Second-order kernel online convex optimization with adaptive sketching

Daniele Calandriello, Alessandro Lazaric, Michal Valko
SequeL, Inria Lille – Nord Europe

ICML 2017

Linear Online Convex Optimization (LOCO)

Online game between learner and adversary, at each round $t \in [T]$

1. the adversary reveals a new point $\mathbf{x}_t \in \mathcal{X}$
2. the learner chooses a function $f_{\mathbf{w}_t}$ and predicts $f_{\mathbf{w}_t}(\mathbf{x}_t) = \mathbf{x}_t^\mathsf{T}\mathbf{w}_t$,
3. the adversary reveals the curved loss ℓ_t,
4. the learner suffers $\ell_t(\mathbf{x}_t^\mathsf{T}\mathbf{w}_t)$ and observes the associated gradient $\mathbf{g}_t$.

Kernel Online Convex Optimization (KOCO)

Online game between learner and adversary, at each round $t \in [T]$

1. the adversary reveals a new point $\mathbf{x}_t \in \mathcal{X}$
2. the learner chooses a function $f_{\mathbf{w}_t}$ and predicts $f_{\mathbf{w}_t}(\mathbf{x}_t) = \mathbf{x}_t^\mathsf{T}\mathbf{w}_t$,
3. the adversary reveals the curved loss ℓ_t,
4. the learner suffers $\ell_t(\mathbf{x}_t^\mathsf{T}\mathbf{w}_t)$ and observes the associated gradient $\mathbf{g}_t$.

Kernel

$\varphi(\cdot) : \mathcal{X} \to \mathcal{H}$ is the high-dimensional (possibly infinite) map

Kernel Online Convex Optimization (KOCO)

Online game between learner and adversary, at each round $t \in [T]$

1. the adversary reveals a new point $\varphi(\mathbf{x}_t) = \phi_t \in \mathcal{H}$
2. the learner chooses a function $f_{\mathbf{w}_t}$ and predicts $f_{\mathbf{w}_t}(\mathbf{x}_t) = \varphi(\mathbf{x}_t)^\mathsf{T}\mathbf{w}_t$,
3. the adversary reveals the curved loss ℓ_t,
4. the learner suffers $\ell_t(\phi_t^\mathsf{T}\mathbf{w}_t)$ and observes the associated gradient $\mathbf{g}_t$.

Kernel

$\varphi(\cdot) : \mathcal{X} \to \mathcal{H}$ is the high-dimensional (possibly infinite) map

Kernel Online Convex Optimization (KOCO)

Online game between learner and adversary, at each round $t \in [T]$

1. the adversary reveals a new point $\varphi(\mathbf{x}_t) = \phi_t \in \mathcal{H}$
2. the learner chooses a function $f_{\mathbf{w}_t}$ and predicts $f_{\mathbf{w}_t}(\mathbf{x}_t) = \varphi(\mathbf{x}_t)^\mathsf{T}\mathbf{w}_t$,
3. the adversary reveals the curved loss ℓ_t,
4. the learner suffers $\ell_t(\phi_t^\mathsf{T}\mathbf{w}_t)$ and observes the associated gradient $\mathbf{g}_t$.

Kernel

$\varphi(\cdot) : \mathcal{X} \to \mathcal{H}$ is the high-dimensional (possibly infinite) map

$\mathbf{\Phi}_t = [\phi_1, \ldots, \phi_t]$, $\mathbf{\Phi}_t^\mathsf{T}\mathbf{\Phi}_t = \mathbf{K}_t$ (kernel trick)

$\mathbf{g}_t = \ell_t'(\phi_t^\mathsf{T}\mathbf{w}_t)\phi_t := \dot{g}_t\phi_t$

Kernel Online Convex Optimization (KOCO)

Online game between learner and adversary, at each round $t \in [T]$

1. the adversary reveals a new point $\varphi(\mathbf{x}_t) = \phi_t \in \mathcal{H}$
2. the learner chooses a function $f_{\mathbf{w}_t}$ and predicts $f_{\mathbf{w}_t}(\mathbf{x}_t) = \varphi(\mathbf{x}_t)^\mathsf{T}\mathbf{w}_t$,
3. the adversary reveals the curved loss ℓ_t,
4. the learner suffers $\ell_t(\phi_t^\mathsf{T}\mathbf{w}_t)$ and observes the associated gradient $\mathbf{g}_t$.

Kernel

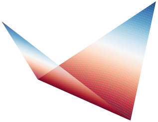
$\varphi(\cdot) : \mathcal{X} \to \mathcal{H}$ is the high-dimensional (possibly infinite) map

$\mathbf{\Phi}_t = [\phi_1, \ldots, \phi_t]$, $\mathbf{\Phi}_t^\mathsf{T}\mathbf{\Phi}_t = \mathbf{K}_t$ (kernel trick)

$\mathbf{g}_t = \ell_t'(\phi_t^\mathsf{T}\mathbf{w}_t)\phi_t := \dot{g}_t\phi_t$

Optimization to minimize regret $R(\mathbf{w}) = \sum_{t=1}^{T} \ell_t(\phi_t\mathbf{w}_t) - \ell_t(\phi_t\mathbf{w})$
and compete with best-in-hindsight $\mathbf{w}^* := \arg\min_{\mathbf{w}\in\mathcal{H}} \sum_{t=1}^{T} \ell_t(\phi_t\mathbf{w})$

Kernel Online Convex Optimization (KOCO)



convex

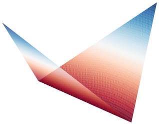
First order (GD) Zinkevich 2003, Kivinen et al. 2004:

↳ $\mathcal{O}(d)/\mathcal{O}(t)$ time/space per-step but slow rate

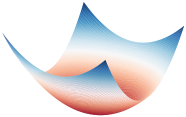
approximation avoids $\mathcal{O}(t)$ runtime dependency

↳ but introduce approximation error (potentially $\mathcal{O}(T)$ regret)

Kernel Online Convex Optimization (KOCO)



convex



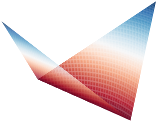
strongly convex

First order (GD) Hazan, Rakhlin, et al. 2008:

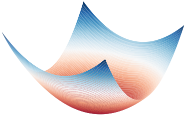
↳ $\mathcal{O}(d)/\mathcal{O}(t)$ time/space per-step and fast rate

but not satisfied in practice (e.g. $(y_t - \phi_t^\intercal \mathbf{w}_t)^2$)

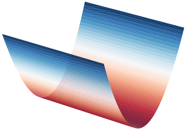
Kernel Online Convex Optimization (KOCO)



convex



strongly convex

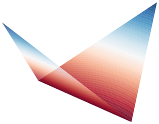


σ **curved**

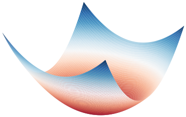
First order (GD)

↳ $\mathcal{O}(d)/\mathcal{O}(t)$ time/space per-step but slow rate

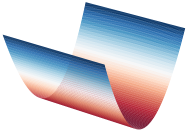
Kernel Online Convex Optimization (KOCO)



convex



strongly convex



σ **curved**

First order (GD)

↳ $\mathcal{O}(d)/\mathcal{O}(t)$ time/space per-step but slow rate

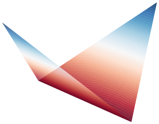
Second order Hazan, Kalai, et al. 2006, Zhdanov and Kalnishkan 2010:

↳ fast rate but $\mathcal{O}(d^2)/\mathcal{O}(t^2)$ time/space per-step

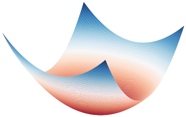
fast approximations for linear case Luo et al. 2016

no approximate methods for kernel case

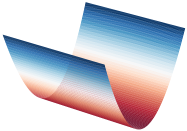
Kernel Online Convex Optimization (KOCO)



convex



strongly convex



σ **curved**

First order (GD)

↳ $\mathcal{O}(d)/\mathcal{O}(t)$ time/space per-step but slow rate

Second order Hazan, Kalai, et al. 2006, Zhdanov and Kalnishkan 2010:

↳ fast rate but $\mathcal{O}(d^2)/\mathcal{O}(t^2)$ time/space per-step

fast approximations for linear case Luo et al. 2016

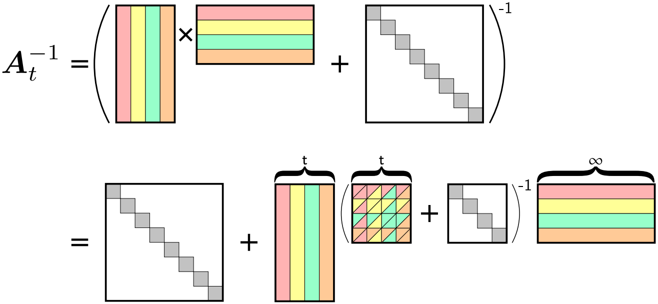
no approximate methods for kernel case

How to reduce computational cost without losing fast rate?

Second-Order KOCO (Kernel-Online Newton Step)

Second-Order Gradient Descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{A}_t^{-1}\mathbf{g}_t, \qquad \mathbf{A}_t = \mathbf{A}_{t-1} + \sigma \mathbf{g}_t \mathbf{g}_t^{\mathsf{T}}, \qquad \mathbf{A}_0 = \alpha \mathbf{I}$$



Second-Order KOCO (Kernel-Online Newton Step)

Second-Order Gradient Descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{A}_t^{-1}\mathbf{g}_t, \qquad \mathbf{A}_t = \mathbf{A}_{t-1} + \sigma \mathbf{g}_t \mathbf{g}_t^\mathsf{T}, \qquad \mathbf{A}_0 = \alpha \mathbf{I}$$

$$R(\mathbf{w}) \leq \mathcal{O}\left(\sum_{t=1}^{T} \mathbf{g}_t^\mathsf{T} \mathbf{A}_t^{-1} \mathbf{g}_t\right) \leq \mathcal{O}\left(L \sum_{t=1}^{T} \phi_t^\mathsf{T} \left(\mathbf{\Phi}_t \mathbf{\Phi}_t^\mathsf{T} + \alpha \mathbf{I}\right)^{-1} \phi_t\right)$$
$$\leq \mathcal{O}(\log(\mathrm{Det}(\mathbf{K}_T + \alpha \mathbf{I})))$$

Second-Order KOCO (Kernel-Online Newton Step)

Second-Order Gradient Descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{A}_t^{-1}\mathbf{g}_t, \qquad \mathbf{A}_t = \mathbf{A}_{t-1} + \sigma \mathbf{g}_t \mathbf{g}_t^\mathsf{T}, \qquad \mathbf{A}_0 = \alpha \mathbf{I}$$

$$R(\mathbf{w}) \leq \mathcal{O}\left(\sum_{t=1}^{T} \mathbf{g}_t^\mathsf{T} \mathbf{A}_t^{-1} \mathbf{g}_t\right) \leq \mathcal{O}\left(L \sum_{t=1}^{T} \phi_t^\mathsf{T} \left(\mathbf{\Phi}_t \mathbf{\Phi}_t^\mathsf{T} + \alpha \mathbf{I}\right)^{-1} \phi_t\right)$$
$$\leq \mathcal{O}(\log(\mathrm{Det}(\mathbf{K}_T + \alpha \mathbf{I}))) \leq \boxed{?}$$

Second-Order KOCO (Kernel-Online Newton Step)

Second-Order Gradient Descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{A}_t^{-1}\mathbf{g}_t, \qquad \mathbf{A}_t = \mathbf{A}_{t-1} + \sigma \mathbf{g}_t \mathbf{g}_t^\mathsf{T}, \qquad \mathbf{A}_0 = \alpha \mathbf{I}$$

$$R(\mathbf{w}) \leq \mathcal{O}\left(\sum_{t=1}^{T} \mathbf{g}_t^\mathsf{T} \mathbf{A}_t^{-1} \mathbf{g}_t\right) \leq \mathcal{O}\left(L \sum_{t=1}^{T} \underbrace{\phi_t^\mathsf{T} \left(\mathbf{\Phi}_t \mathbf{\Phi}_t^\mathsf{T} + \alpha \mathbf{I}\right)^{-1} \phi_t}_{\text{Ridge Leverage Score } \tau_{t,t}}\right)$$

$$\leq \mathcal{O}(\log(\mathrm{Det}(\mathbf{K}_T + \alpha \mathbf{I}))) \leq ?$$

Second-Order KOCO (Kernel-Online Newton Step)

Second-Order Gradient Descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{A}_t^{-1}\mathbf{g}_t, \qquad \mathbf{A}_t = \mathbf{A}_{t-1} + \sigma \mathbf{g}_t \mathbf{g}_t^\mathsf{T}, \qquad \mathbf{A}_0 = \alpha \mathbf{I}$$

$$R(\mathbf{w}) \leq \mathcal{O}\left(\sum_{t=1}^{T} \mathbf{g}_t^\mathsf{T} \mathbf{A}_t^{-1} \mathbf{g}_t\right) \leq \mathcal{O}\left(L \sum_{t=1}^{T} \underbrace{\phi_t^\mathsf{T} \left(\mathbf{\Phi}_t \mathbf{\Phi}_t^\mathsf{T} + \alpha \mathbf{I}\right)^{-1} \phi_t}_{\text{Ridge Leverage Score } \tau_{t,t}}\right)$$

$$\leq \mathcal{O}(\log(\mathrm{Det}(\mathbf{K}_T + \alpha \mathbf{I}))) \leq ?$$

Lemma: $\mathcal{O}(\log(\mathrm{Det}(\mathbf{K}_T + \alpha \mathbf{I}))) \leq 2\mathbf{d}_{\text{eff}}^{\mathbf{T}}(\alpha)\log(\mathbf{T}/\alpha)$.

Effective Dimension

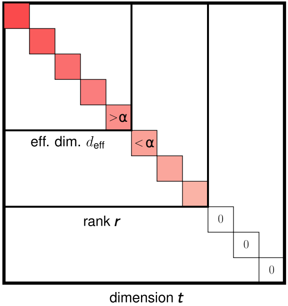
Formally $d_{\text{eff}}^{T}(\alpha)$ is an α soft-thresholded version of the rank defined as

$$d_{\text{eff}}^{T}(\alpha) = \operatorname{Tr}\left(\mathbf{K}_T\left(\mathbf{K}_T + \alpha \mathbf{I}\right)^{-1}\right) = \sum_{t=1}^{T} \frac{\lambda_t(\mathbf{K}_T)}{\lambda_t(\mathbf{K}_T) + \alpha} \leq \operatorname{Rank}(\mathbf{K}_T) = r$$

Effective Dimension

Formally $d_{\text{eff}}^{T}(\alpha)$ is an α soft-thresholded version of the rank defined as

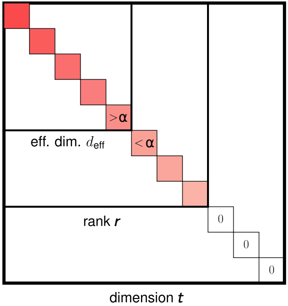
$$d_{\text{eff}}^{T}(\alpha) = \operatorname{Tr}\left(\mathbf{K}_T\left(\mathbf{K}_T + \alpha\mathbf{I}\right)^{-1}\right) = \sum_{t=1}^{T} \frac{\lambda_t(\mathbf{K}_T)}{\lambda_t(\mathbf{K}_T) + \alpha} \leq \operatorname{Rank}(\mathbf{K}_T) = r$$



Effective Dimension

Formally $d_{\text{eff}}^{T}(\alpha)$ is an α soft-thresholded version of the rank defined as

$$d_{\text{eff}}^{T}(\alpha) = \operatorname{Tr}\left(\mathbf{K}_T\left(\mathbf{K}_T + \alpha\mathbf{I}\right)^{-1}\right) = \sum_{t=1}^{T}\frac{\lambda_t(\mathbf{K}_T)}{\lambda_t(\mathbf{K}_T) + \alpha} \leq \operatorname{Rank}(\mathbf{K}_T) = r$$

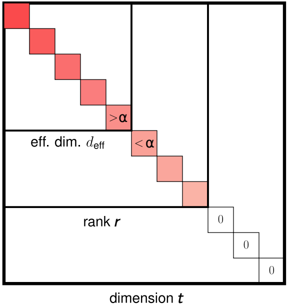


Intuitively, it quantifies the number of relevant orthogonal directions played by the adversary.

Effective Dimension

Formally $d_{\text{eff}}^{T}(\alpha)$ is an α soft-thresholded version of the rank defined as

$$d_{\text{eff}}^{T}(\alpha) = \operatorname{Tr}\left(\mathbf{K}_T\left(\mathbf{K}_T + \alpha\mathbf{I}\right)^{-1}\right) = \sum_{t=1}^{T} \frac{\lambda_t(\mathbf{K}_T)}{\lambda_t(\mathbf{K}_T) + \alpha} \leq \operatorname{Rank}(\mathbf{K}_T) = r$$

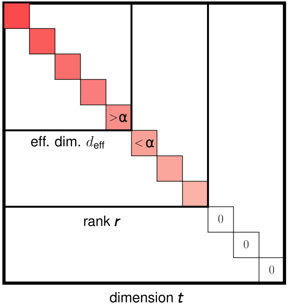


A direction (eigenvector) is relevant if its importance (eigenvalue) is larger than the regularization α

Effective Dimension

Formally $d_{\text{eff}}^T(\alpha)$ is an α soft-thresholded version of the rank defined as

$$d_{\text{eff}}^T(\alpha) = \operatorname{Tr}\left(\mathbf{K}_T\left(\mathbf{K}_T + \alpha\mathbf{I}\right)^{-1}\right) = \sum_{t=1}^{T} \frac{\lambda_t(\mathbf{K}_T)}{\lambda_t(\mathbf{K}_T) + \alpha} \leq \operatorname{Rank}(\mathbf{K}_T) = r$$



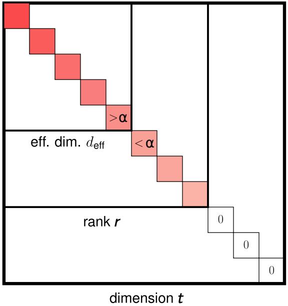
If all ϕ_t are orthogonal

$$d_{\text{eff}}^T(\alpha) \sim T/\alpha$$

Effective Dimension

Formally $d_{\text{eff}}^{T}(\alpha)$ is an α soft-thresholded version of the rank defined as

$$d_{\text{eff}}^{T}(\alpha) = \operatorname{Tr}\left(\mathbf{K}_T\left(\mathbf{K}_T + \alpha\mathbf{I}\right)^{-1}\right) = \sum_{t=1}^{T} \frac{\lambda_t(\mathbf{K}_T)}{\lambda_t(\mathbf{K}_T) + \alpha} \leq \operatorname{Rank}(\mathbf{K}_T) = r$$



If all ϕ_t are orthogonal

$$d_{\text{eff}}^{T}(\alpha) \sim T/\alpha$$

If all ϕ_t come from a bounded distribution or a finite set and $\alpha = 1$ then

$$d_{\text{eff}}^{T}(1) \sim \mathcal{O}(1) \leq r$$

is constant in T

Goal

How to maintain $d_{\text{eff}}^{T}(\alpha)\log(T)$ regret and reduce computational costs?

Goal

How to maintain $d_{\text{eff}}^{T}(\alpha)\log(T)$ regret and reduce computational costs?

Computation scales with number of vectors $\mathbf{g}_t\mathbf{g}_t$ added to $\mathbf{A}_t$

↳ skip some of the additions

Goal

How to maintain $d_{\text{eff}}^{T}(\alpha)\log(T)$ regret and reduce computational costs?

Computation scales with number of vectors $\mathbf{g}_t\mathbf{g}_t$ added to $\mathbf{A}_t$

↳ skip some of the additions

Regret is large when $\tau_{t,t}$ is large

↳ Update $\mathbf{A}_t$ only when $\tau_{t,t}$ is large

Goal

How to maintain $d_{\text{eff}}^T(\alpha)\log(T)$ regret and reduce computational costs?

Computation scales with number of vectors $\mathbf{g}_t\mathbf{g}_t$ added to $\mathbf{A}_t$

↳ skip some of the additions

Regret is large when $\tau_{t,t}$ is large

↳ Update $\mathbf{A}_t$ only when $\tau_{t,t}$ is large

Computing $\tau_{t,t}$ is expensive

↳ Propose Kernel Online Row Sampling (KORS)
to approximate $\tau_{t,t}$ efficiently

Naive Approach

Unbiased estimator:

$$\widetilde{\mathbf{A}}_t = \widetilde{\mathbf{A}}_{t-1} + (\mathbb{I}\{\text{coin flip w.p. } p_t\}/\mathbf{p_t})\sigma \mathbf{g}_t \mathbf{g}_t^\mathsf{T}$$

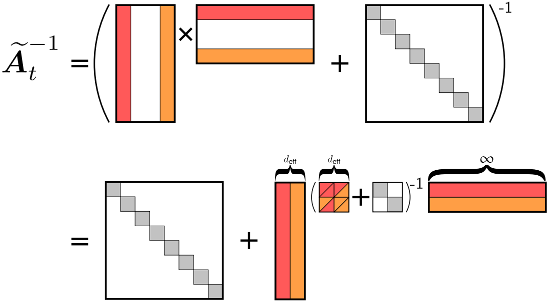
with $p_t \propto \widetilde{\tau}_{t,t}$

Naive Approach

Unbiased estimator:

$$\widetilde{\mathbf{A}}_t = \widetilde{\mathbf{A}}_{t-1} + (\mathbb{I}\{\text{coin flip w.p. } p_t\}/\mathbf{p_t})\sigma \mathbf{g}_t \mathbf{g}_t^\mathsf{T}$$

with $p_t \propto \widetilde{\tau}_{t,t}$



Naive Approach

Unbiased estimator:

$$\widetilde{\mathbf{A}}_t = \widetilde{\mathbf{A}}_{t-1} + (\mathbb{I}\{\text{coin flip w.p. } p_t\}/\mathbf{p_t})\sigma \mathbf{g}_t \mathbf{g}_t^\mathsf{T}$$

with $p_t \propto \widetilde{\tau}_{t,t}$

Pros:

w.h.p. $\widetilde{A}_t$ updated only $d_{\text{eff}}^T(\alpha)\log^2(T)$ times

$\widetilde{\mathcal{O}}(d_{\text{eff}}^T(\alpha)^2 + t)$ per-step space/time complexity

Cons:

Expected regret $d_{\text{eff}}^T(\alpha)\log(T)$

The weights $1/p_t \sim 1/\widetilde{\tau}_{t,t}$ can be large

↳ large updates $\widetilde{\mathbf{A}}_t - \widetilde{\mathbf{A}}_{t-1}$ and large variance

Naive Approach

biased estimator:

$$\widetilde{\mathbf{A}}_t = \widetilde{\mathbf{A}}_{t-1} + (\mathbb{I}\{\text{coin flip w.p. } p_t\} \quad)\sigma \mathbf{g}_t \mathbf{g}_t^\intercal$$

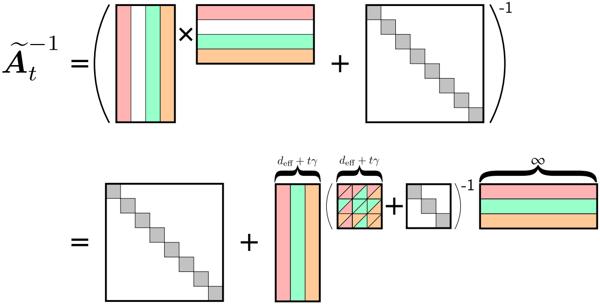
with $p_t \propto \widetilde{\tau}_{t,t}$

Sketched-KONS

SKETCHED-KONS:

$$\widetilde{\mathbf{A}}_t = \widetilde{\mathbf{A}}_{t-1} + (\mathbb{I}\{\text{coin flip w.p. } p_t\} \quad)\sigma \mathbf{g}_t \mathbf{g}_t^{\mathsf{T}}$$

with $p_t \propto \max\{\gamma, \widetilde{\tau}_{t,t}\}$



Sketched-KONS

SKETCHED-KONS:

$$\widetilde{\mathbf{A}}_t = \widetilde{\mathbf{A}}_{t-1} + (\mathbb{I}\{\text{coin flip w.p. } p_t\} \quad)\sigma \mathbf{g}_t \mathbf{g}_t^\mathsf{T}$$

with $p_t \propto \max\{\gamma, \widetilde{\tau}_{t,t}\}$

Theorem: with high probability SKETCHED-KONS

(1) achieves $d_{\text{eff}}^T(\alpha)\log(T)/\max\{\gamma, \tau_{\min}\}$ regret

(2) requires only $\widetilde{\mathcal{O}}(\mathbf{d}_{\text{eff}}^{\mathbf{T}}(\alpha)^{\mathbf{2}} + \mathbf{t}^{\mathbf{2}}\gamma^{\mathbf{2}} + t)$ per-step space/time

Trade-off $1/\gamma$ increase in regret for γ^2 space/time improvement
Still dependent on t

What next?

Can we get rid of dependency on t without losing fast rates?

Skipping updates not enough:

↳ we propose a counterexample

Different approaches?

Learn how to remove old $\mathbf{g}_s\mathbf{g}_s^\top$ from $\mathbf{A}_t$?

Instead of approximating $\mathbf{A}_t$, approximate ϕ_t

↳ Random feature not strong enough yet
Avron et al. ICML 17 for batch setting)
Ongoing work using RLS sampling and Nyström embeddings

Bibliography I

Elad Hazan, Adam Kalai, Satyen Kale, and Amit Agarwal. "Logarithmic regret algorithms for online convex optimization". In: Conference on Learning Theory. Springer, 2006, pp. 499–513.

Elad Hazan, Alexander Rakhlin, and Peter L Bartlett. "Adaptive online gradient descent". In: Advances in Neural Information Processing Systems. 2008, pp. 65–72.

J. Kivinen, A.J. Smola, and R.C. Williamson. "Online Learning with Kernels". en. In: IEEE Transactions on Signal Processing 52.8 (Aug. 2004). (Visited on 02/18/2017).

Haipeng Luo, Alekh Agarwal, Nicolo Cesa-Bianchi, and John Langford. "Efficient second-order online learning via sketching". In: Neural Information Processing Systems. 2016.

Bibliography II

Fedor Zhdanov and Yuri Kalnishkan. “An Identity for Kernel Ridge Regression”. en. In: Algorithmic Learning Theory. Ed. by Springer. Lecture Notes in Computer Science. Oct. 2010, pp. 405–419.

Martin Zinkevich. “Online Convex Programming and Generalized Infinitesimal Gradient Ascent”. In: International Conference on Machine Learning. 2003, pp. 928–936.