Online PCA with Optimal Regrets*

Jiazhong Nie[1], Wojciech Kotłowski[2], and Manfred K. Warmuth[1]

[1] Department of Computer Science, University of California, Santa Cruz, CA 95064
{niejiazhong,manfred}@cse.ucsc.edu
[2] Institute of Computing Science, Poznań University of Technology, Poland
wkotlowski@cs.put.poznan.pl

Abstract. We carefully investigate the online version of PCA, where in each trial a learning algorithm plays a k-dimensional subspace, and suffers the compression loss on the next instance when projected into the chosen subspace. In this setting, we give regret bounds for two popular online algorithms, Gradient Descent (GD) and Matrix Exponentiated Gradient (MEG). We show that both algorithms are essentially optimal in the worst-case when the regret is expressed as a function of the number of trials. This comes as a surprise, since MEG is commonly believed to perform sub-optimally when the instances are sparse. This different behavior of MEG for PCA is mainly related to the non-negativity of the loss in this case, which makes the PCA setting qualitatively different from other settings studied in the literature. Furthermore, we show that when considering regret bounds as a function of a loss budget, MEG remains optimal and strictly outperforms GD.

Next, we study a generalization of the online PCA problem, in which the Nature is allowed to play with dense instances, which are positive matrices with bounded largest eigenvalue. Again we can show that MEG is optimal and strictly better than GD in this setting.

Keywords: Online learning, regret bounds, expert setting, k-sets, PCA, Gradient Descent and Matrix Exponentiated Gradient algorithms.

1 Introduction

In Principal Component Analysis (PCA), the n-dimensional data is projected / compressed onto a k-dimensional subspace so that the total quadratic compression loss is minimized. The problem of (centered) PCA is equivalent to finding the eigenvectors of the k largest eigenvalues of the covariance matrix. Here the data points $\boldsymbol{x}_t$ are arbitrary unit vectors in $\mathbb{R}^n$ and the instances of the PCA problem are the outer products $\boldsymbol{x}_t\boldsymbol{x}_t^\top$. The covariance matrix $\boldsymbol{C} = \sum_t \boldsymbol{x}_t\boldsymbol{x}_t^\top$ is the sum of the instances.

In this paper we consider the online version of centered PCA [17], where in each trial $t = 1, \ldots, T$, the algorithm chooses (based on the previously observed

* This research was supported by the NSF grant IIS-0917397, while the second author was visiting UC Santa Cruz.

S. Jain et al. (Eds.): ALT 2013, LNAI 8139, pp. 98–112, 2013.

points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}$) a subspace of dimension k described by a projection matrix $\boldsymbol{P}_t$ of rank k. Then a next point $\boldsymbol{x}_t$ (or instance $\boldsymbol{x}_t\boldsymbol{x}_t^\top$) is revealed and the algorithm incurs the "compression loss":

$$\|\boldsymbol{x}_t - \boldsymbol{P}_t\boldsymbol{x}_t\|_2^2 \;=\; \operatorname{tr}((\boldsymbol{I} - \boldsymbol{P}_t)\boldsymbol{x}_t\boldsymbol{x}_t^\top). \tag{1}$$

The goal is to obtain an online algorithm whose cumulative loss over trials $t = 1, \ldots, T$ is close to the cumulative loss of the best rank k projection matrix chosen in hindsight after seeing all T instances. The difference between the cumulative losses of the algorithm and the best off-line comparator is called the *regret*.

There are two main families of algorithms in online learning: The Gradient Descent (GD)[5, 20] family which is based on regularizing with the squared Euclidean distance, and the Exponentiated Gradient (EG)[11] family which use the relative entropy as their regularization. The first family leads to *additive updates* of the parameter vector/matrix. When there are no constraints on the parameter space, then the parameter vector/matrix is a linear combination of the instances. However when there are constraints, then after the update the parameter is projected onto the constraints (via a Bregman projection w.r.t. the squared Euclidean distance). As we shall discuss in the conclusions (Section 4), projections w.r.t. inequality constraints introduce all kinds of subtle problems for GD. The second family leads to *multiplicative update* algorithms. For that family the non-negativity constraints on the parameters are already enforced and less projections are needed.

In [17], a matrix version of the multiplicative update was applied to PCA, whose regret bound is logarithmic in the dimension n. This algorithm is based on regularizing with the quantum relative entropy and is called the *Matrix Exponentiated Gradient* (MEG) algorithm [16]. Beginning with some of the early work on linear regression [11], it is known that multiplicative updates are especially useful when the instances are *dense*. In the matrix context this means that the symmetric positive semi-definite instance matrix $\boldsymbol{X}_t \in \mathbb{R}^{n\times n}$ processed at trial t has maximum eigenvalue of say one. However in the PCA context, the instance matrices are the outer products, i.e. $\boldsymbol{X}_t = \boldsymbol{x}_t\boldsymbol{x}_t^\top$. Such instances (also called *dyads*) are *s*parse in the sense that their trace norm is one, independent of the dimension n of the instance matrix. Thus, one may suspect that MEG is not able to fully exploit the sparsity of the instance matrices. On the other hand for linear regression, GD is known to have the advantage when the instances are sparse and consistently with that, when GD is used for PCA, then its regret is bounded by a term that is *independent* of the dimension of the instances. The advantage of GD in the sparse case is also supported by a general survey of Mirror Descent algorithms (to which GD and MEG belong) for the case when the loss vectors (which have negative components) lie in certain symmetric norm balls [14].

Surprisingly, the situation is quite different for PCA: We show that MEG achieves the same regret bound as GD for online PCA (despite the sparseness of the instance matrices) and the regret bounds for both algorithms are within a constant factor of our new lower bound that holds for any online PCA algorithm.

This surprising performance of MEG comes from the fact that the losses in the PCA case are restricted to be non-negative, and therefore our results are qualitatively different from the cases studied in [14] where loss vectors are within a $p-$norm ball, i.e. symmetric around zero.

Actually, there are two kinds of regret bounds in the literature: bounds expressed as a function of the time horizon T and bounds that depend on an *upper bound* on the loss of the best comparator (which we call a *loss budget* following [2]). In typical applications for PCA, there exists a low dimensional subspace which captures most of the variance in the data and guarding against the worst-case loss that grows with T is not useful. We can show that when considering regret bounds as a function of a loss budget, MEG is optimal and strictly better than GD by a factor of $\sqrt{k}$. This suggests that the multiplicative updates algorithm is the best choice for prediction problems, in which the parameters are mixture of projection matrices and the losses are non-negative. Note that by upper bounds on the regret, we mean upper bounds for particular algorithms. However, the matching lower bounds are always proved against any algorithm that solves the problem.

Related Work and Our Contribution: The comparison of the GD and MEG algorithms has quite an extensively history (see, e.g. [11, 19, 15, 14]). It is simplest to compare algorithms in the case when the loss is linear. Linear losses are the least convex losses and in the regret bounds, convex losses are often approximated by first-order Taylor approximations [20] which are linear, and the gradient of the loss functions as the loss vector. Note that in this case the assumptions on the gradient of the loss are typically symmetric.

In the case when the parameter space and the space of loss vectors are convex and symmetric, the regret bounds are as expected: EG is optimal when the parameter space is 1-norm bounded and the loss vectors are infinity norm bounded, and GD is optimal when the both spaces are 2-norm bounded [15, 14]. However, none of the previous work exploits the special PCA setup, where the loss matrices (here the instances) are non-negative and sparse (see (1)). In this paper we carefully study this case.

We also made significant technical progress on the lower bounds. Previous lower bounds focused on the non-sparse case [17, 12]. Lower bounds were proved as a function of a loss budget. In this paper we prove lower bounds as a function of time. These lower bounds harbor the budget case as a special case.

For the time dependent case, lower bounds were previously shown for the expert setting [7, 4, 1]. However, these lower bounds rely on the Central Limit Theorem and only hold in the limit (as $T, n \to \infty$). In contrast our lower bounds use a probabilistic bounding argument for the minimum of n random variables and the resulting bounds are non-asymptotic.

In summary, our contribution consists of proving tighter regret bounds for two online PCA algorithms, as well as proving lower bounds on the regret of any algorithm for online PCA. From that we get the following conclusions: MEG's and GD's regret bounds are independent of the dimension n of the problem and are tight within a constant factor when the time horizon T is fixed, which implies that both algorithms are essentially optimal in this case. If we fix the loss budget instead, MEG remains optimal, while GD is proved to be suboptimal.

Furthermore, for a generalization of the PCA setting to the dense instance case, we improve the known regret bound significantly by switching from a loss version to a gain version of MEG. It turns out that MEG is optimal in the dense setting as well, whereas GD is not.

Outline of the Paper: In Section 2 we describe the MEG and GD algorithms and prove regret bounds for them. In Section 3 we prove lower bounds for both the sparse and the dense setting. We conclude with an open problem about the Incremental Off-line version of GD.

2 The Online Algorithms

The GD and MEG algorithms are both examples of the *Mirror Descent* algorithm [14]. Mirror Descent updates its parameters by minimizing a trade-off between a divergence to the parameter at the end of the last trial and the loss on the current single instance, followed by a projection into the parameter set. The divergence is always a Bregman divergence. In Machine Learning these updates were discovered in [11, 9]. If we choose the quantum relative entropy as the Bregman divergence, then we get the matrix version of a multiplicative update algorithm known as *Matrix Exponentiated Gradient* algorithm (here denoted as MEG). Similarly, the squared Frobenius norm results in an additive update algorithm known as *Gradient Descent* (GD).[1]

Sparse and Dense Instances: We call a symmetric positive semi-definite matrix *sparse* iff its trace norm (sum of the eigenvalues) is at most one. Note that the instance matrices in our online PCA setup are sparse since they are outer products of unit vectors. We also generalize our subspace learning problem to *dense* instance matrices, which are symmetric positive semi-definite matrices with maximum eigenvalues at most one.

2.1 The MEG Algorithms

In the online PCA problem, the algorithm predicts at trial t with a projection matrix $\boldsymbol{P}_t$ of rank k and incurs the compression loss $\|\boldsymbol{x}_t - \boldsymbol{P}_t\boldsymbol{x}_t\|_2^2$ upon receiving the next point $\boldsymbol{x}_t$. This loss is equivalent to the linear loss $\operatorname{tr}((\boldsymbol{I}-\boldsymbol{P}_t)\boldsymbol{X}_t)$, where $\boldsymbol{X}_t = \boldsymbol{x}_t\boldsymbol{x}_t^\top$ is the instance matrix. Actually, $\boldsymbol{I}-\boldsymbol{P}_t$ is a complementary projection matrix which has rank $m = n-k$. Since

$$\operatorname{tr}((\boldsymbol{I}-\boldsymbol{P}_t)\boldsymbol{X}_t) = \operatorname{tr}(\boldsymbol{X}_t) - \operatorname{tr}(\boldsymbol{P}_t\boldsymbol{X}_t),$$

there are always two versions of the algorithms: one that produces projection matrices of rank $m = n-k$ and minimizes the compression loss $\operatorname{tr}((\boldsymbol{I}-\boldsymbol{P}_t)\boldsymbol{X}_t)$

[1] We avoided the name "Matrix" Gradient Descent, since Gradient Descent updates are motivated by regularizing with the squared Euclidean distance and the Frobenius norm of a matrix is simply the Euclidean norm of "vectorized" matrix.

and one that produces projection matrices of rank k and maximizes the gain $\operatorname{tr}(\boldsymbol{P}_t\boldsymbol{X}_t)$ (or minimizes $-\operatorname{tr}(\boldsymbol{P}_t\boldsymbol{X}_t)$). As we shall see, for the MEG algorithm the loss and the gain versions (referred to as *Loss MEG* and *Gain MEG* throughout the paper) are different, whereas for GD, both versions collapse to the same algorithm.

We allow the algorithms to choose their projection matrix at random. Thus the algorithms maintain a mixture of projection matrices of rank k or $m = n-k$, respectively, as their parameter matrix $\boldsymbol{W}_t$. These mixtures are *generalized density matrices*, which are symmetric, positive definite matrices with eigenvalues upper bounded by 1, and trace equal to k or m, respectively [17]. We use $\mathcal{W}_k$ and $\mathcal{W}_m$ to denote the parameter space of all such matrices, respectively. Now we define the update of the Loss MEG and Gain MEG as follows:

$$\begin{aligned}
\text{Loss MEG:}\quad \boldsymbol{W}_{t+1} &= \operatorname*{argmin}_{\boldsymbol{W}\in\mathcal{W}_m} \left(\Delta(\boldsymbol{W},\boldsymbol{W}_t) + \eta\,\operatorname{tr}(\boldsymbol{W}\boldsymbol{X}_t)\right),\\
\text{Gain MEG:}\quad \boldsymbol{W}_{t+1} &= \operatorname*{argmin}_{\boldsymbol{W}\in\mathcal{W}_k} \left(\Delta(\boldsymbol{W},\boldsymbol{W}_t) - \eta\,\operatorname{tr}(\boldsymbol{W}\boldsymbol{X}_t)\right),
\end{aligned}$$

where $\Delta(\boldsymbol{W},\boldsymbol{W}') = \operatorname{tr}\big(\boldsymbol{W}(\log\boldsymbol{W} - \log\boldsymbol{W}'\big)$ is the quantum relative entropy, and $\eta > 0$ is a learning rate. Also $\operatorname{tr}(\boldsymbol{W}\boldsymbol{X}_t)$ for $\boldsymbol{W} \in \mathcal{W}_m$ is the expected loss in trial t of the random projection matrix of rank m drawn from the mixture summarized by $\boldsymbol{W} \in \mathcal{W}_m$. Similarly, $\operatorname{tr}(\boldsymbol{W}\boldsymbol{X}_t)$ for $\boldsymbol{W} \in \mathcal{W}_k$ is the expected gain in trial t of the random projection matrix of rank k drawn from the mixture summarized by $\boldsymbol{W} \in \mathcal{W}_k$, Note that the loss version of MEG corresponds to the original MEG algorithm developed in [17], where it was shown to have the following regret bound:

$$\text{regret}_{\text{Loss MEG}} \le \sqrt{2Bm\log\frac{n}{m}} + m\log\frac{n}{m}. \tag{2}$$

This bound holds for any sequence of instance matrices (dense as well as sparse) for which the total compression loss of the best rank k subspace does not exceed the loss budget B. With a similar analysis, the regret of Gain MEG can be bounded by

$$\text{regret}_{\text{Gain MEG}} \le \sqrt{2kG\log\frac{n}{k}}.$$

This bound holds for any sequence of instance matrices (dense as well as sparse) for which the total gain of the best rank k subspace does not exceed the gain budget G.

Budget dependent upper bounds on the regret always lead to time dependent regret bounds (as exploited in the proof of the below Theorem). Note that for PCA, the gain bound G is usually much larger than the loss bound B and therefore Gain MEG is not that useful for PCA. However as we shall see for dense instances, Gain MEG is sometimes better than Loss MEG. Incidentally, for lower bounds on the regret the implication is reversed in that time dependent regret bounds imply budget dependent regret bounds.

Theorem 1. *For sparse instance sequences of length T, the regret of the Loss MEG and Gain MEG algorithms is upper bounded by:*

$$\begin{aligned}\mathrm{regret}_{\text{Loss MEG}} &\leq m\sqrt{\frac{2T}{n}\log\frac{n}{m}} + m\log\frac{n}{m} \leq \sqrt{\frac{2kmT}{n}} + k \qquad (3)\\ \mathrm{regret}_{\text{Gain MEG}} &\leq \sqrt{2kT\log\frac{n}{k}}.\end{aligned}$$

Similarly, for dense instances, the following regret bounds hold:

$$\begin{aligned}\mathrm{regret}_{\text{Loss MEG}} &\leq m\sqrt{2T\log\frac{n}{m}} + m\log\frac{n}{m}\\ \mathrm{regret}_{\text{Gain MEG}} &\leq k\sqrt{2T\log\frac{n}{k}}.\end{aligned}$$

Proof. The best rank k subspace picks k eigendirections of the covariance matrix $\boldsymbol{C} = \sum_{t=1}^{T}\boldsymbol{X}_t$ with the largest eigenvalues. Hence the total compression loss equals the sum of the smallest m eigenvalues of $\boldsymbol{C}$. If $\lambda_1, \ldots, \lambda_n$ denote the eigenvalues of $\boldsymbol{C}$ then:

$$\sum_{i=1}^{n}\lambda_i = \mathrm{tr}(\boldsymbol{C}) = \sum_{t=1}^{T}\mathrm{tr}\left(\boldsymbol{X}_t\right) \leq \begin{cases} T & \text{for sparse instances,}\\ Tn & \text{for dense instances.}\end{cases}$$

This implies that the total compression loss of the comparator is upper bounded by $\frac{Tm}{n}$ and Tm, respectively. Plugging these values into (2) results in the bounds for Loss MEG. The second inequality in (3) follows from

$$m\log\frac{n}{m} = m\log\left(1+\frac{k}{m}\right) \leq k.$$

For the regret bounds of Gain MEG, we use the fact that G is upper bounded by T when instances are sparse and upper bounded by kT when the instances are dense. □

Note that in light of previous results for MEG, it is actually surprising that the regret bound (3) for Loss MEG with sparse instances is independent of the dimension n of the problem.

We now discuss in detail which version of MEG has a better regret bound for the dense instance case. We claim that this depends on the value of k, the dimension of the chosen subspace. Consider the ratio of the regret bounds of Loss MEG over Gain MEG. When $T \geq k$, then we can ignore the $m\log\frac{n}{m}$ term in the Loss MEG bound since this term is at most k. In this case the ratio becomes:

$$\Theta\left(\sqrt{\frac{k^2}{m^2}\frac{\ln\frac{n}{k}}{\ln\frac{n}{m}}}\right).$$

When $k \leq \frac{n}{2}$, $\ln \frac{n}{m} = \ln(1 + \frac{k}{m}) = \Theta(\frac{k}{m})$, and the ratio simplifies to $\Theta\left(\sqrt{\frac{\ln \frac{n}{k}}{\frac{n}{k}}}\right)$. Therefore, when $\frac{n}{k}$ grows, the regret bound for the Loss MEG is less than the regret bound for the Gain MEG. Similarly, when $k \geq \frac{n}{2}$, the ratio becomes $\Theta\left(\sqrt{\frac{\frac{n}{m}}{\ln \frac{n}{m}}}\right)$ and the regret bound for the Gain MEG is better in this case.

2.2 The GD Algorithm

In this section we consider the GD algorithm (see e.g. [6, 20]) which is motivated by the squared Frobenius norm (The loss and gain versions are the same in this case and we use the loss version below):

$$\boldsymbol{W}_{t+1} = \underset{\boldsymbol{W} \in \mathcal{W}_m}{\operatorname{argmin}} \ \left(\tfrac{1}{2}\|\boldsymbol{W} - \boldsymbol{W}_t\|_F^2 + \eta \operatorname{tr}(\boldsymbol{W}\boldsymbol{X}_t)\right).$$

This algorithm is simple and a time dependent regret bound has been proved for arbitrary convex losses [20, 14]. By applying this bound to PCA we obtain:

$$\text{regret}_{GD} \leq \left(\max_{1 \leq t \leq T} \|\boldsymbol{X}_t\|_F\right) \sqrt{T\|\boldsymbol{W}_1 - \boldsymbol{W}_c\|_F^2} = \left(\max_{1 \leq t \leq T} \|\boldsymbol{X}_t\|_F\right) \sqrt{\frac{mkT}{n}},$$

where $\boldsymbol{W}_c$ is any comparator in the parameter space $\mathcal{W}_m$. For sparse instances, $\|\boldsymbol{X}_t\|_F = \sqrt{\operatorname{tr}(\boldsymbol{X}_t\boldsymbol{X}_t^\top)} \leq 1$, the regret is bounded by $\sqrt{mkT/n} = \sqrt{(n-k)kT/n}$. This is the same as the regret bound for Loss MEG (3) except for an additional $\sqrt{2}$ factor bound for the Loss MEG. When instances are dense, $\|\boldsymbol{X}_t\|_F \leq n$, resulting in regret bound of $\sqrt{mkT}$. To see that this bound is worse than the MEG bound for dense instances, we can consider the ratio of the regret bound for GD over the regret bound for the appropriate version of MEG. It is easy to check that when $k \leq \frac{n}{2}$, the ratio is $\Theta(\sqrt{\frac{\frac{m}{k}}{\ln(\frac{2m}{k})}})$, and when $k \geq \frac{n}{2}$, the ratio is $\Theta\left(\sqrt{\frac{\frac{k}{m}}{\ln(\frac{2k}{m})}}\right)$. In both case, the regret bound for MEG is better by more than a constant factor.

We now conclude this section by investigating budget bounds for GD. Since GD achieves the same time horizon dependent regret bound as Loss MEG, we first conjectured that this is also the case for budget dependent regret bounds. However, this is not true: we will now show in remainder of this section a $k\sqrt{B}$ *lower bound* on the regret of GD for instance sequences with budget B. Since Loss MEG has regret at most $\sqrt{kB}$ in this case, this lower bound shows that GD is suboptimal by a factor of $\sqrt{k}$.

It suffices to prove the lower bound on a restricted data set. As already observed in [17], the PCA problem has the *m-set* problem as a special case. In this problem, all instance matrices are diagonal (i.e. the eigenvectors are the standard basis vectors) and therefore the algorithm can restrict itself to choosing subspaces that are subsets of the standard basis vectors. In other words, PCA collapses to learning a subset of $m = n - k$ experts. The algorithm chooses a subset of m out of n experts in each trial, the loss of all experts is given as a vector $\boldsymbol{\ell} \in [0,1]^n$, and the loss of a set is the total loss of all experts in the set. The algorithm maintains uncertainty over the m-sets by means of a weight vector $\boldsymbol{w} \in [0,1]^n$, such that $\sum_{i=1}^n w_i = m$. We denote the set of all such weight vectors as $\mathbb{S}_m$. The above GD algorithm for PCA specializes to the following algorithm for learning sets:

$$\begin{array}{ll} \text{Gradient Descent step:} & \hat{\boldsymbol{w}}_{t+1} = \boldsymbol{w}_t - \eta \boldsymbol{\ell}_t \\ \text{Projection step:} & \boldsymbol{w}_{t+1} = \operatorname{argmin}_{\boldsymbol{w} \in \mathbb{S}_m} \|\boldsymbol{w} - \hat{\boldsymbol{w}}_{t+1}\|^2. \end{array} \qquad (4)$$

The projection step is a convex optimization problem with inequality constraints and can be analyzed using its KKT conditions. We only describe the projection step in two cases needed for the lower bound. Let $\boldsymbol{w}_t = (w^1, \cdots, w^n)$ be the weight of GD at trial t. Our lower bound is for the sparse case. In the set problem this means that the loss vectors $\boldsymbol{\ell}_t$ are standard basis vectors. Let $\boldsymbol{\ell}_t = \boldsymbol{e}_{i_t}$. In the simplest case, the descent step decreases the weight of expert i by η and the projection step adds $\frac{\eta}{n}$ to all n weights so that the total weight remains m:

$$\widehat{\boldsymbol{w}}_{t+1} = (w^1, \ldots, w^i - \eta, \ldots, w^n),\, \boldsymbol{w}_{t+1} = \Big(w^1 + \frac{\eta}{n}, \ldots, w^i - \frac{(n-1)\eta}{n}, \ldots, w^n + \frac{\eta}{n}\Big). \quad (5)$$

Two problems may happen with the additive adjustment: $w_i - \eta + \frac{\eta}{n}$ might be negative or some of the weights $w_j + \frac{\eta}{n}$ might be larger than 1. The projection step is slightly more involved when this happens. In our lower bound, we only need the following additional case:

$$\boldsymbol{w}_t = \big(\underbrace{w^1, \ldots, w^{i-1}}_{\text{all} \leq 1 - \frac{\eta}{n}}, \underbrace{w^i}_{< \frac{n-1}{n}\eta}, \underbrace{1-\delta, \ldots, 1-\delta}_{\text{for } \delta < \frac{w_i}{n}}\big).$$

One can show that in this case the projection sets w^i to 0, it sets the $n-i$ weights of size $1-\delta$ to 1, and it adds $\frac{w^i - (n-i)\delta}{i-1}$ to the first $i-1$ weights which are not capped. That is, in this case the projections produces the following updated weight vector:

$$\boldsymbol{w}_{t+1} = \Big(w^1 + \frac{w^i - (n-i)\delta}{i-1}, \ldots, w^{i-1} + \frac{w^i - (n-i)\delta}{i-1}, \underbrace{0}_{\text{capped at } 0}, \underbrace{1, \ldots, 1}_{\text{capped at } 1}\Big). \quad (6)$$

Note that the total weight of the projected weight vector $\boldsymbol{w}_{t+1}$ is again m.

Now we are ready to give our regret lower bound for GD.

Theorem 2. *For any $k \leq n/2$ and any learning rate η, there is a sparse loss sequence for online PCA which has budget B and forces the GD algorithm* (4) *to incur regret at least $\Omega(k\sqrt{B})$.*

Proof. W.l.o.g., assume all the experts have the same initial weight $m/n = (n-k)/n \geq 1/2$. Call the first k experts *bad experts*, the $(k+1)$st expert the *faulty expert* and the last $m-1$ experts *good experts*. Let $\eta' = \min\{\eta, 1\}$.

The loss sequence consists of two phases. We will show that the algorithm suffers loss at least $\Omega(\frac{k}{\eta})$ in the first phase and essentially loss at least $B + \Omega(kB\eta)$ in the second phase. The optimum trade-off between these two term give the lower bound.

More precisely, in the first phase unit losses are given to bad experts and in each trial the algorithm suffers the current weight of the chosen bad expert. The phase ends when each of the good experts and the faulty expert have weights at least $1 - \frac{\eta'}{4m}$. To show that the algorithm suffers loss at least $\Omega(\frac{k}{\eta})$ in this phase, first notice that for a particular bad expert, its weight decreases by at most $\frac{n-1}{n}\eta$ when it receives a unit of loss and increases in all of the other trials (see (5) and (6)). So when it receives loss for the s-th time, its weight is lower bounded by

$$\max\left\{\frac{m}{n} - (s-1)\frac{n-1}{n}\eta, 0\right\} \geq \max\left\{\frac{1}{2} - (s-1)\eta, 0\right\} \tag{7}$$

The sum of (7) with $s = 1, 2, \ldots$ is a lower bound on the loss of the algorithm when this particular bad expert incurred a unit of loss. Note that (7) is the term of an arithmetic series that is capped from below by zero. One can show that as long as there is a constant gap between the first and last term of the summation, then the sum of these terms is at least $\Omega(\frac{1}{\eta})$. In our case, this gap is at least 1/4 since the first term (initial weight) is at least 1/2 and the last term, upper bounded by this bad expert's weight after phase one, is less than

$$\underbrace{m}_{\substack{\text{sum of the weights}\\ \text{of all experts}}} - \underbrace{m(1 - \frac{\eta'}{4m})}_{\substack{\text{sum of the weights}\\ \text{of faulty and good experts}}} \leq \frac{1}{4}.$$

Since we have k bad experts, we obtained a $\Omega(\frac{k}{\eta})$ lower bound on the loss of the algorithm during the first phase.

We now describe the second phase which lasts for B rounds, where each round consists of several trials. At the beginning of each round, the faulty expert receives one unit of loss. Its weight after the gradient descent step is at most $1-\eta$ and after the projection step it can be shown to be at most $\max\{1-\eta/2, 0\}$ (see (6)). Notice that after the first trial of each round, all good experts will have weight 1 since they are at least $1 - \frac{\eta'}{4m}$ after phase one. In the following trials of this round, unit losses are given to bad experts until the faulty expert recovers its weight to at least $1 - \frac{\eta'}{4m}$. Since all the weights of good experts are capped at 1, the re-balancing of weights only occurs between the faulty and the bad experts. This means that in each trial, the faulty expert can only recover at most $1/(k+1)$ of the loss incurred by the algorithm in this trial. Thus we can lower bound the loss of algorithm in a given round as follows:

$$\begin{aligned} 1 - \frac{\eta'}{4m} + (k+1)\Big(1 - \frac{\eta'}{4m} - \max\Big\{1 - \frac{\eta}{2}, 0\Big\}\Big) &\geq 1 - \frac{\eta'}{4m} + \frac{k+1}{4}\min\{\eta, 1\} \\ &= 1 + \Omega(k\eta'). \end{aligned}$$

After B such rounds, algorithm suffers loss at least $B + \Omega(kB\eta')$: When $\eta \geq 1$, this is $B + \Omega(kB)$ and when $\eta \leq 1$, then summing up the bounds of the two phases, gives an $\Omega(k/\eta) + B + \Omega(Bk\eta)$ lower bound on the loss of the algorithm. The latter is minimized at $\eta = \Theta(1/\sqrt{B})$ and for this choice of η, the algorithm suffers loss at least $B + \Omega(k\sqrt{B})$. The theorem now follows, since the best off-line m-set for the loss sequence consists of the faulty expert, which suffers total loss B, and all $m-1$ good experts, which incur no loss. □

3 Lower Bounds and Optimal Algorithms

In the previous section, we showed a lower bound on the regret of GD as a function of the budget B of the sequence. In this section we show regret bounds for any algorithm that solves the problem. In particular, we show regret lower bounds for online PCA and its generalization to the dense instance case. As argued in Section 2.2, it suffices to prove our lower bounds for the m-set problem which is the vector version of online PCA and its generalization to dense instances. We prove lower bound on the minimax regret, i.e. the minimum worst case regret any algorithm can achieve against the best set:

$$\min_{\substack{\text{alg. A with}\\ \boldsymbol{w}_t \in \mathbb{S}_m}} \quad \max_{\substack{\text{sparse/dense loss seq.}\\ \boldsymbol{\ell}_{1\ldots T} \text{ of length } T}} \quad \underbrace{R(A, \boldsymbol{\ell}_{1\ldots \mathrm{T}})}_{\substack{\text{loss of alg. A - loss of best set}\\ \text{on loss sequence } \boldsymbol{\ell}_{1\ldots T}}}$$

Recall that $\mathbb{S}_m$ were vectors in $[0,1]^n$ of total weight m that represent mixtures of sets of size m. Our lower bounds will match the uppers bounds on the regret of MEG (within constant factors) that we proved in the previous section for both online PCA and its generalization to dense instances. The lower bounds rely on the following probabilistic bounding technique for the minimax regret:

$$\begin{aligned}
\min_{\substack{\text{alg. A with}\\ \boldsymbol{w}_t \in \mathbb{S}_m}} \max_{\substack{\text{loss seq. } \boldsymbol{\ell}_{1\ldots T}\\ \text{of length T}}} R(A, \boldsymbol{\ell}_{1\ldots T}) &\geq \min_{\substack{\text{alg. A with}\\ \boldsymbol{w}_t \in \mathbb{S}_m}} \mathbb{E}_{\boldsymbol{\ell}_{1\ldots T} \sim \mathcal{P}} \left[\, R(A, \boldsymbol{\ell}_{1\ldots T}) \,\right] \\
&= \min_{\substack{\text{alg. A with}\\ \boldsymbol{w}_t \in \mathbb{S}_m}} \mathbb{E}_{\boldsymbol{\ell}_{1\ldots T} \sim \mathcal{P}} \left[\, L_A \,\right] - \mathbb{E}_{L \sim \mathcal{P}} \left[\, L_C \,\right], \quad (8)
\end{aligned}$$

where $\mathcal{P}$ is any distribution on loss sequences, and L_A and L_C are the cumulative losses of the algorithm and the best m-set, respectively.

3.1 Lower Bounds for PCA with Sparse Instances

Recall that for the vectorized version of PCA, the loss vectors $\boldsymbol{\ell}_t$ are restricted to be standard basis vectors. We start with the following technical lemma for two experts.

Lemma 1. *Let $p \in [0,1]$ be such that $p \leq 1/4$ and $Tp \geq 1/2$. Assume that in a two expert game, one of the experts is randomly chosen to suffer one unit of loss with probability $2p$ in each trial, and with probability $1-2p$ none of the experts suffers any loss. Then, after T independent trials,*

$$\mathbb{E}\left[\, \textit{Loss of the winner} \,\right] \leq \; Tp - c\sqrt{Tp},$$

for a constant c independent of T and p.

Due to the space limit, we omit the proof of this lemma. We are now ready to prove a lower bound for PCA. We first consider the case when $k \leq \frac{n}{2}$.

Theorem 3. *For $T \geq k$ and $k \leq \frac{n}{2}$, in the T trial online PCA problem with sparse instances, any online algorithm suffers worst case regret at least $\Omega(\sqrt{kT})$.*

Proof. At each trial, a randomly chosen expert out of the first $2k$ experts receives a unit of loss. To show an upper bound on the loss of the comparator, we group these $2k$ experts into k pairs and notice that the losses of each expert pair have a joint distribution as described in Lemma 1 with $p = \frac{1}{2k}$. Hence, the expected loss of the winner in each pair is at most $T/2k - c\sqrt{T/k}$, and the total expected loss for the k winners from all k pairs is upper bounded by $T/2 - c\sqrt{kT}$. Since the last $n-2k$ experts are loss-free, this is also an upper bound on the expected loss of the comparator, because the comparator will pick $n-2k$ loss-free experts and k best experts among the remaining $2k$ experts. On the other hand, since losses are generated independently between trials, any online algorithm suffers loss at least $T/2$. Taking the difference between two bounds concludes the proof. □

Noting that $m\sqrt{\ln(n/m)T/n} \leq m\sqrt{(k/m)T/n} \leq \sqrt{kT}$, the lower bound in Theorem 3 matches the upper bound of Loss MEG algorithm in Theorem 1 for the case $k \leq \frac{n}{2}$. For the case $k \geq \frac{n}{2}$, we need the following lemma, which is a generalization of Lemma 1 to n experts. In the proof we upper bound the minimum loss of the experts by the loss of the winner of a tournament among the experts. The tournament winner does not necessarily have the lowest loss. However as we shall see later, its expected loss is close enough to the expected loss of the best expert to make this bounding techniques useful for obtaining lower bounds on the regret.

Lemma 2. *Choose any n, S and T, such that $n = 2^S$ and S divides T. If the loss sequence of length T is generated from a distribution $\mathcal{P}$, such that:*

- *at each trial t, the distribution of losses on n experts is exchangeable,*
- *the distribution of losses is i.i.d. between trials,*

then,

$$\begin{aligned}&\mathbb{E}\left[\,\textit{Minimum loss of } n \textit{ experts in } T \textit{ trials}\,\right]\\ &\qquad\qquad \leq S\,\mathbb{E}\left[\,\textit{Loss of the winner among two experts in } T/S \textit{ trials}\,\right].\end{aligned}$$

Proof. Due to the space limit, we only sketch the proof. The key idea in the proof is to upper bound the loss of the best expert by the loss of the expert that wins a *tournament* with S rounds. In each round, the experts are paired and compared against their partners, using the sum of their losses in the next T/S consecutive trails. The winner of each local pair competition survives to the next round. The winners are again paired and winners among those continue with the tournament until one expert is left from the original $n = 2^S$ experts. We call this expert the tournament winner. The expected loss of the best expert is upper bounded by the expected loss of the tournament winner, which curiously enough equals the number of rounds times the expected loss of the two expert case:

$$\begin{aligned}
&\mathbb{E}\,[\text{ Minimum loss of all } n \text{ experts in all } T \text{ trials }] \\
&\leq \quad \mathbb{E}\left[\begin{array}{c}\text{Loss of the } \textit{tournament} \text{ winner in the} \\ \text{S rounds tournament among the } n = 2^S \text{ experts}\end{array}\right] \\
&\overset{\text{since expectations sum}}{=} \sum_{\text{rounds } 1 \leq s \leq S} \mathbb{E}\,[\text{ Loss of } \textit{tournament} \text{ winner in round } s\,] \\
&\overset{\text{i.i.d. loss btw. trials}}{=} S\,\mathbb{E}\,[\text{ Loss of the } \textit{tournament} \text{ winner in one round }] \\
&\overset{\text{def. of local tournament}}{=} S\,\mathbb{E}\,[\text{ Loss of winner among two experts in } T/S \text{ trials }]\,.
\end{aligned}$$

The last equality would be trivial if the distribution $\mathcal{P}$ on the sequence of loss vectors was i.i.d. between experts. An additional argument is needed to show the equality with the weaker assumption of exchangeability. □

We now consider the uncommon case when $k \geq \frac{n}{2}$:

Theorem 4. *For $T \geq n \log_2(n/m)$ and $k \geq \frac{n}{2}$, in the T trial online PCA problem with sparse instances, any online algorithm suffers worst case regret at least $\Omega(m\sqrt{\ln(n/m)T/n})$.*

Proof. At each trial, a randomly chosen expert out of n experts receives a unit of loss. To show an upper bound on the loss of the comparator, we partition the n experts into m groups and notice that the losses of the n/m experts in each group are exchangeable. By applying Lemma 2 to each group, we obtain:

$$\begin{aligned}
&\mathbb{E}\,[\text{ Loss of the winner in a given group in } T \text{ trials }] \\
&\quad \leq \log_2\!\left(\frac{n}{m}\right)\ \mathbb{E}\left[\text{Loss of winner of two experts in } \frac{T}{\log_2(\frac{n}{m})} \text{ trials}\right]. \qquad (9)
\end{aligned}$$

We bound the last expectation that deals with the 2 experts case by applying Lemma 1 with $p = 1/n$ and $T/\log_2(n/m)$. This lets us replace the expectation by the upper bound

$$\frac{T}{\log_2(n/m)n} - \sqrt{\frac{T}{\log_2(n/m)n}}.$$

Plugging this into (9) gives a $T/n - \sqrt{\log_2(n/m)T/n}$ upper bound on the expected loss of a winner in a given group. We upper bound the expected loss of the comparator by the total loss of m winners from the m groups, which in expectation is at most $mT/n - m\sqrt{\log_2(n/m)T/n}$.

Finally the loss of the algorithm is bounded as follows. Since every expert suffers loss $1/n$ in expectation at each trail and losses are i.i.d. between trials, any online algorithm suffers loss at least mT/n. This concludes the proof. □

Combining this lower bound with the upper bounds proved in Section 2.1 on the regret of Loss MEG for the sparse instance case results in the following corollary:

Corollary 1. *For online PCA with sparse instances, the regret* $\Theta(m\sqrt{\frac{T\ln(n/m)}{n}})$ *of Loss MEG is within a constant factor of the minimax regret.*

3.2 Lower Bound for PCA with Dense Instances

The following lower bound again employs Lemma 2 which was proved using a tournament.

Theorem 5. *For* $T \geq \log_2(\frac{d}{\min\{k,m\}})$*, in the* T *trial online PCA problem with dense instances, any online algorithm suffers worst case regret at least*

$$\Omega(m\sqrt{\ln(n/m)T}) \text{ when } m \leq \frac{n}{2} \quad \text{or} \quad \Omega(k\sqrt{\ln(n/k)T}) \text{ when } m \geq \frac{n}{2}.$$

Proof. The proof is similar to the proof of Theorem 4, except that at each trial, unit losses are independently given to all the experts with probability $\frac{1}{2}$. For such a distribution over losses, any algorithm suffers cumulative loss at least $mT/2$ in expectation. We now upper bound the comparator's expected loss by distinguishing two cases: When $m \leq n/2$, we group the experts into m groups and upper bound the comparator loss using the m winners, one from each of the groups. This gives an $mT/2 - cm\sqrt{\ln(n/m)T}$ upper bound, and results in a $\Omega(m\sqrt{\ln(n/m)T})$ lower bound for the regret.

When $m \geq n/2$, we group the experts into k groups and consider a *loser* out of each group, i.e. the expert which suffers the *largest* loss in each group. One can flip around the content of Lemma 2 to show that the loser's loss in a group of n/k experts is lower bounded by $T/2 + c\sqrt{\ln(n/k)T}$, so that the expected loss of all k losers is lower bounded by $kT/2 + ck\sqrt{\ln(n/k)T}$. The claimed regret bounds now follows from the fact that the cumulative loss of the comparator is upper bounded by the total expected loss of all experts ($nT/2$) minus the total expected loss of all k losers. This completes the sketch proof. □

Combining this lower bound with the upper bounds on the regret of Loss MEG and Gain MEG for dense instance case proved in Section 2 gives a following corollary, which basically states that the Loss MEG is optimal for $m \leq \frac{n}{2}$ while the Gain MEG is optimal for $m \geq \frac{n}{2}$.

Corollary 2. *Consider online PCA with dense instances.*

- *When* $m \leq \frac{n}{2}$*, the regret* $\Theta(m\sqrt{T\log\frac{n}{m}})$ *of Loss MEG is within a constant factor of the minimax regret.*
- *When* $m \geq \frac{n}{2}$*, the regret* $\Theta(k\sqrt{T\log\frac{n}{k}})$ *of Gain MEG is within a constant factor of the minimax regret.*

Minimax regret for sequences with a budget. One can also show the minimax regret for a prediction game in which the budget B is fixed, rather than the time horizon T. In this case, no matter if the instances are dense or sparse, we get the following corollary establishing the optimality of Loss MEG:

Corollary 3. *Let $B \geq m \log_2 \frac{n}{m}$. For online PCA with both sparse and dense instances, the regret $\Theta(\sqrt{m \ln(n/m) B})$ of Loss MEG is within a constant factor the minimax budget regret.*

Proof. Since the instance matrices have eigenvalues bounded by one, the minimax regret is upper bounded by $O(\sqrt{m \ln(n/m) B})$, the regret bound of the Loss MEG algorithm given in (2), Section 2.1. On the other hand, we now reason that for any algorithm we can construct a sparse instance sequence of budget B incurring regret at least $\Omega(\sqrt{m \ln(n/m) B})$. This instance sequence is constructed via Theorem 3 and Theorem 4: For any algorithm, these theorems provide a sparse instance sequence of length T with regret at least $\Omega(m\sqrt{\frac{T \ln(n/m)}{n}})$. We apply these theorems with $T = \frac{n}{m} B \geq n \log_2 \frac{n}{m}$. Since the produced sequence is sparse and has length $\frac{n}{m} B$, its budget is at most B. Finally plugging $T = \frac{n}{m} B$ into the regret bounds guaranteed by the theorems results in the regret $\Omega(\sqrt{m \ln(n/m) B})$. □

4 Conclusion

We showed in this paper that GD is non-optimal for various problems. However, our lower bounds are for the Mirror Descent version of GD that trades off the loss on the last example with a divergence to the last capped parameter matrix. There is an alternate algorithm: the Incremental Off-line [3] or Follow the Perturbed Leader algorithm [10] that in its motivation trades off the total loss on all examples against the divergence to the initial distribution. Note that both versions follow their update with a projection into the parameter space. We conjecture that the Incremental Off-line version of GD is strictly better than the commonly studied Mirror Descent version. The advantage of processing all examples versus just the last one has now shown up in a number of different contexts: in Boosting it led to better algorithms [18] and it also was crucial for obtaining a kernelizable online PCA algorithm [13]. When there are only equality constraints and the loss is linear, then the two versions of the algorithm are provably the same (See e.g. [8]). However when there are inequality constraints that are not enforced by the divergence, then the projection steps of the Mirror Descent version of the algorithm "forgets" information about the past examples whenever the algorithm runs into the boundaries of the inequality constraints.

More concretely, we conjecture that the Incremental Off-line version of GD has the optimal budget regret bound for online PCA (as Mirror Descent MEG does which enforces the non-negativity constraints with its divergence). If this conjecture is true, then this would be the first case where there is a provable gap between processing just the last versus all past examples. If the conjecture is false, then Mirror Descent MEG is truly better than both versions of GD. Both outcomes would be in important step forward in our understanding of online algorithms. Note that our $k\sqrt{B}$ lower bound for GD specifically exploits the forgetting effect and consequently only applies to the Mirror Descent version of the GD algorithm.

References

[1] Abernethy, J., Agarwal, A., Bartlett, P.L., Rakhlin, A.: A stochastic view of optimal regret through minimax duality. In: COLT (2009)

[2] Abernethy, J., Warmuth, M.K., Yellin, J.: When random play is optimal against an adversary. In: COLT, pp. 437–446 (2008)

[3] Azoury, K.S., Warmuth, M.K.: Relative loss bounds for on-line density estimation with the exponential family of distributions. Machine Learning 43(3), 211–246 (2001)

[4] Cesa-Bianchi, N., Freund, Y., Haussler, D., Helmbold, D.P., Schapire, R.E., Warmuth, M.K.: How to use expert advice. J. ACM 44(3), 427–485 (1997)

[5] Cesa-Bianchi, N., Long, P.M., Warmuth, M.K.: Worst-case quadratic loss bounds for prediction using linear functions and gradient descent. IEEE Trans. Neural Netw. Learning Syst. 7(3), 604–619 (1996)

[6] Cesa-Bianchi, N., Long, P.M., Warmuth, M.K.: Worst-case quadratic loss bounds for prediction using linear functions and gradient descent. IEEE Trans. Neural Netw. Learning Syst. 7(3), 604–619 (1996)

[7] Cesa-Bianchi, N., Lugosi, G.: Prediction, learning, and games. Cambridge University Press (2006)

[8] Helmbold, D.P., Warmuth, M.K.: Learning permutations with exponential weights. Journal of Machine Learning Research 10, 1705–1736 (2009)

[9] Herbster, M., Warmuth, M.K.: Tracking the best linear predictor. Journal of Machine Learning Research 1, 281–309 (2001)

[10] Kalai, A.T., Vempala, S.: Efficient algorithms for online decision problems. J. Comput. Syst. Sci. 71(3), 291–307 (2005)

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

[12] Koolen, W.M., Warmuth, M.K., Kivinen, J.: Hedging structured concepts. In: COLT, pp. 93–105 (2010)

[13] Kuzmin, D., Warmuth, M.K.: Online kernel PCA with entropic matrix updates. In: ICML, pp. 465–472 (2007)

[14] Srebro, N., Sridharan, K., Tewari, A.: On the universality of online mirror descent. In: NIPS, pp. 2645–2653 (2011)

[15] Sridharan, K., Tewari, A.: Convex games in banach spaces. In: Proceedings of the 23nd Annual Conference on Learning Theory (COLT) (2010)

[16] Tsuda, K., Rätsch, G., Warmuth, M.K.: Matrix exponentiated gradient updates for on-line learning and Bregman projections. Journal of Machine Learning Research 6, 995–1018 (2005)

[17] Warmuth, M.K., Kuzmin, D.: Randomized online PCA algorithms with regret bounds that are logarithmic in the dimension. Journal of Machine Learning Research 9, 2287–2320 (2008)

[18] Warmuth, M.K., Liao, J., Rätsch, G.: Totally corrective boosting algorithms that maximize the margin. In: ICML, pp. 1001–1008 (2006)

[19] Warmuth, M.K., Vishwanathan, S.V.N.: Leaving the span. In: Auer, P., Meir, R. (eds.) COLT 2005. LNCS (LNAI), vol. 3559, pp. 366–381. Springer, Heidelberg (2005)

[20] Zinkevich, M.: Online convex programming and generalized infinitesimal gradient ascent. In: Fawcett, T., Mishra, N. (eds.) ICML, pp. 928–936. AAAI Press (2003)