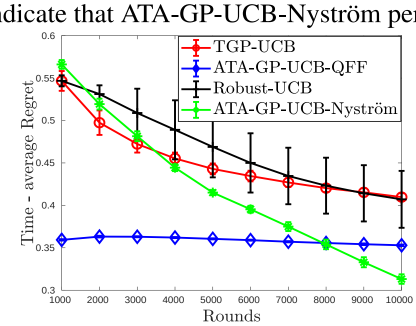
We thank all the reviewers for their careful reviews and insightful feedbacks.

Reviewer 1: **1.** Analysis of TGP-UCB depends on a Hoeffding type concentration bound for self-normalized processes and analysis of ATA-GP-UCB depends on a Bernstein type concentration bound in each direction. Thus, if we can derive a Bernstein type bound for self-normalized processes which depends on the $(1+\alpha)$-th moments of rewards, then we may not have to switch to the feature space to get optimal regret. **2.** Instead of truncating raw observations to build a robust estimator of f, one can use a median of means type estimator to obtain a slightly better bound, though not optimal, for TGP-UCB. **3.** ATA-GP-UCB requires knowledge of the horizon T, but one can use a standard doubling trick to get around this. **4.** It is indeed an interesting direction to explore when α and v are misspecified. In practice, as suggested by our real data experiments, one can estimate α and v from data and use those in our algorithms.

Reviewer 2: **1. Computational complexity for continuum arm sets:** One of our main aims is to quantify the achievable statistical efficiency of nonparametric algorithms for optimization under heavy-tailed noise from a theoretical standpoint. This is the reason why we do not delve in detail into the specifics of how the function $g_t(x) := \tilde{\mu}_{t-1}(x) + \beta_t \tilde{\sigma}_{t-1}(x)$ over all $x \in \mathcal{X} \subset \mathbb{R}^d$ is optimized. This practical issue also arises in other well-known bandit optimization settings such as finite dimensional linear bandits – the celebrated LinUCB or OFUL algorithms do not address how to solve the UCB optimization problem over a continuum set. In this regard, we mention that it is well known in the BO literature that one can approximately maximize g_t by grid search / Branch and Bound methods such as DIRECT (Brochu et al. (2010)). In fact g_t can be maximized within $O(\epsilon)$ accuracy by making $O(\epsilon^{-d})$ calls to it, yielding a per-step time complexity of $O(m_t^2(t+\epsilon^{-d}))$. Another viewpoint for considering a continuum arm set is that it serves to model the case of a large, finite set of arms which share regularity structure with respect to their rewards, enforced through a kernel function, and makes the analysis of the algorithm cleaner due to results in Gaussian process theory. **2. Versus the "bandits with heavy tail" paper:** Compared to this paper's setting, "bandits with heavy tail" indeed makes weaker assumptions (i.e., no regularity structure on arms' rewards) and shows a more general but weaker regret bound (especially if the number of arms is very large) which is not surprising – more structure allows for lower regret. Our results show how smoothness in the arms' rewards (which is common in practice) can be exploited to achieve better regret. Numerical comparisons of the Robust-UCB algorithm (with truncated mean estimator) of "bandits with heavy tail" paper with our algorithms on real datasets (figure is given for lightsensor data) indicate that ATA-GP-UCB-Nyström performs much better than Robust-UCB, suggesting that it is indeed able



to capture the smoothness structure present in the data. Theoretically if there are only K arms, the cumulative regret of ATA-GP-UCB will be better than that of Robust-UCB as long as $\gamma_T \leq K^{\frac{\alpha}{1+\alpha}}$. This holds if $K^{\frac{\alpha}{1+\alpha}} \geq (\ln T)^d$ for SE kernel and if $K^{\frac{\alpha}{1+\alpha}} \geq T^{\frac{1}{1+\nu}}$ for Matérn kernel (on $\mathbb{R}$). This is typically true if K is large and in fact, for a continuous set of arms the analysis of Robust-UCB yields a trivial regret upper bound of infinity. This introduces additional challenges that require a different set of ideas and is quite representative of real world problems, e.g., hyperparameter tuning in ML. **3. More details on time and space complexity:** For SE kernel γ_t is poly-logarithmic in t (i.e., $\gamma_t << t$), and since $m_t = \tilde{O}(\gamma_t)$, per step time complexity, in practice, for continuous $\mathcal{X}$ is $\tilde{O}(t+\epsilon^{-d})$. For Matérn kernel, the complexity is $\tilde{O}(t^p(t+\epsilon^{-d})), 1<p<2$. Now for finite $\mathcal{X}$, we only need to store the number of times each arm has been played so far and thus per step space complexity does not grow linearly with t. For continuous $\mathcal{X}$, we indeed need to store all previously chosen arms, but this linear dependence on t is subsumed by the larger $|\mathcal{X}|$ term. Thus in practice, the space needed is $O(t + m_t(m_t+\epsilon^{-d})) = O(m_t(m_t+\epsilon^{-d}))$ for small enough ϵ. So for continuous $\mathcal{X}$ and for practical purposes, time and space complexities can be obtained by replacing $|\mathcal{X}|$ with ϵ^{-d} in those given in the paper. **6, 8, 9:** These are typos. **7:** $\mathcal{X}_t$ is the set $\{x_1, \ldots, x_t\}$. **10.** Setting $\delta = 1/T$, we can achieve expected cumulative regret of same order (upto some constant factor). **11:** This holds, for example, Matérn kernel on $\mathbb{R}^2$ with $\nu = 3.5$ when variance of the rewards is finite. **13:** True, our algorithm is not optimal for Matérn kernel as mentioned in Remark 7. **14:** We meant to say that existing BO algorithms like GP-UCB fails under heavy-tailed noise (figure 1(f)).

Reference: Brochu, Eric, Cora, Vlad M., and de Freitas, Nando. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. *CoRR*, 2010.

Reviewer 3: **1.** Setting $y_t = 0$ if $|y_t| > b_t$ and blowing up the confidence width of GP-UCB by b_t, together help us to construct a robust estimate of f and a "good" confidence set around this estimate which contains f (figure 1(f)). Instead of truncating, one can also use a median of means type estimator in the feature space and obtain optimal regret. **2.** The result will hold as long as $\bar{m} > 1/l^2$ and $\bar{m} = \Theta(2\log_{4/e}(T^3))$. **3.** We compared for different choices of $\bar{m}$ and as long as $\bar{m}$ satisfies the above constraints, we found that increasing $\bar{m}$ improves the performance to some extent. **4.** We discretized the set $[0,1]$ into 100 evenly spaced points and generated a random sample uniformly from those 100 points.