Online learning with kernel losses

Aldo Pacchiano [*1] **Niladri S. Chatterji** [*1] **Peter L. Bartlett** [1]

Abstract

We present a generalization of the adversarial linear bandits framework, where the underlying losses are kernel functions (with an associated reproducing kernel Hilbert space) rather than linear functions. We study a version of the exponential weights algorithm and bound its regret in this setting. Under conditions on the eigen-decay of the kernel we provide a sharp characterization of the regret for this algorithm. When we have polynomial eigen-decay ($\mu_j \leq \mathcal{O}(j^{-\beta})$), we find that the regret is bounded by $\mathcal{R}_n \leq \mathcal{O}(n^{\beta/2(\beta-1)})$. While under the assumption of exponential eigen-decay ($\mu_j \leq \mathcal{O}(e^{-\beta j})$) we get an even tighter bound on the regret $\mathcal{R}_n \leq \tilde{\mathcal{O}}(n^{1/2})$. When the eigen-decay is polynomial we also show a *non-matching* minimax lower bound on the regret of $\mathcal{R}_n \geq \Omega(n^{(\beta+1)/2\beta})$ and a lower bound of $\mathcal{R}_n \geq \Omega(n^{1/2})$ when the decay in the eigenvalues is exponentially fast.

We also study the full information setting when the underlying losses are kernel functions and present an adapted exponential weights algorithm and a conditional gradient descent algorithm.

1. Introduction

In adversarial online learning, a player interacts with an unknown and arbitary adversary in a sequence of rounds. At each round, the player chooses an action from an action space and incurs a loss associated with that chosen action. The loss functions are determined by the adversary and are fixed at the beginning of each round. After choosing an action the player observes some feedback, which can help guide the choice of actions in subsequent rounds. The most common feedback model is the *full information* model, where the player has access to the entire loss function at the end of each round. Another, more challenging feedback model is the *partial information* or *bandit* feedback model where the player at the end of the round just observes the loss associated with the action chosen in that particular round. There are also other feedback models in between and beyond the full and bandit information models, many of which have also been studied in detail. A figure of merit that is often used to judge online learning algorithms is the notion of *regret*, which compares the players actions to the best single action in hindsight (defined formally in Section 1.2).

When the underlying action space is a continuous and compact (possibly convex) set and the losses are linear or convex functions over this set; there are many algorithms known that attain sub-linear and sometimes optimal regret in both these feedback settings. In this work we present a generalization of the well studied adversarial online linear learning framework. In our paper, at each round the player selects an action $a \in \mathcal{A} \subset \mathbb{R}^d$. This action is mapped to an element in a reproducing kernel Hilbert space (RKHS) generated by a mapping $\mathcal{K}(\cdot,\cdot)$. The function $\mathcal{K}(\cdot,\cdot)$ is a kernel map, that is, it can thought as an inner product of an appropriate Hilbert space $\mathcal{H}$. The kernel map can be expressed as $\mathcal{K}(x,y) = \langle \Phi(x), \Phi(y) \rangle_{\mathcal{H}}$, where $\Phi(\cdot) \in \mathbb{R}^D$ is the associated feature map.

Thus at each round the loss is $\langle \Phi(a), w \rangle_{\mathcal{H}}$, where $w \in \mathcal{H}$ is the adversary's action. In the full information setting, as feedback, the player has access to the entire adversarial loss function $\langle \cdot, w \rangle_{\mathcal{H}}$. In the bandit setting the player is just presented with the value of the loss, $\langle \Phi(a), w \rangle_{\mathcal{H}}$.

Notice that this class of losses is much more general than ordinary linear losses and includes potentially non-linear and non-convex losses like:

1. Linear Losses: $\langle a, w \rangle_{\mathcal{H}} = a^\top w$. This loss is well studied in both the bandit and full information setting. We shall see that our regret bounds will match the bounds established in the literature for these losses.
2. Quadratic Losses: $\left\langle \phi(a), \binom{W}{b} \right\rangle_{\mathcal{H}} = a^\top W a + b^\top a$, where W is a symmetric matrix and b is a vector. Convex quadratic losses have been well studied under full information feedback as the online eigenvector decomposition problem. Our work establishes regret bounds

[*]Equal contribution [1]University of California Berkeley. Correspondence to: Aldo Pacchiano <pacchiano@berkeley.edu>, Niladri S. Chatterji <chatterji@berkeley.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

in the full information setting and also under the mostly unexplored bandit feedback.

3. Gaussian Losses: $\langle \Phi(a), \Phi(y) \rangle_{\mathcal{H}} = \exp\left(-\|a-y\|_2^2/2\sigma^2\right)$. We provide regret bounds of kernel losses not commonly studied before like Gaussian losses that provide a different loss profile than a linear or convex loss.

4. Polynomial Losses: $\langle \Phi(a), \Phi(y) \rangle_{\mathcal{H}} = (1 + a^\top y)^2$ for example. We also provide regret bounds for polynomial kernel losses which are potentially (non-convex) under both partial and full information settings. Specifically in the full information setting we study posynomial losses (discussion in Section 4.3).

1.1. Related Work

Adversarial online convex bandits that was introduced and first studied by (33; 22). The problem most closely related to our work is the case when the losses are linear introduced earlier by (35; 7). In this setting (20; 18; 13) proposed the EXP 2 (Expanded Exp) algorithm under different choices of exploration distributions. (20) worked with the uniform distribution over the barycentric spanner of the set, in (18) this distribution was the uniform distribution over the set and in (13) they use the exploration distribution given by John's theorem that leads to a regret bound of $\mathcal{O}((dn\log(N))^{1/2})$, where N is the number of actions, n is the number of rounds and d is the dimension of the losses. For this same problem when the set $\mathcal{A}$ is convex and compact, (1) analyzed Mirror descent to get a regret bound of $\mathcal{O}(d\sqrt{\theta n \log(n)})$ for some $\theta > 0$. For the case with general convex losses with bandit feedback recently (15) proposed a poly-time algorithm that has a regret guarantee of $\tilde{\mathcal{O}}(d^{9.5}\sqrt{n})$, which is optimal in its dependence on the number of rounds n. Previous work on this problem includes, (2; 41; 27; 21; 14; 28) in the adversarial setting under different assumptions on the structure of the convex losses and by (3) who studied this problem in the stochastic setting[1]. (46) study stochastic kernelized contextual bandits with a modified UCB algorithm to obtain a regret bound similar to ours, $\mathcal{R}_n \leq \sqrt{\tilde{d}n}$ where $\tilde{d}$ is the effective dimension dependent on the eigen-decay of the kernel. This problem was also studied previously for loss functions drawn from Gaussian processes in (44). Online learning under bandit feedback has also been studied when the losses are non-parametric, for example when the losses are Lipschitz (16; 40).

In the full information case, the online optimization framework with convex losses was first introduced by (49). The conditional gradient descent algorithm (a modification of which we study in this work) for convex losses in this setting was introduced and analyzed by (31) and then improved subsequently by (26). The exponential weights algorithm which we modify and use multiple times in this paper has a rich history and has been applied to various online as well as offline settings. The particular with the losses being convex quadratic functions has been well studied in the full information setting. This problem is also called online eigenvector decomposition or online PCA. Very recently (4) established a regret bound of $\tilde{\mathcal{O}}(\sqrt{n})$ for the problem by presented an efficient algorithm that achieves this rate – a modified exponential weights strategy, follow the compressed leader. Previous results for this problem were established in both adversarial and stochastic settings by modifications of exponential weights, gradient descent and follow the perturbed leader algorithms (6; 45; 47; 48; 32; 23).

In the full information setting there has also been work on analyzing gradient descent and mirror descent in RKHS spaces (36; 8). However, in these papers the player is allowed to play any action in a bounded set in Hilbert space, while in our paper the player is constrained to only play rank one actions, that is the player chooses an action in $\mathcal{A}$ which gets mapped to an action in the RKHS.

Contributions

Our primary contribution is to extend the linear bandits framework to more general classes of kernel losses. We present an algorithm in this setting and provide a regret bound for the same. We provide a more detailed analysis of the regret when we make assumptions on the eigen-decay of the kernel. Particularly when we assume the polynomial eigen-decay of the kernel ($\mu_j \leq \mathcal{O}(j^{-\beta})$) we can guarantee the regret is bounded as $\mathcal{R}_n \leq \mathcal{O}(n^{\frac{\beta}{2(\beta-1)}})$. Under exponential eigendecay we can guarantee an even sharper bound on the regret of $\mathcal{R}_n \leq \tilde{\mathcal{O}}(n^{1/2})$. We also provide a minimax lower bound on the regret of $\mathcal{R}_n \geq \Omega(n^{(\beta+1)/2\beta})$ and $\mathcal{R}_n \geq \Omega(n^{1/2})$ under the polynomial and exponential decay eigen-decay assumptions respectively. We analyze an exponential weights algorithm and a conditional gradient algorithm for the full information case where we don't need to assume any conditions on the eigen-decay. Finally we provide a couple of applications of our framework – (i) general quadratic losses (not necessarily convex) with linear terms which we can solve efficiently in the full information setting, (ii) we provide a computationally efficient algorithm when the underlying losses are posynomial (special class of polynomials).

Organization of the Paper

In the next section we introduce the notation and definitions. In Section 2 we present our algorithm under bandit feedback and present regret bounds for this algorithm. In Section 3 we study the problem in the full information setting. In Section

[1] For an extended bibliography of the work on online convex bandits see (15).

4 we present two applications of our framework, and prove that our algorithms are computationally efficient in these settings. All the proofs, technical details and experiments are relegated to the appendix.

1.2. Notation, main definitions and setting

Here we introduce definitions and notational conventions used throughout the paper.

In each round $t = \{1, \dots, n\}$, the player chooses an action vector $\{a_t\}_{t=1}^n \in \mathcal{A} \subset \mathbb{R}^d$. The underlying kernel function at each round is $\mathcal{K}(\cdot, \cdot)$ which is a map from $\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ such that it is a *kernel map* and has an associated separable reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ with an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ (for more properties of kernel maps and RKHS see (42)). Let $\Phi(\cdot) : \mathbb{R}^d \mapsto \mathbb{R}^D$ denote a *feature map* of $\mathcal{K}(\cdot, \cdot)$ such that for every x, y we have $\mathcal{K}(x, y) = \langle \Phi(x), \Phi(y) \rangle_{\mathcal{H}}$. Note that the dimension of the RKHS, D could be infinite (for example in the Gaussian kernel over $[0, 1]^d$).

We let the adversary choose a vector in $\mathcal{H}$, $w_t \in \mathcal{W} \subset \mathbb{R}^D$ and at each round the loss incurred by the player is $\langle \Phi(a_t), w_t \rangle_{\mathcal{H}}$. We assume that the adversary is oblivious, that is, it is a function of the previous actions of the player $(a_1, \dots, a_{t-1})$ but unaware of the randomness used to generate a_t. We let the size of the sets $\mathcal{A}, \mathcal{W}$ be bounded[2] in *kernel norm*, that is,

$$\sup_{a \in \mathcal{A}} \mathcal{K}(a, a) \le \mathcal{G}^2 \qquad \text{and,} \quad \sup_{w \in \mathcal{W}} \langle w, w \rangle_{\mathcal{H}} \le \mathcal{G}^2. \tag{1}$$

Throughout this paper we assume a *rank-one learner*, that is, in each round the player can pick a vector $v \in \mathcal{H}$, such that $v = \Phi(a)$ for some $a \in \mathbb{R}^d$. We now formally define the notion of expected regret.

Definition 1 (Expected regret) *The expected regret of an algorithm $\mathcal{M}$ after n rounds is defined as*

$$\mathcal{R}_n = \mathbb{E}_{\mathcal{M}} \left[\sum_{t=1}^n \langle \Phi(a_t), w_t \rangle_{\mathcal{H}} - \sum_{t=1}^n \langle \Phi(a^*), w_t \rangle_{\mathcal{H}} \right] \tag{2}$$

where $a^* = \inf_{a \in \mathcal{A}} \{ \sum_{t=1}^n \langle \Phi(a), w_t \rangle_{\mathcal{H}} \}$ and the expectation is over the randomness in the algorithm.

Essentially this amounts to comparing against the *best single action* a^* in hindsight. Our hope will be to find a randomized strategy such that the regret grows sub-linearly with the number of rounds n. In what follows we will omit the subscript $\mathcal{H}$ from the subscript of the inner product whenever it is clear from the context that it refers to the RKHS inner product.

[2] We set the bound on the size of both sets to be the same for ease of exposition, but they could be different and would only change the constants in our results.

To establish regret guarantees we will find that it is essential to work with finite dimensional kernels when working under bandit feedback (more details regarding this in the proof of the regret bound of Algorithm 2.3). General kernel maps can have infinite dimensional feature maps thus we will require the construction of a finite dimensional kernel that uniformly approximates the original kernel $\mathcal{K}(\cdot, \cdot)$. This motivates the definition of ϵ-approximate kernels.

Definition 2 (ϵ-approximate kernels) *Let $\mathcal{K}_1$ and $\mathcal{K}_2$ be two kernels over $\mathcal{A} \times \mathcal{A}$ and let $\epsilon > 0$. We say $\mathcal{K}_2$ is an ϵ-approximation of $\mathcal{K}_1$ if for all $x, y \in \mathcal{A}$, $|\mathcal{K}_1(x, y) - \mathcal{K}_2(x, y)| \le \epsilon$.*

2. Bandit Feedback Setting

In this section we present our results on *kernel bandits*. In the bandit setting we assume the player knows the underlying kernel function $\mathcal{K}(\cdot, \cdot)$, however, at each round after the player plays a vector a_t only the value of the loss associated with that action is revealed to the player – $\langle \Phi(a_t), w_t \rangle_{\mathcal{H}}$ – and not the action of the adversary w_t. We also assume that the player's action set $\mathcal{A}$ has finite cardinality[3].This is a generalization of the well studied adversarial linear bandits problem. As we will see in subsequent sections to guarantee a bound on the regret in the bandit setting our algorithm will build an estimate of adversary's action w_t. This becomes impossible if w_t is infinite dimensional ($D \to \infty$). To circumvent this, we will construct a finite dimensional proxy kernel that is an ϵ-approximation of $\mathcal{K}$.

Whenever no approximate kernel is needed, for example when $D < \infty$ we allow the adversary to be able to choose an action $w_t \in \mathcal{W} \subset \mathbb{R}^D$ without imposing extra requirements on the set $\mathcal{W}$ other than being bounded in $\mathcal{H}$ norm. When D is infinite we impose an additional constraint on the adversary to also select *rank-one* actions at each round, that is, $w_t = \Phi(y_t)$ where $y_t \in \mathbb{R}^d$. Next we present a discussion of the procedure to construct a finite kernel that approximates the original kernel well.

2.1. Construction of the finite dimensional kernel

To construct the finite dimensional kernel we will rely crucially on Mercer's theorem. We first recall a couple of useful definitions.

Definition 3 *Let $\mathcal{A} \subset \mathbb{R}^d$ and $\mathbb{P}$ a probability measure supported over $\mathcal{A}$. We denote by $L_2(\mathcal{A}; \mathbb{P})$ the space of square integrable functions over $\mathcal{A}$ and measure $\mathbb{P}$, $L_2(\mathcal{A}; \mathbb{P}) :=$*

[3] This assumption can be relaxed to let $\mathcal{A}$ be a compact set when $\mathcal{K}$ is Lipschitz continuous. In this setting we can instead work with an appropriately fine approximating cover over the set $\mathcal{A}$.

$$\left\{ f : \mathcal{A} \to \mathbb{R} \middle| \int_{\mathcal{A}} f^2(x) d\mathbb{P}(x) < \infty \right\}.$$

Definition 4 *A kernel $\mathcal{K} : \mathcal{A} \times \mathcal{A} \to \mathbb{R}$ is square integrable with respect to measure $\mathbb{P}$ over $\mathcal{A}$, if $\int_{\mathcal{A}\times\mathcal{A}} \mathcal{K}^2(x,y) d\mathbb{P}(x) d\mathbb{P}(y) < \infty$.*

Now we are ready to present Mercer's theorem (38) (see (19)).

Theorem 5 (Mercer's Theorem) *Let $\mathcal{A} \subset \mathbb{R}^d$ be compact and $\mathbb{P}$ be a finite Borel measure with support $\mathcal{A}$. Suppose $\mathcal{K}$ is a continuous square integrable positive definite kernel on $\mathcal{A}$, and define a positive definite operator $T_{\mathcal{K}} : L_2(\mathcal{A};\mathbb{P}) \mapsto L_2(\mathcal{A};\mathbb{P})$ by*

$$\left(T_{\mathcal{K}} f\right)(\cdot) := \int_{\mathcal{A}} \mathcal{K}(\cdot, x) f(x) \, d\mathbb{P}.$$

Then there exists a sequence of eigenfunctions $\{\phi_i\}_{i=1}^{\infty}$ that form an orthonormal basis of $L_2(\mathcal{A};\mathbb{P})$ consisting of eigenfunctions of $T_{\mathcal{K}}$, and an associated sequence of non-negative eigenvalues $\{\mu_j\}_{j=1}^{\infty}$ such that $T_{\mathcal{K}}(\phi_j) = \mu_j \phi_j$ for $j = 1, 2, \ldots$. Moreover the kernel function can be represented as

$$\mathcal{K}(u, v) = \sum_{i=1}^{\infty} \mu_i \phi_i(u) \phi_i(v) \tag{3}$$

where the convergence of the series holds uniformly.

Mercer's theorem yields a natural way to construct a feature map $\Phi(x)$ for $\mathcal{K}$ by defining the i^{th} component of the feature map to be $\Phi(x)_i := \sqrt{\mu_i}\phi_i(x)$. With this choice of feature map the eigenfunctions $\{\phi_i\}_{i=1}^{\infty}$ are orthogonal under the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$[4]. Armed with Mercer's theorem we first present a simple deterministic procedure to obtain a finite dimensional ϵ-approximate kernel of $\mathcal{K}$. Essentially when the eigenfunctions of the kernel are uniformly bounded, $\sup_{x\in\mathcal{A}} |\phi_j(x)| \le \mathcal{B}$ for all j, and if the eigenvalues decay at a suitable rate we can truncate the series in (3) to get a finite dimensional approximation.

Lemma 6 *Given $\epsilon > 0$, let $\{\mu_j\}_{j=1}^{\infty}$ be the Mercer operator eigenvalues of $\mathcal{K}$ under a finite Borel measure $\mathbb{P}$ with support $\mathcal{A}$ and eigenfunctions $\{\phi_j\}_{j=1}^{\infty}$ with $\mu_1 \ge \mu_2 \ge \cdots$. Further assume that $\sup_{j\in\mathbb{N}} \sup_{x\in\mathcal{A}} |\phi_j(x)| \le \mathcal{B}$ for some $\mathcal{B} < \infty$. Let $m(\epsilon)$ be such that $\sum_{j=m+1}^{\infty} \mu_j \le \frac{\epsilon}{4\mathcal{B}^2}$. Then the kernel induced by a truncated feature map,*

$$\Phi_m^o(x) := \begin{cases} \sqrt{\mu_i}\phi_i(x) & \text{if } i \le m \\ 0 & \text{o.w.} \end{cases} \tag{4}$$

induces a kernel $\hat{\mathcal{K}}_m^o := \langle \Phi_m^o(x), \Phi_m^o(y) \rangle_{\mathcal{H}} = \sum_{j=1}^{m} \mu_j \phi_j(x) \phi_j(y)$, for all $(x, y) \in \mathcal{A} \times \mathcal{A}$ that is an $\epsilon/4$-approximation of $\mathcal{K}$.

The Hilbert space induced by the $\hat{\mathcal{K}}_m^o$ is a subspace of the original Hilbert space $\mathcal{H}$. The proof of this lemma is a simple application of Mercer's theorem and is relegated to Appendix C. If we have access to the eigenfunctions of $\mathcal{K}$ we can construct and work with $\hat{\mathcal{K}}_m^o$ because as Lemma 6 shows $\hat{\mathcal{K}}_m^o$ is an $\epsilon/4$-approximation to $\mathcal{K}$. Additionally, $\hat{\mathcal{K}}_m^o$ also has the same first m Mercer eigenvalues and eigenfunctions under $\mathbb{P}$ as $\mathcal{K}$. Unfortunately, in most applications of interest the analytical computation of the eigenfunctions $\{\phi_i\}_{i=1}^{\infty}$ is not possible. We can get around this by building an estimate of the eigenfunctions using samples from $\mathbb{P}$ by leveraging results from kernel principal component analysis (PCA).

Definition 7 *Let S_m be the subspace of $\mathcal{H}$ spanned by the first m eigenvectors of the covariance matrix $\mathbb{E}_{x\sim\mathbb{P}}\left[\Phi(x)\Phi(x)^\top\right]$.*

This corresponds to the span of the eigenfunctions $\phi_1, \phi_2, \ldots, \phi_m$ in $\mathcal{H}$ [5] . Define the linear projection operator $P_{S_m} : \mathcal{H} \mapsto \mathcal{H}$ that projects onto the subspace S_m; where $P(S_m)(v + v^{\perp}) = v$, if $v \in S_m$ and $v^{\perp} \in S_m^{\perp}$.

Remark 8 *The feature map $\Phi_m^o(x)$ is a projection of the complete feature map to this subspace, $\Phi_m^o(x) = P_{S_m}(\Phi(x))$.*

Let $x_1, x_2, \ldots, x_p \sim \mathbb{P}$ be p i.i.d. samples and construct the sample (kernel) covariance matrix, $\hat{\Sigma} := \frac{1}{p}\sum_{i=1}^{p} \Phi(x_i)\Phi(x_i)^\top$. Let $\hat{S}_m$ be the subspace spanned by the m top eigenvectors of $\hat{\Sigma}$. Define the stochastic feature map, $\Phi_m(x) := P_{\hat{S}_m}(\Phi(x))$, the feature map defined by projecting $\Phi(x)$ to the random subspace $\hat{S}_m$. Intuitively we would expect that if the number of samples p is high enough, then the kernel defined by the feature map $\Phi_m(x)$, $\hat{\mathcal{K}}_m(x, y) = \langle \Phi_m(x), \Phi_m(y) \rangle_{\mathcal{H}}$ will also be an ϵ-approximation for the original kernel $\mathcal{K}$. Formalizing this claim is the following theorem.

Theorem 9 *Let $m, \mathbb{P}$ be defined as in Lemma 6. Define the m-th level eigen-gap as $\delta_m = \frac{1}{2}(\mu_m - \mu_{m+1})$. Also let $B_m = \frac{2\mathcal{G}}{\delta_m}\left(1 + \sqrt{\frac{\alpha}{2}}\right)$, $2\delta_m > \sqrt{\epsilon} > 0$ and $p \ge \frac{2B_m^2\mathcal{G}^2}{\sqrt{\epsilon}}$. Then the finite dimensional kernels $\hat{\mathcal{K}}_m^o$ and $\hat{\mathcal{K}}_m$ satisfy the following properties with probability $1 - e^{-\alpha}$,*

1. $\sup_{x,y\in\mathcal{A}} |\mathcal{K}(x, y) - \hat{\mathcal{K}}_m(x, y)| \le \epsilon$.

[4] To see this observe that the function ϕ_i can be expressed as a vector in the RKHS as a vector v_i with ϕ_i in the i^{th} coordinate and zeros everywhere else. So for any two v_i and v_j with $i \ne j$ we have $\langle v_i, v_j \rangle_{\mathcal{H}} = 0$.

[5] This holds as the i^{th} eigenvector of the covariance matrix has ϕ_i as the i^{th} coordinate and zero everywhere else combined with the fact that $\{\phi_i\}_{i=1}^{\infty}$ are orthonormal under the $L(\mathcal{A};\mathbb{P})$ inner product.

Algorithm 1 Finite dimensional proxy construction

Input : Kernel $\mathcal{K}$, effective dimension m, set $\mathcal{A}$, measure $\mathbb{P}$, bias tolerance $\epsilon > 0$, number of samples p.

Function : Finite proxy feature map $\Phi_m(\cdot)$

sample $x_1, \cdots, x_p \sim \mathbb{P}$.

construct sample Gram matrix $\hat{\mathbb{K}}_{i,j} = \frac{1}{p}\mathcal{K}(x_i, x_j)$.

calculate the top m eigenvectors of $\hat{\mathbb{K}} \to \{\omega_1, \omega_2, \ldots, \omega_m\}$.

for $j = 1, \ldots, m$ **do**

  Set $v_j = \sum_{k=1}^{p} \omega_{jk} \Phi(x_k)$, ($\omega_{jk}$ is the k^{th} entry of ω_j)

end

define the feature map

$$\Phi_m(\cdot) := \begin{bmatrix} \langle v_1, \Phi(x)\rangle_{\mathcal{H}} \\ \vdots \\ \langle v_m, \Phi(x)\rangle_{\mathcal{H}} \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^{p} \omega_{1k}\mathcal{K}(x_k, x) \\ \vdots \\ \sum_{k=1}^{p} \omega_{mk}\mathcal{K}(x_k, x) \end{bmatrix}.$$

2. *The Mercer eigenvalues* $\mu_1^{(p)} \geq \cdots \geq \mu_m^{(p)}$ *and* $\mu_1 \geq \cdots \geq \mu_m$ *of* $\hat{\mathcal{K}}_m$ *and* $\hat{\mathcal{K}}_m^o$ *are close,* $\sup_{i=1,\cdots,m} |\mu_i^{(p)} - \mu_i| \leq \sqrt{\epsilon}/2$.

Theorem 9 shows that given $\epsilon > 0$ the finite dimensional proxy $\hat{\mathcal{K}}_m$ is a ϵ-approximation of $\mathcal{K}$ with high probability as long as sufficiently large number of samples are used. Furthermore, the top m eigenvalues of the second moment matrix of $\mathcal{K}$ are at most $\sqrt{\epsilon}/2$-away from the eigenvalues of the second moment matrix of $\hat{\mathcal{K}}_m$ under $\mathbb{P}$.

To construct $\Phi_m(\cdot)$ we need to calculate the top m eigenvectors of the sample covariance matrix $\hat{\Sigma}$, however, it is equivalent to calculate the top m eigenvectors of the sample Gram matrix $\mathbb{K}$ and use them to construct the eigenvectors of $\hat{\Sigma}$ (for more details see Appendix B where we review the basics of kernel PCA).

2.2. Bandits Exponential Weights

In this section we present a modified version of exponential weights adapted to work with kernel losses. Exponential weights has been analyzed extensively applied to linear losses under bandit feedback (20; 17; 13). Two technical challenges make it hard to directly adapt their algorithms to our setting.

The first challenge is that at each round we need to estimate the adversarial action w_t. If the feature map of the kernel is finite dimensional this is easy to handle, however when the feature map is infinite dimensional, this becomes challenging and we need to build an approximate feature map $\Phi_m(\cdot)$ using Algorithm 2.1. This introduces a bias in our estimate of the adversarial action w_t and we will need to control the contribution of the bias in our regret analysis. The second challenge will be to lower bound the minimum eigenvalue of the kernel covariance matrix as we will need to invert this matrix to estimate w_t. For general kernels which are infinite dimensional, the minimum eigenvalue is zero. To resolve this we will again turn to our construction of a finite dimensional proxy kernel.

2.3. Bandit Algorithm and Regret Bound

In our exponential weights algorithm we first build the finite dimensional proxy kernel $\hat{\mathcal{K}}_m$ using Algorithm 2.1. The rest of the algorithm is then almost identical to the exponential weights algorithm (EXP 2) studied for linear bandits in (20; 17; 13). In Algorithm 2.3 we set the exploration distribution $\nu_{\mathcal{J}}^{\mathcal{A}}$ to be such that it induces John's distribution ($\nu_{\mathcal{J}}$) over $\Phi_m(\mathcal{A}) := \{\Phi_m(a) \in \mathbb{R}^m : a \in \mathcal{A}\}$ (first introduced as an exploration distribution in (13); also a short discussion is presented in Appendix H.1). Note that for finite sets it is possible to build minimal volume ellipsoid containing $conv(\Phi_m(\mathcal{A}))$–John's ellipsoid and John's distribution in polynomial time (24)[6]. We assume without loss of generality that the center of the set $\mathcal{A}$ is such that the John's ellipsoid is centered at the origin.

If we know beforehand the behavior of the eigen-decay of the Mercer eigenvalues of $\mathcal{K}$ under measure μ we will be able to choose our tuning parameters optimally. In our algorithm we also build and invert the exact covariance matrix $\Sigma_m^{(t)}$, however this can be relaxed and we can work with a sample covariance matrix instead. We analyze the required sample complexity and error introduced by this additional step in Appendix D. We now state the main result of this paper which is an upper bound on the regret of Algorithm 2.3.

Theorem 10 *Let* μ_i *be the* i*-th Mercer operator eigenvalue of* $\mathcal{K}$ *for the uniform measure* μ *over* $\mathcal{A}$*. Let* m, p, α *and* ϵ *be chosen as specified by the conditions in Theorem 9. Let the mixing coefficient be chosen such that* $\gamma = \eta\mathcal{G}^4 m$*. Then Algorithm 2.3 with probability* $1 - e^{-\alpha}$ *has regret bounded by*

$$\mathcal{R}_n \leq \gamma n + (e-2)\mathcal{G}^4\eta m n + 3\epsilon n + \frac{1}{\eta}\log(|\mathcal{A}|).$$

We prove this theorem in Appendix A. Note that this is similar to the regret rate attained for adversarial linear bandits in (20; 18; 13) with an additional term $3\epsilon n$ that accounts for the bias in our loss estimates $\hat{w}_t$. In our regret bounds the parameter m plays the role of the effective dimension and will be determined by the rate of the eigen-decay of the kernel. When the underlying Hilbert space is finite dimensional (as is the case when the losses are linear) our regret

[6] It is thus possible to construct $\nu_{\mathcal{J}}$ over $\Phi_m(\mathcal{A})$ in polynomial time. However, as $\mathcal{A}$ is a finite set, using $\Phi_m(\cdot)$ and $\nu_{\mathcal{J}}$ it is also possible to construct $\nu_{\mathcal{J}}^{\mathcal{A}}$ efficiently.

Algorithm 2 Bandit Information: Exponential Weights

Input : Set $\mathcal{A}$, learning rate $\eta > 0$, mixing coefficient $\gamma > 0$, number of rounds n, uniform distribution μ over $\mathcal{A}$, exploration distribution $\nu_{\mathcal{J}}^{\mathcal{A}}$ over $\mathcal{A}$, kernel map $\mathcal{K}$, effective dimension $m(\epsilon)$, number of samples p.

Build kernel $\hat{\mathcal{K}}_m$ with feature map $\Phi_m(\cdot)$ using Algorithm 2.1 with kernel $\mathcal{K}$, dimension m, distribution μ, bias tolerance ϵ and number of samples p.

set $q_1(a) = \nu_{\mathcal{J}}^{\mathcal{A}}$.

for $t = 1, \ldots, n$ **do**

set $p_t = \gamma \nu_{\mathcal{J}}^{\mathcal{A}} + (1-\gamma) q_t$

choose $a_t \sim p_t$

observe $\langle \Phi(a_t), w_t \rangle_{\mathcal{H}}$

build the covariance matrix

$$\Sigma_m^{(t)} = \mathbb{E}_{x \sim p_t}\left[\Phi_m(x)\Phi_m(x)^\top\right]$$

compute the estimate $\hat{w}_t = \Sigma_m^{-1}\Phi_m(a_t)\langle \Phi(a_t), w_t\rangle_{\mathcal{H}}$.

update $q_{t+1}(a) \propto q_t(a) \cdot \exp\left(-\eta \cdot \langle \hat{w}_t, \Phi_m(a)\rangle_{\mathcal{H}}\right)$

end

bound recovers exactly the results of previous work (that is, $\epsilon = 0$ and $m = d$). Next we state the following different characteristic eigenvalue decay profiles.

Definition 11 (Eigenvalue decay) *Let the Mercer operator eigenvalues of a kernel $\mathcal{K}$ with respect to a measure $\mathbb{P}$ over a set $\mathcal{A}$ be denoted by $\mu_1 \geq \mu_2 \geq \ldots$.*

1. *$\mathcal{K}$ is said to have (C, β)-**polynomial eigenvalue decay** (with $\beta > 1$) if for all $j \in \mathbb{N}$ we have $\mu_j \leq C j^{-\beta}$.*
2. *$\mathcal{K}$ is said to have (C, β)-**exponential eigenvalue decay** if for all $j \in \mathbb{N}$ we have $\mu_j \leq C e^{-\beta j}$.*

Under assumptions on the eigen-decay we can establish bounds on the *effective dimension* m and μ_m, so that the condition stated in Lemma 6 is satisfied and we are guaranteed to build an ϵ-approximate kernel $\hat{\mathcal{K}}_m$. We establish bounds on m in Proposition 33 presented in Appendix C.1.

Corollary 12 *Let the conditions stated in Theorem 10 hold. Then Algorithm 2.3 has its regret bounded by the following rates with probability $1 - e^{-\alpha}$.*

1. *If $\mathcal{K}$ has (C, β)-polynomial eigenvalue decay under measure μ, with $\beta > 2$. Then by choosing $\eta = \frac{1}{3^{\frac{1}{2\beta-1}}} \cdot \left[\frac{\beta-1}{4C\mathcal{B}^2}\right]^{1/2\beta-1} \cdot \left[\frac{\log(|\mathcal{A}|)}{((e-1)\mathcal{G}^4)^{\frac{\beta-1}{\beta}} n}\right]^{\frac{\beta}{2\beta-1}}$ and $m = \left[\frac{4C\mathcal{B}^2}{(\beta-1)\epsilon}\right]^{1/\beta-1}$ where $\epsilon = \left(\frac{(e-1)\eta\mathcal{G}^4}{3}\right)^{(\beta-1)/\beta}\left[\frac{4C\mathcal{B}^2}{\beta-1}\right]^{1/\beta}$, the expected regret is bounded by*

$$\mathcal{R}_n \leq 3\left[\frac{4C\mathcal{B}^2}{\beta-1}\right]^{\frac{1}{2\beta-1}} \left(e\mathcal{G}^4 \log\left(|\mathcal{A}|\right)\right)^{\frac{\beta-1}{2\beta-1}} \cdot n^{\frac{\beta}{2(\beta-1)}}.$$

2. *If $\mathcal{K}$ has (C, β)-exponential eigenvalue decay under measure μ. Then by choosing $\eta = \left(\frac{\beta \log(|\mathcal{A}|)}{(e-1)\mathcal{G}^4 \log\left(\frac{4C\mathcal{B}^2}{\beta}\right)\cdot n}\right)^{1/2}$ and $m = \frac{1}{\beta}\log\left(\frac{4C\mathcal{B}^2}{\beta\epsilon}\right)$ where $\epsilon = \frac{(e-1)\mathcal{G}^4\eta}{3\beta}\log\left(\frac{4C\mathcal{B}^2}{\beta}\right)$, with n large enough so that $\epsilon < 1$, the expected regret is bounded by*

$$\mathcal{R}_n \leq \tilde{\mathcal{O}}\left(\left[\frac{18\mathcal{G}^4 \log\left(|\mathcal{A}|\right)\cdot n}{\beta \log\left(\frac{4C\mathcal{B}^2}{\beta}\right)}\right]^{1/2}\right).$$

Remark 13 *Under (C, β)-polynomial eigen-decay condition we have that the regret is upper bounded by $\mathcal{R}_n \leq \mathcal{O}(n^{\frac{\beta}{2(\beta-1)}})$. While when we have (C, β)-exponential eigen-decay we almost recover the adversarial linear bandits regret rate (up to logarithmic factors), with $\mathcal{R}_n \leq \mathcal{O}(n^{1/2}\log(n))$.*

One way to interpret the results of Corollary 12 in contrast to the regret bounds obtained for linear losses is the following. We introduce additional parameters into our analysis to handle the infinite dimensionality of our feature vectors – the effective dimension m and bias of our estimate ϵ. When the effective dimension m is chosen to be large we get can build an estimate of the adversarial action $\hat{w}_t$ which has low bias, however this estimate would have large variance ($\mathcal{O}(m)$). On the other hand if we choose m to be small we can build a low variance estimate of the adversarial action but with high bias (ϵ is large). We trade these off optimally to get the regret bounds established above. In the case of exponential decay we obtain that the choice $m = \mathcal{O}(\log(n))$ is optimal, hence the regret bound only degrades by a logarithmic factor in terms of n as compared to linear losses (where m would be a constant). When we have polynomial decay, the effective dimension is higher $m = \mathcal{O}(n^{\frac{1}{2(\beta-1)}})$ which leads to worse bounds on the expected regret. Note that asymptotically as $\beta \to \infty$ the regret bound goes to $n^{1/2}$ which aligns well with the intuition that the effective dimension is small. While when $\beta \to 2$ (the effective dimension $m \to \infty$) the regret bound becomes close to linear in n.

We can also show a minimax lower bound for these two settings that are close to matching the upper bound.

Proposition 14 *(informal) For any algorithm used by the player, there exist a strategy for the adversary such that $\mathcal{R}_n \geq \Omega\left(n^{\frac{\beta+1}{2\beta}}\right)$ whenever $\mu_j = \tilde{\mathcal{O}}(j^{-\beta})$, while when the decay is exponential $\mathcal{R}_n \geq \Omega\left(n^{1/2}\right)$.*

Algorithm 3 Full Information: Exponential Weights

Input : Set $\mathcal{A}$, learning rate $\eta > 0$, number of rounds n.
Set $p_1(a)$ uniform distribution over $\mathcal{A}$.
for $t = 1, \ldots, n$ **do**
  choose $a_t \sim p_t$
  observe w_t
  update $p_{t+1}(a) \propto p_t(a) \cdot \exp\left(-\eta \cdot \langle w_t, \Phi(a)\rangle_{\mathcal{H}}\right)$
end

The lower bound follows by a modification of the arguments used to prove a lower bound linear bandits. For a complete proof see Appendix E.

3. Full Information Setting

3.1. Full information Exponential Weights

We begin by presenting a version of the exponential weights algorithm, Algorithm 3 adapted to our setup. In each round we sample an action vector $a_t \in \mathcal{A}$ from the exponential weights distribution p_t. After observing the loss, $\langle \Phi(a_t), w_t\rangle_{\mathcal{H}}$ we update the distribution by a multiplicative factor, $\exp(-\eta\langle w_t, \Phi(a)\rangle_{\mathcal{H}})$. In the algorithm presented we choose the initial distribution $p_1(a)$ to be uniform over the set $\mathcal{A}$, however we note that alternate initial distributions with support over the whole set could also be considered. We can establish a sub-linear regret of $\mathcal{O}(\sqrt{n})$ for the exponential weights algorithm.

Theorem 15 *Assume that in Algorithm 3 the step size η is chosen to be, $\eta = \sqrt{\frac{\log(vol(\mathcal{A}))}{e-2}} \cdot \frac{1}{\mathcal{G}^2 n^{1/2}}$, with n large enough such that $\sqrt{\frac{\log(vol(\mathcal{A}))}{e-2}} \frac{1}{n^{1/2}} \leq 1$. Then the expected regret after n rounds is bounded by,*

$$\mathcal{R}_n \leq \sqrt{(e-2)\log(vol(\mathcal{A}))}\mathcal{G}^2 n^{1/2}.$$

We prove this regret bound in Appendix F.1.

3.2. Conditional Gradient Descent

Next we present an online conditional gradient (Frank-Wolfe) method (26) adapted for kernel losses. The conditional gradient method is also a well studied algorithm studied in both the online and offline setting (for a review see (25)). The main advantage of the conditional gradient method is that as opposed to projected gradient descent and related methods, the projection step is avoided. At each round the conditional gradient method involves the optimization of a linear (kernel) objective function over $\mathcal{A}$ to get a point $v_t \in \mathcal{A}$. Next we update the *optimal mean* action X_{t+1} by re-weighting the previous mean action X_t by $(1-\gamma_t)$ and weight our new action v_t by γ_t. Note that this construction also automatically suggests a distribution over $a_1, v_1, v_2, \ldots, v_t \in \mathcal{A}$ such that, X_{t+1} is a convex combination of $\Phi(a_1), \Phi(x_1), \ldots, \Phi(a_t)$. For this algorithm we can prove a regret bound of $\mathcal{O}(n^{3/4})$ (presented in Appendix F.2.).

Algorithm 4 Full Information: Conditional Gradient

Input : Set $\mathcal{A}$, number of rounds n, initial action $a_1 \in \mathcal{A}$, inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$, learning rate η, mixing rates $\{\gamma_t\}_{t=1}^n$.
$X_1 = \Phi(a_1)$
choose $\mathcal{D}_1$ such that $\mathbb{E}_{x\sim\mathcal{D}_1}\Phi(x) = X_1$
for $t = 1, 2, \ldots, n$ **do**
  sample $a_t \sim \mathcal{D}_t$
  observe the adversarial action w_t
  define $F_t(Y) \triangleq \eta \sum_{s=1}^{t-1}\langle w_s, Y\rangle_{\mathcal{H}} + \|Y - X_1\|_{\mathcal{H}}^2$
  compute $v_t = \operatorname{argmin}_{a\in\mathcal{A}}\langle \nabla F_t(X_t), \Phi(a)\rangle_{\mathcal{H}}$
  update mean $X_{t+1} = (1-\gamma_t)X_t + \gamma_t\Phi(v_t)$
  choose $\mathcal{D}_{t+1}$ s.t. $\mathbb{E}_{x\sim\mathcal{D}_{t+1}}[\Phi(x)] = X_{t+1}$.
end

Theorem 16 *Let the step size be $\eta = \frac{1}{2n^{3/4}}$. Also let the mixing rates be $\gamma_t = \min\{1, 2/t^{1/2}\}$, then Algorithm 4 attains regret of $\mathcal{R}_n \leq 8\mathcal{G}^2 n^{3/4}$.*

4. Applications

4.1. General Quadratic Losses

The first example of losses that we present are general quadratic losses. At each round the adversary can choose a symmetric (not necessarily positive semi-definite matrix) $A \in \mathbb{R}^{d\times d}$, and a vector $b \in \mathbb{R}^d$, with a constraint on the norm of the matrix and vector such that $\|A\|_F^2 + \|b\|_2^2 \leq \mathcal{G}^2$. If we embed this pair into a Hilbert space defined by the feature map (A, b) we get a kernel loss defined as – $\langle \Phi(x), (A, b)\rangle_{\mathcal{H}} = x^\top Ax + b^\top x$, where $\Phi(x) = (xx^\top, x)$ is the associated feature map for any $x \in \mathcal{A}$ and the inner product in the Hilbert space is defined as the concatenation of the trace inner product on the first coordinate and the Euclidean inner product on the second coordinate. The cumulative loss that the player would aspire to minimize is, $\sum_{t=1}^n x_t^\top A_t x_t + b_t^\top x_t$. The setting without the linear term, that is when $b_t = 0$ with positive semidefinite matrices A_t is previously well studied in (47; 48; 23; 4). However when the matrix is not positive semi-definite (making the losses non-convex) and there is a linear term, regret guarantees and tractable algorithms have not been studied even in the full information case.

As this is a kernel loss we have regret bounds for these losses. We demonstrate in the subsequent sections in the full information case it is also possible to run our algorithms efficiently. First for exponential weights we show sampling is efficient for these losses.

Lemma 17 (Proof in Appendix F.1) *Let $B \in \mathbb{R}^{d\times d}$ be a symmetric matrix and $b \in \mathbb{R}^d$. Sampling from $q(a) \propto \exp(a^\top Ba + a^\top b)$ for $\|a\|_2 \leq 1, a \in \mathbb{R}^d$ is tractable in $\tilde{\mathcal{O}}(d^4)$ time.*

4.2. Guarantees for Conditional Gradient Descent

We now demonstrate that conditional gradient descent also can be run efficiently when the adversary plays a general quadratic loss. At each round the conditional gradient descent requires the player to solve the optimization problem, $v_t = \operatorname{argmin}_{a\in\mathcal{A}}\langle \nabla F_t(X_t), \Phi(a)\rangle_{\mathcal{H}}$. When the set of actions is $\mathcal{A} = a \in \mathbb{R}^d : \|a\|_2 \leq 1$ then under quadratic losses this problem becomes,

$$v_t = \operatorname*{argmin}_{a\in\mathcal{A}} a^\top Ba + b^\top a, \tag{5}$$

for an appropriate matrix B and b that can be calculated by aggregating the adversary's actions up to step t. Observe that the optimization problem (5) is a quadratically constrained quadratic program (QCQP) given our choice of $\mathcal{A}$. The dual problem is the (semi-definite program) SDP,

$$\begin{aligned} &max - t - \mu \\ \text{s. t.} \\ &\begin{bmatrix} B + \mu I & b/2 \\ b/2 & t \end{bmatrix} \succ 0. \end{aligned}$$

For this particular program with a norm ball constraint set it is known the duality gap is zero provided Slater's condition holds, that is, strong duality holds (see Annex B.1 (12)).

4.3. Posynomial Losses

In this section we will define a *posynomial game*, by introducing posynomial losses and prove that these losses can also be viewed as kernel inner products. We will use the connection between posynomials and *Geometric programs* to prove that conditional gradient descent can be run efficiently on this family of losses.

Definition 18 (Monomial) *A function $f : \mathbb{R}^d_+ \mapsto \mathbb{R}$ defined as*

$$f(x) = cx_1^{\alpha_1}x_2^{\alpha_2}\cdots x_d^{\alpha_d},$$

where $c > 0$ and $\alpha_i \in \mathbb{R}$, is called a monomial function.

A sum of monomials is a posynomial.

Definition 19 (Posynomial) *A function $f : \mathbb{R}^d_+ \mapsto \mathbb{R}$ defined as*

$$f(x) = \sum_{k=1}^{m} c_k x_1^{\alpha_{1k}}x_2^{\alpha_{2k}}\cdots x_d^{\alpha_{dk}}$$

where $c_k > 0$ and $\alpha_{ik} \in \mathbb{R}$, is called a posynomial function.

Note that posynomial functions are closed under addition, multiplication and non-negative scaling. If we assume the adversary at each round plays a vector of dimension m with all non-negative entries, $w_t = (c_1, c_2, \cdots, c_m)$, while the player chooses a vector $x \in \mathbb{R}^d_+$. This vector is then partitioned into m parts,

$$x = (\underbrace{x_1, x_2}_{s_1}, \ldots, \underbrace{x_{d-2}, x_{d-1}, x_d}_{s_m}),$$

and the feature vector is defined as

$$\Phi(x) = \begin{bmatrix} x_1^{\alpha_1}x_2^{\alpha_2} \\ \vdots \\ x_{d-2}^{\alpha_{d-2}}x_{d-1}^{\alpha_{d-1}}x_d^{\alpha_d} \end{bmatrix}.$$

Where the i^{th} component of $\Phi(\cdot)$ is only a function of the i^{th} partition of the coordinates s_i. Then the loss obtained on the evaluation of the inner product between the adversary and player action is a posynomial loss function,

$$\langle w_t, \Phi(x)\rangle_{\mathcal{H}} = \sum_{k=1}^{m} c_k x_1^{\alpha_{k1}}\cdots x_d^{\alpha_{kd}}.$$

A number of scenarios can be modeled as a minimization/maximization problem over posynomial functions (see (11) for a detailed list of examples). We now show that conditional gradient descent can be run efficiently over posynomial losses. If we again assume that the set of actions $\mathcal{A} = \{a \in \mathbb{R}^d : \|a\|_2 \leq 1\}$. Additionally we all choose the initial action to be the solution to the optimization problem,

$$a_1 = \operatorname*{argmin}_{a\in\mathcal{A}} \sum_{k=1}^{d} \Phi(a)_i.$$

Observe that the objective function is a posynomial subject to a posynomial inequality constraint. This is a geometric program that can be solved efficiently by changing variables and converting into a convex program (Section 2.5 in (11)). At each round of the conditional gradient descent algorithm requires us to solve the optimization problem,

$$v_t = \operatorname*{argmin}_{a\in\mathcal{A}} \langle \eta \sum_{s=1}^{t-1} w_t + 2(X_t - \Phi(a_1)), \Phi(a)\rangle_{\mathcal{H}}. \tag{6}$$

Given that posynomials are closed under addition, and given our choice of a_1, the objective function (6) is still a posynomial and the constraint is a posynomial inequality. This can again be cast as a geometric program that can be solved efficiently at each round.

Conclusion

It would be interesting to explore and study more kernel losses for which we have regret guarantees and for which our algorithms are also computationally efficient.

References

[1] Jacob Duncan Abernethy, Elad Hazan, and Alexander Rakhlin. An efficient algorithm for bandit linear optimization. In *21st Annual Conference on Learning Theory*, 2008.

[2] Alekh Agarwal and Ofer Dekel. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *Conference on Learning Theory*, pages 28–40, 2010.

[3] Alekh Agarwal, Dean P Foster, Daniel J Hsu, Sham M Kakade, and Alexander Rakhlin. Stochastic convex optimization with bandit feedback. In *Advances in Neural Information Processing Systems*, pages 1035–1043, 2011.

[4] Zeyuan Allen-Zhu and Yuanzhi Li. Follow the compressed leader: Faster algorithms for matrix multiplicative weight updates. *arXiv preprint arXiv:1701.01722*, 2017.

[5] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: A meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[6] Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 227–236. ACM, 2007.

[7] Baruch Awerbuch and Robert D Kleinberg. Adaptive routing with end-to-end feedback: Distributed learning and geometric approaches. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 45–53. ACM, 2004.

[8] Maximilian Balandat, Walid Krichene, Claire Tomlin, and Alexandre Bayen. Minimizing regret on reflexive Banach spaces and learning Nash equilibria in continuous zero-sum games. *arXiv preprint arXiv:1606.01261*, 2016.

[9] Keith Ball. An elementary introduction to modern convex geometry. *Flavors of geometry*, 31:1–58, 1997.

[10] Peter Bartlett. Learning in sequential decision problems. Lecture Notes Stat 260/CS 294-102, 2014.

[11] Stephen Boyd, Seung-Jean Kim, Lieven Vandenberghe, and Arash Hassibi. A tutorial on geometric programming. *Optimization and engineering*, 8(1):67, 2007.

[12] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[13] Sébastien Bubeck, Nicolo Cesa-Bianchi, and Sham Kakade. Towards minimax policies for online linear optimization with bandit feedback. In *Annual Conference on Learning Theory*, volume 23, pages 41–1. Microtome, 2012.

[14] Sébastien Bubeck, Ofer Dekel, Tomer Koren, and Yuval Peres. Bandit convex optimization: $\sqrt{T}$ regret in one dimension. In *Conference on Learning Theory*, pages 266–278, 2015.

[15] Sébastien Bubeck, Ronen Eldan, and Yin Tat Lee. Kernel-based methods for bandit convex optimization. *arXiv preprint arXiv:1607.03084*, 2016.

[16] Nicolò Cesa-Bianchi, Pierre Gaillard, Claudio Gentile, and Sébastien Gerchinovitz. Algorithmic chaining and the role of partial feedback in online nonparametric learning. In *Conference on Learning Theory*, pages 465–481, 2017.

[17] Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

[18] Nicolo Cesa-Bianchi and Gábor Lugosi. Combinatorial bandits. *Journal of Computer and System Sciences*, 78(5):1404–1422, 2012.

[19] Nello Cristianini and John Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[20] Varsha Dani, Sham M Kakade, and Thomas P Hayes. The price of bandit information for online optimization. In *Advances in Neural Information Processing Systems*, pages 345–352, 2008.

[21] Ofer Dekel, Ronen Eldan, and Tomer Koren. Bandit smooth convex optimization: Improving the bias-variance tradeoff. In *Advances in Neural Information Processing Systems*, pages 2926–2934, 2015.

[22] Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 385–394. Society for Industrial and Applied Mathematics, 2005.

[23] Dan Garber, Elad Hazan, and Tengyu Ma. Online learning of eigenvectors. In *International Conference on Machine Learning*, pages 560–568, 2015.

[24] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

[25] Elad Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

[26] Elad Hazan and Satyen Kale. Projection-free online learning. *arXiv preprint arXiv:1206.4657*, 2012.

[27] Elad Hazan and Kfir Levy. Bandit convex optimization: Towards tight bounds. In *Advances in Neural Information Processing Systems*, pages 784–792, 2014.

[28] Elad Hazan and Yuanzhi Li. An optimal algorithm for bandit convex optimization. *arXiv preprint arXiv:1603.04350*, 2016.

[29] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

[30] Alan J Hoffman and Helmut W Wielandt. The variation of the spectrum of a normal matrix. In *Selected Papers Of Alan J Hoffman: With Commentary*, pages 118–120. World Scientific, 2003.

[31] Martin Jaggi. Convex optimization without projection steps. *arXiv preprint arXiv:1108.1170*, 2011.

[32] Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

[33] Robert D Kleinberg. Nearly tight bounds for the continuum-armed bandit problem. In *Advances in Neural Information Processing Systems*, pages 697–704, 2005.

[34] László Lovász and Santosh Vempala. The geometry of log-concave functions and sampling algorithms. *Random Structures & Algorithms*, 30(3):307–358, 2007.

[35] H Brendan McMahan and Avrim Blum. Online geometric optimization in the bandit setting against an adaptive adversary. In *International Conference on Computational Learning Theory*, pages 109–123. Springer, 2004.

[36] H Brendan McMahan and Francesco Orabona. Unconstrained online linear learning in Hilbert spaces: Minimax algorithms and normal approximations. In *Conference on Learning Theory*, pages 1020–1039, 2014.

[37] Shahar Mendelson, Alain Pajor, et al. On singular values of matrices with independent rows. *Bernoulli*, 12(5):761–773, 2006.

[38] James Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character*, 209:415–446, 1909.

[39] Jiazhong Nie, Wojciech Kotłowski, and Manfred K Warmuth. Online PCA with optimal regrets. In *International Conference on Algorithmic Learning Theory*, pages 98–112. Springer, 2013.

[40] Alexander Rakhlin and Karthik Sridharan. Online nonparametric regression with general loss functions. *arXiv preprint arXiv:1501.06598*, 2015.

[41] Ankan Saha and Ambuj Tewari. Improved regret guarantees for online smooth convex optimization with bandit feedback. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 636–642, 2011.

[42] Bernhard Scholkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[43] Shai Shalev-Shwartz and Yoram Singer. A primal-dual perspective of online learning algorithms. *Machine Learning*, 69(2-3):115–142, 2007.

[44] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

[45] Koji Tsuda, Gunnar Rätsch, and Manfred K Warmuth. Matrix exponentiated gradient updates for online learning and Bregman projection. *Journal of Machine Learning Research*, 6(Jun):995–1018, 2005.

[46] Michal Valko, Nathaniel Korda, Rémi Munos, Ilias Flaounas, and Nelo Cristianini. Finite-time analysis of kernelised contextual bandits. *arXiv preprint arXiv:1309.6869*, 2013.

[47] Manfred K Warmuth and Dima Kuzmin. Online variance minimization. In *International Conference on Computational Learning Theory*, pages 514–528. Springer, 2006.

[48] Manfred K Warmuth and Dima Kuzmin. Randomized online PCA algorithms with regret bounds that are logarithmic in the dimension. *Journal of Machine Learning Research*, 9(Oct):2287–2320, 2008.

[49] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936, 2003.

[50] Laurent Zwald and Gilles Blanchard. On the convergence of eigenspaces in kernel principal component analysis. In *Advances in neural information processing systems*, pages 1649–1656, 2006.