Online Linear Regression using Burg Entropy

Prateek Jain, Brian Kulis, and Inderjit Dhillon

Technical Report TR-07-08
University of Texas at Austin
Austin, TX 78712

February 14, 2007

Abstract

We consider the problem of online prediction with a linear model. In contrast to existing work in online regression, which regularizes based on squared loss or KL-divergence, we regularize using divergences arising from the Burg entropy. We demonstrate regret bounds for our resulting online gradient-descent algorithm; to our knowledge, these are the first online bounds involving Burg entropy. We extend this analysis to the matrix case, where our algorithm employs LogDet-based regularization, and discuss an application to online metric learning. We demonstrate empirically that using Burg entropy for regularization is useful in the presence of noisy data.

1 Introduction

Recently, online learning has received significant attention, and several recent results have demonstrated strong provable online bounds. In the online linear regression problem, considered in this paper, the algorithm receives an instance $\boldsymbol{x}_t$ ($\boldsymbol{x}_t \in \Re_+^m$) at time t and predicts $\hat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$; $\boldsymbol{w}_t$ is the current model used by the algorithm. We denote w_t^i as the i-th component of the model at the t-th step. The error incurred at step t is $l_t(\boldsymbol{w}_t) = (y_t - \hat{y}_t)^2$, where y_t is the true/target distance, and the goal is to minimize the total loss over all time steps $\sum_t l_t(\boldsymbol{w}_t)$.

If there is no correlation between the input $\boldsymbol{x}_t$ and the output y_t, then the cumulative loss of the algorithm may be unbounded. Hence, online algorithms are traditionally compared to the performance of the best possible offline solution, where all the input and output instances are provided beforehand. The goal is to prove a relative loss bound on how the online algorithm compares to the optimal offline algorithm. Given a T-trial sequence $S = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_T, y_T)\}$, the optimal offline solution is given by

$$\begin{aligned} \boldsymbol{u} = \operatorname*{argmin}_{\boldsymbol{w}} \quad & \sum_{t=1}^{T} l_t(\boldsymbol{w}) \\ \text{s.t.} \quad & w^i > 0 \ \ \forall\ i \end{aligned}$$

A typical approach to solving the online linear regression problem uses the standard gradient descent method with regularization for minimizing the loss at each step. Specifically, the algorithm tries to minimize the following function at each step:

$$U(\boldsymbol{w}) = \overbrace{D(\boldsymbol{w}, \boldsymbol{w}_t)}^{\text{Regularization Term}} + \eta_t \overbrace{L(y_t, \boldsymbol{w} \cdot \boldsymbol{x}_t)}^{\text{Loss Term}}, \tag{1}$$

where η_t is learning rate at t-th step and $d(\boldsymbol{w}, \boldsymbol{w}_t)$ is some distance measure between the new weight vector $\boldsymbol{w}$ and the current weight vector $\boldsymbol{w}_t$.

In the above objective function, the regularization term favors weight vectors which are "close" to the current model $\boldsymbol{w}_t$, representing a tendency for conservativeness. On the other hand, the loss term is minimized

when $\boldsymbol{w}$ is updated to exactly satify $y_t = \hat{y}_t$; hence the loss term has a tendency to satisfy target outputs for recent examples. The tradeoff between these two quantities is handled by the learning rate η_t, an important parameter in online learning problems.

The performance of any method following this approach will depend heavily on the choice of the regularization term and the loss function. Kivinen et. al[6, 8] show that the performance of any regularization term in turn depends heavily on the problem domain. For example, for sparse inputs, relative entropy as a regularization term often works better than squared Euclidean distance. For dense inputs, relative entropy's performance is generally very poor compared to that of squared Euclidean distance.

In this paper, we study the performance of online linear regression when the Burg entropy is considered as the regularization term:

$$D_{Burg}(\boldsymbol{w}_{t+1}, \boldsymbol{w}_t) = \sum_{i=1}^{N} \left(\frac{w_{t+1}^i}{w_t^i} - \log\left(\frac{w_{t+1}^i}{w_t^i} \right) \right).$$

The relative Burg entropy (or Itakura-Saito distance) is a special case of a *Bregman divergence*, and such measures have been shown to be useful distance measures for various machine learning tasks [1] (relative entropy and squared Euclidean distance are other special cases of Bregman divergences). The Burg entropy in particular has recently been shown to be useful for matrix approximations and kernel learning [7]. Our study of Burg entropy is also motivated by the fact that online information-theoretic metric learning reduces to online linear regression problem with the LogDet divergence—the natural generalization of the Burg entropy as regularization term [3]. The results in this paper complement those of our work on information-theoretic metric learning.

We present an algorithm for using Burg relative entropy for regularization and prove an associated regret bound. We also generalize the above approach to the matrix case when the input matrices are rank-one; this is precisely the case needed for information-theoretic metric learning. In section 5, we empirically show how the Burg entropy performs compared to other standard regularization terms.

2 The Vector Case

In this study, we select the Bregman divergence corresponding to the Burg entropy as our regularization term for (1):

$$D_{Burg}(\boldsymbol{w}_{t+1}, \boldsymbol{w}_t) = \sum_{i=1}^{N} \left(\frac{w_{t+1}^i}{w_t^i} - \log\left(\frac{w_{t+1}^i}{w_t^i} \right) \right).$$

We derive Algorithm 1 from standard gradient descent; in particular, if we differentiate $U(\boldsymbol{w})$, we obtain the update given as (2) in Algorithm 1.

Algorithm 1 Burg Gradient (BG) Algorithm

1: **Initialize:** Set $\eta_t = \frac{1}{4mR^2}$ and $w_0^i = \frac{1}{N}$, $\forall\ i$.
2: **Prediction:** Upon receiving the t-th instance $\boldsymbol{x}_t$, give the prediction $\hat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$.
3: **Update:** Upon receiving the t-th outcome y_t, update the weights according to the rule,

$$w_{t+1}^i = ((w_t^i)^{-1} + 2\eta_t(\hat{y}_t - y_t)x_t^i)^{-1}, \tag{2}$$

where

$$\eta_t = \begin{cases} n_{t-1} & \text{if } \hat{y}_t - y_t > 0 \\ \min\left(n_{t-1}, \frac{1}{-2(\hat{y}_t - y_t)x_t^i} \left(\frac{1}{w_t^i} - 1 \right) \right) & \text{if } \hat{y}_t - y_t < 0. \end{cases} \tag{3}$$

Note that the divergence D_{Burg} is only defined over positive-valued weight vectors. To maintain positivity, we use an adaptive learning rate η_t, which guarantees that the resulting vector $\boldsymbol{w}$ is positive. Assuming $\eta_{t-1} > 0$, then $\eta_t > 0$ since η_t is the minimum of two positive quantities. Also, if $\hat{y}_t - y_t > 0$ then clearly

$\boldsymbol{w}_{t+1} > 0$ (assuming $\boldsymbol{w}_t > 0$). On the other hand, if $\hat{y}_t - y_t < 0$, then since $\eta_t \leq \frac{1}{-2(\hat{y}_t - y_t)x_t^i}\left(\frac{1}{w_t^i} - 1\right)$, we can conclude that $\boldsymbol{w}_{t+1} > 0$. Hence, we maintain positivity.

2.1 Regret Bounds

Now we demonstrate regret bounds for the proposed algorithm. Let the total loss of Algorithm 1 be $Loss_{BG} = \sum_t l_t(\mathbf{w}_t)$, and let $Loss_{\mathbf{u}} = \sum_t l_t(\mathbf{u})$ be the loss of the best offline solution. First we bound the loss of the algorithm at a single trial in terms of the loss of a comparison vector $\boldsymbol{u}$ at that trial and the progress of the algorithm towards $\boldsymbol{u}$.

Lemma 1.

$$a_t(\hat{y}_t - y_t)^2 - b_t(\boldsymbol{u} \cdot \boldsymbol{x}_t - y_t)^2 \leq D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_t) - D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_{t+1}),$$

where a_t and b_t are positive constants and $\boldsymbol{u}$ is the optimal offline solution.

Proof.

$$\begin{aligned} D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_t) - D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_{t+1}) &= \sum_{i=1}^{N}\left(u^i\left(\frac{1}{w_t^i} - \frac{1}{w_{t+1}^i}\right) + \ln\frac{w_t^i}{w_{t+1}^i}\right) \\ &= -2\eta_t(\hat{y}_t - y_t)\sum_{i=1}^{N} u^i x_t^i + \sum_{i=1}^{N}\ln(1 + 2\eta_t(\hat{y}_t - y_t)w_t^i x_t^i) \\ &= -2\eta_t(\hat{y}_t - y_t)\boldsymbol{u} \cdot \boldsymbol{x}_t + \sum_{i=1}^{N}\ln(1 + 2\eta_t(\hat{y}_t - y_t)w_t^i x_t^i). \end{aligned}$$

Consider the following two cases:

1. $\hat{y}_t - y_t \geq 0$: Since $w_t^i > 0$ and $x_t^i > 0$, it follows that $2\eta_t(\hat{y}_t - y_t)w_t^i x_t^i \geq 0$.
2. $\hat{y}_t - y_t < 0$: Since $\eta_t < \frac{1}{-2(\hat{y}_t - y_t)x_t^i}\left(\frac{1}{w_t^i} - 1\right)$, it follows that $2\eta_t(\hat{y}_t - y_t)w_t^i x_t^i > \frac{1}{w_t^i} - 1$. Now, $w_t^i > 0$ and so $2\eta_t(\hat{y}_t - y_t)w_t^i x_t^i > -1$.

Thus, in either case, $2\eta_t(\hat{y}_t - y_t)w_t^i x_t^i > -1$. We can therefore apply the inequality

$$\ln(1 + z) \geq z - z^2/2,$$

which holds for all $z > -1$. Hence

$$D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_t) - D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_{t+1}) \geq -2\eta_t(\hat{y}_t - y_t)\boldsymbol{u} \cdot \boldsymbol{x}_t + \sum_{i=1}^{N}\left(2\eta_t(\hat{y}_t - y_t)w_t^i x_t^i - 2\eta_t^2(\hat{y}_t - y_t)^2(w_t^i x_t^i)^2\right).$$

Assume $x_t^i \leq R$ and $\sum_{i=1}^{N} w_t^i \leq m$; the above inequality simplifies to

$$D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_t) - D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_{t+1}) \geq -2\eta_t(\hat{y}_t - y_t)\boldsymbol{u} \cdot \boldsymbol{x}_t + 2\eta_t(\hat{y}_t - y_t)\hat{y}_t - 2\eta_t^2(\hat{y}_t - y_t)^2 mR^2.$$

Let $r = \boldsymbol{u} \cdot \boldsymbol{x}_t$. Proving the lemma amounts to showing that

$$\begin{aligned} D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_t) - D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_{t+1}) &\geq -2\eta_t(\hat{y}_t - y_t)r + 2\eta_t(\hat{y}_t - y_t)\hat{y}_t - 2\eta_t^2(\hat{y}_t - y_t)^2 mR^2 \quad (4) \\ &\geq a_t(\hat{y}_t - y_t)^2 - b_t(r - y_t)^2, \quad (5) \end{aligned}$$

for some positive constants a and b. Consider the function

$$F(r) = a_t(\hat{y}_t - y_t)^2 - b_t(r - y_t)^2 + 2\eta_t(\hat{y}_t - y_t)r - 2\eta_t(\hat{y}_t - y_t)\hat{y}_t + 2\eta_t^2(\hat{y}_t - y_t)^2 mR^2.$$

Equation 5 is equivalent to $F(r) \leq 0, \forall r$. It can be easily seen that $F(r)$ is maximized when $r = y_t + \frac{\eta_t}{b_t}(\hat{y}_t - y_t)$. Substituting for r in $F(r)$ and simplifying, we get:

$$a_t(\hat{y}_t - y_t)^2 + \frac{\eta_t^2}{b_t}(\hat{y}_t - y_t)^2 - 2\eta_t(\hat{y}_t - y_t)^2 + 2\eta_t^2(\hat{y}_t - y_t)^2 mR^2.$$

Hence, we need to prove that

$$0 \geq a_t(\hat{y}_t - y_t)^2 + \frac{\eta_t^2}{b_t}(\hat{y}_t - y_t)^2 - 2\eta_t(\hat{y}_t - y_t)^2 + 2\eta_t^2(\hat{y}_t - y_t)^2 mR^2.$$

Since $(\hat{y}_t - y_t)^2 \geq 0$, this amounts to showing

$$0 \geq a_t b_t + \eta_t^2 - 2\eta_t b_t + 2\eta_t^2 mR^2 b_t = Q(\eta_t).$$

Now, $Q(\eta_t)$ is minimized for $\eta_t = \frac{b_t}{1+2mR^2 b_t}$, which implies $b_t = \frac{\eta_t}{1-2\eta_t mR^2}$. Since, $\eta_t \leq \eta_0 = \frac{1}{4mR^2}$, we know that $b_t > 0$. $Q(\eta_t) < 0$ iff $a_t \leq \frac{b_t}{1+2mR^2 b_t} = \eta_t$. Hence, for $0 \leq a_t \leq \eta_t$ and $b_t = \frac{\eta_t}{1-2\eta_t mR^2}$, Lemma 1 holds. □

Using Lemma 1, we can sum the loss over all time steps to get an overall bound on the loss of the Burg entropy-based online learning algorithm.

Theorem 1.

$$L_{BG} \leq \frac{1}{2\eta_T mR^2} L_{\boldsymbol{u}} + \frac{1}{\eta_T} D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_0),$$

where L_{BG} is the loss incurred by the Burg Gradient descent algorithm and $L_{\boldsymbol{u}}$ is the loss incurred by the optimal offline algorithm.

Proof. By Lemma 1, for each trial t,

$$\eta_t(\hat{y}_t - y_t)^2 - \frac{\eta_t}{1 - 2\eta_t mR^2}(\boldsymbol{u} \cdot \boldsymbol{x}_t - y_t)^2 \leq D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_t) - D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_{t+1}).$$

Hence, adding over all trials we get

$$\sum_{t=1}^{T} \eta_t(\hat{y}_t - y_t)^2 \leq \sum_{t=1}^{T} \frac{\eta_t}{1 - 2\eta_t mR^2}(\boldsymbol{u} \cdot \boldsymbol{x}_t - y_t)^2 + D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_0) - D_{Burg}(\boldsymbol{u}, \boldsymbol{w}_T)$$

Thus, since $\eta_t \leq \eta_T$ for all t, we have:

$$L_{BG} \leq \frac{1}{2\eta_T mR^2} L_{\boldsymbol{u}} + \frac{1}{\eta_T} D_{Burg}(\boldsymbol{u}, \mathbf{w}_0).$$

□

Note that similar bounds can be obtained using the framework presented in Gordon[5], but the resulting bound is substantially weaker than our bound and the proof is much more involved.

3 The Matrix Case

In the matrix case, at the t-th step, the algorithm receives an instance X_t ($X_t \in \Re_+^{m \times m}$) and predicts $\hat{y}_t = \mathrm{tr}(W_t X_t)$. W_t is the current model used by the algorithm. The error incurred at the t-th step is $l_t(W_t) = (y_t - \hat{y}_t)^2$.

Similarly to the vector case, for a given T-trial sequence $S = \{(X_1, y_1), (X_2, y_2), \ldots, (X_T, y_T)\}$, the optimal offline solution is given by

$$\begin{aligned} U = \operatorname*{argmin}_{W} \quad & \sum_{t=1}^{T} l_t(W) \\ \text{s.t.} \quad & W \succ 0. \end{aligned}$$

As a result, the goal here is to compare the loss of the online algorithm with the optimal offline solution, just as in the vector case. For the matrix case, we consider the LogDet divergence as the regularization in the objective. The LogDet divergence is the generalization of D_{Burg} to matrices, given by

$$D_{ld}(W, W_t) = \mathrm{tr}(W W_t^{-1}) - \log\det(W W_t^{-1}).$$

3.1 Rank-One Input

We consider the special case where each input X_t is a rank-1 matrix. This assumption holds in the case of information-theoretic metric learning (discussed later), where the inputs are rank-one constraints. We further assume that input is symmetric; thus, the input matrix X_t can be written as

$$X_t = \boldsymbol{z}_t \boldsymbol{z}_t^T.$$

Using LogDet as the regularization term, we derive Algorithm 2 analogously as in the vector case. We set η_t carefully at each step so that the algorithm ensures positive definiteness of the weight matrix W_t. Lemma 2 proves this invariant.

Algorithm 2 Matrix Burg Gradient (BG) Algorithm

1: **Initialize:** Set $\eta_t = \frac{1}{4R^2}$ and $W_0 = \frac{1}{N} I$, $\forall\ i$.
2: **Prediction:** Upon receiving the t-th instance X_t, give the prediction $\hat{y}_t = \mathrm{tr}(W_t X_t)$.
3: **Update:** Upon receiving the t-th outcome y_t, update the weights according to the rule

$$W_{t+1} = \left(W_t^{-1} + 2\eta_t(\hat{y}_t - y_t)X_t\right)^{-1}, \tag{6}$$

where

$$\eta_t = \begin{cases} n_{t-1} & \text{if } \hat{y}_t - y_t > 0 \\ \min\left(n_{t-1}, \frac{1}{2(y_t - \hat{y}_t)}\left(\frac{1}{\mathrm{tr}((I + (W_t^{-1} - I)^{-1})W_t X_t)}\right)\right) & \text{if } \hat{y}_t - y_t < 0. \end{cases} \tag{7}$$

Lemma 2. *Algorithm 2 preserves the invariant* $0 \prec W_t \prec I$.

Proof. We prove this by induction over t. The base case holds trivially for $N > 1$. By the induction hypothesis, $0 \prec W_t \prec I$. Assume $X_t = \boldsymbol{z}\boldsymbol{z}^T$ as X_t is a symmetric positive definite rank-one matrix. Thus, using the update rule and the Sherman Morrison-Woodbury formula[4]:

$$W_{t+1} = W_t - \frac{2\eta_t(\hat{y}_t - y_t)W_t \boldsymbol{z}\boldsymbol{z}^T W_t}{1 + 2\eta_t(\hat{y}_t - y_t)\boldsymbol{z}^T W_t \boldsymbol{z}}. \tag{8}$$

We break up the proof into the following two cases:

Case 1. $\hat{y}_t - y_t \geq 0$: Consider the update $W_{t+1} = \left(W_t^{-1} + 2\eta_t(\hat{y}_t - y_t)X_t\right)^{-1}$. Since $\eta_t > 0$ and $\hat{y}_t - y_t > 0$, then $2\eta_t(\hat{y}_t - y_t)X_t \succ 0$. Now, W_{t+1}^{-1} is sum of two symmetric positive definite matrices, and so W_{t+1} is positive definite.

Using Equation 8,

$$I - W_{t+1} = I - W_t + \frac{2\eta_t(\hat{y}_t - y_t)W_t \boldsymbol{z}\boldsymbol{z}^T W_t}{1 + 2\eta_t(\hat{y}_t - y_t)\boldsymbol{z}^T W_t \boldsymbol{z}}.$$

Since, W_t is symmetric,

$$W_t \boldsymbol{z}\boldsymbol{z}^T W_t = W_t \boldsymbol{z}\boldsymbol{z}^T W_t^T = \boldsymbol{v}\boldsymbol{v}^T \succ 0,$$

where $\boldsymbol{v} = W\boldsymbol{z}$. Also, $\eta_t > 0$ and $\hat{y}_t - y_t > 0$, and thus $\frac{2\eta_t(\hat{y}_t - y_t)W_t \boldsymbol{z}\boldsymbol{z}^T W_t}{1 + 2\eta_t(\hat{y}_t - y_t)\boldsymbol{z}^T W_t \boldsymbol{z}} \succ 0$. Furthermore, $W_t \prec I$, and so $I - W_t \succ 0$. Thus, $I - W_{t+1}$ is sum of two symmetric positive definite matrices, implying $I - W_{t+1} \succ 0$. We conclude that $W_{t+1} \prec I$.

Case 2. $\hat{y}_t - y_t < 0$: By Equation 8,

$$\begin{aligned} W_{t+1} &= W_t - \frac{2\eta_t(\hat{y}_t - y_t)W_t \boldsymbol{z}\boldsymbol{z}^T W_t}{1 + 2\eta_t(\hat{y}_t - y_t)\boldsymbol{z}^T W_t \boldsymbol{z}} \\ &= W_t + \frac{2\eta_t(y_t - \hat{y}_t)W_t \boldsymbol{z}\boldsymbol{z}^T W_t}{1 - 2\eta_t(y_t - \hat{y}_t)\boldsymbol{z}^T W_t \boldsymbol{z}}. \end{aligned}$$

The update rule guarantees that $\eta_t < \frac{1}{2(y_t - \hat{y}_t)\mathrm{tr}(W_t X_t)}$. Since

$$\mathrm{tr}(W_t X_t) = \mathrm{tr}(W_t \boldsymbol{z}\boldsymbol{z}^T) = \mathrm{tr}(\boldsymbol{z}^T W_t \boldsymbol{z}) = \boldsymbol{z}^T W_t \boldsymbol{z}, \tag{9}$$

we know that $1 - 2\eta_t(y_t - \hat{y}_t) z^T W_t z > 0$, implying that W_{t+1} is sum of two positive definite matrices hence $W_{t+1} \succ 0$. Write W_t as its eigendecomposition $W_t = Q\Sigma Q^T$. Since W_t is positive definite matrix, Σ is a diagonal matrix with positive diagonal. Using this decomposition, the update for W_{t+1} is expressed as

$$W_{t+1} = Q\Sigma Q^T - \frac{2\eta_t(\hat{y}_t - y) Q\Sigma Q^T \boldsymbol{z}\boldsymbol{z}^T Q\Sigma Q^T}{1 + 2\eta_t(\hat{y}_t - y)\boldsymbol{z}^T Q\Sigma Q^T \boldsymbol{z}}. \tag{10}$$

Since $W_t \prec I$, then $I - \Sigma \succ 0$. Using Equation 10, we have that

$$\begin{aligned}
I - W_{t+1} &= I - Q\Sigma Q^T - \frac{2\eta_t(y_t - \hat{y}_t) Q\Sigma Q^T \boldsymbol{z}\boldsymbol{z}^T Q\Sigma Q^T}{1 + 2\eta_t(\hat{y}_t - y_t)\boldsymbol{z}^T Q\Sigma Q^T \boldsymbol{z}} \\
&= Q\left(I - \Sigma - \frac{2\eta_t(y_t - \hat{y}_t)\Sigma Q^T \boldsymbol{z}\boldsymbol{z}^T Q\Sigma}{1 + 2\eta_t(\hat{y}_t - y_t)\boldsymbol{z}^T Q\Sigma Q^T \boldsymbol{z}}\right) Q^T \\
&= Q(I-\Sigma)^{\frac{1}{2}}\left(I - \frac{2\eta_t(y_t - \hat{y}_t)(I-\Sigma)^{-\frac{1}{2}}\Sigma Q^T \boldsymbol{z}\boldsymbol{z}^T Q\Sigma(I-\Sigma)^{-\frac{1}{2}}}{1 + 2\eta_t(\hat{y}_t - y_t)\boldsymbol{z}^T Q\Sigma Q^T \boldsymbol{z}}\right)(I-\Sigma)^{\frac{1}{2}} Q^T.
\end{aligned}$$

Let $G = Q(I-\Sigma)^{\frac{1}{2}}$. If the middle term is symmetric positive definite, then it can be written as EE^T. This would lead to $I - W_{t+1} = GEE^T G^T = GE(GE)^T \succ 0$, since for any A, $AA^T \succ 0$.

Thus to prove that $W_{t+1} \prec I$, we need to prove that

$$\left(I - \frac{2\eta_t(y_t - \hat{y}_t)(I-\Sigma)^{-\frac{1}{2}}\Sigma Q^T \boldsymbol{z}\boldsymbol{z}^T Q\Sigma(I-\Sigma)^{-\frac{1}{2}}}{1 + 2\eta_t(\hat{y}_t - y_t)\boldsymbol{z}^T Q\Sigma Q^T \boldsymbol{z}}\right) \succ 0. \tag{11}$$

To do this, we calculate the eigenvalues of this matrix, and show that they are greater than zero. Let $\boldsymbol{v} = (I-\Sigma)^{-\frac{1}{2}}\Sigma Q^T \boldsymbol{z}$. Then $(I-\Sigma)^{-\frac{1}{2}}\Sigma Q^T \boldsymbol{z}\boldsymbol{z}^T Q\Sigma(I-\Sigma)^{-\frac{1}{2}} = \boldsymbol{v}\boldsymbol{v}^T$. Since, $\boldsymbol{v}\boldsymbol{v}^T$ is a rank-one matrix, it has exactly one positive eigenvalue, and that eigenvalue is equal to its trace $\mathrm{tr}((I-\Sigma)^{-\frac{1}{2}}\Sigma Q^T \boldsymbol{z}\boldsymbol{z}^T Q\Sigma(I-\Sigma)^{-\frac{1}{2}})$. Expanding the trace yields

$$\begin{aligned}
\mathrm{tr}((I-\Sigma)^{-\frac{1}{2}}\Sigma Q^T \boldsymbol{z}\boldsymbol{z}^T Q\Sigma(I-\Sigma)^{-\frac{1}{2}}) &= \mathrm{tr}(Q\Sigma(I-\Sigma)^{-1}\Sigma Q^T \boldsymbol{z}\boldsymbol{z}^T) \\
&= \mathrm{tr}(Q(\Sigma^{-1} - I)^{-1}\Sigma Q^T \boldsymbol{z}\boldsymbol{z}^T) \\
&= \mathrm{tr}((W_t^{-1} - I)^{-1} W_t X_t).
\end{aligned}$$

To prove Equation 11, we need to show that the eigenvalues are positive; equivalently, we can show that

$$1 - \frac{2\eta_t(y_t - \hat{y}_t)\mathrm{tr}((W_t^{-1} - I)^{-1} W_t X_t)}{1 - 2\eta_t(y - \hat{y}_t)\mathrm{tr}(W_t X_t)} > 0. \tag{12}$$

By the update rule, $\eta_t < \frac{1}{2(y_t - \hat{y}_t)\mathrm{tr}(W_t X_t)}$, and hence the denominator of the second term is positive. Also, $\eta_t < \frac{1}{2(y_t - \hat{y}_t)}\left(\frac{1}{\mathrm{tr}((I + (W_t^{-1} - I)^{-1})W_t X_t)}\right)$, so Equation 12 holds. Thus $W_{t+1} \prec I$. □

3.2 Regret Bound

Similar to the vector case, we bound loss at each step incurred by algorithm in terms of loss incurred by the optimal offline solution. Assume $\|\boldsymbol{z}\|_2 \leq R$.

Lemma 3.

$$a_t(\hat{y}_t - y_t)^2 - b_t(tr(UX_t) - y_t)^2 \leq D_{ld}(U, W_t) - D_{ld}(U, W_{t+1}),$$

where a_t and b_t are positive constants and U is the optimal offline solution.

Proof.

$$D_{ld}(U, W_t) - D_{ld}(U, W_{t+1}) = \log\left(\frac{\det(W_t)}{\det(W_{t+1})}\right) + \mathrm{tr}((W_t^{-1} - W_{t+1}^{-1})U).$$

Since $\log(\det(A)) = \mathrm{tr}(\log(A))$ and $\mathrm{tr}(AB) = \mathrm{tr}(BA)$, we have that

$$\begin{aligned} D_{ld}(U, W_t) - D_{ld}(U, W_{t+1}) &= \mathrm{tr}(\log(W_t) - \log(W_{t+1})) + \mathrm{tr}((W_t^{-1} - W_{t+1}^{-1})U) \\ &= \mathrm{tr}(\log(W_t W_{t+1}^{-1})) - 2\eta_t(\hat{y}_t - y_t)\mathrm{tr}(X_t U) \\ &= \mathrm{tr}(\log(I + 2\eta_t(\hat{y}_t - y_t)W_t X_t)) - 2\eta_t(\hat{y}_t - y_t)\mathrm{tr}(U X_t). \end{aligned}$$

Using the Taylor series expansion and ignoring lower order terms,

$$D_{ld}(U, W_t) - D_{ld}(U, W_{t+1}) \geq 2\eta_t(\hat{y}_t - y_t)\mathrm{tr}(W_t X_t) - 2\eta_t^2(\hat{y}_t - y_t)^2\mathrm{tr}((W_t X_t)^2) - 2\eta_t(\hat{y}_t - y_t)\mathrm{tr}(U X_t).$$

We write X as $\boldsymbol{z}\boldsymbol{z}^T$, so $\mathrm{tr}((W_t X_t)^2) = \mathrm{tr}(W_t \boldsymbol{z}\boldsymbol{z}^T W_t \boldsymbol{z}\boldsymbol{z}^T) = (\boldsymbol{z}^T W_t \boldsymbol{z})\mathrm{tr}(W \boldsymbol{z}\boldsymbol{z}^T) = \mathrm{tr}(W_t X_t)^2$. Hence,

$$\begin{aligned} D_{ld}(U, W_t) - D_{ld}(U, W_{t+1}) &\geq 2\eta_t(\hat{y}_t - y_t)\mathrm{tr}(W_t X_t) - 2\eta_t^2(\hat{y}_t - y_t)^2\mathrm{tr}((W_t X_t)^2) - 2\eta_t(\hat{y}_t - y_t)\mathrm{tr}(U X_t) \\ &\geq 2\eta_t(\hat{y}_t - y_t)\mathrm{tr}(W_t X_t) - 2\eta_t^2(\hat{y}_t - y_t)^2\mathrm{tr}(W_t X_t)^2 - 2\eta_t(\hat{y}_t - y_t)\mathrm{tr}(U X_t). \end{aligned}$$

Since $W_t \prec I$ by Lemma 2, $\mathrm{tr}(W_t X_t) = \boldsymbol{z}^T W_t \boldsymbol{z} \leq \boldsymbol{z}^T \boldsymbol{z} \leq R^2$. As a result,

$$\begin{aligned} D_{ld}(U, W_t) - D_{ld}(U, W_{t+1}) &\geq 2\eta_t(\hat{y}_t - y_t)\mathrm{tr}(W_t X_t) - 2\eta_t^2(\hat{y}_t - y_t)^2 R^2 - 2\eta_t(\hat{y}_t - y_t)\mathrm{tr}(U X_t) \\ &= 2\eta_t(\hat{y}_t - y_t)\hat{y}_t - 2\eta_t^2(\hat{y}_t - y_t)^2 R^2 - 2\eta_t(\hat{y}_t - y_t)r, \end{aligned}$$

where $r = \mathrm{tr}(U_t X_t)$. Thus, to prove Lemma 3, we need to prove the same inequality as in Equation 5, with mR^2 replaced by R^2. Hence, for $0 \leq a_t \leq \eta_t$ and $b_t = \frac{\eta_t}{1 - 2\eta_t R^2}$, Lemma 3 holds. □

As in the vector case, we can obtain the final bound by summing the loss at each step.

Theorem 2.

$$L_{BG} \leq \frac{1}{2\eta_T R^2} L_U + \frac{1}{\eta_T} H_{ld}(U),$$

where L_{MBG} is the loss incurred by the Matrix Burg Gradient descent algorithm, L_U is loss incurred by optimal offline algorithm and $H_{ld}(U) = -\log det(U)$.

Proof. The proof is very similar to that of the vector case. □

4 Application: Online Metric Learning

Selecting an appropriate distance measure (or metric) is fundamental to many learning algorithms such as k-means, nearest neighbor searches, and others. However, choosing such a measure is highly problem-specific and ultimately dictates the success—or failure—of the learning algorithm. To this end, there have been several recent approaches that attempt to learn distance functions. These methods work by exploiting distance information that is intrinsically available in many learning settings. For example, in the problem of semi-supervised clustering, points are constrained to be either similar (i.e. the distance between them should be relatively small) or dissimilar (the distance should be larger). In information retrieval settings, constraints between pairs of distances can be gathered from click-through feedback. In fully supervised settings, constraints can be inferred so that points in the same class have smaller distances to each other than to points in different classes.

Given a set of n points $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$ in $\mathbb{R}^d$, we seek a positive definite matrix A which parameterizes the Mahalanobis distance[1]:

$$d_A(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T A(\boldsymbol{x}_i - \boldsymbol{x}_j). \tag{13}$$

[1]The term 'Mahalanobis distance' frequently refers to the squared Mahalanobis distance, even though it is $\sqrt{d_A(\boldsymbol{x}, \boldsymbol{y})}$ that is a metric. This difference does not affect our method, hence we drop the qualification *squared* when referring to (13)

We can quantify the measure of "closeness" between two Mahalanobis matrices A and A_0 via a natural information-theoretic approach. There exists a simple bijection (up to a scaling function) between the set of Mahalanobis distances and the set of equal-mean multivariate Gaussian distributions (without loss of generality, we can assume the Gaussians have zero-mean $\boldsymbol{\mu}$). Given a Mahalanobis distance parameterized by A, we express its corresponding multivariate Gaussian as $p(\boldsymbol{x}; A) = \frac{1}{Z}\exp\left(-\frac{1}{2}d_A(\boldsymbol{x}, \boldsymbol{\mu})\right)$, where Z is a normalizing constant. Using this bijection, we measure the distance between two Mahalanobis distance functions parameterized by A and A_0 by the (differential) relative entropy between their corresponding multivariate Gaussians:

$$\mathrm{KL}(p(\boldsymbol{x}; A)\|p(\boldsymbol{x}; A_0)) = \int p(\boldsymbol{x}; A)\log\frac{p(\boldsymbol{x}; A)}{p(\boldsymbol{x}; A_0)}\,d\boldsymbol{x}. \tag{14}$$

The distance (14) provides a well-founded measure of "closeness" between two Mahalanobis distance functions and forms the basis of information-theoretic metric learning. It is known that the differential relative entropy between two multivariate Gaussians can be expressed as the convex combination of a Mahalanobis distance between mean vectors and the LogDet divergence between the covariance matrices. Assuming the means to be the same, we have

$$\mathrm{KL}(p(\boldsymbol{x}; A)\|p(\boldsymbol{x}; A_0)) = \frac{1}{2}D_{ld}(A, A_0), \tag{15}$$

We therefore select $d(A, A_t) = D_{ld}(A, A_t)$ as the regularizer for (1), using the above equivalence 15 to obtain the update

$$A_{t+1} = \operatorname*{argmin}_A D_{ld}(A, A_t) + \eta_t(d_t - \hat{d}_t)^2.$$

We can view the problem of online metric learning in the following manner: at each time step, we receive a matrix X_t and a target y_t. These encode the distance between two vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. If we let $\boldsymbol{z} = \boldsymbol{x}_i - \boldsymbol{x}_j$, then $\mathrm{tr}(A_tX_t) = \boldsymbol{z}^TA_t\boldsymbol{z} = (\boldsymbol{x}_i - \boldsymbol{x}_j)^TA_t(\boldsymbol{x}_i - \boldsymbol{x}_j) = d_{A_t}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Thus, the prediction $\mathrm{tr}(A_tX_t)$ captures the Mahalanobis distance between the vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ using the current Mahalanobis matrix A_t. The resulting loss $(\mathrm{tr}(A_tX_t) - y_t)^2$, where y_t is the target distance between the two vectors, tells us how close the current Mahalanobis matrix A_t is in terms of satisfying the given distance constraint.

As a result, the rank-one online regression algorithm using the LogDet divergence can be used as method for learning a Mahalanobis matrix A, where at every time step two points, along with their target distance, are presented to the algorithm. The regret bound proven in the previous section applies directly to this case.

See[2, 3] for further details on information-theoretic metric learning, including the analysis of an offline algorithm.

5 Experimental Results

In section 2.1 and 3.2, we showed worst case bounds for our algorithms. To simulate the online learning problem, we now generate simple artificial data. First, we generate a random vector $\boldsymbol{u}$ drawn from S^{m-1}. Then, we generate some random inputs $\boldsymbol{x}_t$, also drawn from S^{m-1}. We output y as $\boldsymbol{u}\cdot\boldsymbol{x}_t$ mixed with some noise, i.e,

$$y = (\boldsymbol{u}\cdot\boldsymbol{x}_t)(1 + \nu Z),$$

where Z is an uniformly distributed random number over $[-0.5, 0.5]$ and ν is the noise level.

We generate this artificial data for $N = 100$ and $T = 10000$ iterations. Figure 1 compares performance of the Burg entropy-based algorithm with those using relative entropy and squared Euclidean distance as the regularization term, when the noise level is set to 0. Clearly, the Burg algorithm converges slower compared to the other methods. But, Figures 2 and 3 show that as noise increases, the Burg performance improves in comparison to both relative entropy and squred Euclidean distance. Figure 4 shows that when noise is around 30% then the Burg algorithm performs comparatively to relative entropy, and squred Euclidean performs poorly.

6 Conclusion

In this paper, we introduced online regression based on using a Burg entropy-based regularization, leading to new online regret bounds. An extension to the rank-one matrix case was also analyzed. The main application

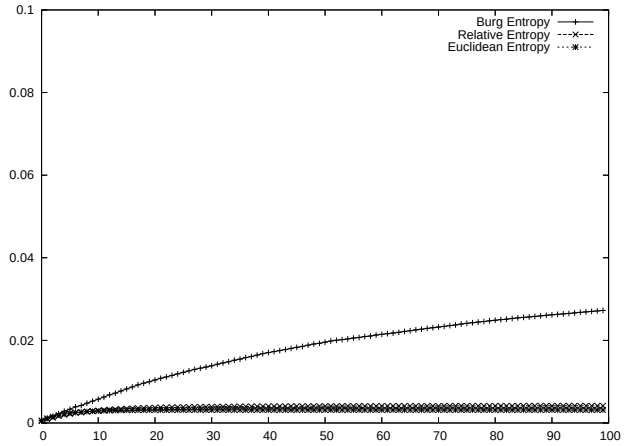

Figure 1: Comparison of performance of burg, relative and euclidean distance measure with noise=0%

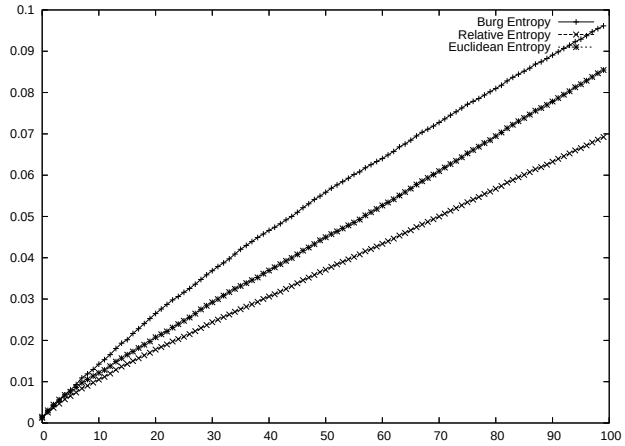

Figure 2: Comparison of performance of burg, relative and euclidean distance measure with noise=10%

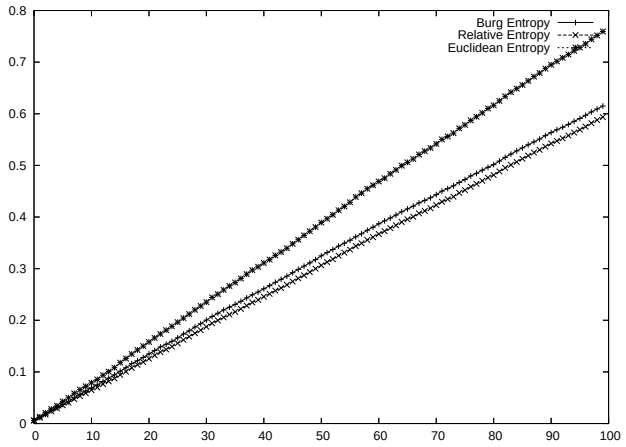

Figure 3: Comparison of performance of burg, relative and euclidean distance measure with noise=20%

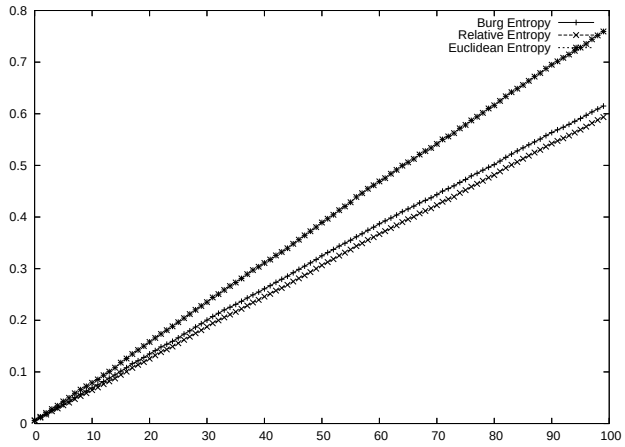

Figure 4: Comparison of performance of burg, relative and euclidean distance measure with noise=30%

is to online metric learning, which can be cast as an online regression problem for rank-one matrices; this metric learning formulation has an information-theoretic motivation, and results in [2] have shown that it outperforms existing methods. Future work includes improving the bounds so that they do not depend on the learning rate, and extending the matrix analysis to the general case.

References

[1] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh, *Clustering with bregman divergences.*, in SDM, 2004, pp. 234–245.

[2] J. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, *Information-theoretic metric learning*, in International Conference on Metric Learning, 2007, p. to appear.

[3] J. Davis, B. Kulis, S. Sra, and I. Dhillon, *Information-theoretic metric learning*, in NIPS 2006 Workshop on Learning to Compare Examples, 2006.

[4] G. H. Golub and C. F. van Loan, *Matrix Computations*, Johns Hopkins Series in the Mathematical Sciences, Johns Hopkins Univ. Press, second ed., 1989.

[5] G. J. Gordon, *Regret bounds for prediction problems*, in COLT, 1999, pp. 29–40.

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

[7] B. Kulis, M. Sustik, and I. S. Dhillon, *Learning low-rank kernel matrices.*, in ICML, 2006, pp. 505–512.

[8] K. Tsuda, G. Raetsch, and M. K. Warmuth, *Matrix exponentiated gradient updates of online learning and bregman projection*, Journal of Machine Learning Research, 6 (2005).