# Dimension-Free Exponentiated Gradient

**Authored by:**

Francesco Orabona

**Abstract**

We present a new online learning algorithm that extends the exponentiated gradient to infinite dimensional spaces. Our analysis shows that the algorithm is implicitly able to estimate the $L_2$ norm of the unknown competitor, $U$, achieving a regret bound of the order of $O(U log (U T+1))sqrt{T})$, instead of the standard $O((U2 +1) sqrt{T})$, achievable without knowing $U$. For this analysis, we introduce novel tools for algorithms with time-varying regularizers, through the use of local smoothness. Through a lower bound, we also show that the algorithm is optimal up to $sqrt{log T}$ term for linear and Lipschitz losses.

## 1 Paper Body

Online learning provides a scalable and flexible approach for solving a wide range of prediction problems, including classification, regression, ranking, and portfolio management. These algorithms work in rounds, where at each round a new instance is given and the algorithm makes a prediction. After the true label of the instance is revealed, the learning algorithm updates its internal hypothesis. The aim of the classifier is to minimize the cumulative loss it suffers due to its prediction, such as the total number of mistakes. Popular online algorithms for classification include the standard Perceptron and its many variants, such as kernel Perceptron [6], and p-norm Perceptron [7]. Other online algorithms, with properties different from those of the standard Perceptron, are based on multiplicative (rather than additive) updates, such as Winnow [10] for classification and Exponentiated Gradient (EG) [9] for regression. Recently, Online Mirror Descent (OMD)1 and has been proposed as a general meta-algorithm for online learning, parametrized by a regularizer [16]. By appropriately choosing the regularizer, most online learning algorithms are recovered as special cases of OMD. Moreover, performance guarantees can also be derived simply by instantiating the general OMD bounds to the specific regularizer being used. So, for all the ? first-order online learning algorithms, it is possible to prove regret bounds of the order of O(f (u) T ), where T is the number of rounds and f (u) is the regularizer used in OMD, evaluated on the competitor vector u. Hence, different choices of the regularizer will give rise to different algorithms and guar-

antees. For example, p-norm algorithms can be derived from the squared Lp -norm regularization, while EG can be derived from the one. In particular ? ? entropic 2 for the Euclidean regularizer ? T kuk2 , we have a regret bound of O( T (kuk /? + ?)). Knowing ? kuk it is possible to tune? ? to have a O(kuk T ) bound, that is optimal [1]. On the other hand, EG has a regret bound of O( T log d), where d is the dimension of the space. In this paper, I use OMD to extend EG to infinite dimensional spaces, through the use of a carefully designed time-varying regularizer. The algorithm, that I call Dimension-Free Exponentiated Gradient (DFEG), does not need direct access to single components of the vectors, rather it only requires 1 The algorithm should be more correctly called Follow the Regularized Leader, however here I follow Shalev-Shwartz in [16], and I will denote it by OMD.

1

to access them through inner products. Hence, DFEG can be used with kernels too, extending for the ? first time EG to the kernel domain. I prove a regret bound of O(kuk log(kukT + 1) T ). Up to logarithmic terms, the bound of DFEG is equal to the optimal bound obtained through the knowledge of kuk, but it does not require the tuning of any parameter. I built upon ideas of [19], but I designed my new algorithm as an instantiation of OMD, rather than using an ad-hoc analysis. I believe that this route increases the comprehension of the inner working of the algorithm, its relation to other algorithms, and it makes easier to extend it in other directions as well. In order to analyze DFEG, I also introduce new and general techniques to cope with timevarying regularizers for OMD, using the local smoothness of the dual of the regularization function, that might be of independent interest. ? I also extend and improve the lower bound in [19], to match the upper bound of DFEG up to a log T term, and to show an implicit trade-off on the regret versus different competitors. 1.1

Related works

Exponentiated gradient algorithms have been proposed by [9]. The algorithms have multiplicative updates and regret bounds that depend logarithmically on the dimension of the input space. In particular, they proposed a version of EG where the weights are not normalized, called EGU. A closer algorithm to mine is the epoch-free in [19]. Indeed, DFEG is equivalent to theirs when used on one dimensional problems. However, the extension to infinite dimensional spaces is nontrivial and very different in nature from their extension to d-dimensional problems, that consists on running a copy of the algorithm independently on each coordinate. Their regret bound depends on the dimension of the space and can neither be used with infinite dimensional spaces nor with kernels. ? Vovk proposed two algorithms for square loss, with regret bounds of O((kuk + Y ) T ) and ? O(kuk T ) respectively, where Y is an upper bound on the range of the target values [20]. A matching lower bound is also presented, proving the optimality of the second algorithm. However, the algorithms seem specific to the square loss and it is not possible to adapt them p to other losses. Indeed, the lower bound I prove shows that for linear and Lipschitz losses a log(kukT ) term is unavoidable. Moreover, the second algorithm, being an instantiation of the Aggregating Algorithm [21], does not seem to have an efficient

implementation. My algorithm also shares similarities in spirit with the family of self-confident algorithms [2, 7, 15], in which the algorithm self-tunes its parameters based on internal estimates. From the point of view of the proof technique, the primal-dual analysis of OMD is due to [15, 17]. Starting from the work of [8], it is now clear that OMD can be easily analyzed using only a few basic convex duality properties. See the recent survey [16] for a lucid description of these developments. The time-varying regularization for OMD has been explored in [4, 12, 15], but in none of these works does the negative terms in the bound due to the time-varying regularizer play a decisive role. The use of the local estimates of strong smoothness is new, as far as I know. A related way to have a local analysis is through the local norms [16], but my approach is better tailored to my needs.

2

Problem setting and definitions

In the online learning scenario the learning algorithms work in rounds [3]. Let X a Euclidean vector space2 , at each round t, an instance xt ? X, is presented to the algorithm, which then predicts a label y?t ? R. Then, the correct label yt is revealed, and the algorithm pays a loss ‘(? yt , yt ), for having predicted y?t , instead of yt . The aim of the online learning algorithm is to minimize the cumulative sum of the losses, on any sequence of data/labels {(xt , yt )}Tt=1 . Typical examples of loss functions are, for example, the absolute loss, —? yt ? yt —, and the hinge loss, max(1 ? y?t yt , 0). Note that the loss function can change over time, so in the following I will denote by ‘t : R ? R the generic loss function received by the algorithm at time t. In this paper I focus on linear prediction of the form y?t = hwt , xt i, where wt ? X represents the hypothesis of the online algorithm at time t. 2 All the theorems hold also in general Hilbert spaces, but for simplicity of exposition I consider a Euclidean setting.

2

Algorithm 1 Dimension-Free Exponentiated Gradient. Parameters: 0.882 ? a ? 1.109, L ¿ 0, ? ¿ 0. Initialize: ? 1 = 0 ? X, H0 = ? for t = 1, 2, . . . do Receive kxt k, where xt ? X

Set Ht = Ht?1 + L2 max kxt k, kxt k2 3 ? Set ?t = a Ht , ?t = Ht2 if k? t k == 0 then choose wt= 0 ?t else choose wt = ?t k? exp k??ttk tk Suffer loss ‘t (hwt , xt i) Update ? t+1 = ? t ? ¿t (hwt , xt i)xt end for We strive to design online learning algorithms for which it is possible to prove a relative regret bound. Such analysis bounds the regret, that is the difference between the cumulative loss of the PT PT algorithm, t=1 ‘t (hwt , xt i), and the one of an arbitrary and fixed competitor u, t=1 ‘t (hu, xt i). We will consider L-Lipschitz losses, that is —‘t (y) ? ‘t (y 0 )— ? L—y ? y 0 —, ?y, y 0 . I now introduce some basic notions of convex analysis that are used in the paper. I refer to [14] for definitions and terminology. I consider functions f : X ? R that are closed and convex. Given a ? closed and convex function f with domain S ? X, its Fenchel conjugate f : X ? R is defined as f ? (u) = supv?S hv, ui ? f (v) . The Fenchel-Young inequality states that f (u) + f ? (v) ? hu, vi for all v, u. A vector x is a subgradient of a convex function f at v if f (u) ? f (v) ? hu ? v,

xi for any u in the domain of f . The differential set of f at v, denoted by ?f (v), is the set of all the subgradients of f at v. If f is also differentiable at v, then ?f (v) contains a single vector, denoted by ?f (v), which is the gradient of f at v. Strong convexity and strong smoothness are key properties in the design of online learning algorithms, they are defined as follows. A function f is ?-strongly convex with respect to a norm k?k if for any u, v in its domain, and any x ? ?f (u), ? 2 f (v) ? f (u) + hv ? u, xi + ku ? vk . 2 The Fenchel conjugate f ? of a ?-strongly convex function f is everywhere differentiable and strongly smooth [8], this means that for all u, v ? X, f ? (v) ? f ? (u) + hv ? u, ?f ? (u)i +

1 ?-

1 2 ku ? vk? . 2?

In the remainder of the paper all the norms considered will be the L2 ones.

3

Dimension-Free Exponentiated Gradient

In this section I describe the DFEG algorithm. The pseudo-code is in Algorithm 1. It shares some similarities with the exponentiated gradient with unnormalized weights algorithm [9], to the selftuning variant of exponentiated gradient in [15], and to the epoch-free algorithm in [19]. However, note that it does not access to single coordinates of wt and xt , but only their inner products. Hence, we expect the algorithm not to depend on the dimension of X, that can be even infinite. In other words, DFEG can be used with kernels as well, on contrary of all the mentioned algorithms above. For the DFEG algorithm we have the following regret bound, that will be proved in Section 4. Theorem 1. Let 0.882 ? a ? 1.109, ? ¿ 0, then, for any sequence of input vectors {xt }Tt=1 , any sequence of L-Lipschitz convex losses {‘t (?)}Tt=1 , and any u ? X, the following bound on the regret holds for Algorithm 1 T X

‘t (hwt , xt i) ?

t=1

where HT = ? +

T X t=1

PT

t=1

‘t (hu, xt i) ?

3

p 4 exp(1 + a1 ) ? + akuk HT ln HT2 kuk ? 1 , L ?

L2 max(kxt k, kxt k2 ). 3

The bound has a logarithmic part, typical of the family of exponentiated gradient algorithms, but instead of depending on the dimension, it depends on the norm of the competitor, kuk. Hence, the regret bound of DFEG holds for infinite dimensional spaces as well, that is, it is dimension-free. It is interesting to compare this bound with the usual bound for online learning using an L2 regular? t izer. Using a time-varying regularizer ft (w) = ? kwk2 it is easy to see, e.g. [15], that the bound ? would be3 O((kuk2 /? + ?) T ). If an upper ? bound U on kuk is known, we can use it to tune ? to obtain an upper bound of ? the order of O(U T ). On the other hand, we obtain for DFEG a bound of O(kuk log(kukT + 1) T ), that is optimal bound, up to logarithmic terms,

without knowing U . So my bound goes to constant if the norm of the competitor goes to zero. However, note that, for any fixed competitor, the gradient descent bound is asymptotically better. The lower bound on the range of a we get comes from technical details of the analysis. The parameter a is directly linked to the leading constant of the regret bound; therefore, it is intuitive that the range of acceptable values must have a lower bound different from zero. This is also confirmed by the lower bound in Theorem 2 below. Notice that the bound is data-dependent because it depends on the sequence of observed input vectors xt . A data-independent bound can be easily obtained from the upper bound on the norm of the input vectors. The use of the function max(kxt k, kxt k2 ) is necessary to have such a data-dependent bound and it seems that it cannot be avoided in order to prove the regret bound. It is natural to ask if the log term in the bound can be avoided. Extending Theorem 7 in [19], we can reply in the negative to this question. In particular, the following theorem shows that the regret of any online learning algorithm has a satisfy to a trade-off between the guarantees against the competitor with norm equal to zero and the ones against other competitors. A similar trade-off has been proven in the expert settings [5]. Theorem 2. Fix a non-trivial vector space X , a specific online learning algorithm, and let the sequence of losses be composed by linear losses. If the algorithm guarantees a zero regret against the competitor with zero L2 norm, then there exists a sequence of T vectors in X , such that the regret against any other competitor is ?(T ). On the other hand, if the algorithm guarantees a regret at most of ¿ 0 against the competitor with zero L2 norm, then, for any 0 ¡ ? ¡ 1, there exists a T0 and a sequence of T ? T0 unitary norm vectors z t ? X , and a vector u ? X such that v u ? ! r T T X X 1 u ?kuk T t hwt , z t i ? (1 ? ?)kuk hu, z t i ? ?2. T log log 2 3 t=1 t=1 The proof can be found in the supplementary material. It isqpossible to show that the optimal ? is of the order of log1 T , so that the leading constant approaches log1 2 ? 1.2011 when T goes to infinity. ? It is also interesting to note ? that an L2 regularizer suffers a loss of O( T ) against a competitor with zero norm, that cancels the log T term.

4

Analysis

In this section I prove my main result. I will first briefly introduce the general OMD algorithm with time-varying regularizers on which my algorithm is based. 4.1

Online mirror descent and local smoothness

Algorithm 2 is a generic meta-algorithm for online learning. Most of the online learning algorithms can be derived from it, choosing the functions ft and the vectors z t . The following lemma, that is a generalization of Corollary 4 in [8], Corollary 3 in [4], and Lemma 1 in [12], is the main tool to prove the regret bound for the DFEG algorithm. The proof is in the supplementary material. ?

3

Despite what claimed in Section 1 of [19], the use of the time-varying regularizer ft (w) = guarantees a sublinear regret for unconstrained online convex optimization, for any ? ¿ 0.

4

t kwk2 ?

Algorithm 2 Time-varying Online Mirror Descent Parameters: A sequence of convex functions f1 , f2 , . . . defined on S ? X. Initialize: ? 1 = 0 ? X for t = 1, 2, . . . do Choose wt = ?ft? (? t ) Observe z t ? X Update ? t+1 = ? t + z t end for Lemma 1. Assume Algorithm 2 is run with functions f1 , f2 , . . . defined on a common domain S ? X. Then for any w0t , u ? S we have T X

hz t , u ? w0t i ? fT (u) +

t=1

T X

? ft? (? t+1 ) ? ft?1 (? t ) ? hw0t , z t i ,

t=1

where we set f0? (w01 ) = max0?? ?1 k?2 ft? (? t + ? z t )k

0. Moreover, if f1? , f2? , . . . are twice differentiable, and ? ?t , then we have

? ? ft? (? t+1 ) ? ft?1 (? t ) ? hwt , z t i ? ft? (? t ) ? ft?1 (? t ) +

?t 2 kz t k . 2

Note that the above Lemma is usually stated assuming the strong convexity of ft , that is equivalent to the strong smoothness of ft? , that in turns for twice differentiable functions is equivalent to a global bound on the norm of the Hessian of ft? (see Theorem 2.1.6 in [11]). Here I take a different route, assuming the functions ft? to be twice differentiable, but using the weaker hypothesis of local boundedness of the Hessian of ft? . Hence, for twice differentiable conjugate functions, this bound is always tighter than the ones in [4, 8, 12]. Indeed, in our case, the global strong smoothness cannot be used to prove any meaningful regret bound. We derive the Dimension-Free Exponentiated Gradient from the general OMD above. Set in Algorithm 2 ft (w) = ?t kwk(log(?t kwk) ? 1), where ?t and ?t are defined in Algorithm 1, and z t = ?¿t (hwt , xt i)xt . The proof idea of my theorem is the following. First, assume that we are on a round where we have a local upper bound on the norm of the Hessian ft? .?The usual approach in these kind of proof is to have a regularizer that is growing over time as t, so that the ? (? t ) are negative and can be safely discarded. At the same time the sum of the terms ft? (? t ) ? ft?1 ? ? squared norms of the gradients will typically be of the order of O( T ), giving us a O( T ) regret bound (see for example the proofs in [4]). However, following this approach in ? DFEG we would have that the sum of norms of the squared gradients grows much faster than O( T ). This is due to the fact that the global strong smoothness is too small. Hence I introduce a different proof method. In the following, I will show the surprising result that with my choice of the regularizers ft , the ? terms ft? (? t ) ? ft?1 (? t ) and the squared norm of the gradient cancel out. Notice that already in [12, 13] it has been advocated not to discard those terms to obtain tighter bounds. Here the same terms play a major role in the proof and they are present thanks to the time-varying regularization. This is in agreement with Theorem 9 in [19] that rules out algorithms with a fixed regularizer to obtain regret bounds like Theorem 1. It remains to bound the regret in the rounds where we do not have an upper bound on the norm of the Hessian. In these

rounds I show that the norm of wt (and ? t ) is small enough so that the regret is still bounded, thanks to the choice of ?t . 4.2

Proof of the main result

We start defining the new regularizer and show its properties in the following Lemma (proof in the supplementary material). Note the similarities with EGU, where the regularizer is Pd d i=1 wi (log(wi ) ? 1), w ? R , wi ? 0 [9]. Lemma 2. Define f (w) = ?kwk(ln(?kwk) ? 1), for ?, ? ¿ 0. The following properties hold ? f ? (?) =

? ?

exp k?k ? . 5

? ?f ? (?) =

? ?k?k

? k?2 f ? (?)k2 ?

exp k?k ? . 1 ? min(k?k,?)

exp( k?k ? ).

Equipped with a local upper bound on the Hessian of f ? , we can now use Lemma 1. We notice that?Lemma 1 also guides ? us in the choice of the sequences ?t . In fact if we want the regret to be ? ? O( T ), ?t must be O( T ) too. In the proof of Theorem 1 we also use the following three technical lemmas, whose proofs are in the supplementary material. The first two are used to upper bound the exponential function with quadratic functions. Lemma 3. Let M ¿ 0, then for any p+

exp(M )?p 2 x M2

exp(M ) M 2 +1

? p ? exp(M ), and 0 ? x ? M , we have exp(x) ?

.

Lemma 4. Let M ¿ 0, then for any 0 ? x ? M , we have exp(x) ? 1 + x + Lemma 5. For any p, q ¿ 0 we have that

?2 p

?

?2 p+q

?

q 3

(p+q) 2

exp(M )?1?M 2 x . M2

.

Proof of Theorem 1. In the following denote by n(x) := max(kxk, kxk2 ). We will use Lemma 1 to upper bound the regret of DFEG. Hence, using the notation in Algorithm 1, set z t = ?¿t (hwt , xt i)xt , and ft (w) = ?t kwk(log(?t kwk) ? 1). Observe that, by the hypothesis on ‘t , we have kz t k ? Lkxt k. We first consider two cases, based on the norm of ? t . Case 1: k? t k ¿ ?t + kz t k. With this assumption, and using the third property of Lemma 2, we have

k? t k+kz t k zt k exp exp k?t +? ? ? t t ? max k?2 ft? (? t + ? z t )k ? max . 0?? ?1 0?? ?1 ?t min(k? t + ? z t k, ?t ) ?t ? t 2

? We now use the second statement of Lemma 1. We have that ?t kz2 t k + ft? (? t ) ? ft?1 (? t ) can be upper bounded by

kz t k2 k? t k + kz t k ?t k? t k ?t?1 k? t k exp + exp ? exp 2?t ?t ?t ?t ?t ?t?1 ?t?1

kz t k2 k? t k + kz t k ?t k? t k ?t?1 k? t k ? exp + exp ? exp 2?t ?t ?t ?t ?t ?t?1 ?t

k? t k kz t k kz t k2 a a = exp exp + ? . (1) ?t 2aHt2 ?t Ht Ht?1 We will now prove that the term in the parenthesis of (1) is negative. It can be rewritten as

kz t k2 Ht?1 exp kz?ttk ? 2a2 Ht?1 L2 n(xt ) ? 2a2 L4 (n(xt ))2 kz t k a a kz t k2 exp + ? = , 2aHt2 ?t Ht Ht?1 2aHt2 Ht?1 kz t k 1 2 ?t ? a , so we now use Lemma 3 with p = 2a and 1 exp( a ) because 1 +1 ? 2a2 ? exp( a1 ), ? 0.825 ? a ? 1.109, as it a2

and from the expression of ?t we have that M = 1/a. These are valid settings can be verified numerically.

kz t k2 kz t k a a exp + ? 2 2aHt ?t Ht Ht?1

2 2 2 2 kz t k Ht?1 2a + a (exp( a1 ) ? 2a2 ) kz?t2k ? 2a2 Ht?1 L2 n(xt ) ? 2a2 L4 (n(xt ))2 t ? 2aHt2 Ht?1

2 2 tk L2 kxt k2 Ht?1 2a2 + a2 (exp( a1 ) ? 2a2 ) L akx ? 2a2 Ht?1 L2 kxt k2 ? 2a2 L4 kxt k2 2H t ? 2aHt2 Ht?1 L4 kxt k4 (exp( a1 ) ? 4a2 ) ? ? 0, (2) 2aHt2 Ht?1 6

where in last step we used the fact that exp( a1 ) ? 4a2 , ? a ? 0.882, as again it can be verified numerically. Case 2: k? t k ? ?t + kz t k. We use the first statement of Lemma 1, setting w0t = wt if k?k = 6 0, and w0t = 0 otherwise. In this 0 way, from the second property of Lemma 2, we have that kwt k ? ?1t exp( k??ttk ). Note that any other choice of w0t satisfying the the previous relation on the norm of w0t would have worked as well.

?t k? t+1 k ?t?1 k? t k ? ? ft (? t+1 ) ? ft?1 (? t ) = exp ? exp ?t ?t ?t?1 ?t?1

k

exp kz ?t Ht?1 ? Ht k? t k ?t kz t k ?t?1 k? t k a Ht ? exp exp ? = a exp . (3) ?t ?t ?t ?t?1 ?t Ht?1 Ht Remembering that kz?ttk ? a1 , and using Lemma 4 with M = a1 , we have

kz t k Lkxt k ? ? Ht?1 ? L2 n(xt ) ? Ht?1 exp ? ? Ht?1 ? L2 kxt k2 Ht?1 exp a Ht a Ht

1 1 L2 kxt k2 Lkxt k 2 + a exp ?1? ? Ht?1 1 + ? ? Ht?1 ? L2 kxt k2 a a a2 Ht a Ht

1 LHt?1 kxt k 1 L2 Ht?1 kxt k2 ? + exp = ?1? ? L2 kxt k2 a a Ht a Ht

1 1 LHt?1 kxt k LHt?1 kxt k 2 2 ? ? + L kxt k exp ?2? ? , (4) ? a a a Ht a Ht where in the last step we used the fact that exp( a1 ) ? 2 ? a1 ? 0, ? a ? 0.873, verified numerically. Putting together (3) and (4), we have

k? t k Lkxt k ? ? hw0t , z t i ft? (? t+1 ) ? ft?1 (? t ) ? hw0t , z t i ? exp 3 ?t 2 H

t

k? t k Lkxt k k? t k Lkxt k k? t k Lkxt k 0 ? exp + Lkwt kkxt k ? exp + exp 3 3 ?t ?t ?t ?t Ht2 Ht2

1 2 exp(1 + a )Lkxt k k? t k Lkxt k ? , (5) = 2 exp 3 3 ?t Ht2 Ht2 where in the second inequality we used the Cauchy-Schwarz inequality and the

Lipschitzness of ‘t , in the third the bound on the norm of w0t , and in the last inequality the fact that k? t k ? ?t + kz t k implies exp( k??ttk ) ? exp(1 + a1 ). Putting together (2) and (5) and summing over t, we have T X

T T X 2 exp(1 + a1 )Lkxt k 2 X exp(1 + a1 )L2 kxt k ? ft? (? t+1 ) ? ft?1 (? t ) ? hw0t , z t i ? ? P 3 L t=1 ( tj=1 L2 kxt k + ?) 32 Ht2 t=1 t=1 ? ? T 4 exp(1 + a1 ) X 4 exp(1 + a1 ) 1 1 ?q ?? ? q ? ? , Pt Pt?1 2 L L ? L2 kx k + ? L kx k + ? t=1 j=1

t

j=1

t

where in the third inequality we used Lemma 5. The stated bound can be obtained observing that ‘t (hwt , xt i) ? ‘t (hu, xt i) ? hu ? wt , z t i, from the convexity of ‘t and the definition of z t .

5

Experiments

A full empirical evaluation of DFEG is beyond the scope of this paper. Here I just want to show the empirical effect of some of its theoretical properties. In all the experiments I used the absolute loss, 7

18 16

cadata dataset

4

Synthetic dataset 2

20 DFEG Kernel GD, eta=0.05 Kernel GD, eta=0.1 Kernel GD, eta=0.2

x 10

1.95

cpusmall dataset 14000

DFEG Kernel GD, various eta

DFEG Kernel GD, various eta 12000

1.9 1.85

14

10000

10

1.8 1.75

8

1.7

6

1.65

4

1.6

2

1.55

0 0

Total loss

Total loss

Regret

12

8000
6000
4000
2000
4000 6000 Numer of samples
8000
10000
1.5 ?2 10
?1
10
0
10 eta
1
10
2
10
2000 ?1 10
0
1
10
10
2
10
eta

Figure 1: Left: regret versus number of input vectors on synthetic dataset. Center and Right: total loss for DFEG and Kernel GD on the cadata and cpusmall dataset respectively.

so L = 1, a is set to the minimal value allowed by Theorem 1 and ? = 1. I denote by Kernel GD ? the OMD with the regularizer ?t kwk2 . First, I generated synthetic data as in the proof of Theorem 2, that is the input vectors are all the same and the yt is equal to 1 for the t even and ?1 for the others. In this case we know that the optimal predictor has norm equal to zero and we can exactly calculate the value of the regret. Figure 1(left) I have plotted the regret as a function of the number of input vectors. As ? predicted by the theory, DFEG has a constant regret, while Kernel GD has a regret of the form O(? T ). Hence, it can have a constant regret only when ? is set to zero, and this can be done only with prior knowledge of kuk, that is impossible in practical applications. For the second experiment, I analyzed the behavior of DFEG on two real word regression datasets, cadata and cpusmall4 . I used the Gaussian kernel with variance equal to the average distance between training input vectors. I have plotted in Figure 1(central) the final cumulative loss of DFEG and the ones of GD with varying values of ?. We see that, while the performance of Kernel GD can be better of the one of DFEG, as predicted by the theory, its performance varies greatly in relation to ?. On the other hand the performance of DFEG is close to the optimal one without the need to tune any parameters. It is also worth noting the catastrophic result we can get from

a wrong tuning of ? in GD. Similar considerations hold for the cpusmall dataset in Figure 1(right).

6

Discussion

I have presented a new algorithm for online learning, the first one in the family of exponentiated gradient to be dimension-free. Thanks ? to new analysis tools, I have proved that DFEG attains a regret bound of O(U log(U T + 1)) T ), without any ? parameter to tune. I also proved a lower bound that shows that the algorithm is optimal up to log T term for linear and Lipschitz losses. The problem of deriving a regret bound that depends on the sequence of theq gradients, rather than PT on the xt , remains open. Resolving this issue would result in the tighter O( t=1 ‘t (hw t , xt i)) regret bounds in the case that the ‘t are smooth [18]. The difficulty in proving these kind of bounds seem to lie in the fact that (2) is negative only because Ht ? Ht?1 is bigger than kz t k2 . Acknowledgments I am thankful to Jennifer Batt for her help and support during the writing of this paper, to Nicol‘o Cesa-Bianchi for the useful comments on an early version of this work, and to Tamir Hazan for his writing style suggestions. I also thank the anonymous reviewers for their precise comments, which helped me to improve the clarity of this manuscript.

4

http://www.csie.ntu.edu.tw/?cjlin/libsvmtools/datasets/

8

# 2 References

[1] J. Abernethy, A. Agarwal, P. L. Bartlett, and A. Rakhlin. A stochastic view of optimal regret through minimax duality. In COLT, 2009. [2] P. Auer, N. Cesa-Bianchi, and C. Gentile. Adaptive and self-confident on-line learning algorithms. J. Comput. Syst. Sci., 64(1):48?75, 2002. [3] N. Cesa-Bianchi and G. Lugosi. Prediction, learning, and games. Cambridge University Press, 2006. [4] J. C. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, 12:2121?2159, 2011. [5] E. Even-Dar, M. Kearns, Y. Mansour, and J. Wortman. Regret to the best vs. regret to the average. In N. H. Bshouty and C. Gentile, editors, COLT, volume 4539 of Lecture Notes in Computer Science, pages 233?247. Springer, 2007. [6] Y. Freund and R. E. Schapire. Large margin classification using the Perceptron algorithm. Machine Learning, pages 277?296, 1999. [7] C. Gentile. The robustness of the p-norm algorithms. Machine Learning, 53(3):265?299, 2003. [8] S. M. Kakade, S. Shalev-Shwartz, and A. Tewari. Regularization techniques for learning with matrices. CoRR, abs/0910.0610, 2009. [9] J. Kivinen and M. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. Information and Computation, 132(1):1?63, January 1997. [10] N. Littlestone. Learning quickly when irrelevant attributes abound: a new linear-threshold algorithm. Machine Learning, 2(4):285?318, 1988. [11] Y. Nesterov. Introductory lectures on convex optimization: A basic

course, volume 87. Springer, 2003. [12] F. Orabona and K. Crammer. New adaptive algorithms for online classification. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, Advances in Neural Information Processing Systems 23, pages 1840?1848. 2010. [13] F. Orabona, K. Crammer, and N. Cesa-Bianchi. A generalized online mirror descent with applications to classification and regression, 2013. arXiv:1304.2994. [14] R. T. Rockafellar. Convex Analysis (Princeton Mathematical Series). Princeton University Press, 1970. [15] S. Shalev-Shwartz. Online learning: Theory, algorithms, and applications. Technical report, The Hebrew University, 2007. PhD thesis. [16] S. Shalev-Shwartz. Online learning and online convex optimization. Foundations and Trends in Machine Learning, 4(2), 2012. [17] S. Shalev-Shwartz and Y. Singer. A primal-dual perspective of online learning algorithms. Machine Learning Journal, 2007. [18] N. Srebro, K. Sridharan, and A. Tewari. Smoothness, low noise and fast rates. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, Advances in Neural Information Processing Systems 23, pages 2199?2207. 2010. [19] M. Streeter and B. McMahan. No-regret algorithms for unconstrained online convex optimization. In P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, Advances in Neural Information Processing Systems 25, pages 2411?2419. 2012. [20] V. Vovk. On-line regression competitive with reproducing kernel hilbert spaces. In Jin-Yi Cai, S.Barry Cooper, and Angsheng Li, editors, Theory and Applications of Models of Computation, volume 3959 of Lecture Notes in Computer Science, pages 452?463. Springer Berlin Heidelberg, 2006. [21] V. G. Vovk. Aggregating strategies. In COLT, pages 371?386, 1990.

9