &\leq \mathbb{D}_F(wx_t|y_t) \\ &= l_t(w) \end{aligned}$$

so Theorem 3 applies with $\lambda = 1$. (To achieve the effect of varying λ we can simply rescale the gauge s.)

For example, if f is the identity prediction link (so that $F(y) = \frac{1}{2}y^{\mathrm{T}}y$ and we can take s to be Euclidean distance) and $||x_t||_2 \leq 1$ (so that we can take r to be Euclidean distance also), then $g(w)$ will be the matrix 2-norm $||w||_2$. If we now take $l_0(w) = \frac{k}{2}\sum_{i,j} w_{ij}^2$, then $\frac{1}{k}D_{l_0}(v|w) \geq \frac{1}{2}g(v - w)^2$, so we can take $\eta_1 = \frac{1}{k}$.

Often we will take each coordinate of f to be one of the one-dimensional link functions described above. This kind of link function decomposes the multiple-output prediction problem into several single-output problems which share a parameter vector. On the other hand, sometimes we may know about dependencies among the components of the output vector. In this case we can take advantage of our knowledge by picking a prediction link function that encodes these dependencies. For example, if we have reason to believe that the output vector has covariance matrix Σ, we can select the link $\hat{y} = \Sigma p$ with its matching loss $\frac{1}{2}\hat{y}^{\mathrm{T}}\Sigma^{-1}\hat{y} - \hat{y}^{\mathrm{T}}p + \frac{1}{2}p^{\mathrm{T}}\Sigma p$.

9 Linear regression algorithms

In this section we will analyze two gradient-descent-like algorithms for linear regression: standard gradient descent and the exponentiated gradient algorithm from [KW97]. These algorithms are both generalized linear regression algorithms, and therefore MAP algorithms.

In linear regression problems, the loss on trial $t \geq 1$ is $l_t(w) = \mathbb{D}_{l_t}(w|0) = \frac{1}{2}(y_t - x_t \cdot w)^2$. This is the loss function for a generalized linear regression model using the identity prediction link with its matching loss function, the squared error. The algorithms differ only in their choice of prior loss l_0.

We will bound the regret of each algorithm by applying Theorem 3. Because l_t for $t \geq 1$ always has the form given above, we can take $\lambda = 1$ in Theorem 3; so the main part of the analysis of each algorithm will be to find appropriate seminorms g and g° with which to measure the parameter vectors w_t and the input vectors x_t.

First consider the gradient descent algorithm for linear regression, defined by the update

$$w_{t+1} = w_t + \eta(y_t - w_t \cdot x_t)x_t$$

Gradient descent is a GGD algorithm, given by the choice $l_0(w) = \frac{1}{2\eta}||w||_2^2$ (or $l_0 = \frac{1}{2\eta}||w - w_1||_2^2$ if we want a starting weight vector $w_1 \neq 0$). We showed above that if $||x_t||_2 \leq X$ for all t then we can take $g^\circ(x) = \frac{1}{X}||x||_2$ and $\eta_t = X^2\eta$ in Theorem 3. The result is that

$$\sum_{t=1}^{T} l_t(w_t) \leq \frac{1}{1 - X^2\eta}\sum_{t=1}^{T} l_t(u) + \frac{1}{\eta(1 - X^2\eta)}||u||_2^2$$

Next consider the exponentiated gradient algorithm. EG is a GGD algorithm given by the choice $l_0(w) = \frac{1}{\eta}H(w)$, so its update is

$$v_{t+1,i} = w_{t,i}\exp(\eta(y - w_t \cdot x_t)x_{t,i})$$

$$w_{t+1,i} = \frac{v_{t+1,i}}{\sum_j v_{t+1,j}}$$

To analyze EG, we will set X to be the maximum span of any of the input vectors, that is, $||x_t||_{\mathrm{sp}} \leq X$ where

$$||x||_{\mathrm{sp}} = \max_i x_i - \min_i x_i$$

It is easy to check that $||\cdot||_{\mathrm{sp}}$ is a seminorm. We can bound the polar of the span seminorm by splitting its argument vector into components parallel and perpendicular to $e = (1, 1, \ldots, 1)^{\mathrm{T}}$. We have $||e||_{\mathrm{sp}} = 0$, so $||e||_{sp}^\circ = \infty$. On the other hand, if x has no component along e, then $||x||_{\mathrm{sp}} \geq ||x||_\infty$, so $||x||_{\mathrm{sp}}^\circ \leq ||x||_1 \leq ||x||_2$. That means that, for any v and w and $a \in \partial H(w)$,

$$\mathbb{D}_H(v|a) \geq 2(||v - w||_{\mathrm{sp}}^\circ)^2$$

To see why, remember that we have assumed that $\partial H(w)$ is nonempty, so w must be in the unit simplex. So, depending on whether v is in the plane containing the unit simplex, either $v - w$ has a nonzero component along e, in which case $\mathbb{D}_H(v|a)$ is infinite, or $v - w$ is perpendicular to e, in which case $\mathbb{D}_H \geq 2||v - w||_2^2$. In either case the result follows. So, we can take $g^\circ(x) = \frac{1}{X}||x||_{\mathrm{sp}}$ and $\eta_t = \frac{1}{4}X^2\eta$ in Theorem 3 and conclude that

$$\sum_{t=1}^{T} l_t(w_t) \leq \frac{1}{1 - \frac{1}{4}X^2\eta}\sum_{t=1}^{T} l_t(u) + \frac{1}{\eta(1 - \frac{1}{4}X^2\eta)}H(u)$$

The above results can be compared to to Lemmas 5.2 (for GD) and 5.9 (for EG) in [KW97]. Unfortunately, our bounds here are slightly weaker than the ones in [KW97]. We do not believe that this is due to a weakness in our framework; instead we believe that with some additional work our theorems could be sharpened so that they are a strict generalization of the known results for linear regression with GD and EG.

After deriving the results mentioned above, the authors of [KW97] perform an additional step: they adjust the learning rate η so that the two terms in the regret bound have comparable coefficients. We have not taken a similar step.

As a final example consider the $\mathrm{EG}^\pm$ algorithm. Just as in [KW97], we could prove bounds on $\mathrm{EG}^\pm$ by reducing it to EG. Instead, we will sketch how to find the prior l_0 that yields the $\mathrm{EG}^\pm$ algorithm. Finding this prior is important both because it increases our understanding of $\mathrm{EG}^\pm$ and because it is a good first step towards a direct proof of the regret bound for $\mathrm{EG}^\pm$.

The $\mathrm{EG}^\pm$ algorithm can be defined by its parameter link function, which is (up to scaling) given by the mapping $w = f(x)$ defined as

$$v_i = \exp(x_i) - \exp(-x_i)$$

$$w_i = \frac{v_i}{\sum_j v_j}$$

The prior loss function l_0 for $\mathrm{EG}^\pm$, and its convex dual l_0^*, are determined up to scaling by this link function. We can find $l_0^*(x)$ by integrating f along any path from the origin to x.

A plot of the resulting function for two-dimensional inputs is shown in Figure 5; it looks like a rounded-off version of the L_∞ norm. The characterization of $l_0^*(x)$ as a rounded-off version of $||x||_\infty$ makes sense, since $\mathrm{EG}^\pm$ restricts w_t to have bounded L_1 norm and the dual of $\delta(x \mid ||x||_1 \leq 1)$ is the L_∞ norm.

10 Conclusion

We have presented a unified framework for deriving worst-case regret bounds for a wide class of learning algorithms.

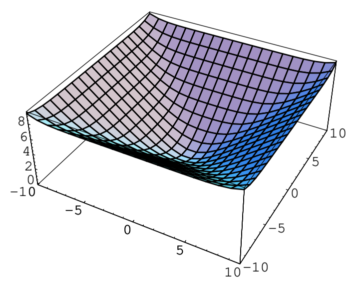

Figure 5: The dual of the potential function for EG$^{\pm}$.

These algorithms include weighted majority; gradient descent and generalizations of gradient descent such as exponentiated gradient; linear and logistic regression; and inference of the natural parameter in an exponential family. Because we have wherever possible avoided assumptions such as differentiability of the loss functions, our framework also includes a wide variety of new algorithms which we have not fully explored.

Our unified treatment clarifies the relationships among these methods, and it provides a recipe for designing and studying new learning algorithms. For example, we showed that both the gradient descent and exponentiated gradient algorithms for linear regression are MAP algorithms. By casting them in this common framework, we revealed that the only difference between these two algorithms is their choice of prior loss function. In addition to allowing a common proof of the regret bounds for these algorithms, this analysis suggests that we can design new linear regression algorithms simply by picking new priors. These priors can express known bounds on the parameter vector (for example, the prior $kw^2 + \delta(w|C)$ yields the gradient projection algorithm with domain C) or preferences for particular kinds of parameter vectors (for example, the prior of the EG$^{\pm}$ algorithm prefers vectors with low L_1 norm).

Our results also suggest new applications for old algorithms. By avoiding assumptions such as independence of training examples, we have justified the use of these algorithms in situations where they might not have been considered before.

References

[AHW96] Peter Auer, Mark Herbster, and Manfred Warmuth. Exponentially many local minima for single neurons. In D. Touretzky, M. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8. MIT Press, 1996.

[CBFH$^+$95] Nicoló Cesa-Bianchi, Yoav Freund, David P. Helmboldt, David Haussler, Robert E. Schapire, and Manfred K. Warmuth. How to use expert advice. Technical Report UCSC-CRL-95-19, University of California Santa Cruz, 1995.

[CLR90] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. McGraw-Hill, 1990.

[Gor99] Geoffrey J. Gordon. *Approximate Solutions to Markov Decision Processes*. PhD thesis, Carnegie Mellon University, 1999.

[KW96] Jyrki Kivinen and Manfred K. Warmuth. Relative loss bounds for multidimensional regression problems. In D. Touretzky, M. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8. MIT Press, 1996.

[KW97] Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, 1997. Preliminary version appeared as tech report UCSC-CRL-94-16; extended abstract appeared in 27th STOC.

[LW92] Nick Littlestone and Manfred Warmuth. The weighted majority algorithm. Technical Report UCSC-CRL-91-28, University of California Santa Cruz, 1992.

[MN83] P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman & Hall, London, 2nd edition, 1983.

[OR70] James Ortega and W. C. Rheinboldt. *Iterative solution of nonlinear equations in several variables*. Academic Press, New York, 1970.

[Roc70] R. Tyrell Rockafellar. *Convex Analysis*. Princeton University Press, New Jersey, 1970.

[SO91] A. Stuart and J. K. Ord. *Kendall's Advanced Theory of Statistics*. Oxford University Press, New York, 5th edition, 1991.

[Vov90] Volodimir Vovk. Aggregating strategies. In *Proc. 3rd Ann. Workshop on Computational Learning Theory*, pages 371–383. Morgan Kaufmann, 1990.