Nearly Optimal Algorithms with Sublinear Computational Complexity for Online Kernel Regression

Junfan Li [1] **Shizhong Liao** [1]

Abstract

The trade-off between regret and computational cost is a fundamental problem for online kernel regression, and previous algorithms worked on the trade-off can not keep optimal regret bounds at a sublinear computational complexity. In this paper, we propose two new algorithms, AOGD-ALD and NONS-ALD, which can keep nearly optimal regret bounds at a sublinear computational complexity, and give sufficient conditions under which our algorithms work. Both algorithms dynamically maintain a group of nearly orthogonal basis used to approximate the kernel mapping, and keep nearly optimal regret bounds by controlling the approximate error. The number of basis depends on the approximate error and the decay rate of eigenvalues of the kernel matrix. If the eigenvalues decay exponentially, then AOGD-ALD and NONS-ALD separately achieves a regret of $O(\sqrt{L(f)})$ and $O(\mathrm{d}_{\mathrm{eff}}(\mu)\ln T)$ at a computational complexity in $O(\ln^2 T)$. If the eigenvalues decay polynomially with degree $p \geq 1$, then our algorithms keep the same regret bounds at a computational complexity in $o(T)$ in the case of $p > 4$ and $p \geq 10$, respectively. $L(f)$ is the cumulative losses of f and $\mathrm{d}_{\mathrm{eff}}(\mu)$ is the effective dimension of the problem. The two regret bounds are nearly optimal and are not comparable.

1. Introduction

Online kernel learning in the regime of the square loss is an important non-parametric online learning method (Kivinen et al., 2004; Vovk, 2006; Sahoo et al., 2014). The learning protocol can be formulated as a game between a learner and an adversary. Before the game, the learner selects a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ induced by a positive semidefinite kernel function (Aronszajn, 1950; Shawe-Taylor & Cristianini, 2004). At each round $t = 1, 2, \ldots,$, the adversary sends an instance $\mathbf{x}_t \in \mathbb{R}^d$ to the learner. Then the learner chooses a hypothesis $f_t \in \mathbb{H} \subset \mathcal{H}$ and output $f_t(\mathbf{x}_t)$. After that the adversary reveals the true output y_t. The learner suffers a loss $\ell(f_t(\mathbf{x}_t), y_t)$. The goal is to minimize the *regret* defined as follows

$$\forall f \in \mathbb{H},\ \mathrm{Reg}(f) = \sum_{t=1}^{T}[\ell(f_t(\mathbf{x}_t), y_t) - \ell(f(\mathbf{x}_t), y_t)]. \quad (1)$$

One of the challenges minimizing the regret is to balance the computational cost. Kernel online gradient descent (KOGD) enjoys a regret of $O(\sqrt{L(f)})$ at a computational complexity (space and per-round time) in $O(dT)$ (Zinkevich, 2003; Srebro et al., 2010; Zhang et al., 2019), where $L(f) = \sum_{t=1}^{T} \ell(f(\mathbf{x}_t), y_t)$. $O(\sqrt{L(f)})$ implies the "small-loss" bound (Wang et al., 2020; Zhang et al., 2022). Kernel online Newton step (KONS) (Calandriello et al., 2017b) enjoys a regret of $O(\mu\|f\|_{\mathcal{H}}^2 + \mathrm{d}_{\mathrm{eff}}(\mu)\ln T)$ at a computational complexity in $O(T^2)$, where $\mu > 0$ is a regularization parameter and $\mathrm{d}_{\mathrm{eff}}(\mu)$ is called *effective dimension* depending on the decay rate of eigenvalues of the kernel matrix (Caponnetto & Vito, 2007; Rudi et al., 2015). The KAAR algorithm (Gammerman et al., 2004) and kernel ridge regression algorithm (Zhdanov & Kalnishkan, 2013) enjoy the same regret bound and computational complexity with KONS. If the eigenvalues decay exponentially, then $\mathrm{d}_{\mathrm{eff}}(\mu) = O(\ln \frac{T}{\mu})$ (Li et al., 2019). If the eigenvalues decay polynomially with degree $p \geq 1$, then $\mathrm{d}_{\mathrm{eff}}(\mu) = O((T/\mu)^{1/p})$ (Jézéquel et al., 2019a).

The $O(dT)$ and $O(T^2)$ computational complexities are prohibitive. Some approximate algorithms reduce the computational complexity at the expense of regret (Lu et al., 2016; Calandriello et al., 2017b;a). The FOGD algorithm approximating KOGD, achieves a regret of $\tilde{O}(\sqrt{TL(f)/D})$ at a computational complexity in $O(dD)$ where D is a tunable parameter (Lu et al., 2016). Achieving the optimal regret bound requires $D = \Omega(T)$. The Sketched-KONS algorithm approximating KONS, reduces the computational complexity by a factor of γ^{-2}, but increases the regret by $\gamma > 1$ (Calandriello et al., 2017b). The PROS-N-KONS

[1]College of Intelligence and Computing, Tianjin University, Tianjin 300350, China. Correspondence to: Shizhong Liao <szliao@tju.edu.cn>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

algorithm approximating KONS, increases the regret by a factor of $\tilde{O}(\mathrm{d}_{\mathrm{eff}}(\alpha))$ and suffers a space complexity in $\tilde{O}(\mathrm{d}_{\mathrm{eff}}(\alpha)^2)$ and an average per-round time complexity in $\tilde{O}(\mathrm{d}_{\mathrm{eff}}(\alpha)^2 + \mathrm{d}_{\mathrm{eff}}(\alpha)^4/T)$ (Calandriello et al., 2017a), where $\alpha > 0$. Although Sketched-KONS and PROS-N-KONS can ensure a $o(T)$ computational complexity, they can not achieve the optimal regret bound. The PKAWV algorithm keeps the regret of KONS at a computational complexity in $\tilde{O}(T\mathrm{d}_{\mathrm{eff}}(\alpha) + \mathrm{d}^2_{\mathrm{eff}}(\alpha))$ (Jézéquel et al., 2019a). Although PKAWV reduces the $O(T^2)$ computational complexity, it can not ensure a $o(T)$ computational complexity. Besides, PKAWV must store all of the observed examples.

In summary, existing approximate algorithms can not achieve nearly optimal regret bounds and a $o(T)$ computational complexity simultaneously. It is important to rise the question: *Is it possible to achieve nearly optimal regret bounds at a computational complexity in $o(T)$?* To be specific, the question is equivalent to the following two. (1) Is it possible to achieve a regret of $O(\sqrt{L(f)})$ at a $o(T)$ computational complexity? $O(\sqrt{L(f)})$ matches the lower bound in the stochastic setting (Srebro et al., 2010). (2) Is it possible to achieve a regret of $O(\mu\|f\|^2_{\mathcal{H}} + \mathrm{d}_{\mathrm{eff}}(\mu)\ln T)$ at a $o(T)$ computational complexity? The regret bound is optimal up to $\ln T$ (Jézéquel et al., 2019a). If the eigenvalues of the kernel matrix decay exponentially, then $O(\mu\|f\|^2_{\mathcal{H}} + \mathrm{d}_{\mathrm{eff}}(\mu)\ln T) = O(\ln^2 T)$. If the eigenvalues decay polynomially with degree $p \geq 1$, then $O(\mu\|f\|^2_{\mathcal{H}} + \mathrm{d}_{\mathrm{eff}}(\mu)\ln T) = O(T^{\frac{1}{1+p}}\ln T)$ where $\mu = T^{\frac{1}{1+p}}$.

1.1. Main Results

In this paper, we propose two algorithms, AOGD-ALD and NONS-ALD, and give conditions under which the answers are affirmative. The computational complexities of both algorithms depend on the decay rate of eigenvalues of the kernel matrix. If the eigenvalues decay exponentially, then AOGD-ALD and NONS-ALD separately achieves a regret of $O(\sqrt{L(f)})$ and $O(\mathrm{d}_{\mathrm{eff}}(\mu)\ln T)$ at a computational complexity in $O(\ln^2 T)$. If the eigenvalues decay polynomially with degree $p \geq 1$, then AOGD-ALD keeps the same regret bound at a computational complexity in $O(\min\{dT^{\frac{2}{p}} + T^{\frac{4}{p}}, dT\})$, and NONS-ALD achieves a regret of $O(T^{\frac{1}{1+p}}\ln T)$ at a space complexity in $O(T^{\frac{2(1+5p)}{p(1+p)}})$ and an average per-round time complexity in $O(T^{\frac{2(1+5p)}{p(1+p)}} + T^{\frac{4(1+5p)}{p(1+p)}-1})$. AOGD-ALD and NONS-ALD achieve a computational complexity in $o(T)$ in the case of $p > 4$ and $p \geq 10$, respectively. We summary the related results in Table 1.

1.2. Technical Contributions

AOGD-ALD approximates KOGD and NONS-ALD approximates KONS. We use the approximate linear dependence condition (Engel et al., 2004) to dynamically maintain a group of nearly orthogonal basis. The computational complexities of our algorithms have a quadratic dependence on the number of basis which depends on the decay rate of eigenvalues of the kernel matrix (Li & Liao, 2022). For AOGD-ALD, we use the orthogonal basis to approximate the gradients. For NONS-ALD, we use the Nyström projection with the orthogonal basis to construct explicit feature mapping. Since the number of basis may grow with t, the feature mapping must change dynamically. The first technical challenge is how to incrementally update the model parameter $\mathbf{w}_t$ and the covariance matrix $\mathbf{A}_t$ when the explicit feature mapping changes. Our first technical contribution is a projection scheme which projects $\mathbf{w}_t \in \mathbb{R}^j$ onto $\mathbb{R}^{j+1}$, and projects $\mathbf{A}_t \in \mathbb{R}^{j\times j}$ onto $\mathbb{R}^{(j+1)\times(j+1)}$. The regret analysis is also challenging, since it requires to control the regret induced by the projection. Our second technical contribution is a non-trivial and novel analysis for the regret induced by the projection. We proved that it only depends on the error related to the ALD condition and can be omitted by controlling the error. The approximate scheme of NONS-ALD provides a new approach for both online and offline kernel learning which might be of independent interest.

2. Preliminary and Problem Setting

Let $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^d | \|\mathbf{x}\|_2 < \infty\}$ and $\mathcal{I}_T = \{(\mathbf{x}_t, y_t)_{t\in[T]}\}$ be a sequence of examples, where $[T] = \{1, \ldots, T\}$, $\mathbf{x}_t \in \mathcal{X}, |y_t| \leq Y$. Let $\kappa(\cdot,\cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ be a positive semidefinite kernel function. We assume that κ is normalized and $\kappa(\mathbf{x}, \mathbf{x}) = 1$. Denote by $\mathcal{H}$ the RKHS associated with κ, such that (i) $\langle f, \kappa(\mathbf{x}, \cdot)\rangle_{\mathcal{H}} = f(\mathbf{x})$; (ii) $\mathcal{H} = \overline{\mathrm{span}(\kappa(\mathbf{x}_t, \cdot) : t \in [T])}$. We define $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ as the inner product in $\mathcal{H}$, which induces the norm $\|f\|_{\mathcal{H}} = \sqrt{\langle f, f\rangle_{\mathcal{H}}}$. Denote by $\mathbb{H} = \{f \in \mathcal{H} | \|f\|_{\mathcal{H}} \leq U\}$. U is a constant. The square loss function is $\ell(f(\mathbf{x}), y) = (f(\mathbf{x}) - y)^2$.

2.1. Effective Dimension

κ induces an implicit feature mapping $\phi(\cdot) : \mathcal{X} \to \mathbb{R}^n$, where n may be infinite. The orthogonality of $\{\phi(\mathbf{x}_t)\}_{t=1}^T$ characterizes the hardness of the data. A usual measure of the orthogonality is the effective dimension (Calandriello et al., 2017b).

Definition 2.1 (μ-effective dimension)**.** Given instances $\{\mathbf{x}_\tau\}_{\tau=1}^T$, a kernel function κ and a regularization parameter $\mu > 0$, the ridge leverage scores (RLS) of $\mathbf{x}_\tau$ is defined by

$$r_{T,\tau}(\mu) = \mathbf{e}_{T,\tau}^\top \mathbf{K}_T(\mathbf{K}_T + \mu\mathbf{I}_T)^{-1}\mathbf{e}_{T,\tau},\ \tau = 1, \ldots, T,$$

where $\mathbf{K}_T$ is the kernel matrix and $\mathbf{e}_{T,\tau} \in \{0,1\}^T$. Only the τ-th element of $\mathbf{e}_{T,\tau}$ is one. The μ-effective dimension is $\mathrm{d}_{\mathrm{eff}}(\mu) := \sum_{\tau=1}^T r_{T,\tau}(\mu) = \mathrm{tr}(\mathbf{K}_T(\mathbf{K}_T + \mu\mathbf{I}_T)^{-1})$.

Let $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_T$ be the eigenvalues of $\mathbf{K}_T$. If

Eigenvalues condition	Algorithm	Regret bound	Computational complexity	#Buffer
decay exponentially: $\exists r \in (0,1), R_0 = \Theta(T)$, s.t. $\forall i \in [T], \lambda_i \leq R_0 r^i$	PKAWV	$O(\ln^2 T)$	$O(T \ln^3 T)$	T
	Sketched-KONS	$O(\gamma \ln^2 T)$	$O(T^2/\gamma^2)$	$O(T/\gamma)$
	Pros-N-KONS	$O(\ln^5 T)$	$O(\ln^6 T)$	$O(\ln^3 T)$
	FOGD	$\tilde{O}(\sqrt{TL(f)/D})$	$O(dD)$	0
	AOGD-ALD	$O(\sqrt{L(f)})$	$O(\ln^2 T)$	$O(\ln T)$
	NONS-ALD	$O(\ln^2 T)$	$O(\ln^2 T)$	$O(\ln T)$
decay polynomially: $\exists p \geq 1, R_0 = \Theta(T)$, s.t. $\forall i \in [T], \lambda_i \leq R_0 i^{-p}$	PKAWV	$O(T^{\frac{1}{1+p}} \ln T)$	$\tilde{O}(T^{2r} + T^{1+r}), r = \frac{2p}{p^2-1}$	T
	Sketched-KONS	$O(\gamma T^{\frac{1}{1+p}} \ln T)$	$O(T^2/\gamma^2)$	$O(T/\gamma)$
	Pros-N-KONS	$O(T^{\frac{3p+1}{(1+p)^2}} \ln^2 T)$	$O(T^{\frac{4p}{(1+p)^2}} \ln^4 T)$	$O(T^{\frac{2p}{(1+p)^2}} \ln^2 T)$
	FOGD	$\tilde{O}(\sqrt{TL(f)/D})$	$O(dD)$	0
	AOGD-ALD	$O(\sqrt{L(f)})$	$O(dT^{\frac{2}{p}} + T^{\frac{4}{p}}), p > 4$	$O(T^{\frac{2}{p}})$
	NONS-ALD	$O(T^{\frac{1}{1+p}} \ln T)$	$O(T^{\frac{2(1+5p)}{p(1+p)}}), p \geq 10$	$O(T^{\frac{1+5p}{p(1+p)}})$

Table 1. Regret bound and computational complexity (space complexity and (averaged) per-round time complexity) of online kernel regression algorithms. $\{\lambda_i\}_{i=1}^T$ are the eigenvalues of the kernel matrix. #Buffer is the number of stored examples.

λ_i decays exponentially, then $\mathrm{d}_{\mathrm{eff}}(\mu) = O(\ln \frac{T}{\mu})$ (Li et al., 2019). If λ_i decays polynomially with degree $p \geq 1$, then $\mathrm{d}_{\mathrm{eff}}(\mu) = O((T/\mu)^{1/p})$ (Jézéquel et al., 2019a).

2.2. Online Kernel Regression

The protocol of online kernel regression is as follows: at any round t, an adversary sends an instance $\mathbf{x}_t \in \mathcal{X}$. An learner chooses a hypothesis $f_t \in \mathcal{H}$, and makes the prediction $\hat{y}_t = f_t(\mathbf{x}_t)$. Then the adversary reveals the true output y_t. We aim to minimize the regret w.r.t. any $f \in \mathbb{H}$, denoted by $\mathrm{Reg}(f)$ defined in (1). It is worth mentioning that competing with $f \in \mathbb{H}$ does not weaken the definition of regret. Note that $|y_t| \leq Y$. It is natural to require $|f(\mathbf{x}_t)| \leq Y$. Since $f(\mathbf{x}_t) \leq \|f\|_{\mathcal{H}} \cdot \|\kappa(\mathbf{x}_t, \cdot)\|_{\mathcal{H}} \leq \|f\|_{\mathcal{H}}$. We only need to consider all f such that $\|f\|_{\mathcal{H}} \leq Y$. To this end, we can define $U \geq Y$.

3. Approximating KOGD

In this section, we propose a deterministic approximation of KOGD, named AOGD-ALD.

3.1. Algorithm

According to the protocol of online kernel regression, the key is to compute f_{t+1} from f_t. KOGD (Zinkevich, 2003) executes the following update rule,

$$\bar{f}_{t+1} = f_t - \eta_t \nabla \ell(f_t(\mathbf{x}_t), y_t), \tag{2}$$

$$f_{t+1} = \min\left\{1, \frac{U}{\|\bar{f}_{t+1}\|_{\mathcal{H}}}\right\} \bar{f}_{t+1}, \tag{3}$$

where $\nabla \ell(f_t(\mathbf{x}_t), y_t) = \ell'(f_t(\mathbf{x}_t), y_t)\kappa(\mathbf{x}_t, \cdot)$ and η_t is a time-variant learning rate. Note that $\ell'(f_t(\mathbf{x}_t), y_t) = 2(f_t(\mathbf{x}_t) - y_t)$. f_{t+1} can be recursively rewritten as $f_{t+1} = \sum_{\tau=1}^t a_\tau \kappa(\mathbf{x}_\tau, \cdot)$. To store f_{t+1}, we must store some observed examples, denoted by $S_{t+1} = \{(\mathbf{x}_\tau, y_\tau), \tau \leq t : a_\tau \neq 0\}$. For simplicity, we call S_{t+1} the buffer. The computational complexity is $O(dt)$. To reduce the computational cost, we must limit the size of S_{t+1}. Next, we use the approximate linear dependence (ALD) condition (Engel et al., 2004) to maintain S_{t+1}.

At the beginning of round t, let S_t be the buffer. If $\nabla \ell(f_t(\mathbf{x}_t), y_t) \neq 0$, then we must decide whether $(\mathbf{x}_t, y_t)$ will be added into S_t. The ALD condition measures whether $\kappa(\mathbf{x}_t, \cdot)$ is approximate linear dependence with $\mathbf{\Phi}_{S_t} = (\kappa(\mathbf{x}, \cdot)_{\mathbf{x} \in S_t})$. We compute the projection error

$$\left(\min_{\boldsymbol{\beta} \in \mathbb{R}^{|S_t|}} \|\mathbf{\Phi}_{S_t}\boldsymbol{\beta} - \kappa(\mathbf{x}_t, \cdot)\|_{\mathcal{H}}^2\right) =: \alpha_t. \tag{4}$$

The solution [1] is

$$\boldsymbol{\beta}_t^* = \mathbf{K}_{S_t}^{-1} \mathbf{\Phi}_{S_t}^\top \kappa(\mathbf{x}_t, \cdot),$$

where $\mathbf{K}_{S_t}$ is the kernel matrix defined on S_t. We introduce a threshold for α_t and define the ALD condition as follows

$$\mathrm{ALD}_t : \alpha_t \leq \alpha, \ \alpha \in (0, 1]. \tag{5}$$

If ALD_t holds, then $\kappa(\mathbf{x}_t, \cdot)$ can be well approximated by $\mathbf{\Phi}_{S_t}\boldsymbol{\beta}_t^*$. Thus we replace (2) with (6),

$$\bar{f}_{t+1} = f_t - \eta_t \ell'(f_t(\mathbf{x}_t), y_t) \cdot \mathbf{\Phi}_{S_t}\boldsymbol{\beta}_t^*. \tag{6}$$

In this case, we do not add $(\mathbf{x}_t, y_t)$ into S_t, i.e., $S_{t+1} = S_t$.

If ALD_t does not hold, that is, $\kappa(\mathbf{x}_t, \cdot)$ can not be well approximated by $\mathbf{\Phi}_{S_t}\boldsymbol{\beta}_t^*$, then we still execute (2). In this case, we add $(\mathbf{x}_t, y_t)$ into S_t, i.e., $S_{t+1} = S_t \cup \{(\mathbf{x}_t, y_t)\}$.

[1] If $S_t = \emptyset$, then we set $\boldsymbol{\beta}_t^* = 0$ and $\alpha_t = 1$.

Algorithm 1 AOGD-ALD

Input: U, α, B_0.
Initialize: $f_1 = 0$
for $t = 1, \ldots, T$ **do**
  Receive $\mathbf{x}_t$
  Compute $\hat{y}_t = f_t(\mathbf{x}_t)$
  Compute η_t
  if $|S_t| < B_0$ **then**
    Compute α_t
    if ALD_t holds **then**
      Compute f_{t+1} following (6) and (3)
    else
      Compute f_{t+1} following (2) and (3)
      Update $S_{t+1} = S_t \cup \{(\mathbf{x}_t, y_t)\}$
    end if
  else
    Compute f_{t+1} following (2) and (3)
    Update $S_{t+1} = S_t \cup \{(\mathbf{x}_t, y_t)\}$
  end if
end for

The computational complexity is $O(d|S_t| + |S_t|^2)$. It has been proved that $|S_t|$ depends on the decay rate of eigenvalues of the kernel matrix $\mathbf{K}_T$. If the eigenvalues decay slowly, then it is possible that $|S_t| \gg \sqrt{T}$. In this case, the computational complexity is $\Omega(T)$. To address this issue, we set a threshold B_0 for $|S_t|$. If $|S_t| \geq B_0$, then we always execute (2).

The learning rate η_t is defined as follows

$$\eta_t = \frac{U}{\sqrt{1 + \sum_{\tau=1}^{t} \|\hat{\nabla}_\tau\|_{\mathcal{H}}^2}},$$

$$\hat{\nabla}_\tau = \begin{cases} \ell'(f_\tau(\mathbf{x}_\tau), y_\tau) \cdot \mathbf{\Phi}_{S_\tau} \boldsymbol{\beta}_\tau^*, & \text{if } \mathrm{ALD}_\tau \text{ holds}, \\ \ell'(f_\tau(\mathbf{x}_\tau), y_\tau) \cdot \kappa(\mathbf{x}_\tau, \cdot), & \text{otherwise}. \end{cases}$$

We name this algorithm AOGD-ALD (Approximating kernelized Online Gradient Descent by the ALD condition), and give the pseudo-code in Algorithm 1.

3.2. Regret Bound

We first give the size of buffer maintained by the ALD condition.

Lemma 3.1 (Li & Liao (2022)). *Let $S_1 = \emptyset$ and ALD_t be defined in (5). For all $t \leq T-1$, if ALD_t does not hold, then $S_{t+1} = S_t \cup \{(\mathbf{x}_t, y_t)\}$. Otherwise, $S_{t+1} = S_t$. Let $\{\lambda_i\}_{i=1}^T$ be the eigenvalues of $\mathbf{K}_T$ sorted in decreasing order. If $\{\lambda_i\}_{i=1}^T$ decay exponentially, that is, there is a constant $R_0 > 0$ and $0 < r < 1$ such that $\lambda_i \leq R_0 r^i$, then $|S_T| \leq 2\frac{\ln(\frac{C_1 R_0}{\alpha})}{\ln r^{-1}}$. If $\{\lambda_i\}_{i=1}^T$ decay polynomially, that is, there is a constant $R_0 > 0$ and $p \geq 1$, such that $\lambda_i \leq R_0 i^{-p}$, then $|S_T| \leq \mathrm{e}(\frac{C_2 R_0}{\alpha})^{\frac{1}{p}}$. In both cases, C_1 and C_2 are constants, and $R_0 = \Theta(T)$.*

Next we give the regret bound and the computational complexity of AOGD-ALD.

Theorem 3.2. *Let $B_0 = \lfloor(\sqrt{d^2 + 4dT} - d)/2\rfloor$ and $\alpha = T^{-1}$. For any $\mathcal{I}_T$ satisfying $T > \ln^2 T$, the regret of AOGD-ALD satisfies,*

$$\forall f \in \mathbb{H}, \quad \mathrm{Reg}(f) = O\left(U\sqrt{L(f) + U} + U^2\right).$$

If $\{\lambda_i\}_{i=1}^T$ decay exponentially, then the computational complexity is $O(d \ln T + \ln^2 T)$. If $\{\lambda_i\}_{i=1}^T$ decay polynomially with degree $p \geq 1$, then the computational complexity is $O\left(\min\{dT^{\frac{2}{p}} + T^{\frac{4}{p}}, dT\}\right)$.

Let $f^* = \mathrm{argmin}_{f \in \mathbb{H}} L(f)$. $O(\sqrt{L(f^*)})$ is called "small-loss" bound (Orabona et al., 2012; Lykouris et al., 2018; Lee et al., 2020; Wang et al., 2020; Zhang et al., 2022). The data-dependent bound is never worse than the worst-case bound i.e., $O(\sqrt{T})$. If we select a good kernel function such that $L(f^*) \ll T$, then we can obtain a regret of $o(\sqrt{T})$. If $L_T(f^*) = 0$, then we obtain a regret of $O(1)$.

3.3. Comparison with Previous Results

The challenge of obtaining a regret of $O(\sqrt{L(f)})$ is the computational cost. KOGD achieves this regret bound at a computational complexity in $O(dT)$ (Zinkevich, 2003). With probability at least $1 - \delta$, FOGD (Lu et al., 2016) achieves a regret of $O(\sqrt{L(f)} + \frac{\sqrt{TL(f)\ln\frac{1}{\delta}}}{\sqrt{D}})$ at a computational complexity in $O(dD)$. We can define $D = o(T)$ which yields a suboptimal regret bound. For completeness, we reanalyze the regret of FOGD in the Appendix. Theorem 3.2 shows that if the eigenvalues decay exponentially or polynomially with degree $p > 4$, AOGD-ALD achieves the optimal regret at a computational complexity in $o(T)$.

Note that $L(f^*)$ depends on $\{(\mathbf{x}_t, y_t)\}_{t=1}^T$, while $\mathrm{d}_{\mathrm{eff}}(\mu)$ depends on $\{\mathbf{x}_t\}_{t=1}^T$. In general, they are not comparable. Thus it is not intuitive to compare AOGD-ALD with Pros-N-KONS (Calandriello et al., 2017a) and PKAWV (Jézéquel et al., 2019a). We just explain that AOGD-ALD provides a new regret-computational cost trade-off. Table 1 shows that the computational complexity of Pros-N-KONS can be smaller than AOGD-ALD, but its regret bound is worse in the case of $p \leq 2 + \sqrt{5}$. The computational complexity of PKAWV is always larger than AOGD-ALD, but its regret bound may be better for $p > 1$. In the case of $L(f^*) \ll T$, the regret bound of AOGD-ALD is also very small.

4. Approximating KONS

The square loss function is exp-concave. Thus second-order algorithms, such as KONS, can obtain a regret of $O(\mu + \mathrm{d}_{\mathrm{eff}}(\mu)\ln T)$. In this section, we propose a deterministic approximation of KONS, named NONS-ALD.

Algorithm 2 KONS

Input: $\mathbf{A}_0 = \mu\mathbf{I}$, $f_1 = 0$.
for $t = 1, \ldots, T$ **do**
  Receive $\mathbf{x}_t$
  Compute $\hat{y}_t = f_t(\mathbf{x}_t)$
  Update $\mathbf{A}_t = \mathbf{A}_{t-1} + \eta_t \nabla\ell(f_t(\mathbf{x}_t), y_t)(\nabla\ell(f_t(\mathbf{x}_t), y_t))^\top$
  Compute $f_{t+1} = f_t - \mathbf{A}_t^{-1}\nabla\ell(f_t(\mathbf{x}_t), y_t)$
end for

4.1. Kernelized ONS

For simplicity, we use the hypothesis space $\mathcal{H}$. At the end of round t, the KONS algorithm (Calandriello et al., 2017b) compute f_{t+1} by the following rule,

$$\begin{aligned} \mathbf{A}_t =& \mathbf{A}_{t-1} + \eta_t \nabla\ell(f_t(\mathbf{x}_t), y_t)(\nabla\ell(f_t(\mathbf{x}_t), y_t))^\top, \\ f_{t+1} =& f_t - \mathbf{A}_t^{-1}\nabla\ell(f_t(\mathbf{x}_t), y_t), \end{aligned} \tag{7}$$

where $\mathbf{A}_0 = \mu\mathbf{I}$. We give the pseudo-code in Algorithm 2.

KONS nearly stores all of the observed examples. At any round t, the computational complexity is $O(dt + t^2)$. To reduce the computational complexity, a natural idea is to use the ALD condition to maintain S_t. However, such a approach still has a $O(t \cdot |S_t|)$ computational complexity. Next we briefly explain the reason.

At any round t, if ALD_t holds, then we can approximate $\kappa(\mathbf{x}_t, \cdot)$ by $\mathbf{\Phi}_{S_t}\boldsymbol{\beta}_t^*$ and S_t keeps unchanged. Then we have $\mathbf{A}_t = \mathbf{A}_{t-1} + \eta_t(\ell'(f_t(\mathbf{x}_t), y_t))^2 \mathbf{\Phi}_{S_t}\boldsymbol{\beta}_t^*(\mathbf{\Phi}_{S_t}\boldsymbol{\beta}_t^*)^\top$. The key is to compute $f_{t+1}(\mathbf{x}_{t+1})$.

Theorem 4.1. *Let* $g_t = \ell'(f_t(\mathbf{x}_t), y_t)$ *and*

$$\hat{\phi}(\mathbf{x}_t) = \begin{cases} \phi(\mathbf{x}_t) = \kappa(\mathbf{x}_t, \cdot) & \text{if } \mathrm{ALD}_t \text{ does not hold,} \\ \mathbf{\Phi}_{S_t}\boldsymbol{\beta}_t^* & \text{otherwise.} \end{cases}$$

Let $\hat{\nabla}_t = g_t\hat{\phi}(\mathbf{x}_t)$ *and* $\hat{\mathbf{\Phi}}_t = (\sqrt{\eta_1}\hat{\nabla}_1, \ldots, \sqrt{\eta_t}\hat{\nabla}_t)$. *Then*

$$\begin{aligned} f_{t+1}(\mathbf{x}_{t+1}) = & \frac{-1}{\mu}\sum_{\tau=1}^{t} g_\tau \hat{\phi}(\mathbf{x}_\tau)^\top \phi(\mathbf{x}_{t+1}) + \\ & \frac{1}{\mu}\sum_{\tau=1}^{t} g_\tau \hat{\phi}(\mathbf{x}_\tau)^\top \hat{\mathbf{\Phi}}_\tau (\hat{\mathbf{\Phi}}_\tau^\top \hat{\mathbf{\Phi}}_\tau + \mu\mathbf{I})^{-1} \hat{\mathbf{\Phi}}_\tau^\top \phi(\mathbf{x}_{t+1}). \end{aligned}$$

Computing the first term requires time in $O(d|S_t|)$. Computing the second term requires time in $O(t|S_t|)$. The computational challenge comes from that KONS runs in the implicit feature space $\mathbb{R}^n$ in which we can not explicitly store and incrementally update $\mathbf{A}_t$. To address this issue, we use the Nyström projection to approximate the kernel mapping $\phi(\cdot)$, and run online Newton step (ONS) in an explicit feature space. To be specific, let $\phi_j(\cdot) : \mathcal{X} \to \mathbb{R}^j$ be an approximate kernel mapping. Then $\mathbf{A}_t = \mathbf{A}_{t-1} + \eta_t g_t^2 \phi_j(\mathbf{x}_t)\phi_j^\top(\mathbf{x}_t)$. We only store $\mathbf{A}_t \in \mathbb{R}^{j\times j}$, $j < \infty$ and can incrementally update $\mathbf{A}_t$. The computational complexity is $O(j^2)$.

4.2. Nyström Projection

We briefly introduce how the Nyström projection constructs explicit feature mapping (Williams & Seeger, 2001).

We select j columns from $\mathbf{K}_T$ to form a matrix $\mathbf{K}_{T,j} \in \mathbb{R}^{T\times j}$, and select the corresponding j rows from $\mathbf{K}_T$ to form a matrix $\mathbf{K}_{T,j}^\top \in \mathbb{R}^{j\times T}$. Let $S(j)$ contain the selected instances and $\mathbf{K}_{S(j)} \in \mathbb{R}^{j\times j}$ be the crossing matrix whose SVD is $\mathbf{K}_{S(j)} = \mathbf{U}_{S(j)}\Sigma_{S(j)}\mathbf{U}_{S(j)}^\top$. The Nyström projection approximates $\mathbf{K}_T$ by

$$\begin{aligned} \mathbf{K}_T \approx & \mathbf{K}_{T,j}\mathbf{K}_{S(j)}^{+}\mathbf{K}_{T,j}^\top \\ = & (\Sigma_{S(j)}^{-\frac{1}{2}}\mathbf{U}_{S(j)}^\top \mathbf{\Phi}_{S(j)}^\top \mathbf{\Phi}_T)^\top \Sigma_{S(j)}^{-\frac{1}{2}}\mathbf{U}_{S(j)}^\top \mathbf{\Phi}_{S(j)}^\top \mathbf{\Phi}_T. \end{aligned}$$

$\mathbf{\Phi}_T = (\phi(\mathbf{x}_t)_{t\in[T]}) \in \mathbb{R}^{n\times T}$ and $\mathbf{\Phi}_{S(j)} = (\phi(\mathbf{x})_{\mathbf{x}\in S(j)}) \in \mathbb{R}^{n\times j}$. Denote by $\mathcal{P}_{S(j)} = \mathbf{\Phi}_{S(j)}\mathbf{U}_{S(j)}\Sigma_{S(j)}^{-1}\mathbf{U}_{S(j)}^\top\mathbf{\Phi}_{S(j)}^\top$ the projection matrix onto the column space of $\mathbf{\Phi}_{S(j)}$. The approximate scheme defines an explicit feature mapping

$$\phi_j(\cdot) : \mathcal{X} \to \mathcal{P}_{S(j)}^{\frac{1}{2}}\phi(\cdot) \in \mathbb{R}^j,$$

in which $\mathcal{P}_{S(j)}^{\frac{1}{2}} = \Sigma_{S(j)}^{-\frac{1}{2}}\mathbf{U}_{S(j)}^\top\mathbf{\Phi}_{S(j)}^\top$. It is obvious that the approximation error depends on the selected j columns, or the crossing matrix. In the next subsection, we will use the ALD condition to select columns.

4.3. Column Selecting by the ALD Condition

At the beginning of the t-th round, assuming that $|S_t| = j$. Denote by $S_t = S(j)$. We first decide whether $\mathbf{x}_t$ will be added into $S(j)$. Solving (4), we obtain

$$\boldsymbol{\beta}_j^*(t) = \mathbf{K}_{S(j)}^{-1}\mathbf{\Phi}_{S(j)}^\top\phi(\mathbf{x}_t).$$

If the ALD_t condition holds, that is

$$\alpha_t = \left\|\mathbf{\Phi}_{S(j)}\boldsymbol{\beta}_j^*(t) - \kappa(\mathbf{x}_t, \cdot)\right\|_{\mathcal{H}}^2 \le \alpha,$$

then $S(j)$ keeps unchanged. We use $\mathbf{K}_{S(j)}$ as the crossing matrix and defined $\phi_j(\mathbf{x}_t) = \mathcal{P}_{S(j)}^{\frac{1}{2}}\phi(\mathbf{x}_t)$. If the ALD_t condition does not hold, then we execute $S_{t+1} = S_t \cup \{(\mathbf{x}_t, y_t)\}$, and denote by $S_{t+1} = S(j+1)$. We will construct explicit feature $\phi_{j+1}(\mathbf{x}_t)$.

4.4. Algorithm

Let $\mathbb{W}_t = \{f \in \mathcal{H} : |f(\mathbf{x}_t)| \le U\}$ (Luo et al., 2016; Calandriello et al., 2017a). For each $f \in \mathbb{H}$, $|f(\mathbf{x}_t)| \le \|f\|_{\mathcal{H}}\|\phi(\mathbf{x}_t)\|_{\mathcal{H}} \le U$. Thus $\mathbb{H} \subseteq \mathbb{W}_t$. Our algorithm will run in $\{\mathbb{W}_t\}_{t=1}^T$ not $\mathbb{H}$, since projection onto $\mathbb{W}_t$ is computationally more efficient.

We divide the time horizon $\{1, \ldots, T\}$ into different epochs.

$$\begin{aligned} T_0 &= \left\{s_1, \ldots, s_j \ldots, s_J : \mathrm{ALD}_{s_j} \text{ does not hold}\right\}, \\ T_j &= \{s_j, s_j + 1, \ldots, s_{j+1} - 1\},\ j = 1, 2, \ldots, J, \end{aligned}$$

where we define $s_1 = 1$ and $s_{J+1} - 1 = T$. Thus $\{1, \ldots, T\} = \cup_{j=1}^{J} T_j$. For any $t \in T_j$, let $S_t = S(j) = \{\mathbf{x}_{s_1}, \ldots, \mathbf{x}_{s_j}\}$. $\forall j \in [J], t \in T_j \setminus \{s_j\}$, the ALD_t condition holds. Besides, it is obvious that $\kappa(\mathbf{x}_{s_j}, \cdot) \in \mathbf{\Phi}_{S(j)}$.

The main idea of our algorithm is to run ONS on $T_j, j \in [J]$. Next we consider a fixed epoch T_j. At the beginning of round t, we compute $\phi_j(\mathbf{x}_t)$. Our algorithm maintains a linear hypothesis $f_{j,t}(\cdot) = \mathbf{w}_j^\top(t)\phi_j(\cdot)$, where $\mathbf{w}_j(t) \in \mathbb{R}^j$. The prediction is given by $\hat{y}_t = f_{j,t}(\mathbf{x}_t)$. For simplicity, let $g_j(t) = 2(f_{t,j}(\mathbf{x}_t) - y_t)$, and

$$\nabla_j(t) = \nabla \ell(f_{j,t}(\mathbf{x}_t), y_t) = g_j(t)\phi_j(\mathbf{x}_t).$$

We execute the following updating

$$\begin{cases} \mathbf{A}_j(t) = \mathbf{A}_j(t-1) + \eta_t g_j^2(t)\phi_j(\mathbf{x}_t)\phi_j^\top(\mathbf{x}_t), \\ \tilde{\mathbf{w}}_j(t+1) = \mathbf{w}_j(t) - \mathbf{A}_j^{-1}(t)\phi_j(\mathbf{x}_t) \in \mathbb{R}^j, \\ \mathbf{w}_j(t+1) = \mathcal{P}_{\mathbb{W}_{t+1}}(\tilde{\mathbf{w}}_j(t+1)), \end{cases}$$

where $\mathcal{P}_{\mathbb{W}_{t+1}}(\cdot)$ is a projection operator defined as follows

$$\mathbf{w}_j(t+1) = \operatorname*{arg\,min}_{\mathbf{w} \in \mathbb{W}_{t+1}} \|\mathbf{w} - \tilde{\mathbf{w}}_j(t+1)\|_{\mathbf{A}_j(t)}^2. \quad (8)$$

The initial configurations are denoted by $\mathbf{A}_j(s_j - 1)$ and $\mathbf{w}_j(s_j)$. When we enter T_j from T_{j-1}, the dimension of explicite feature mapping changes from $j-1$ to j which induces a technical challenge on initializing the configurations. To be specific, we can not use $f_{j-1,s_j} = \mathbf{w}_{j-1}^\top(s_j)\phi_{j-1}(\cdot)$ to prediction $\mathbf{x}_t, t \in T_j$. To address this issue, a simple approach is the restart technique. We just need to run a new ONS in T_j, which implies $\mathbf{A}_j(s_j - 1) = \alpha\mathbf{I}$ and $\mathbf{w}_j(s_j) = \mathbf{0}$. This idea is adopted by PROS-N-KONS (Calandriello et al., 2017a). The simple restart technique increases the regret by a factor of $O(J)$. Intuitively, the restart technique discards all of the information contained in $\mathbf{w}_{j-1}(s_j) \in \mathbb{R}^{j-1}$ and $\mathbf{A}_r(s_{r+1} - 1) \in \mathbb{R}^{r \times r}$, $r \leq j-1$. Next we redefine the initial configurations. The main idea is to project $\mathbf{w}_{j-1}(s_j)$ onto $\mathbb{R}^j$ and project $\mathbf{A}_r(s_{r+1} - 1)$ onto $\mathbb{R}^{j \times j}$, $r \leq j-1$.

The definition of $\mathbf{A}_j(s_j - 1)$ is intuitive. For any $t \in T_j$, the updating rule of ONS is as follows,

$$\mathbf{A}_j(t) = \mathbf{A}_j(s_j - 1) + \sum_{\tau=s_j}^{t} \eta_\tau g_j^2(\tau)\phi_j(\mathbf{x}_\tau)\phi_j^\top(\mathbf{x}_\tau).$$

The ideal value of $\mathbf{A}_j(s_j - 1)$ should be

$$\mathbf{A}_j(s_j - 1) = \mu\mathbf{I} + \sum_{r=1}^{j-1} \sum_{t \in T_r} \eta_t g_r^2(t)\phi_j(\mathbf{x}_t)\phi_j^\top(\mathbf{x}_t),$$

where $\phi_j(\mathbf{x}_t) = \mathcal{P}_{S(j)}^{\frac{1}{2}}\phi(\mathbf{x}_t)$. However, such an approach must store $\{\mathbf{x}_t\}_{t=1}^{s_j-1}$ which induce a $O(dT)$ computational complexity. Recalling that the ALD condition guarantees that $\phi(\mathbf{x}_t) \approx \mathbf{\Phi}_{S(r)}\boldsymbol{\beta}_r^*(t)$. It is natural to define

$$\forall t \in T_r, \quad \tilde{\phi}_j(\mathbf{x}_t) = \mathcal{P}_{S(j)}^{\frac{1}{2}}\mathbf{\Phi}_{S(r)}\boldsymbol{\beta}_r^*(t). \quad (9)$$

We can define $\mathbf{A}_j(s_j - 1)$ as follows,

$$\mathbf{A}_j(s_j - 1) = \mu\mathbf{I} + \sum_{r=1}^{j-1} \sum_{t \in T_r} \eta_t g_r^2(t)\tilde{\phi}_j(\mathbf{x}_t)\tilde{\phi}_j^\top(\mathbf{x}_t). \quad (10)$$

In this way, we only use the instances in S_t. The computational complexity is $O(d|S_t| + |S_t|^2)$.

It is less intuitive to define $\mathbf{w}_j(s_j)$. The projection of any $f \in \mathbb{H}$ onto the column space of $\mathbf{\Phi}_{S(j-1)}$ and $\mathbf{\Phi}_{S(j)}$ are $f_{j-1} = \mathcal{P}_{S(j-1)}f$ and $f_j = \mathcal{P}_{S(j)}f$, respectively. Denote by $f_{j-1} = \mathbf{w}_{j-1}^\top\phi_{j-1}(\cdot)$ and $f_j = \mathbf{w}_j^\top\phi_j(\cdot)$. We can prove that $\mathbf{w}_{j-1} = \mathcal{P}_{S(j-1)}^{\frac{1}{2}}(\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top\mathbf{w}_j$. Thus it must be

$$\mathbf{w}_{j-1}(s_j) = \mathcal{P}_{S(j-1)}^{\frac{1}{2}}(\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top\mathbf{w}_j(s_j). \quad (11)$$

Besides, at the $(s_j - 1)$-th round, $\mathbf{w}_{j-1}(s_j)$ must be the solution of the following projection

$$\mathbf{w}_{j-1}(s_j) = \mathcal{P}_{\mathbb{W}_{s_j}}(\tilde{\mathbf{w}}_{j-1}(s_j)). \quad (12)$$

To this end, we need to compute $\phi_{j-1}(\mathbf{x}_{s_j})$. Note that ALD_{s_j} does not hold. Although $\kappa(\mathbf{x}_{s_j}, \cdot)$ can not be well approximated by $\mathbf{\Phi}_{S(j-1)}$, the goal of (12) is just to ensure $\mathbf{w}_j(s_j) \in \mathbb{W}_{s_j}$. Both the property in (11) and (12) are critical to the regret analysis.

We name this algorithm NONS-ALD (Nyström Online Newton Step using the ALD condition), and give the pseudo-code in Algorithm 3.

4.5. Theoretical Analysis

4.5.1. Regret analysis

We first show an equivalent definition of (10).

Lemma 4.2. *For any $j = 1, \ldots, J$, the approximate scheme (10) is equivalent to the following scheme*

$$\mathbf{A}_j(s_j - 1) = \mu\mathbf{I} + Q_{j,j-1}(\mathbf{A}_{j-1}(s_j - 1) - \mu\mathbf{I})Q_{j,j-1}^\top$$

where $Q_{j,j-1} = \mathcal{P}_{S(j)}^{\frac{1}{2}}(\mathcal{P}_{S(j-1)}^{\frac{1}{2}})^\top$.

Storing $\mathbf{A}_j(s_j - 1)$ and $Q_{j,j-1}$ requires space in $O(j^2)$, and computing $\mathbf{A}_j(s_j - 1)$ requires time in $O(j^3)$.

Lemma 4.3. *For any $j = 1, \ldots, J$, let $\mathbf{w}_{j-1}(s_j)$ satisfy (12), and*

$$\mathbf{w}_j(s_j) = \mathcal{P}_{S(j)}^{\frac{1}{2}}(\mathcal{P}_{S(j-1)}^{\frac{1}{2}})^\top\mathbf{w}_{j-1}(s_j).$$

Then $\mathbf{w}_j(s_j) \in \mathbb{W}_{s_j}$ and (11) is satisfied.

Algorithm 3 NONS-ALD

Input: μ, α, U, Y
Initialize: $j=0, \mathbf{w}_1(1)=0, \mathbf{A}_1(0)=\mu, S(0)=\emptyset$, flag $=1$
for $t=1,\dots,T$ **do**
  Receive $\mathbf{x}_t$
  Compute $\boldsymbol{\beta}_j^*(t)=\arg\min_{\boldsymbol{\beta}\in\mathbb{R}^j}\|\phi(\mathbf{x}_t)-\boldsymbol{\Phi}_{S(j)}\boldsymbol{\beta}\|_{\mathcal{H}}^2$
  Compute $\alpha_t=\kappa(\mathbf{x}_t,\mathbf{x}_t)-\phi(\mathbf{x}_t)^\top\boldsymbol{\Phi}_{S(j)}\boldsymbol{\beta}_j^*(t)$
  if $\alpha_t>\alpha$ **then**
    $S(j+1)=S(j)\cup\{(\mathbf{x}_t,y_t)\}$
    flag $=1$
    $j=j+1$
  else
    if flag $==1$ **then**
      $s_j=t$
      $(\mathbf{U}_{S(j)},\Sigma_{S(j)})\leftarrow\mathrm{SVD}(\mathbf{K}_{S(j)})$
      flag $=0$
      Compute $Q_{j,j-1}=\mathcal{P}_{S(j)}^{\frac{1}{2}}(\mathcal{P}_{S(j-1)}^{\frac{1}{2}})^\top$
      Compute $\mathbf{A}_j(s_j-1)$ follows Lemma 4.2
      Compute $\mathbf{w}_j(s_j)=\mathcal{P}_{S(j)}^{\frac{1}{2}}(\mathcal{P}_{S(j-1)}^{\frac{1}{2}})^\top\mathbf{w}_{j-1}(s_j)$
    end if
    Compute $\phi_j(\mathbf{x}_t)=\Sigma_{S(j)}^{-\frac{1}{2}}\mathbf{U}_{S(j)}^\top\boldsymbol{\Phi}_{S(j)}^\top\phi(\mathbf{x}_t)$
    Output $\hat{y}_t=\mathbf{w}_j^\top(t)\phi_j(\mathbf{x}_t)$
    Compute $\nabla_j(t)=\ell'(\hat{y}_t,y_t)\cdot\phi_j(\mathbf{x}_t)$
    Update $\mathbf{A}_j(t)=\mathbf{A}_j(t-1)+\eta_t\nabla_j(t)\nabla_j^\top(t)$
    Compute $\tilde{\mathbf{w}}_j(t+1)=\mathbf{w}_j(t)-\mathbf{A}_j^{-1}(t)\nabla_j(t)$
    Compute $\phi_j(\mathbf{x}_{t+1})=\Sigma_{S(j)}^{-\frac{1}{2}}\mathbf{U}_{S(j)}^\top\boldsymbol{\Phi}_{S(j)}^\top\phi(\mathbf{x}_{t+1})$
    Compute $\mathbf{w}_j(t+1)$ following (8)
  end if
end for

Remark 4.4. An empirical version of Pros-N-KONS (Calandriello et al., 2017a), named CON-KNOS, uses a different $\mathbf{w}_j(s_j)$. CON-KNOS uses $\mathbf{w}_{j-1}(s_j-1)$ to construct $\mathbf{w}_j(s_j)$, while our algorithm uses $\mathbf{w}_{j-1}(s_j)$ to construct $\mathbf{w}_j(s_j)$. Our regret analysis shows that $\mathbf{w}_{j-1}(s_j)$ is necessary for obtaining the nearly optimal regret bound.

Next we measure the quality of columns selected by the ALD condition using spectral norm error bounds.

Lemma 4.5 (Spectral Norm Error Bound)**.** *Let $\alpha\le 1$. For all $j=1,\dots,J$, let $\boldsymbol{\Phi}_{T_j}=(\phi(\mathbf{x}_t))_{t\in T_j}$ and $\mathcal{P}_{S(j)}$ be the projection matrix onto the column space of $\boldsymbol{\Phi}_{S(j)}$.*

$$\forall j\in[J],\ \left\|\boldsymbol{\Phi}_{T_j}^\top\boldsymbol{\Phi}_{T_j}-\boldsymbol{\Phi}_{T_j}^\top\mathcal{P}_{S(j)}\boldsymbol{\Phi}_{T_j}\right\|_2\le|T_j|\cdot\alpha. \quad (13)$$

Let $\tilde{\boldsymbol{\Phi}}_T=\left((\tilde{\phi}_J(\mathbf{x}_t))_{t\in T_1},\dots,(\tilde{\phi}_J(\mathbf{x}_t))_{t\in T_J}\right)\in\mathbb{R}^{J\times T}$, where $\tilde{\phi}_J(\cdot)$ follows (9). Then

$$\left\|\mathbf{K}_T-\tilde{\boldsymbol{\Phi}}_T^\top\tilde{\boldsymbol{\Phi}}_T\right\|_2\le T\sqrt{\alpha}. \quad (14)$$

We call (14) global spectral norm error bound. We call (13) local spectral norm error bound. According to Lemma 4.5, we can prove that the regret induced by our projection scheme (i.e., projecting $\mathbf{A}_{j-1}(s_j-1)$ and $\mathbf{w}_{j-1}(s_j)$) is controlled by the parameter α. Thus optimizing α will yield the desired regret bounds.

Lemma 4.5 gives deterministic spectral norm error bounds, while most of previous results only hold in a high probability, such as the uniform column sampling (Drineas & Mahoney, 2005; Jin et al., 2013) and the RSL sampling (Calandriello et al., 2017a). If the instances could be observed beforehand, such as offline learning, then we can obtain a goal spectral norm error bound stated in (13). Such a result might be of independent interest. In this case, previous work only proved a global spectral norm error bound of $O(T\sqrt{\alpha})$ (Sun et al., 2012).

Theorem 4.6. *Let $U\ge Y$ and $\eta_t=\frac{1}{4(U^2+Y^2)}$ for all $t\in[T]$. Assuming that $|S_T|=J$. For any $f\in\mathbb{H}$, the regret of NONS-ALD satisfies*

$$\mathrm{Reg}(f)\le(\frac{\mu}{2}+T\alpha)\|f\|_{\mathcal{H}}^2+\frac{1}{2}\mathrm{d}_{\mathrm{eff}}(\frac{\mu}{2})\left(1+\ln\frac{2T+\mu}{\mu}\right)+\frac{T^2\sqrt{\alpha}}{\sqrt{2}\mu}+\sqrt{8(U^2+Y^2)}\|f\|_{\mathcal{H}}\cdot T\sqrt{\alpha}.$$

The space complexity is $O(dJ+J^2)$. The average per-round time complexity is $O(dJ+J^2+\frac{J^4}{T})$.

We will omit the factor $O(dJ)$ in the discussion on computational complexity. Next we give the values of μ and α and derive nearly optimal regret bounds.

Corollary 4.7. *Let $\alpha=\frac{\ln^4 T}{T^4}$ and $\mu>0$ be a constant. If $\{\lambda_i\}_{i=1}^T$ decay exponentially, i.e., $\lambda_i\le R_0 r^i$, $R_0=\Theta(T)$, then the regret of NONS-ALD satisfies*

$$\forall f\in\mathbb{H},\ \mathrm{Reg}(f)=O(\|f\|_{\mathcal{H}}^2+\ln^2 T).$$

The space and average per-round time complexity is $O(\ln^2 T)$.

If $\{\lambda_i\}_{i=1}^T$ decay polynomially, then we must tune μ and α.

Corollary 4.8. *If $\{\lambda_i\}_{i=1}^T$ decay polynomially with degree $p\ge 1$, i.e., $\lambda_i\le R_0 i^{-p}$, $R_0=\Theta(T)$, then let $\mu=T^{\frac{1}{1+p}}$ and $\alpha=T^{-\frac{4p}{1+p}}$. The regret of NONS-ALD satisfies*

$$\forall f\in\mathbb{H},\ \mathrm{Reg}(f)=O\left(T^{\frac{1}{1+p}}\ln T\right).$$

The space complexity is $O(T^{\frac{2(1+5p)}{p(1+p)}})$, and the average per-round time complexity is $O(T^{\frac{2(1+5p)}{p(1+p)}}+T^{\frac{4(1+5p)}{p(1+p)}-1})$.

It is worth mentioning that for all $p\ge 10$, the space complexity and the average per-round time complexity is $O(T^{\frac{2(1+5p)}{p(1+p)}})=o(T)$. This is the first algorithm that achieves a nearly optimal regret bound at a sublinear computational complexity. However, the computational complexity becomes worse for $p<10$. It is left to further work

to achieve the same regret bound at a $o(T)$ computational complexity in the case of $p < 10$.

The regret bounds in Corollary 4.7 and Corollary 4.8 recovery the regret bounds of KONS (Calandriello et al., 2017b). Our regret bounds are optimal up to $\ln T$. The most important improvement is the computational complexity. KONS requires a $O(T^2)$ computational complexity.

4.5.2. Comparison with more results

We compare our algorithm with Pros-N-KONS (Calandriello et al., 2017a) and PKAWV (Jézéquel et al., 2019a).

With probability at least $1-\delta$, Pros-N-KONS achieves

$$\forall f \in \mathbb{H}, \mathrm{Reg}(f) \leq \frac{\mu}{2} J \|f\|_{\mathcal{H}}^2 + J \cdot \mathrm{d}_{\mathrm{eff}}(\mu) \ln(T) + \frac{T\alpha}{\mu},$$

where $J = O(\mathrm{d}_{\mathrm{eff}}(\alpha) \cdot \ln^2 \frac{T}{\delta})$. The space complexity is $O(J^2)$. Pros-N-KONS executes the SVD operations J times. Thus the average per-round time complexity is $O(J^2 + \frac{J^4}{T})$. The factor $O(\ln^2 \frac{T}{\delta})$ on J is induced by the RLS sampling (see Proposition 1 in Calandriello et al. (2017a)) which is a random method. Thus $O(\ln^2 \frac{T}{\delta})$ is unavoidable. Our algorithm uses the ALD condition which is a deterministic method, and does not have the factor.

If $\{\lambda_i\}_{i=1}^T$ decay exponentially, then $\mathrm{d}_{\mathrm{eff}}(\alpha) = O(\ln \frac{T}{\alpha})$. Let μ be a constant and $\alpha = \frac{\mu}{T}$. Pros-N-KONS enjoys a regret of $O(\ln^5 T)$ at a computational complexity (space complexity and average time-complexity) in $O(\ln^6 T)$. Our algorithm enjoys a regret of $O(\ln^2 T)$ at a computational complexity in $O(\ln^2 T)$.

If $\{\lambda_i\}_{i=1}^T$ decay polynomially, then $\mathrm{d}_{\mathrm{eff}}(\alpha) = O((\frac{T}{\alpha})^{\frac{1}{p}})$. We solve the following two equations

$$\mu = \left(\frac{T}{\mu}\right)^{\frac{1}{p}}, \quad \mu \left(\frac{T}{\alpha}\right)^{\frac{1}{p}} = \frac{T\alpha}{\mu}.$$

The solutions are $\mu = T^{\frac{1}{1+p}}$ and $\alpha = T^{-\frac{p^2-2p-1}{(p+1)^2}}$. Pros-N-KONS enjoys a regret of $O(T^{\frac{3p+1}{(1+p)^2}} \ln^3 T)$ at a computational complexity in $O(T^{\frac{4p}{(1+p)^2}} \ln^4 T)$. Although Pros-N-KONS ensures a computational complexity in $o(T)$ for $p > 1$, its regret bound is far from optimal.

With probability at least $1-\delta$, PKAWV achieves

$$\forall f \in \mathbb{H}, \quad \mathrm{Reg}(f) \leq \frac{\mu}{2} \|f\|_{\mathcal{H}}^2 + \mathrm{d}_{\mathrm{eff}}(\mu) \ln(T) + \frac{JT\alpha}{\mu}.$$

The computational complexity is $O(TJ + J^2)$, where $J = O(\mathrm{d}_{\mathrm{eff}}(\alpha) \cdot \ln^2 \frac{T}{\delta})$ (see Algorithm 2 in the Supplementary material of Jézéquel et al. (2019a), or see Section H in Jézéquel et al. (2019b)). Besides, at each round t, PKAWV must store the pervious examples $\{(\mathbf{x}_\tau, y_\tau)_{\tau=1}^t\}$. Both our algorithm and Pros-N-KONS only store J examples.

If $\{\lambda_i\}_{i=1}^T$ decay exponentially, then PKAWV enjoys a regret of $O(\ln^2 T)$ at a computational complexity in $O(T \ln^3 T)$. Our algorithm enjoys the same regret bound only at a computational complexity in $O(\ln^2 T)$.

If $\{\lambda_i\}_{i=1}^T$ decay polynomially, then PKAWV also enjoys a regret of $O(T^{\frac{1}{1+p}} \ln T)$. PKAWV suffers a computational complexity in $O(T^{\frac{4p}{p^2-1}} \ln^2 T + T^{1+\frac{2p}{p^2-1}} \ln^4 T)$ which can not be $o(T)$ for all $p \geq 1$. In the case of $p \geq 10$, our algorithm enjoys a computational complexity in $o(T)$.

Finally, we note that Pros-N-KONS can compare with $f \in \mathbf{H} = \{f \in \mathcal{H} : \forall t \in [T], |f(\mathbf{x}_t)| \leq U\}$ and PKAWV can compare with $f \in \mathcal{H}$, while our algorithm only compares with $f \in \mathbb{H}$. It should be that $\mathbb{H} \subseteq \mathbf{H} \subseteq \mathcal{H}$. From the perspective of the size of hypothesis space, our algorithm is weaker than Pros-N-KONS and PKAWV. As explained in Section 2.2, it is enough to compare with hypotheses in $\mathbb{H}$.

4.5.3. Computational Complexity analysis

At each round t, the main time cost is to compute the projection (8), $\mathbf{A}_j(s_j - 1)$, $\mathbf{A}_j^{-1}(s_j)$ and the SVD of $\mathbf{K}_{S(j)}$.

The solution of projection (8) is as follows.

Theorem 4.9 (Luo et al. (2016))**.** *At each round t,*

$$\mathbf{w}_j(t+1) = \tilde{\mathbf{w}}_j(t+1) - \frac{m(\tilde{y}_{t+1}) \mathbf{A}_j^{-1}(t) \phi_j(\mathbf{x}_{t+1})}{\phi_j^\top(\mathbf{x}_{t+1}) \mathbf{A}_j^{-1}(t) \phi_j(\mathbf{x}_{t+1})}$$

where $\tilde{y}_{t+1} = \tilde{\mathbf{w}}_j^\top(t+1)\phi_j(\mathbf{x}_{t+1})$ and $m(\tilde{y}_{t+1}) = \mathrm{sign}(\tilde{y}_{t+1}) \max\{|\tilde{y}_{t+1}| - U, 0\}$.

For any invertible $\mathbf{B} \in \mathbb{R}^{j \times j}$ and $\mathbf{a}, \mathbf{b} \in \mathbb{R}^j$, we have

$$(\mathbf{B} + \mathbf{a}\mathbf{b}^\top)^{-1} = \mathbf{B}^{-1} - \frac{\mathbf{B}^{-1}\mathbf{a}\mathbf{b}^\top\mathbf{B}^{-1}}{1 + \mathbf{b}^\top\mathbf{B}^{-1}\mathbf{a}}.$$

Let $\mathbf{B} = \mathbf{A}_j(t-1)$ and $\mathbf{a} = \mathbf{b} = \sqrt{\eta_t} \nabla_j(t)$. In this way, $\mathbf{A}_j^{-1}(t))$ can be computed incrementally in time $O(j^2)$. The time complexity over T rounds is $O(TJ^2)$.

Computing $\mathbf{A}_j(s_j - 1)$, $\mathbf{A}_j^{-1}(s_j - 1)$ and the SVD of $\mathbf{K}_{S(j)}$ requires time in $O(j^3)$. Such operations are only executed J times. The time complexity over T rounds is $O(J^4)$.

Thus the total complexity is $O(TJ^2 + J^4)$. Thus the average per-round time complexity is $O(J^2 + \frac{J^4}{T})$. The space complexity is always $O(J^2)$.

5. Conclusion

In this paper, we have studied the trade-off between regret and computational cost for online kernel regression, and proposed two algorithms that achieve two types of nearly optimal regret bounds at a sublinear computational complexity for the first time. The two regret bounds are data-dependent

and not comparable. The computational complexities of our algorithms depend on the decay rate of eigenvalues of the kernel matrix, and are sublinear if the eigenvalues decay fast enough. We empirically verified that our algorithms can balance the prediction performance and computational cost better than previous algorithms can do.

The two algorithms use the ALD condition to dynamically maintain a group of nearly orthogonal basis which are used to approximate the kernel mapping. Compared with other basis selecting schemes, such as uniform sampling and the RLS sampling, both the number of basis and the approximate error bound can be smaller. The ALD condition can be a better basis selecting scheme for designing computationally efficient online and offline kernel learning algorithms.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under grants No. 62076181. We thank all anonymous reviewers for their valuable comments and suggestions.

References

Aronszajn, N. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.

Calandriello, D., Lazaric, A., and Valko, M. Efficient second-order online kernel learning with adaptive embedding. *Advances in Neural Information Processing Systems*, 30:6140–6150, 2017a.

Calandriello, D., Lazaric, A., and Valko, M. Second-order kernel online convex optimization with adaptive sketching. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 645–653, 2017b.

Caponnetto, A. and Vito, E. D. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.

Drineas, P. and Mahoney, M. W. On the nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005.

Engel, Y., Mannor, S., and Meir, R. The kernel recursive least-squares algorithm. *IEEE Transactions on Signal Processing*, 52(8):2275–2285, 2004.

Gammerman, A., Kalnishkan, Y., and Vovk, V. On-line prediction with kernels and the complexity approximation principle. In *Proceedings of the 20th Conference in Uncertainty in Artificial Intelligence*, pp. 170–176, 2004.

Hazan, E., Agarwal, A., and Kale, S. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2):169–192, 2007.

Jézéquel, R., Gaillard, P., and Rudi, A. Efficient online learning with kernels for adversarial large scale problems. *Advances in Neural Information Processing Systems*, 32: 9427–9436, 2019a.

Jézéquel, R., Gaillard, P., and Rudi, A. Efficient online learning with kernels for adversarial large scale problems. *CoRR*, arXiv:1902.09917v2, 2019b. URL https://arxiv.org/abs/1902.09917v2.

Jin, R., Yang, T., Mahdavi, M., Li, Y., and Zhou, Z. Improved bounds for the nyström method with application to kernel classification. *IEEE Transactions on Information Theory*, 59(10):6939–6949, 2013.

Kivinen, J., Smola, A. J., and Williamson, R. C. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52(8):2165–2176, 2004.

Lee, C.-W., Luo, H., and Zhang, M. A closer look at small-loss bounds for bandits with graph feedback. In *Proceedings of the 33rd Annual Conference on Learning Theory*, pp. 2516–2564, 2020.

Li, J. and Liao, S. Improved kernel alignment regret bound for online kernel learning. *CoRR*, abs/2212.12989, 2022. URL https://arxiv.org/abs/2212.12989.

Li, Z., Ton, J.-F., Oglic, D., and Sejdinovic, D. Towards a unified analysis of random Fourier features. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 3905–3914, 2019.

Lu, J., Hoi, S. C. H., Wang, J., Zhao, P., and Liu, Z. Large scale online kernel learning. *Journal of Machine Learning Research*, 17(47):1–43, 2016.

Luo, H., Agarwal, A., Cesa-Bianchi, N., and Langford, J. Efficient second order online learning by sketching. *Advances in Neural Information Processing Systems*, 29: 902–910, 2016.

Lykouris, T., Sridharan, K., and Tardos, É. Small-loss bounds for online learning with partial information. In *Proceedings of the 31st Annual Conference on Learning Theory*, pp. 979–986, 2018.

Orabona, F., Cesa-Bianchi, N., and Gentile, C. Beyond logarithmic bounds in online learning. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, pp. 823–831, 2012.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. *Advances in Neural Information Processing Systems*, 20:1177–1184, 2007.

Rudi, A., Camoriano, R., and Rosasco, L. Less is more: Nyström computational regularization. *Advances in Neural Information Processing Systems*, 28:1657–1665, 2015.

Sahoo, D., Hoi, S. C. H., and Li, B. Online multiple kernel regression. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*, pp. 293–302, 2014.

Shawe-Taylor, J. and Cristianini, N. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, 2004.

Srebro, N., Sridharan, K., and Tewari, A. Smoothness, low noise and fast rates. *Advances in Neural Information Processing Systems*, 23:2199–2207, 2010.

Sun, Y., Gomez, F. J., and Schmidhuber, J. On the size of the online kernel sparsification dictionary. In *Proceedings of the 29th International Conference on Machine Learning*, pp. 329–336, 2012.

Vovk, V. On-line regression competitive with reproducing kernel hilbert spaces. In *Proceedings of the 3rd International Conference on Theory and Applications of Models of Computation*, pp. 452–463, 2006.

Wang, G., Lu, S., Hu, Y., and Zhang, L. Adapting to smoothness: A more universal algorithm for online convex optimization. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pp. 6162–6169, 2020.

Williams, C. K. I. and Seeger, M. Using the nyström method to speed up kernel machines. *Advances in Neural Information Processing Systems*, 13:682–688, 2001.

Zhang, L., Liu, T.-Y., and Zhou, Z.-H. Adaptive regret of convex and smooth functions. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 7414–7423, 2019.

Zhang, L., Wang, G., Yi, J., and Yang, T. A simple yet universal strategy for online convex optimization. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 26605–26623, 2022.

Zhdanov, F. and Kalnishkan, Y. An identity for kernel ridge regression. *Theoretical Computer Science*, 473:157–178, 2013.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning*, pp. 928–936, 2003.

Table 2. Basic information of datasets. #num is the number of examples. #fea is the number of features.

Dataset	#num	#fea	Dataset	#num	#fea	Dataset	# num	#fea	Dataset	#num	#fea
parkinson	5875	16	elevators	16599	18	cpusmall	8192	12	bank	8192	32
ailerons	13750	40	calhousing	14000	8	Year	51630	90	TomsHardware	28179	96

A. Experiments

In this section, we verify the following three goals.

G 1 NONS-ALD enjoys the best prediction performance.
Corollary 4.7 and Corollary 4.8 show that the regret bounds of NONS-ALD are optimal up to $\ln T$. We expect that NONS-ALD performs best.

G 2 If the kernel function is well tuned, then AOGD-ALD and NONS-ALD only store few examples.
The number of stored examples depends on the decay rate of eigenvalues of the kernel matrix. We can tune the kernel function such that the eigenvalues of the kernel matrix decay fast.

G 3 If the kernel function is well tuned, then AOGD-ALD and NONS-ALD are computationally efficient.
We have proved that if the eigenvalues of the kernel matrix decay fast, then AOGD-ALD and NONS-ALD achieve a $o(T)$ computational complexity. If the eigenvalues decay exponentially, the computational complexity is $O(\ln^2 T)$.

A.1. Experimental Setting

We adopt the Gaussian kernel $\kappa(\mathbf{x}, \mathbf{v}) = \exp(-\frac{\|\mathbf{x}-\mathbf{v}\|^2}{2\varsigma^2})$ and use 8 regression datasets from WEKA and UCI machine learning repository [2]. The information of datasets is given in Table 2. The target variables and features of all datasets are rescaled to fit in $[0, 1]$ and $[-1, 1]$ respectively. We randomly permutate the instances in the datasets 10 times and report the average results. All algorithms are implemented with R on a Windows machine with 2.8 GHz Core(TM) i7-1165G7 CPU [3].

The baseline algorithms include two first-order algorithms, FOGD and NOGD (Lu et al., 2016) and two second-order algorithms, PROS-N-KONS and CON-KONS (Calandriello et al., 2017a). CON-KONS which is an empirical variant of PROS-N-KONS, sets $\mathbf{A}_j(s_j - 1) = Q_{j,j-1}\mathbf{A}_{j-1}(s_j - 1)Q_{j,j-1}^\top$ and $\mathbf{w}_j(s_j) = Q_{j,j-1}\mathbf{w}_{j-1}(s_j - 1)$, where $Q_{j,j-1} = \mathcal{P}_{S(j)}^{\frac{1}{2}}(\mathcal{P}_{S(j-1)}^{\frac{1}{2}})^\top$. Note that the two values are different from our initial configurations in Lemma 4.2 and Lemma 4.3. We do not compare with PKAWV (Jézéquel et al., 2019a), since its computational complexity is $O(T)$. The experimental results in (Jézéquel et al., 2019a) also verified that PKAWV runs slower than PROS-N-KONS. For FOGD and NOGD, we tune the stepsize $\eta \in \{\frac{1}{\sqrt{T}}, \frac{10}{\sqrt{T}}, \frac{100}{\sqrt{T}}, \frac{1000}{\sqrt{T}}\}$. We set $D = 400$ for FOGD and $J = 400$ for NOGD in which D is the number of random features and J is the size of buffer (or the number of stored examples). There are five hyper-parameters needed to be tuned in PROS-N-KONS and CON-KONS, i.e., C, β, ε, α and γ. We set $C = 1$, $\beta = 1$, $\varepsilon = 0.5$ following the suggestion in original paper (Calandriello et al., 2017a). To improve the performance of PROS-N-KONS and CON-KONS, we tune $\alpha \in \{1, 5, 15\}$ and $\gamma \in \{0.5, 1, 5, 10\}$. α is a regularization parameter and plays the same role with the parameter μ in NONS-ALD. γ controls the size of buffer. The larger γ is, the smaller the buffer will be, that is, the computational complexity will be smaller. We set $\alpha = \frac{25}{T}$ for AOGD-ALD and NONS-ALD. We set $U = 2$ for AOGD-ALD and $U = 1$ for NONS-ALD. Besides, we tune $\mu \in \{1, 5, 15\}$ for NONS-ALD.

A.2. Experimental Results

Table 3 shows the experimental results. We report the average mean squared error (MSE), the size of buffer (J), the number of random features (D), and the average per-round running time. The MSE is defined as $\text{MSE} = \frac{1}{T}\sum_{t=1}^{T}(\hat{y}_t - y_t)^2$.

As a whole, NONS-ALD enjoys the smallest MSE on all datasets. We first analyze the results of the three second-order algorithms, i.e., NONS-ALD, CON-KONS and PROS-N-KONS. Both NONS-ALD and CON-KONS enjoy much better prediction performance than PROS-N-KONS. The MSE of PROS-N-KONS is even larger than that of the three first-order algorithms. The reason is that PROS-N-KONS uses the restart technique. If the times of restart are large, then the prediction

[2] https://archive.ics.uci.edu/ml/index.php

[3] The codes are available at https://github.com/JunfLi-TJU/OKR.git.

Table 3. Experimental results on benchmark datasets.

Algorithm	parkinson, $\varsigma = 8$			elevator, $\varsigma = 8$		
	MSE	$J\|D$	Time (s)	MSE	$J\|D$	Time (s)
FOGD	0.05590 ± 0.00011	400	0.26 ± 0.01	0.00560 ± 0.00009	400	0.72 ± 0.02
NOGD	0.05711 ± 0.00042	400	1.42 ± 0.03	0.00575 ± 0.00004	400	3.97 ± 0.11
PROS-N-KONS	0.06420 ± 0.00073	33	0.45 ± 0.05	0.00873 ± 0.00023	32	1.16 ± 0.07
CON-KONS	0.05553 ± 0.00024	31	0.46 ± 0.04	0.00452 ± 0.00018	34	1.19 ± 0.14
AOGD-ALD	0.05988 ± 0.00018	13	0.08 ± 0.02	0.00534 ± 0.00003	28	0.27 ± 0.02
NONS-ALD	$\mathbf{0.05514 \pm 0.00008}$	**13**	0.14 ± 0.02	$\mathbf{0.00284 \pm 0.00005}$	**28**	0.71 ± 0.06
Algorithm	cpusmall , $\varsigma = 2$			bank, $\varsigma = 12$		
	MSE	$J\|D$	Time (s)	MSE	$J\|D$	Time (s)
FOGD	0.01269 ± 0.00033	400	0.34 ± 0.02	0.01910 ± 0.00033	400	0.43 ± 0.01
NOGD	0.01388 ± 0.00070	400	1.93 ± 0.04	0.01966 ± 0.00008	400	2.09 ± 0.03
PROS-N-KONS	0.02939 ± 0.00096	42	0.77 ± 0.11	0.02677 ± 0.00015	179	14.05 ± 1.07
CON-KONS	0.01166 ± 0.00080	42	0.77 ± 0.08	0.01663 ± 0.00014	177	14.00 ± 1.15
AOGD-ALD	0.01330 ± 0.00006	44	0.15 ± 0.02	0.01915 ± 0.00009	148	0.66 ± 0.03
NONS-ALD	$\mathbf{0.00703 \pm 0.00024}$	**43**	0.62 ± 0.06	$\mathbf{0.01306 \pm 0.00004}$	**148**	11.18 ± 0.53
Algorithm	ailerons , $\varsigma = 8$			calhousing, $\varsigma = 4$		
	MSE	$J\|D$	Time (s)	MSE	$J\|D$	Time (s)
FOGD	0.00363 ± 0.00009	400	0.73 ± 0.02	0.02690 ± 0.00017	400	0.54 ± 0.01
NOGD	0.00394 ± 0.00013	400	3.63 ± 0.06	0.02800 ± 0.00032	400	3.08 ± 0.05
PROS-N-KONS	0.01509 ± 0.00028	88	4.45 ± 0.40	0.04336 ± 0.00146	45	1.52 ± 0.14
CON-KONS	0.00320 ± 0.00007	84	4.25 ± 0.35	0.02436 ± 0.00010	44	1.49 ± 0.13
AOGD-ALD	0.00345 ± 0.00002	58	0.42 ± 0.03	0.03034 ± 0.00006	29	0.25 ± 0.02
NONS-ALD	$\mathbf{0.00288 \pm 0.00001}$	**58**	1.72 ± 0.12	$\mathbf{0.02215 \pm 0.00011}$	**29**	0.59 ± 0.04
Algorithm	year , $\varsigma = 16$			TomsHardware, $\varsigma = 12$		
	MSE	$J\|D$	Time (s)	MSE	$J\|D$	Time (s)
FOGD	0.01501 ± 0.00004	400	4.04 ± 0.34	0.00080 ± 0.00003	400	2.28 ± 0.08
NOGD	0.01511 ± 0.00013	400	16.49 ± 0.45	0.00085 ± 0.00001	400	10.52 ± 0.27
PROS-N-KONS	0.01967 ± 0.00026	109	22.60 ± 3.33	0.00232 ± 0.00007	105	13.47 ± 1.45
CON-KONS	0.01370 ± 0.00004	107	23.73 ± 3.02	0.00054 ± 0.00001	108	14.88 ± 1.72
AOGD-ALD	0.01499 ± 0.00002	106	3.72 ± 0.23	0.00062 ± 0.00000	100	1.97 ± 0.07
NONS-ALD	$\mathbf{0.01243 \pm 0.00001}$	**106**	31.65 ± 0.61	$\mathbf{0.00043 \pm 0.00000}$	**100**	14.53 ± 0.38

performance will become bad. Both NONS-ALD and CON-KONS use carefully designed projection operations which keep the previous information. Besides, NONS-ALD performs better than CON-KONS which proves that our projection scheme in Lemma 4.2 and Lemma 4.3 is better than that of CON-KONS. All of the first-order algorithms have higher MSE than NONS-ALD and CON-KONS. The results are intuitive, since second-order algorithms use more information of the square loss function. The results very the first goal **G** 1.

Next we analyze the size of buffer. Both AOGD-ALD and NONS-ALD only store few examples. For instance, AOGD-ALD and NONS-ALD only store 13 examples on the *parkinson* dataset, and store 148 examples on the *bank* dataset. NOGD stores 400 examples, but still performs worse than our algorithms. CON-KONS and PROS-N-KONS also store more examples than our algorithms. It is worth mentioning that we must carefully tune the kernel function on each dataset. For instance, we set $\varsigma = 8$ for the *parkinson* dataset, while we set $\varsigma = 16$ for the *year* dataset. The results very the second goal **G** 2.

Finally, we analyze the average per-round running time. As a whole, the running time of AOGD-ALD and NONS-ALD is comparable with all of the baseline algorithms. AOGD-ALD even runs fastest on all datasets except for the *bank* dataset. The per-round time complexity of AOGD-ALD is $O(\min\{dJ + J^2, dT\})$. The average per-round time complexity of NONS-ALD is $O(dJ + J^2 + \frac{J^4}{T})$. The smaller J is, the faster AOGD-ALD and NONS-ALD will run. Note that changing the value of D in FOGD and J in NOGD will balance the prediction performance and computational cost. FOGD and NOGD can not perform better than NONS-ALD by increasing the value of D or J. The reason is that FOGD and NOGD are first-order algorithm. The results very the third goal **G** 3.

B. Reanalyze FOGD

In this section, we reanalyze the regret of FOGD (Lu et al., 2016), and aim to prove a regret of $O(\frac{\sqrt{TL(f)\ln\frac{1}{\delta}}}{\sqrt{D}})$. Our proof is similar with the proof of Theorem 1 in Lu et al. (2016). Thus we just show the critical differences.

For any $f=\sum_{t=1}^{T}a_t\kappa(\mathbf{x}_t,\cdot)\in\mathcal{H}$, we define $\tilde{f}=\sum_{t=1}^{T}a_t\tilde{\phi}(\mathbf{x}_t)$, where $\tilde{\phi}(\cdot)$ is the explicit feature mapping constructed by the random feature technique (Rahimi & Recht, 2007). The regret can be decomposed as follows,

$$\begin{aligned}\mathrm{Reg}(f)=&\sum_{t=1}^{T}\ell(\mathbf{w}_t^\top\tilde{\phi}_t(\mathbf{x}_t),y_t)-\sum_{t=1}^{T}\ell(\tilde{f}(\mathbf{x}_t),y_t)+\sum_{t=1}^{T}\ell(\tilde{f}(\mathbf{x}_t),y_t)-\sum_{t=1}^{T}\ell(f(\mathbf{x}_t),y_t)\\=&\underbrace{\sum_{t=1}^{T}\ell(\mathbf{w}_t^\top\tilde{\phi}_t(\mathbf{x}_t),y_t)-\sum_{t=1}^{T}\ell(\mathbf{w}^\top\tilde{\phi}_t(\mathbf{x}_t),y_t)}_{\mathcal{T}_1}+\underbrace{\sum_{t=1}^{T}\ell(\tilde{f}(\mathbf{x}_t),y_t)-\sum_{t=1}^{T}\ell(f(\mathbf{x}_t),y_t)}_{\mathcal{T}_2}.\end{aligned}$$

Following the original analysis of FOGD, $\mathcal{T}_1$ can be upper bounded as follows,

$$\mathcal{T}_1\le\frac{\|\mathbf{w}\|_2^2}{2\eta}+\frac{\eta}{2}\sum_{t=1}^{T}\ell(\mathbf{w}_t^\top\tilde{\phi}_t(\mathbf{x}_t),y_t)\le\frac{\|\mathbf{w}\|_2^2+1}{2}\sqrt{\sum_{t=1}^{T}\ell(\mathbf{w}_t^\top\tilde{\phi}_t(\mathbf{x}_t),y_t)}\le\frac{\|\mathbf{w}\|_2^2+1}{2}\sqrt{L(\tilde{f})}+\frac{(\|\mathbf{w}\|_2^2+1)^2}{4},$$

where we define the learning rate $\eta=\frac{1}{\sqrt{\sum_{t=1}^{T}\ell(\mathbf{w}_t^\top\tilde{\phi}_t(\mathbf{x}_t),y_t)}}$. Next we analyze $\mathcal{T}_2$. The random feature technique guarantees that, with probability at least $1-2^8(\sigma_pR/\epsilon)^2\exp(-D\epsilon^2/4(d+2))$, $|\tilde{\phi}^\top(\mathbf{x}_\tau)\tilde{\phi}(\mathbf{x}_t)-\kappa(\mathbf{x}_\tau,\mathbf{x}_t)|\le\epsilon$. We further obtain

$$\mathcal{T}_2\le\sum_{t=1}^{T}|\ell'(\tilde{f}(\mathbf{x}_t),y_t)|\cdot\|f\|_1\epsilon\le2\|f\|_1\epsilon\cdot\sqrt{T\sum_{t=1}^{T}\ell(\tilde{f}(\mathbf{x}_t),y_t)}\le2\|f\|_1\epsilon\cdot\sqrt{T\sum_{t=1}^{T}\ell(f(\mathbf{x}_t),y_t)+4T\|f\|_1^2\epsilon^2},$$

where $\|f\|_1=\sum_{t=1}^{T}|a_t|$. It was proved that $\|\mathbf{w}\|_2^2\le(1+\epsilon)\|f\|_1^2$ (Lu et al., 2016). Combining the upper bounds on $\mathcal{T}_1$ and $\mathcal{T}_2$ gives that, with probability at least $1-\delta$,

$$\mathrm{Reg}(f)=O\left(\|f\|_1^2\frac{T}{D}\ln\frac{1}{\delta}+(\|f\|_1^2+1)\sqrt{L(f)}+\frac{\|f\|_1\epsilon\cdot\sqrt{TL(f)}}{\sqrt{D}}\sqrt{\ln\frac{1}{\delta}}\right).$$

We conclude the proof. In Table 1, we omit the term $O(\frac{T}{D})$.

C. Proof of Theorem 3.2

Proof of Theorem 3.2. We first consider the case $|S_t|<\lfloor(\sqrt{d^2+4dT}-d)/2\rfloor$ for all $t=1,\ldots,T$.

$$\begin{aligned}\forall f\in\mathbb{H},\quad\mathrm{Reg}(f)\le&\sum_{t=1}^{T}\langle\hat{\nabla}_t,f_t-f\rangle+\langle\nabla_t-\hat{\nabla}_t,f_t-f\rangle\\\le&\sum_{t=1}^{T}\frac{1}{\eta_t}\langle f_t-\bar{f}_{t+1},f_t-f\rangle+\|f_t-f\|_{\mathcal{H}}\cdot|\ell'(f_t(\mathbf{x}_t),y_t)|\cdot\sqrt{\alpha_t}\cdot\mathbb{I}\{\alpha_t\le\alpha\}\\\le&\sum_{t=1}^{T}\frac{1}{2\eta_t}\left[\|f_t-f\|_{\mathcal{H}}^2-\|\bar{f}_{t+1}-f\|_{\mathcal{H}}^2+\|\bar{f}_{t+1}-f_t\|_{\mathcal{H}}^2\right]+2U\sqrt{\alpha}\sqrt{T\sum_{t=1}^{T}|\ell'(f_t(\mathbf{x}_t),y_t)|^2}\\\le&\frac{3U^2}{2\eta_T}+\frac{1}{2}\sum_{t=1}^{T}\eta_t\|\hat{\nabla}_t\|_{\mathcal{H}}^2+4UT^{\frac{1-\zeta}{2}}\sqrt{\sum_{t=1}^{T}\ell(f_t(\mathbf{x}_t),y_t)},\end{aligned}$$

where we define $\alpha=T^{-\zeta}$, $\zeta\ge0$. Recalling the definition of η_t. It is easy to prove that

$$\sum_{t=1}^{T}\frac{\|\hat{\nabla}_t\|_{\mathcal{H}}^2}{\sqrt{1+\sum_{\tau=1}^{t}\|\hat{\nabla}_\tau\|_{\mathcal{H}}^2}}\le2\sqrt{\sum_{\tau=1}^{T}\|\hat{\nabla}_\tau\|_{\mathcal{H}}^2}.$$

Let $\zeta = 1$. The final regret satisfies

$$\mathrm{Reg}(f) \leq \frac{3U}{2}\sqrt{1+\sum_{\tau=1}^{T}\|\hat{\nabla}_\tau\|_{\mathcal{H}}^2} + U\sqrt{\sum_{\tau=1}^{T}\|\hat{\nabla}_\tau\|_{\mathcal{H}}^2} + 4U\sqrt{\sum_{t=1}^{T}\ell(f_t(\mathbf{x}_t),y_t)} \leq 9U\sqrt{\sum_{t=1}^{T}\ell(f_t(\mathbf{x}_t),y_t)} + 3U.$$

Solving for $\sum_{t=1}^{T}\ell(f_t(\mathbf{x}_t),y_t)$ gives

$$\mathrm{Reg}(f) \leq 9U\sqrt{\sum_{t=1}^{T}\ell(f(\mathbf{x}_t),y_t) + 3U} + 81U^2.$$

Next we consider that there exists a $t_0 < T$ such that $|S_{t_0-1}| \leq \lfloor(\sqrt{d^2+4dT}-d)/2\rfloor$ and $|S_{t_0}| > \lfloor(\sqrt{d^2+4dT}-d)/2\rfloor$. For $t \geq t_0$, our algorithm just runs OGD which is equivalent ALD_t does not hold.

$$\begin{aligned}
\mathrm{Reg}(f) =& \sum_{t=1}^{t_0-1}[\ell(f_t(\mathbf{x}_t),y_t) - \ell(f(\mathbf{x}_t),y_t)] + \sum_{t=t_0}^{T}[\ell(f_t(\mathbf{x}_t),y_t) - \ell(f(\mathbf{x}_t),y_t)] \\
\leq& \sum_{t=1}^{t_0-1}\frac{1}{2\eta_t}\left[\|f_t-f\|_{\mathcal{H}}^2 - \|\bar{f}_{t+1}-f\|_{\mathcal{H}}^2 + \|\bar{f}_{t+1}-f_t\|_{\mathcal{H}}^2\right] + 2UT^{-\frac{\zeta}{2}}\sqrt{(t_0-1)\sum_{t=1}^{t_0-1}|\ell'(f_t(\mathbf{x}_t),y_t)|^2} + \\
& \sum_{t=t_0}^{T}\frac{1}{2\eta_t}\left[\|f_t-f\|_{\mathcal{H}}^2 - \|\bar{f}_{t+1}-f\|_{\mathcal{H}}^2 + \|\bar{f}_{t+1}-f_t\|_{\mathcal{H}}^2\right] \\
\leq& 9U\sqrt{\sum_{t=1}^{T}\ell(f_t(\mathbf{x}_t),y_t)} + 3U.
\end{aligned}$$

Solving for $\sum_{t=1}^{T}\ell(f_t(\mathbf{x}_t),y_t)$ gives the desired result. □

D. Proof of Theorem 4.1

We first give a technical lemma which has been stated in (Calandriello et al., 2017b).

Lemma D.1. *For any $\mathbf{X} \in \mathbb{R}^{n\times m}$ and $\alpha > 0$,*

$$\begin{aligned}
\mathbf{X}\mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top + \alpha\mathbf{I})^{-1} =& \mathbf{X}(\mathbf{X}^\top\mathbf{X} + \alpha\mathbf{I})^{-1}\mathbf{X}^\top, \\
(\mathbf{X}\mathbf{X}^\top + \alpha\mathbf{I})^{-1} =& \frac{1}{\alpha}(\mathbf{I} - \mathbf{X}(\mathbf{X}^\top\mathbf{X} + \alpha\mathbf{I})^{-1}\mathbf{X}^\top).
\end{aligned}$$

Proof of Theorem 4.1. For simplicity, let $g_t = \ell'(f_t(\mathbf{x}_t),y_t)$ and

$$\hat{\phi}(\mathbf{x}_t) = \begin{cases} \phi(\mathbf{x}_t) & \text{if } \mathrm{ALD}_t \text{ does not hold,} \\ \Phi_{S_t}\boldsymbol{\beta}_t^* & \text{otherwise.} \end{cases}$$

Let $\hat{\nabla}_t = g_t\hat{\phi}(\mathbf{x}_t)$ and $\hat{\boldsymbol{\Phi}}_t = (\sqrt{\eta_1}\hat{\nabla}_1,\ldots,\sqrt{\eta_t}\hat{\nabla}_t)$. We can rewrite

$$\mathbf{A}_t = \mathbf{A}_{t-1} + \eta_t\hat{\nabla}_t\hat{\nabla}_t^\top = \mu\mathbf{I} + \sum_{\tau=1}^{t}\eta_\tau\hat{\nabla}_\tau\hat{\nabla}_\tau^\top = \mu\mathbf{I} + \hat{\boldsymbol{\Phi}}_t\hat{\boldsymbol{\Phi}}_t^\top. \tag{15}$$

Recalling that

$$f_{t+1} = f_t - \mathbf{A}_t^{-1}\hat{\nabla}_t = f_{t-1} - \mathbf{A}_{t-1}^{-1}\hat{\nabla}_{t-1} - \mathbf{A}_t^{-1}\hat{\nabla}_t = \ldots = f_1 - \sum_{\tau=1}^{t}\mathbf{A}_\tau^{-1}\hat{\nabla}_\tau,$$

in which $f_1 = 0$. Using Lemma D.1 yields

$$\begin{aligned}
f_{t+1}(\mathbf{x}_{t+1}) =& f_{t+1}^\top \phi(\mathbf{x}_{t+1}) \\
=& -\sum_{\tau=1}^{t} \hat{\nabla}_\tau^\top \mathbf{A}_\tau^{-1} \phi(\mathbf{x}_{t+1}) \\
=& -\sum_{\tau=1}^{t} \hat{\nabla}_\tau^\top \left(\mu\mathbf{I} + \hat{\mathbf{\Phi}}_\tau \hat{\mathbf{\Phi}}_\tau^\top\right)^{-1} \phi(\mathbf{x}_{t+1}) \\
=& \frac{-1}{\mu} \sum_{\tau=1}^{t} \hat{\nabla}_\tau^\top \left(\mathbf{I} - \hat{\mathbf{\Phi}}_\tau (\hat{\mathbf{\Phi}}_\tau^\top \hat{\mathbf{\Phi}}_\tau + \mu\mathbf{I})^{-1} \hat{\mathbf{\Phi}}_\tau^\top\right) \phi(\mathbf{x}_{t+1}) \\
=& \frac{-1}{\mu} \sum_{\tau=1}^{t} g_\tau \cdot \left(\hat{\phi}(\mathbf{x}_\tau)^\top \phi(\mathbf{x}_{t+1}) - \hat{\phi}(\mathbf{x}_\tau)^\top \hat{\mathbf{\Phi}}_\tau (\hat{\mathbf{\Phi}}_\tau^\top \hat{\mathbf{\Phi}}_\tau + \mu\mathbf{I})^{-1} \hat{\mathbf{\Phi}}_\tau^\top \phi(\mathbf{x}_{t+1})\right),
\end{aligned}$$

which concludes the proof. □

E. Proof of Lemma 4.2

Proof of Lemma 4.2. Recalling that

$$\begin{aligned}
\mathbf{A}_j(s_j - 1) - \mu\mathbf{I} =& \sum_{r=1}^{j-1} \sum_{t\in T_r} \eta_t g_r^2(t) \tilde{\phi}_j(\mathbf{x}_t) \tilde{\phi}_j^\top(\mathbf{x}_t) \\
=& \sum_{r=1}^{j-1} \sum_{t\in T_r} \eta_t g_r^2(t) \mathcal{P}_{S(j)}^{\frac{1}{2}} \mathbf{\Phi}_{S(r)} \boldsymbol{\beta}_r^* (\mathbf{\Phi}_{S(r)} \boldsymbol{\beta}_r^*)^\top (\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top \\
=& \sum_{r=1}^{j-1} \sum_{t\in T_r} \eta_t g_r^2(t) \mathcal{P}_{S(j)}^{\frac{1}{2}} \mathbf{\Phi}_{S(r)} (\mathbf{\Phi}_{S(r)}^\top \mathbf{\Phi}_{S(r)})^{-1} \mathbf{\Phi}_{S(r)}^\top \phi(\mathbf{x}_t) (\mathbf{\Phi}_{S(r)} (\mathbf{\Phi}_{S(r)}^\top \mathbf{\Phi}_{S(r)})^{-1} \mathbf{\Phi}_{S(r)}^\top \phi(\mathbf{x}_t))^\top (\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top \\
=& \sum_{r=1}^{j-1} \sum_{t\in T_r} \eta_t g_r^2(t) \mathcal{P}_{S(j)}^{\frac{1}{2}} \mathcal{P}_{S(r)} \phi(\mathbf{x}_t) (\mathcal{P}_{S(r)} \phi(\mathbf{x}_t))^\top (\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top \\
=& \sum_{r=1}^{j-1} \sum_{t\in T_r} \eta_t g_r^2(t) \mathcal{P}_{S(j)}^{\frac{1}{2}} \mathcal{P}_{S(j-1)} \mathcal{P}_{S(r)} \phi(\mathbf{x}_t) (\mathcal{P}_{S(j-1)} \mathcal{P}_{S(r)} \phi(\mathbf{x}_t))^\top (\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top \\
=& \mathcal{P}_{S(j)}^{\frac{1}{2}} \Bigg((\mathcal{P}_{S(j-1)}^{\frac{1}{2}})^\top \sum_{r=1}^{j-2} \sum_{t\in T_r} \eta_t g_r^2(t) \mathcal{P}_{S(j-1)}^{\frac{1}{2}} \mathcal{P}_{S(r)} \phi(\mathbf{x}_t) (\mathcal{P}_{S(r)} \phi(\mathbf{x}_t))^\top (\mathcal{P}_{S(j-1)}^{\frac{1}{2}})^\top \mathcal{P}_{S(j-1)}^{\frac{1}{2}} + \\
& (\mathcal{P}_{S(j-1)}^{\frac{1}{2}})^\top \sum_{t\in T_{j-1}} \eta_t g_{j-1}^2(t) \mathcal{P}_{S(j-1)}^{\frac{1}{2}} \phi(\mathbf{x}_t) (\mathcal{P}_{S(j-1)}^{\frac{1}{2}} \phi(\mathbf{x}_t))^\top \mathcal{P}_{S(j-1)}^{\frac{1}{2}} \Bigg) (\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top \\
=& \mathcal{P}_{S(j)}^{\frac{1}{2}} (\mathcal{P}_{S(j-1)}^{\frac{1}{2}})^\top \left(\mathbf{A}_{j-1}(s_{j-1} - 1) - \mu\mathbf{I} + \sum_{t\in T_{j-1}} \eta_t g_{j-1}^2(t) \phi_{j-1}(\mathbf{x}_t) \phi_{j-1}^\top(\mathbf{x}_t) \right) \mathcal{P}_{S(j-1)}^{\frac{1}{2}} (\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top \\
=& \mathcal{P}_{S(j)}^{\frac{1}{2}} (\mathcal{P}_{S(j-1)}^{\frac{1}{2}})^\top (\mathbf{A}_{j-1}(s_j - 1) - \mu\mathbf{I}) \mathcal{P}_{S(j-1)}^{\frac{1}{2}} (\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top.
\end{aligned}$$

Thus we can obtain

$$\mathbf{A}_j(s_j - 1) = \mu\mathbf{I} + \sum_{r=1}^{j-1} \sum_{t\in T_r} \eta_t g_r^2(t) \tilde{\phi}_j(\mathbf{x}_t) \tilde{\phi}_j^\top(\mathbf{x}_t) = \mu\mathbf{I} + \mathcal{P}_{S(j)}^{\frac{1}{2}} (\mathcal{P}_{S(j-1)}^{\frac{1}{2}})^\top (\mathbf{A}_{j-1}(s_j - 1) - \mu\mathbf{I}) \mathcal{P}_{S(j-1)}^{\frac{1}{2}} (\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top,$$

which concludes the proof. □

F. Proof of Lemma 4.3

Proof of Lemma 4.3. We directly use the definition of $\mathbf{w}_j(s_j)$.

$$\begin{aligned}
&\mathbf{w}_j(s_j) = \mathcal{P}^{\frac{1}{2}}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j-1)})^\top \mathbf{w}_{j-1}(s_j)\\
\Rightarrow&(\mathcal{P}^{\frac{1}{2}}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j-1)})^\top)^\top \mathbf{w}_j(s_j) = (\mathcal{P}^{\frac{1}{2}}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j-1)})^\top)^\top \mathcal{P}^{\frac{1}{2}}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j-1)})^\top \mathbf{w}_{j-1}(s_j)\\
\Rightarrow&\mathcal{P}^{\frac{1}{2}}_{S(j-1)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top \mathbf{w}_j(s_j) = \mathcal{P}^{\frac{1}{2}}_{S(j-1)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top \mathcal{P}^{\frac{1}{2}}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j-1)})^\top \mathbf{w}_{j-1}(s_j)\\
\Rightarrow&\mathcal{P}^{\frac{1}{2}}_{S(j-1)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top \mathbf{w}_j(s_j) = \mathcal{P}^{\frac{1}{2}}_{S(j-1)}\mathcal{P}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j-1)})^\top \mathbf{w}_{j-1}(s_j)\\
\Rightarrow&\mathcal{P}^{\frac{1}{2}}_{S(j-1)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top \mathbf{w}_j(s_j) = \mathbf{w}_{j-1}(s_j),
\end{aligned}$$

where we use the fact $\mathcal{P}^{\frac{1}{2}}_{S(j-1)}(\mathcal{P}^{\frac{1}{2}}_{S(j-1)})^\top = \mathbf{I}$. Next we prove $\mathbf{w}_j(s_j) \in \mathbb{W}_{s_j}$.

$$\begin{aligned}
(\mathbf{w}_j(s_j))^\top \phi_j(\mathbf{x}_{s_j}) = (\mathbf{w}_{j-1}(s_j))^\top \mathcal{P}^{\frac{1}{2}}_{S(j-1)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top \mathcal{P}^{\frac{1}{2}}_{S(j)}\phi(\mathbf{x}_{s_j}) =&(\mathbf{w}_{j-1}(s_j))^\top \mathcal{P}^{\frac{1}{2}}_{S(j-1)}\phi(\mathbf{x}_{s_j})\\
=&(\mathbf{w}_{j-1}(s_j))^\top \phi_{j-1}(\mathbf{x}_{s_j}).
\end{aligned}$$

Since $|(\mathbf{w}_{j-1}(s_j))^\top \phi_{j-1}(\mathbf{x}_{s_j})| \le U$, it must be $\mathbf{w}_j(s_j) \in \mathbb{W}_{s_j}$. Thus we conclude the proof. □

G. Proof of Lemma 4.5

We first prove a technique lemma.

Lemma G.1. *For all $j = 1, \dots, J$, let $\mathcal{P}_{S(j)}$ be the projection matrix onto the column space of $\mathbf{\Phi}_{S(j)}$. For all $t \in T_j$,*

$$0 \le \kappa(\mathbf{x}_t, \mathbf{x}_t) - \phi_j^\top(\mathbf{x}_t)\phi_j(\mathbf{x}_t) \le \alpha,$$

For any $r \in [J]$ and $t \in T_r$, denote by $\tilde{\phi}_J(\mathbf{x}_t) = \mathcal{P}^{\frac{1}{2}}_{S(J)}\mathbf{\Phi}_{S(r)}\boldsymbol{\beta}^*_r(t)$. Then for any $i, j \in [J]$ and for any $t \in T_i, \tau \in T_j$,

$$\kappa(\mathbf{x}_t, \mathbf{x}_\tau) - \tilde{\phi}_J^\top(\mathbf{x}_t)\tilde{\phi}_J(\mathbf{x}_\tau) \le \sqrt{\alpha}.$$

Proof of Lemma G.1. For any $t \in T_j$, $S_t = S(j)$.

$$\begin{aligned}
\kappa(\mathbf{x}_t, \mathbf{x}_t) - \phi_j^\top(\mathbf{x}_t)\phi_j(\mathbf{x}_t) &= \kappa(\mathbf{x}_t, \mathbf{x}_t) - \phi(\mathbf{x}_t)\mathcal{P}_{S(j)}\phi(\mathbf{x}_t)\\
&= \kappa(\mathbf{x}_t, \mathbf{x}_t) - (\mathbf{\Phi}^\top_{S(j)}\kappa(\mathbf{x}_t, \cdot))^\top \mathbf{K}^{-1}_{S(j)}\mathbf{\Phi}^\top_{S(j)}\kappa(\mathbf{x}_t, \cdot)\\
&= \alpha_t \in [0, \alpha],
\end{aligned}$$

where we use the fact that the ALD_t condition holds.
Next we consider $t \in T_i$ and $\tau \in T_j$. Without loss of generality, assuming that $i < j$.

$$\begin{aligned}
\kappa(\mathbf{x}_t, \mathbf{x}_\tau) - \tilde{\phi}_J^\top(\mathbf{x}_t)\tilde{\phi}_J(\mathbf{x}_\tau) =&\phi(\mathbf{x}_t)^\top\phi(\mathbf{x}_\tau) - \left(\mathcal{P}^{\frac{1}{2}}_{S(J)}\mathbf{\Phi}_{S(i)}\boldsymbol{\beta}^*_i(t)\right)^\top \mathcal{P}^{\frac{1}{2}}_{S(J)}\mathbf{\Phi}_{S(j)}\boldsymbol{\beta}^*_j(\tau)\\
=&\phi(\mathbf{x}_t)^\top\phi(\mathbf{x}_\tau) - \left(\mathcal{P}_{S(i)}\phi(\mathbf{x}_t)\right)^\top \mathcal{P}_{S(J)}\mathcal{P}_{S(j)}\phi(\mathbf{x}_\tau)\\
=&\phi(\mathbf{x}_t)^\top\phi(\mathbf{x}_\tau) - \phi(\mathbf{x}_t)^\top\mathcal{P}_{S(i)}\mathcal{P}_{S(J)}\mathcal{P}_{S(j)}\phi(\mathbf{x}_\tau)\\
=&\phi(\mathbf{x}_t)^\top\phi(\mathbf{x}_\tau) - \phi(\mathbf{x}_t)^\top\mathcal{P}_{S(i)}\phi(\mathbf{x}_\tau)\\
\le&\sqrt{\|\phi(\mathbf{x}_t) - \mathcal{P}_{S(i)}\phi(\mathbf{x}_t)\|^2_{\mathcal{H}}} \cdot \|\phi(\mathbf{x}_\tau)\|_{\mathcal{H}}\\
=&\sqrt{\phi(\mathbf{x}_t)^\top\phi(\mathbf{x}_t) - \phi(\mathbf{x}_t)^\top\mathcal{P}_{S(i)}\phi(\mathbf{x}_t)} \cdot \|\phi(\mathbf{x}_\tau)\|_{\mathcal{H}}\\
\le&\sqrt{\alpha},
\end{aligned}$$

which concludes the proof. □

Proof of Lemma 4.5. Denote by $\mathbf{\Phi}_{T_j} = (\phi(\mathbf{x}_{s_j}), \phi(\mathbf{x}_{s_j+1}), \ldots, \phi(\mathbf{x}_{s_{j+1}-1})) \in \mathbb{R}^{n\times|T_j|}$, $\mathbf{K}_{T_j} = \mathbf{\Phi}_{T_j}^\top \mathbf{\Phi}_{T_j}$ and $\tilde{\mathbf{K}}_{T_j} = \mathbf{\Phi}_{T_j}^\top \mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j}$. Let $\mathbf{K}_- = \mathbf{K}_{T_j} - \tilde{\mathbf{K}}_{T_j}$. We first prove that $\mathbf{K}_-$ is a positive semi-definite (PSD) matrix. Let $\mathbf{\Phi}_{[e_j]} = (\phi(\mathbf{x}_{s_1}), \phi(\mathbf{x}_{s_2}), \ldots, \phi(\mathbf{x}_{s_j}), \phi(\mathbf{x}_{s_j+1}), \ldots, \phi(\mathbf{x}_{s_{j+1}-1})) \in \mathbb{R}^{n\times(|T_j|+j-1)}$, $\mathbf{K}_{[e_j]}^{-1} = \mathbf{\Phi}_{[e_j]}^\top \mathbf{\Phi}_{[e_j]}$, and $\mathcal{P}_{[e_j]} = \mathbf{\Phi}_{[e_j]} \mathbf{K}_{[e_j]}^{-1} \mathbf{\Phi}_{[e_j]}^\top$ be the projection matrix on the column space of $\mathbf{\Phi}_{[e_j]}$. We have

$$\begin{aligned}
\mathbf{K}_{T_j} - \tilde{\mathbf{K}}_{T_j} =& \mathbf{\Phi}_{T_j}^\top \mathcal{P}_{[e_j]} \mathbf{\Phi}_{T_j} - \mathbf{\Phi}_{T_j}^\top \mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} \\
=& \mathbf{\Phi}_{I_j}^\top (\mathcal{P}_{[e_j]}^\top \mathcal{P}_{[e_j]} - \mathcal{P}_{S(j)}^\top \mathcal{P}_{S(j)}) \mathbf{\Phi}_{I_j} \\
=& \mathbf{\Phi}_{I_j}^\top (\mathcal{P}_{[e_j]} - \mathcal{P}_{S(j)})^\top (\mathcal{P}_{[e_j]} - \mathcal{P}_{S(j)}) \mathbf{\Phi}_{T_j} \\
=& \left((\mathcal{P}_{[e_j]} - \mathcal{P}_{S(j)}) \mathbf{\Phi}_{T_j}\right)^\top (\mathcal{P}_{[e_j]} - \mathcal{P}_{S(j)}) \mathbf{\Phi}_{T_j},
\end{aligned}$$

where $\mathcal{P}_{[e_j]}$ and $\mathcal{P}_{S(j)}$ satisfy $\mathcal{P}_{[e_j]}^\top \mathcal{P}_{[e_j]} = \mathcal{P}_{[e_j]}$ and $\mathcal{P}_{S(j)}^\top \mathcal{P}_{S(j)} = \mathcal{P}_{S(j)}$. Besides,

$$\mathcal{P}_{[e_j]}^\top \mathcal{P}_{S(j)} = \mathcal{P}_{[e_j]} \mathbf{\Phi}_{S(j)} (\mathbf{\Phi}_{S(j)}^\top \mathbf{\Phi}_{S(j)})^{-1} \mathbf{\Phi}_{S(j)}^\top = \mathcal{P}_{S(j)},$$

where $\mathbf{\Phi}_{S(j)}$ belongs to the column space of $\mathbf{\Phi}_{[e_j]}$. For any $\mathbf{a} \in \mathbb{R}^{|T_j|}$, we have

$$\mathbf{a}^\top \mathbf{K}_- \mathbf{a} = \left((\mathcal{P}_{[e_j]} - \mathcal{P}_{S(j)}) \mathbf{\Phi}_{T_j} \mathbf{a}\right)^\top (\mathcal{P}_{[e_j]} - \mathcal{P}_{S(j)}) \mathbf{\Phi}_{I_j} \mathbf{a} = \left\|(\mathcal{P}_{[e_j]} - \mathcal{P}_{S(j)}) \mathbf{\Phi}_{T_j} \mathbf{a}\right\|_{\mathcal{H}}^2 \geq 0.$$

Thus $\mathbf{K}_-$ is a PSD matrix. Lemma G.1 gives $\mathbf{K}_-[i,i] \in [0, \alpha]$. Thus we have

$$\left\|\mathbf{K}_{T_j} - \tilde{\mathbf{K}}_{T_j}\right\|_2 \leq \operatorname{tr}\left(\mathbf{K}_{T_j} - \tilde{\mathbf{K}}_{T_j}\right) \leq |T_j| \cdot \alpha.$$

Let $\tilde{\mathbf{\Phi}}_T = \left((\tilde{\phi}_J(\mathbf{x}_t))_{t\in T_1}, \ldots, (\tilde{\phi}_J(\mathbf{x}_t))_{t\in T_J}\right)$. The second statement in Lemma G.1 can derive

$$\left\|\mathbf{K}_T - \tilde{\mathbf{\Phi}}_T^\top \tilde{\mathbf{\Phi}}_T\right\|_2 \leq \left\|\mathbf{K}_T - \tilde{\mathbf{\Phi}}_T^\top \tilde{\mathbf{\Phi}}_T\right\|_F \leq T\sqrt{\alpha},$$

which concludes the proof. □

H. Proof of Theorem 4.6

We first give some technical lemmas.

H.1. Technical Lemmas

Lemma H.1. *For any $f \in \mathbb{H}$, let f_j be the projection of f onto the column space of $\mathbf{\Phi}_{S(j)}$, and f_{j+1} be the projection of f onto the column space of $\mathbf{\Phi}_{S(j+1)}$. The following three claims hold: (i) There exist $\mathbf{w}_j = \mathcal{P}_{S(j)}^{\frac{1}{2}} f \in \mathbb{R}^j$ and $\mathbf{w}_{j+1} = \mathcal{P}_{S(j+1)}^{\frac{1}{2}} f \in \mathbb{R}^{j+1}$, such that $f_j(\mathbf{x}) = \mathbf{w}_j^\top \phi_j(\mathbf{x})$ and $f_{j+1}(\mathbf{x}) = \mathbf{w}_{j+1}^\top \phi_{j+1}(\mathbf{x})$, (ii) $\mathbf{w}_j \in \cap_{t=1}^T \mathbb{W}_t$ and $\mathbf{w}_{j+1} \in \cap_{t=1}^T \mathbb{W}_t$, (iii) $\mathbf{w}_j = \mathcal{P}_{S(j)}^{\frac{1}{2}} (\mathcal{P}_{S(j+1)}^{\frac{1}{2}})^\top \mathbf{w}_{j+1}$.*

Proof of Lemma H.1. For any $f \in \mathbb{H}$, the projection of f on the column space of $\mathbf{\Phi}_{S(j+1)}$ and $\mathbf{\Phi}_{S(j)}$ are

$$f_{j+1} = \mathcal{P}_{S(j+1)} f, \quad f_j = \mathcal{P}_{S(j)} f = \mathcal{P}_{S(j)} \mathcal{P}_{S(j+1)} f = \mathcal{P}_{S(j)} f_{j+1}.$$

We have

$$f_j(\mathbf{x}_t) = (\mathcal{P}_{S(j)} f)^\top \phi(\mathbf{x}_t) = f^\top (\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top \mathcal{P}_{S(j)}^{\frac{1}{2}} \phi(\mathbf{x}_t) = (\mathcal{P}_{S(j)}^{\frac{1}{2}} f)^\top \phi_j(\mathbf{x}_t) = \mathbf{w}_j^\top \phi_j(\mathbf{x}_t).$$

Thus we obtain

$$\mathbf{w}_j = \mathcal{P}_{S(j)}^{\frac{1}{2}} f, \quad \mathbf{w}_{j+1} = \mathcal{P}_{S(j+1)}^{\frac{1}{2}} f,$$

which concludes the first claim.
For the second claim, we have

$$\forall t \in [T], \ |\mathbf{w}_j^\top \phi_j(\mathbf{x}_t)| = |(\mathcal{P}_{S(j)}^{\frac{1}{2}} f)^\top \mathcal{P}_{S(j)}^{\frac{1}{2}} \phi(\mathbf{x}_t)| = |f^\top \mathcal{P}_{S(j)} \phi(\mathbf{x}_t)| \leq \|f\|_{\mathcal{H}} \leq U.$$

Thus $\mathbf{w}_j \in \cap_{t=1}^T \mathbb{W}_t$. Similarly, we have $\mathbf{w}_{j+1} \in \cap_{t=1}^T \mathbb{W}_t$.

Since $\mathcal{P}_{S(j)}^{\frac{1}{2}} = \mathcal{P}_{S(j)}^{\frac{1}{2}} \mathcal{P}_{S(j+1)}$, we have,

$$\mathbf{w}_j = \mathcal{P}_{S(j)}^{\frac{1}{2}} \mathcal{P}_{S(j+1)} f = \mathcal{P}_{S(j)}^{\frac{1}{2}} (\mathcal{P}_{S(j+1)}^{\frac{1}{2}})^\top \mathcal{P}_{S(j+1)}^{\frac{1}{2}} f = \mathcal{P}_{S(j)}^{\frac{1}{2}} (\mathcal{P}_{S(j+1)}^{\frac{1}{2}})^\top \mathbf{w}_{j+1}$$

which concludes the third claim. □

Lemma H.2 ((Hazan et al., 2007)). *Let $\mathbf{u}_t \in \mathbb{R}^j$ for $t = 1, \ldots, T$ be a sequence of vectors such that for some $r > 0$, $\|\mathbf{u}_t\| \leq r$. Let $\mu > 0$. Define $\mathbf{V}_t = \sum_{\tau=1}^t \mathbf{u}_\tau \mathbf{u}_\tau^\top + \mu \mathbf{I}$. Then*

$$\sum_{t=1}^T \mathbf{u}_t^\top \mathbf{V}_t^{-1} \mathbf{u}_t \leq \sum_{t=1}^T \ln \frac{\det(\mathbf{V}_t)}{\det(\mathbf{V}_{t-1})} = \ln \frac{\det(\mathbf{V}_T)}{\det(\mathbf{V}_0)} = \ln \det \left(\frac{1}{\mu} \sum_{\tau=1}^T \mathbf{u}_\tau \mathbf{u}_\tau^\top + \mathbf{I} \right),$$

where $\mathbf{V}_0 = \mu \mathbf{I}$.

Lemma H.3. *For any $j = 1, \ldots, J$,*

$$\frac{\mathrm{Det}(\mathbf{A}_j(s_j - 1))}{\mathrm{Det}(\mathbf{A}_{j-1}(s_j - 1))} = \mu.$$

Proof of Lemma H.3. For any $r \leq j$ and $t \in T_r$, let $\bar{\phi}(\mathbf{x}_t) = \sqrt{\eta_t} g_r(t) \phi(\mathbf{x}_t)$. Recalling that

$$\begin{aligned}
\mathbf{A}_{j-1}(s_j - 1) =& \mathbf{A}_{j-1}(s_{j-1} - 1) + \sum_{t \in T_{j-1}} \eta_t g_{j-1}^2(t) \phi_{j-1}(\mathbf{x}_t) \phi_{j-1}^\top(\mathbf{x}_t) \\
=& \mu \mathbf{I} + \sum_{r=1}^{j-2} \sum_{t \in T_r} \eta_t g_r^2(t) \tilde{\phi}_{j-1}(\mathbf{x}_t) \tilde{\phi}_{j-1}^\top(\mathbf{x}_t) + \sum_{t \in T_{j-1}} \eta_t g_{j-1}^2(t) \mathcal{P}_{S(j-1)}^{\frac{1}{2}} \phi(\mathbf{x}_t) (\mathcal{P}_{S(j-1)}^{\frac{1}{2}} \phi(\mathbf{x}_t))^\top \\
=& \mu \mathbf{I} + \sum_{r=1}^{j-1} \sum_{t \in T_r} \mathcal{P}_{S(j-1)}^{\frac{1}{2}} \mathcal{P}_{S(r)} \bar{\phi}(\mathbf{x}_t) (\mathcal{P}_{S(j-1)}^{\frac{1}{2}} \mathcal{P}_{S(r)} \bar{\phi}(\mathbf{x}_t))^\top \\
=& \mu \mathbf{I} + \bar{\mathbf{\Phi}}_{S(j-1)} \bar{\mathbf{\Phi}}_{S(j-1)}^\top,
\end{aligned}$$

where

$$\bar{\mathbf{\Phi}}_{S(j-1)} = \mathcal{P}_{S(j-1)}^{\frac{1}{2}} \left[\left(\mathcal{P}_{S(r)} \bar{\phi}(\mathbf{x}_t) \right)_{t \in T_r} \right]_{r \in [j-1]} \in \mathbb{R}^{(j-1) \times \sum_{r=1}^{j-1} |T_r|}.$$

Similarly, we have

$$\begin{aligned}
\mathbf{A}_j(s_j - 1) =& \mu \mathbf{I} + \sum_{r=1}^{j-1} \sum_{t \in T_r} \eta_t g_r^2(t) \tilde{\phi}_j(\mathbf{x}_t) \tilde{\phi}_j^\top(\mathbf{x}_t) \\
=& \mu \mathbf{I} + \sum_{r=1}^{j-1} \sum_{t \in T_r} \mathcal{P}_{S(j)}^{\frac{1}{2}} \mathcal{P}_{S(r)} \bar{\phi}(\mathbf{x}_t) (\mathcal{P}_{S(j)}^{\frac{1}{2}} \mathcal{P}_{S(r)} \bar{\phi}(\mathbf{x}_t))^\top \\
=& \mu \mathbf{I} + \bar{\mathbf{\Phi}}_{S(j)} \bar{\mathbf{\Phi}}_{S(j)}^\top,
\end{aligned}$$

where we define

$$\bar{\mathbf{\Phi}}_{S(j)} = \mathcal{P}_{S(j)}^{\frac{1}{2}} \left[\left(\mathcal{P}_{S(r)} \bar{\phi}(\mathbf{x}_t) \right)_{t \in T_r} \right]_{r \in [j-1]} \in \mathbb{R}^{j \times \sum_{r=1}^{j-1} |T_r|}.$$

We have the following two facts.

$$\mathrm{rank}(\bar{\mathbf{\Phi}}_{S(j-1)} \bar{\mathbf{\Phi}}_{S(j-1)}^\top) = \mathrm{rank}(\bar{\mathbf{\Phi}}_{S(j-1)}^\top \bar{\mathbf{\Phi}}_{S(j-1)}), \quad \mathrm{rank}(\bar{\mathbf{\Phi}}_{S(j)} \bar{\mathbf{\Phi}}_{S(j)}^\top) = \mathrm{rank}(\bar{\mathbf{\Phi}}_{S(j)}^\top \bar{\mathbf{\Phi}}_{S(j)}).$$

We can prove

$$\begin{aligned}\bar{\mathbf{\Phi}}_{S(j)}^\top\bar{\mathbf{\Phi}}_{S(j)}-\bar{\mathbf{\Phi}}_{S(j-1)}^\top\bar{\mathbf{\Phi}}_{S(j-1)}=&\left(\left[\left(\mathcal{P}_{S(r)}\bar{\phi}(\mathbf{x}_t)\right)_{t\in T_r}\right]_{r\in[j-1]}\right)^\top\mathcal{P}_{S(j)}\left[\left(\mathcal{P}_{S(r)}\bar{\phi}(\mathbf{x}_t)\right)_{t\in T_r}\right]_{r\in[j-1]}-\\&\left(\left[\left(\mathcal{P}_{S(r)}\bar{\phi}(\mathbf{x}_t)\right)_{t\in T_r}\right]_{r\in[j-1]}\right)^\top\mathcal{P}_{S(j-1)}\left[\left(\mathcal{P}_{S(r)}\bar{\phi}(\mathbf{x}_t)\right)_{t\in T_r}\right]_{r\in[j-1]}\\=&\left(\left[\left(\mathcal{P}_{S(r)}\bar{\phi}(\mathbf{x}_t)\right)_{t\in T_r}\right]_{r\in[j-1]}\right)^\top\left[\left(\mathcal{P}_{S(r)}\bar{\phi}(\mathbf{x}_t)\right)_{t\in T_r}\right]_{r\in[j-1]}-\\&\left(\left[\left(\mathcal{P}_{S(r)}\bar{\phi}(\mathbf{x}_t)\right)_{t\in T_r}\right]_{r\in[j-1]}\right)^\top\left[\left(\mathcal{P}_{S(r)}\bar{\phi}(\mathbf{x}_t)\right)_{t\in T_r}\right]_{r\in[j-1]}\\=&0,\end{aligned}$$

in which we use the following facts

$$\begin{aligned}&\mathcal{P}_{S(j)}=(\mathcal{P}_{S(j)}^{\frac{1}{2}})^\top\mathcal{P}_{S(j)}^{\frac{1}{2}},\\&\mathcal{P}_{S(r)}\mathcal{P}_{S(j)}=\mathbf{\Phi}_{S(r)}(\mathbf{\Phi}_{S(r)}^\top\mathbf{\Phi}_{S(r)})\mathbf{\Phi}_{S(r)}^\top\mathcal{P}_{S(j)}=\mathcal{P}_{S(r)},r=1,\ldots,j-1,\\&\mathcal{P}_{S(r)}\mathcal{P}_{S(j-1)}=\mathbf{\Phi}_{S(r)}(\mathbf{\Phi}_{S(r)}^\top\mathbf{\Phi}_{S(r)})\mathbf{\Phi}_{S(r)}^\top\mathcal{P}_{S(j-1)}=\mathcal{P}_{S(r)},r=1,\ldots,j-1.\end{aligned}$$

It must be that $\bar{\mathbf{\Phi}}_{S(j)}\bar{\mathbf{\Phi}}_{S(j)}^\top$ and $\bar{\mathbf{\Phi}}_{S(j-1)}\bar{\mathbf{\Phi}}_{S(j-1)}^\top$ have the same non-zero eigenvalues, denoted by $\bar{\lambda}_1,\bar{\lambda}_2,\ldots,\bar{\lambda}_k$, $k\leq j-1$. We have

$$\frac{\mathrm{Det}(\mathbf{A}_j(s_j-1))}{\mathrm{Det}(\mathbf{A}_{j-1}(s_j-1))}=\frac{\prod_{r=1}^j(\mu+\bar{\lambda}_r)}{\prod_{r=1}^{j-1}(\mu+\bar{\lambda}_r)}=\frac{\prod_{r=1}^k(\mu+\bar{\lambda}_r)\cdot\mu^{j-k}}{\prod_{r=1}^k(\mu+\bar{\lambda}_r)\cdot\mu^{j-k-1}}=\mu,$$

which concludes the proof. □

Proof of Theorem 4.6. Let f_j be the projection of $f\in\mathbb{H}$ onto the column space of $\mathbf{\Phi}_{S(j)}$, $j=1,\ldots,J$. We decompose the regret into two components.

$$\begin{aligned}\forall f\in\mathbb{H},\ \mathrm{Reg}(f)=&\sum_{j=1}^J\sum_{t\in T_j}[\ell(\hat{y}_t,y_t)-\ell(f(\mathbf{x}_t),y_t)]\\=&\sum_{j=1}^J\sum_{t\in T_j}[\ell(\hat{y}_t,y_t)-\ell(f_j(\mathbf{x}_t),y_t)]+\sum_{j=1}^J\sum_{t\in T_j}[\ell(f_j(\mathbf{x}_t),y_t)-\ell(f(\mathbf{x}_t),y_t)]\\=&\underbrace{\sum_{j=1}^J\sum_{t\in T_j}[\ell(\hat{y}_t,y_t)-\ell(\mathbf{w}_j^\top\phi_j(\mathbf{x}_t),y_t)]}_{\mathcal{T}_1}+\underbrace{\sum_{j=1}^J\sum_{t\in T_j}[\ell(f_j(\mathbf{x}_t),y_t)-\ell(f(\mathbf{x}_t),y_t)]}_{\mathcal{T}_2}.\end{aligned}$$

Lemma H.1 proved that there is a $\mathbf{w}_j\in\mathbb{R}^j$ such that $f_j(\mathbf{x}_t)=\mathbf{w}_j^\top\phi_j(\mathbf{x}_t)$.

H.2. Analyze $\mathcal{T}_1$

We consider a fixed epoch T_j. At any round $t\in T_j$, the instantaneous regret can be upper bounded as follows

$$\begin{aligned}&\ell(\hat{y}_t,y_t)-\ell(\mathbf{w}_j^\top\phi_j(\mathbf{x}_t),y_t)\\=&(\hat{y}_t-y_t)^2-(\mathbf{w}_j^\top\phi_j(\mathbf{x}_t)-y_t)^2\\=&2(\hat{y}_t-y_t)(\hat{y}_t-\mathbf{w}_j^\top\phi_j(\mathbf{x}_t))-(\hat{y}_t-\mathbf{w}_j^\top\phi_j(\mathbf{x}_t))^2\\=&\langle\nabla\ell(\mathbf{w}_j^\top(t)\phi_j(\mathbf{x}_t)),\mathbf{w}_j(t)-\mathbf{w}_j\rangle-\frac{1}{4(\hat{y}_t-y_t)^2}\left(\langle\nabla\ell(\mathbf{w}_j^\top(t)\phi_j(\mathbf{x}_t)),\mathbf{w}_j(t)-\mathbf{w}_j\rangle\right)^2\\\leq&\langle\nabla\ell(\mathbf{w}_j^\top(t)\phi_j(\mathbf{x}_t)),\mathbf{w}_j(t)-\mathbf{w}_j\rangle-\frac{1}{8(U^2+Y^2)}\left(\langle\nabla\ell(\mathbf{w}_j^\top(t)\phi_j(\mathbf{x}_t)),\mathbf{w}_j(t)-\mathbf{w}_j\rangle\right)^2.\end{aligned}$$

For simplicity, denote by $\sigma = \frac{1}{8(U^2+Y^2)}$ and $\nabla_j(t) = \nabla \ell(\mathbf{w}_j^\top(t)\phi_j(\mathbf{x}_t)) = \ell'(\hat{y}_t, y_t)\phi_j(\mathbf{x}_t)$.
Lemma H.1 has proved that $\mathbf{w}_j \in \mathbb{W}_{t+1}$. Using the property of projection, we have

$$\begin{aligned}
&\|\mathbf{w}_j(t+1) - \mathbf{w}_j\|^2_{\mathbf{A}_j(t)} - \|\mathbf{w}_j(t) - \mathbf{w}_j\|^2_{\mathbf{A}_j(t)} \\
\leq &\|\tilde{\mathbf{w}}_j(t+1) - \mathbf{w}_j\|^2_{\mathbf{A}_j(t)} - \|\mathbf{w}_j(t) - \mathbf{w}_j\|^2_{\mathbf{A}_j(t)} \\
= &\|\mathbf{w}_j(t) - \mathbf{A}_j^{-1}(t)\nabla_j(t) - \mathbf{w}_j\|^2_{\mathbf{A}_j(t)} - \|\mathbf{w}_j(t) - \mathbf{w}_j\|^2_{\mathbf{A}_j(t)} \\
= &-2\langle \mathbf{w}_j(t) - \mathbf{w}_j, \mathbf{A}_j^{-1}(t)\nabla_j(t)\rangle_{\mathbf{A}_j(t)} + \|\mathbf{A}_j^{-1}(t)\nabla_j(t)\|^2_{\mathbf{A}_j(t)} \\
= &-2\langle \mathbf{w}_j(t) - \mathbf{w}_j, \nabla_j(t)\rangle + \nabla_j^\top(t)\mathbf{A}_j^{-1}(t)\nabla_j(t).
\end{aligned}$$

Let $\eta_t = 2\sigma$. Rearranging terms and summing over $t \in T_j = \{s_j, s_j+1, \ldots, s_{j+1}-1\}$ gives

$$\begin{aligned}
&\sum_{t=s_j}^{s_{j+1}-1} \left(\langle \mathbf{w}_j(t) - \mathbf{w}_j, \nabla_j(t)\rangle - \sigma \left(\langle \nabla_j(t), \mathbf{w}_j(t) - \mathbf{w}_j\rangle\right)^2 \right) \\
\leq &\sum_{t=s_j}^{s_{j+1}-1} \left(\frac{\|\mathbf{w}_j(t) - \mathbf{w}_j\|^2_{\mathbf{A}_j(t)} - \|\mathbf{w}_j(t+1) - \mathbf{w}_j\|^2_{\mathbf{A}_j(t)}}{2} + \frac{\nabla_j^\top(t)\mathbf{A}_j^{-1}(t)\nabla_j(t)}{2} - \sigma \|\mathbf{w}_j(t) - \mathbf{w}_j\|^2_{\nabla_j(t)\nabla_j^\top(t)} \right) \\
= &\frac{\|\mathbf{w}_j(s_j) - \mathbf{w}_j\|^2_{\mathbf{A}_j(s_j)}}{2} - \frac{\|\mathbf{w}_j(s_{j+1}) - \mathbf{w}_j\|^2_{\mathbf{A}_j(s_{j+1}-1)}}{2} + \\
&\sum_{t=s_j}^{s_{j+1}-2} \frac{\|\mathbf{w}_j(t+1) - \mathbf{w}_j\|^2_{\mathbf{A}_j(t+1)} - \|\mathbf{w}_j(j+1) - \mathbf{w}_j\|^2_{\mathbf{A}_j(t)}}{2} + \sum_{t=s_j}^{s_{j+1}-1} \frac{\nabla_j^\top(t)\mathbf{A}_j^{-1}(t)\nabla_j(t)}{2} - \\
&\sum_{t=s_j}^{s_{j+1}-2} \sigma \|\mathbf{w}_j(t+1) - \mathbf{w}_j\|^2_{\nabla_j(j+1)\nabla_j^\top(j+1)} - \sigma \|\mathbf{w}_j(s_j) - \mathbf{w}_j\|^2_{\nabla_j(s_j)\nabla_j^\top(s_j)} \\
= &\sum_{t=s_j}^{s_{t+1}-1} \frac{\nabla_j^\top(t)\mathbf{A}_j^{-1}(t)\nabla_j(t)}{2} + \frac{\|\mathbf{w}_j(s_j) - \mathbf{w}_j\|^2_{\mathbf{A}_j(s_j-1)}}{2} - \frac{\|\mathbf{w}_j(s_{j+1}) - \mathbf{w}_j\|^2_{\mathbf{A}_j(s_{j+1}-1)}}{2},
\end{aligned}$$

where we use the following two facts

$$\begin{aligned}
\mathbf{A}_j(t+1) =& \mathbf{A}_j(t) + 2\sigma \nabla_j(t+1)\nabla_j^\top(t+1), \\
\mathbf{A}_j(s_j - 1) =& \mathbf{A}_j(s_j) - 2\sigma \nabla_j(s_j)\nabla_j^\top(s_j).
\end{aligned}$$

Summing over $j = 1, 2, \ldots, J$, we obtain

$$\mathcal{T}_1 \leq \underbrace{\sum_{j=1}^{J} \sum_{t=s_j}^{s_{j+1}-1} \frac{\nabla_j^\top(t)\mathbf{A}_j^{-1}(t)\nabla_j(t)}{2}}_{\mathcal{T}_{1,1}} + \underbrace{\sum_{j=1}^{J} \frac{\|\mathbf{w}_j(s_j) - \mathbf{w}_j\|^2_{\mathbf{A}_j(s_j-1)}}{2} - \frac{\|\mathbf{w}_j(s_{j+1}) - \mathbf{w}_j\|^2_{\mathbf{A}_j(s_{j+1}-1)}}{2}}_{\mathcal{T}_{1,2}}.$$

The key of our analysis is to prove tighter upper bounds on $\mathcal{T}_{1,2}$ and $\mathcal{T}_{1,1}$ using our initial configurations in Lemma 4.2 and Lemma 4.3. We first give some high-level explanations on why our analysis can give tighter regret bound.

The analysis of PROS-N-KONS (Calandriello et al., 2017a) initializes $\mathbf{w}_j(s_j) = \mathbf{0} \in \mathbb{R}^j$ and $\mathbf{A}_j(s_j - 1) = \alpha \mathbf{I} \in \mathbb{R}^{j \times j}$. A trivial upper bound on $\mathcal{T}_1$ can be derived, i.e.,

$$\mathcal{T}_1 \leq J \cdot \max_{j=1,\ldots,J} \sum_{t=s_j}^{s_{j+1}-1} \frac{\nabla_j^\top(t)\mathbf{A}_j^{-1}(t)\nabla_j(t)}{2} + \sum_{j=1}^{J} \frac{\|\mathbf{w}_j\|^2_{\mathbf{A}_j(s_j-1)}}{2}.$$

It is naturally that the regret bound is linear with J. The analysis can not be improved unless we reset the initial configurations $\mathbf{w}_j(s_j)$ and $\mathbf{A}_j(s_j - 1)$. Intuitively, there is a negative term $-\|\mathbf{w}_j(s_{j+1}) - \mathbf{w}_j\|^2_{\mathbf{A}_j(s_{j+1}-1)}$ in $\mathcal{T}_{1,2}$. Our analysis will use this negative term to cancel with the next positive term $\|\mathbf{w}_{j+1}(s_{j+1}) - \mathbf{w}_{j+1}\|^2_{\mathbf{A}_{j+1}(s_{j+1}-1)}$. To this end, we must carefully design $\mathbf{w}_{j+1}(s_{j+1})$. Finally, we will prove $\mathcal{T}_{1,2} = \frac{1}{2}\|f\|^2_{\mathcal{H}}$ which is independent of J. Similar idea is used to analyze $\mathcal{T}_{1,1}$.

We first analyze $\mathcal{T}_{1,2}$ and then analyze $\mathcal{T}_{1,1}$.

H.2.1. ANALYZING $\mathcal{T}_{1,2}$

Rearranging terms yields

$$\begin{aligned}\mathcal{T}_{1,2} =&\frac{\|\mathbf{w}_1(s_1)-\mathbf{w}_1\|^2_{\mathbf{A}_1(s_1-1)}}{2}-\frac{\|\mathbf{w}_J(s_{J+1})-\mathbf{w}_J\|^2_{\mathbf{A}_J(s_{J+1}-1)}}{2}+\\&\frac{1}{2}\sum_{j=1}^{J}\left[\|\mathbf{w}_{j+1}(s_{j+1})-\mathbf{w}_{j+1}\|^2_{\mathbf{A}_{j+1}(s_{j+1}-1)}-\|\mathbf{w}_j(s_{j+1})-\mathbf{w}_j\|^2_{\mathbf{A}_j(s_{j+1}-1)}\right].\end{aligned}$$

To upper bound the second term, the key is to analyze the relation between $\mathbf{A}_{j+1}(s_{j+1}-1)$ and $\mathbf{A}_j(s_{j+1}-1)$. For any $r\le j$ and $t\in T_r$, let $\bar{\mathbf{\Phi}}_{T_r}=(\bar{\phi}(\mathbf{x}_t))_{t\in T_r}$ where $\bar{\phi}(\mathbf{x}_t)=\sqrt{\eta_t}g_r(t)\phi(\mathbf{x}_t)=\sqrt{2\sigma}g_r(t)\phi(\mathbf{x}_t)$. According to (10), we have

$$\begin{aligned}\mathbf{A}_{j+1}(s_{j+1}-1)=&\mu\mathbf{I}+\sum_{r=1}^{j}\sum_{t\in T_r}\eta_t g_r^2(t)\mathcal{P}^{\frac{1}{2}}_{S(j+1)}\mathbf{\Phi}_{S(r)}\beta_r^*(t)(\mathbf{\Phi}_{S(r)}\beta_r^*(t))^\top(\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top\\=&\mu\mathbf{I}+\sum_{r=1}^{j}\sum_{t\in T_r}\eta_t g_r^2(t)\mathcal{P}^{\frac{1}{2}}_{S(j+1)}\mathcal{P}_{S(r)}\phi(\mathbf{x}_t)(\mathcal{P}_{S(r)}\phi(\mathbf{x}_t))^\top(\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top\\=&\mathcal{P}^{\frac{1}{2}}_{S(j+1)}\left[\mu\mathbf{I}+\left(\mathcal{P}_{S(r)}\bar{\mathbf{\Phi}}_{T_r}\right)_{r\in[j]}\left(\mathcal{P}_{S(r)}\bar{\mathbf{\Phi}}_{T_r}\right)^\top_{r\in[j]}\right](\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top,\\\mathbf{A}_j(s_{j+1}-1)=&\mu\mathbf{I}+\sum_{r=1}^{j}\sum_{t\in T_r}\eta_t g_r^2(t)\mathcal{P}^{\frac{1}{2}}_{S(j)}\mathbf{\Phi}_{S(r)}\beta_r^*(t)(\mathbf{\Phi}_{S(r)}\beta_r^*(t))^\top(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top\\=&\mathcal{P}^{\frac{1}{2}}_{S(j)}\left[\mu\mathbf{I}+\left(\mathcal{P}_{S(r)}\bar{\mathbf{\Phi}}_{T_r}\right)_{r\in[j]}\left(\mathcal{P}_{S(r)}\bar{\mathbf{\Phi}}_{T_r}\right)^\top_{r\in[j]}\right](\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top.\end{aligned}$$

For simplicity, let $\Delta=\mathbf{w}_{j+1}(s_{j+1})-\mathbf{w}_{j+1}$. According to Lemma H.1 and Lemma 4.3, we obtain

$$\begin{aligned}&\|\mathbf{w}_j(s_{j+1})-\mathbf{w}_j\|^2_{\mathbf{A}_j(s_{j+1}-1)}\\=&\|\mathcal{P}^{\frac{1}{2}}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top(\mathbf{w}_{j+1}(s_{j+1})-\mathbf{w}_{j+1})\|^2_{\mathbf{A}_j(s_{j+1}-1)}\\=&\Delta^\top\mathcal{P}^{\frac{1}{2}}_{S(j+1)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top\mathbf{A}_j(s_{j+1}-1)\mathcal{P}^{\frac{1}{2}}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top\Delta\\=&\Delta^\top\mathcal{P}^{\frac{1}{2}}_{S(j+1)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top\mathcal{P}^{\frac{1}{2}}_{S(j)}\left[\mu\mathbf{I}+\left(\mathcal{P}_{S(r)}\bar{\mathbf{\Phi}}_{T_r}\right)_{r\in[j]}\left(\mathcal{P}_{S(r)}\bar{\mathbf{\Phi}}_{T_r}\right)^\top_{r\in[j]}\right](\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top\mathcal{P}^{\frac{1}{2}}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top\Delta\\=&\Delta^\top\mathcal{P}^{\frac{1}{2}}_{S(j+1)}\mathcal{P}_{S(j)}\left[\mu\mathbf{I}+\left(\mathcal{P}_{S(r)}\bar{\mathbf{\Phi}}_{T_r}\right)_{r\in[j]}\left(\mathcal{P}_{S(r)}\bar{\mathbf{\Phi}}_{T_r}\right)^\top_{r\in[j]}\right]\mathcal{P}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top\Delta\\=&\Delta^\top\mathcal{P}^{\frac{1}{2}}_{S(j+1)}\left[\mu\mathbf{I}+\left(\mathcal{P}_{S(r)}\bar{\mathbf{\Phi}}_{T_r}\right)_{r\in[j]}\left(\mathcal{P}_{S(r)}\bar{\mathbf{\Phi}}_{T_r}\right)^\top_{r\in[j]}+\mu\mathcal{P}_{S(j)}-\mu\mathbf{I}\right](\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top\Delta\\=&\|\mathbf{w}_{j+1}(s_{j+1})-\mathbf{w}_{j+1}\|^2_{\mathbf{A}_{j+1}(s_{j+1}-1)}+\mu\Delta^\top\mathcal{P}^{\frac{1}{2}}_{S(j+1)}\left(\mathcal{P}_{S(j)}-\mathbf{I}\right)(\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top\Delta\\=&\|\mathbf{w}_{j+1}(s_{j+1})-\mathbf{w}_{j+1}\|^2_{\mathbf{A}_{j+1}(s_{j+1}-1)}-\mu\Delta^\top\mathcal{P}^{\frac{1}{2}}_{S(j+1)}\left(\mathbf{I}-\mathcal{P}_{S(j)}\right)^\top\left(\mathbf{I}-\mathcal{P}_{S(j)}\right)(\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top\Delta\\=&\|\mathbf{w}_{j+1}(s_{j+1})-\mathbf{w}_{j+1}\|^2_{\mathbf{A}_{j+1}(s_{j+1}-1)}-\mu\left\|\left(\mathbf{I}-\mathcal{P}_{S(j)}\right)(\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top\Delta\right\|^2\\=&\|\mathbf{w}_{j+1}(s_{j+1})-\mathbf{w}_{j+1}\|^2_{\mathbf{A}_{j+1}(s_{j+1}-1)}-\mu\left\|\left(\mathbf{I}-\mathcal{P}_{S(j)}\right)(\mathcal{P}^{\frac{1}{2}}_{S(j+1)})^\top(\mathcal{P}^{\frac{1}{2}}_{S(j+1)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top\mathbf{w}_j(s_{j+1})-\mathcal{P}^{\frac{1}{2}}_{S(j+1)}f)\right\|^2\\=&\|\mathbf{w}_{j+1}(s_{j+1})-\mathbf{w}_{j+1}\|^2_{\mathbf{A}_{j+1}(s_{j+1}-1)}-\mu\left\|\left(\mathbf{I}-\mathcal{P}_{S(j)}\right)(\mathcal{P}_{S(j+1)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top\mathbf{w}_j(s_{j+1})-\mathcal{P}_{S(j+1)}f)\right\|^2\\=&\|\mathbf{w}_{j+1}(s_{j+1})-\mathbf{w}_{j+1}\|^2_{\mathbf{A}_{j+1}(s_{j+1}-1)}-\mu\left\|\left(\mathcal{P}_{S(j)}-\mathbf{I}\right)\mathcal{P}_{S(j+1)}f\right\|^2,\end{aligned}$$

where the last but one equality satisfies

$$\left(\mathbf{I}-\mathcal{P}_{S(j)}\right)\mathcal{P}_{S(j+1)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top=\mathcal{P}_{S(j+1)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top-\mathcal{P}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top=(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top-\mathcal{P}_{S(j)}(\mathcal{P}^{\frac{1}{2}}_{S(j)})^\top=0.$$

Thus we can obtain

$$\begin{aligned}\mathcal{T}_{1,2} \leq & \frac{1}{2}\left(\|\mathbf{w}_1(s_1)-\mathbf{w}_1\|^2_{\mathbf{A}_1(s_1-1)}+\sum_{j=1}^{J}\mu\left\|\left(\mathcal{P}_{S(j)}-\mathbf{I}\right)\mathcal{P}_{S(j+1)}f\right\|^2\right)\\ =&\frac{1}{2}\left(\|\mathbf{w}_1\|^2_{\mu\mathbf{I}}+\sum_{j=1}^{J}\mu f^\top(\mathcal{P}_{S(j+1)}-\mathcal{P}_{S(j)})f\right)\\ \leq&\frac{1}{2}\left(\|\mathbf{w}_1\|^2_{\mu\mathbf{I}}+\mu f^\top(\mathcal{P}_{S(J+1)}-\mathcal{P}_{S(1)})f\right)\\ \leq&\frac{\mu}{2}\left(\|\mathbf{w}_1\|_2^2+\|f\|^2_{\mathcal{H}}-f^\top\mathcal{P}_{S(1)}f\right)\\ \leq&\frac{\mu}{2}\|f\|^2_{\mathcal{H}},\end{aligned}$$

where $\mathbf{w}_1(s_1)=\mathbf{0}$, $\mathbf{A}_1(s_1-1)=\mu\mathbf{I}$ and $\mathbf{w}_1=\mathcal{P}_{S(1)}f$.

H.2.2. ANALYZING $\mathcal{T}_{1,1}$

Recalling that

$$\mathcal{T}_{1,1}=\sum_{j=1}^{J}\sum_{t=s_j}^{s_{j+1}-1}\frac{\nabla_j^\top(t)\mathbf{A}_j^{-1}(t)\nabla_j(t)}{2},$$

where $\nabla_j(t)=\ell'(\hat{y}_t,y_t)\phi_j(\mathbf{x}_t), t\in T_j$. According to (9), we have

$$\forall t\in T_j,\quad \phi_j(\mathbf{x}_t)=\mathcal{P}^{\frac{1}{2}}_{S(j)}\phi(\mathbf{x}_t)=\mathcal{P}^{\frac{1}{2}}_{S(j)}\mathcal{P}_{S(j)}\phi(\mathbf{x}_t)=\mathcal{P}^{\frac{1}{2}}_{S(j)}\mathbf{\Phi}_{S(j)}\boldsymbol{\beta}_j^*(t)=\tilde{\phi}_j(\mathbf{x}_t).$$

For any $r\leq j, t\in T_r$, denote by $\tilde{\nabla}_j(t)=g_r(t)\tilde{\phi}_j(\mathbf{x}_t)$. We can rewrite $\mathbf{A}_j(t)$ as follows

$$\mathbf{A}_j(t)=\mathbf{A}_j(s_j-1)+\sum_{\tau=s_j}^{t}\eta_\tau g_j^2(\tau)\phi_j(\mathbf{x}_\tau)\phi_j^\top(\mathbf{x}_\tau)=\mu\mathbf{I}+2\sigma\sum_{r=1}^{j-1}\sum_{\tau\in T_r}\tilde{\nabla}_j(\tau)\tilde{\nabla}_j^\top(\tau)+2\sigma\sum_{\tau=s_j}^{t}\tilde{\nabla}_j(\tau)\tilde{\nabla}_j^\top(\tau).$$

Using Lemma H.2, we obtain

$$\sum_{t=s_j}^{s_{j+1}-1}\nabla_j^\top(t)\mathbf{A}_j^{-1}(t)\nabla_j(t)=\sum_{t=s_j}^{s_{j+1}-1}\tilde{\nabla}_j^\top(t)\mathbf{A}_j^{-1}(t)\tilde{\nabla}_j(t)\leq\frac{1}{2\sigma}\sum_{t=s_j}^{s_{j+1}-1}\ln\frac{\mathrm{Det}(\mathbf{A}_j(t))}{\mathrm{Det}(\mathbf{A}_j(t-1))}=\frac{\ln\frac{\mathrm{Det}(\mathbf{A}_j(s_{j+1}-1))}{\mathrm{Det}(\mathbf{A}_j(s_j-1))}}{2\sigma}.$$

Summing over $j=1\ldots,J$ yields

$$\begin{aligned}\sum_{j=1}^{J}\sum_{t=s_j}^{s_{j+1}-1}\nabla_j^\top(t)\mathbf{A}_j^{-1}(t)\nabla_j(t)=&\frac{1}{2\sigma}\sum_{j=1}^{J}\ln\frac{\mathrm{Det}(\mathbf{A}_j(s_{j+1}-1))}{\mathrm{Det}(\mathbf{A}_j(s_j-1))}\\ =&\frac{1}{2\sigma}\ln\prod_{j=1}^{J}\frac{\mathrm{Det}(\mathbf{A}_j(s_{j+1}-1))}{\mathrm{Det}(\mathbf{A}_j(s_j-1))}\\ =&\frac{1}{2\sigma}\ln\frac{1}{\mathrm{Det}(\mathbf{A}_1(s_1-1))}\cdot\prod_{j=2}^{J}\frac{\mathrm{Det}(\mathbf{A}_{j-1}(s_j-1))}{\mathrm{Det}(\mathbf{A}_j(s_j-1))}\cdot\mathrm{Det}(\mathbf{A}_J(s_{J+1}-1))\\ \overset{(*)}{=}&\frac{1}{2\sigma}\ln\frac{\mathrm{Det}(\mathbf{A}_J(s_{J+1}-1))}{\mu^{J-1}\mathrm{Det}(\mathbf{A}_1(s_1-1))}\\ \overset{(**)}{=}&\frac{1}{2\sigma}\ln\mathrm{Det}\left(\frac{1}{\mu}\sum_{j=1}^{J}\sum_{t\in T_j}2\sigma\tilde{\nabla}_J(t)\tilde{\nabla}_J^\top(t)+\mathbf{I}\right)\end{aligned}\tag{16}$$

where $(*)$ comes from Lemma H.3, and $(**)$ comes from $\mathbf{A}_1(s_1 - 1) = \mu$.
For simplicity, let

$$\tilde{\mathbf{\Phi}} = \sqrt{2\sigma}\left[(\tilde{\nabla}_J(t))_{t\in T_j}\right]_{j\in[J]} \in \mathbb{R}^{J\times T}, \quad \bar{\mathbf{\Phi}} = \sqrt{2\sigma}\left[(g_j(t)\phi(\mathbf{x}_t))_{t\in T_j}\right]_{j\in[J]} \in \mathbb{R}^{n\times T}.$$

Using the second statement of Lemma G.1, we can obtain

$$\left\|\tilde{\mathbf{\Phi}}^\top\tilde{\mathbf{\Phi}} - \bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}}\right\|_2 \le \left\|\tilde{\mathbf{\Phi}}^\top\tilde{\mathbf{\Phi}} - \bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}}\right\|_F \le \sqrt{2\sigma}\cdot\sqrt{T^2\cdot\max_{i,j\in[J]}|g_i(t)|\cdot|g_j(t)|\alpha} \le T\sqrt{2\alpha}.$$

Thus $\tilde{\mathbf{\Phi}}^\top\tilde{\mathbf{\Phi}} \preceq \bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}} + T\sqrt{2\alpha}\mathbf{I}$. We further obtain

$$\ln\mathrm{Det}\left(\frac{2\sigma}{\mu}\sum_{r=1}^{J}\sum_{t\in T_r}\tilde{\nabla}_J(t)\tilde{\nabla}_J^\top(t) + \mathbf{I}\right) = \ln\mathrm{Det}\left(\frac{\tilde{\mathbf{\Phi}}^\top\tilde{\mathbf{\Phi}}}{\mu} + \mathbf{I}\right) \le \ln\mathrm{Det}\left(\frac{\bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}}}{\mu} + \left(\frac{T\sqrt{2\alpha}}{\mu} + 1\right)\mathbf{I}\right).$$

Let $\bar{\lambda}_1 \ge \bar{\lambda}_2 \ge \ldots \ge \bar{\lambda}_T$ be the eigenvalues of $\bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}}$. Then we have

$$\begin{aligned}
\ln\mathrm{Det}\left(\frac{\bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}}}{\mu} + \left(\frac{T\sqrt{2\alpha}}{\mu} + 1\right)\mathbf{I}\right) =& \ln\left(\prod_{i=1}^{T}\left(\frac{\bar{\lambda}_i}{\mu} + \frac{T\sqrt{2\alpha}}{\mu} + 1\right)\right) \\
\le& \ln\left((1 + \frac{T\sqrt{2\alpha}}{\mu})^T\prod_{i=1}^{T}\left(\frac{\bar{\lambda}_i}{\mu} + 1\right)\right) \\
=& T\ln\left(1 + \frac{T\sqrt{2\alpha}}{\mu}\right) + \sum_{i=1}^{T}\ln\left(\frac{\bar{\lambda}_i}{\mu} + 1\right).
\end{aligned}$$

Let $\mathbf{D} = (\{g_j(t)\sqrt{2\sigma}\}_{t\in\cup_{j=1}^{J}T_j})$. Then $\bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}} = \mathbf{D}\mathbf{K}_T\mathbf{D}$. Since $\ln(1+x) < \frac{x}{1+x}(1+\ln(1+x))$ for all $x > 0$, we have

$$\begin{aligned}
\ln\mathrm{Det}\left(\frac{\bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}}}{\mu} + \left(\frac{T\sqrt{2\alpha}}{\mu} + 1\right)\mathbf{I}\right) \le& T\ln\left(1 + \frac{T\sqrt{2\alpha}}{\mu}\right) + \sum_{i=1}^{T}\frac{\bar{\lambda}_i}{\mu + \bar{\lambda}_i}\left(1 + \max_i\ln\frac{\bar{\lambda}_i + \mu}{\mu}\right) \\
\le& T\ln\left(1 + \frac{T\sqrt{2\alpha}}{\mu}\right) + \mathrm{tr}(\bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}}(\bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}} + \mu\mathbf{I})^{-1})\cdot\left(1 + \ln\frac{\mathrm{tr}(\bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}}) + \mu}{\mu}\right) \\
\overset{(*)}{\le}& T\ln\left(1 + \frac{T\sqrt{2\alpha}}{\mu}\right) + \mathrm{tr}\left(\mathbf{K}_T(\mathbf{K}_T + \frac{\mu}{2}\mathbf{I})^{-1}\right)\cdot\left(1 + \ln\frac{2T + \mu}{\mu}\right).
\end{aligned}$$

$(*)$ follows the proof of Theorem 1 in (Calandriello et al., 2017b) which states

$$\mathrm{tr}\left(\bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}}(\bar{\mathbf{\Phi}}^\top\bar{\mathbf{\Phi}} + \mu\mathbf{I})^{-1}\right) = \mathrm{tr}\left(\mathbf{K}_T(\mathbf{K}_T + \mu\mathbf{D}^{-2})^{-1}\right) \le \mathrm{tr}\left(\mathbf{K}_T(\mathbf{K}_T + \mu\lambda_{\min}(\mathbf{D}^{-2})\mathbf{I})^{-1}\right) = \mathrm{d}_{\mathrm{eff}}\left(\frac{\mu}{2}\right),$$

where $\lambda_{\min}(\mathbf{D}^{-2}) = \frac{1}{2\sigma\max(g_r(t))^2} = \frac{4(U^2+Y^2)}{\max(g_r(t))^2} = \frac{1}{2}$. We obtain

$$\mathcal{T}_{1,1} \le \frac{1}{2}T\ln\left(1 + \frac{T\sqrt{2\alpha}}{\mu}\right) + \frac{1}{2}\mathrm{d}_{\mathrm{eff}}\left(\frac{\mu}{2}\right)\cdot\left(1 + \ln\frac{2T+\mu}{\mu}\right) \le \frac{T^2\sqrt{\alpha}}{\sqrt{2}\mu} + \frac{1}{2}\mathrm{d}_{\mathrm{eff}}\left(\frac{\mu}{2}\right)\cdot\left(1 + \ln\frac{2T+\mu}{\mu}\right),$$

where we use the fact $\ln(1+x) \le x$ for all $x \ge 0$.

H.3. Analyze $\mathcal{T}_2$

Let $\mathbf{Y}_{T_j} = (y_{s_j}, y_{s_j+1}, \ldots, y_{s_{j+1}-1})^\top$. Recalling that $f_j = \mathcal{P}_{S(j)} f$. We have

$$\begin{aligned}
\mathcal{T}_2 =& \sum_{j=1}^{J} \sum_{t \in T_j} \left((\mathcal{P}_{S(j)} f)^\top \phi(\mathbf{x}_t) - y_t \right)^2 - \sum_{t=1}^{T} \left(f^\top \phi(\mathbf{x}_t) - y_t \right)^2 \\
=& \sum_{j=1}^{J} \| f^\top \mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} - \mathbf{Y}_{T_j} \|_2^2 - \sum_{j=1}^{J} \| f^\top \mathbf{\Phi}_{T_j} - \mathbf{Y}_{T_j} \|_2^2 \\
=& \sum_{j=1}^{J} \| f^\top \mathbf{\Phi}_{T_j} - \mathbf{Y}_{T_j} + f^\top (\mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} - \mathbf{\Phi}_{T_j}) \|_2^2 - \sum_{j=1}^{J} \| f^\top \mathbf{\Phi}_{T_j} - \mathbf{Y}_{T_j} \|_2^2 \\
=& \sum_{j=1}^{J} f^\top (\mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} - \mathbf{\Phi}_{T_j})(\mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} - \mathbf{\Phi}_{T_j})^\top f + 2 \sum_{j=1}^{J} \langle f^\top \mathbf{\Phi}_{T_j} - \mathbf{Y}_{T_j}, f^\top (\mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} - \mathbf{\Phi}_{T_j}) \rangle \\
=& \sum_{j=1}^{J} \| f \|_{\mathcal{H}}^2 \cdot \| \mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} - \mathbf{\Phi}_{T_j} \|_2^2 + 2 \sum_{j=1}^{J} \| f^\top \mathbf{\Phi}_{T_j} - \mathbf{Y}_{T_j} \|_2 \cdot \| f \|_{\mathcal{H}} \cdot \| \mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} - \mathbf{\Phi}_{T_j} \|_2.
\end{aligned}$$

Using the first statement of Lemma 4.5, we obtain

$$\begin{aligned}
\| \mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} - \mathbf{\Phi}_{T_j} \|_2^2 =& \| (\mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} - \mathbf{\Phi}_{T_j})^\top (\mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} - \mathbf{\Phi}_{T_j}) \|_2 \\
=& \| \mathbf{\Phi}_{T_j}^\top \mathbf{\Phi}_{T_j} - \mathbf{\Phi}_{T_j}^\top \mathcal{P}_{S(j)} \mathbf{\Phi}_{T_j} \|_2 \\
\leq& |T_j| \alpha.
\end{aligned}$$

Thus we have

$$\mathcal{T}_2 \leq \| f \|_{\mathcal{H}}^2 \cdot T\alpha + 2 \sqrt{\sum_{j=1}^{J} \| f^\top \mathbf{\Phi}_{T_j} - \mathbf{Y}_{T_j} \|_2^2} \cdot \| f \|_{\mathcal{H}} \cdot \sqrt{\sum_{j=1}^{J} |T_j| \alpha} \leq \| f \|_{\mathcal{H}}^2 \cdot T\alpha + \sqrt{8(U^2 + Y^2)} \| f \|_{\mathcal{H}} \cdot T\sqrt{\alpha}.$$

Combining the upper bounds on $\mathcal{T}_{1,1}$, $\mathcal{T}_{1,2}$ and $\mathcal{T}_2$ concludes the proof. □