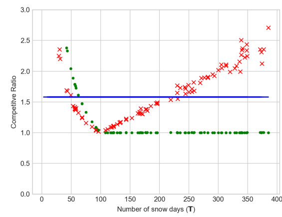
We thank the reviewers for their feedback. Additional feedback and suggestions for improvement from Reviewer #3 is greatly appreciated. We address the issues and concerns raised by the reviewers below.

The data-sets used in experiments [R1& R3]. The treatment of the ski-rental problem with ML-advisor in [1] is germane to our present work. In Fig. 1 we use the same simulation setup as that of [1] (cf. Pg. 8 §4.1 of [1]) to provide a transparent comparison of algorithmic performance. We strongly agree that using non-normal noise is important, but such a discussion within the limited space will significantly deviate from the main idea of the present paper.

The depiction of the range of $\mathbf{T}$ in Fig. 2 was driven by space constraints. Simulating Algorithms 2 and 3 on the range $[1, 4\mathrm{b}]$ instead of $[1, 2.5\mathrm{b}]$ did not provide radically new insights. The graph on the right presents Algorithm 2 (in red) and Algorithm 3 (in green) with $\mathbf{T} \in [1, 4\mathbf{b}]$.



Comparison with classic approaches [R1]. Some discussion on the comparison between classic approaches and the recent ones is presented in [1]. The performance of the classic randomized $e(e-1)^{-1}$-algorithm in the experiments leading to Fig. 2 will be included in the updated version.

Takeaways from Fig. 2 [R3]. (1) In algorithmic implementation, the hyperparameter ρ plays a significant role. We fixed $\rho = 20$ in experiments. This lead to underflow issues when $\mathbf{T} \gg \mathrm{b}$, as demonstrated by the increasing trend in the red ticks in Fig. 2. This suggests, one should vary ρ within an appropriate range. (2) Algorithm 3 avoids this issue at the expense of performance when $\mathbf{T} \ll \mathrm{b}$ (higher values of green ticks in that range). (3) The relatively high competitive ratio of the hedge algorithm, when $\mathbf{T} \ll \mathrm{b}$, corresponds to the to the fixed overhead constant term in Theorem 2.1.

The efficacy of the accuracy measure introduced in this paper [R4]. In statistical prediction, a biased estimator with low variance can be more effective than a consistent estimator with unknown variance. Hypothetically, consider two predictors A and B for ski-rental problem. Suppose, for simplicity, A has a MSE of 3 days and B has a MSE of 5 days. The consistency/robustness framework does not justify why A is a better choice than B. Our accuracy measure provides a justification. A quantitative discussion is presented in lines 44 – 51 of the paper. Moreover, it is unclear how to extend robustness/consistency to the TCP problem where the decision space is $\Delta = [0, 1]^n$. Such an extension will be sensitive to the metric used on Δ (implicit in the definition of η). The (ϵ, α)-accuracy measure extends in this setting with relative ease and conceptual clarity.

The accuracy term doesn't improve the competitve ratio in Theorem 2.2 [R4]. Ensuring that the competitive ratio doesn't inflate significantly due to bad predictions is a central design feature. This fails if the competitive ratio has a direct functional dependence on accuracy. However, Thm 2.2 demonstrates how to hedge against bad predictions while leveraging accurate predictions (alpha = 1) – thus retaining the essence of online algorithms.

The robustness and consistency guarantee of Algorithm 1 coming from two different set of parameter settings. [R4] The guarantees come from two different functions (not parameters) which depend on *a single* parameter λ and the data. The takeaway from Proposition 2.1 and Fig. 1 of the present paper is that using a hyperparameter and the *information* conveyed by the predictor ($\hat{\mathbf{T}}$) significantly outperforms an algorithm just using the hyperparameter alone (cf. Theorem 2.2 and Fig. 2(a) of [1]). The same numerical value of the hyperparameter never optimizes robustness and consistency simultaneously (cf. the $(1 + \lambda^{-1})$-robust and $(1 + \lambda)$-consistent Algorithm 2 of [1]). This is unsurprising because an optimal robust algorithm must also hedge against bad predictions while an optimal consistent algorithm assumes perfect prediction (cf. line 42 of the paper).

Comparing Algorithm 5 to an existing online algorithm [R4]. We are not aware of any prior work on Dynamic TCP Acknowledgment problem with ML prediction. As observed in line 197, Algorithm 5 can improve over the classic $e(e-1)^{-1}$-randomized algorithm only under strict accuracy guarantees from the ML-predictor.

The distinction between this work and the methodology and theoretical results in [1] [R4]. Previous work relevant to the current paper is discussed in lines 31 – 32. In this paper, we present a new accuracy measure and compare it with existing framework of [1] (cf. lines 40 – 56). Algorithm 1 presents a modification of Algorithm 2 of [1]. With the goal of providing a clear and transparent comparison we use the same simulation framework to compare the relative performances of these algorithms. The rest of the present paper is about other algorithms which use a completely different approach from that of [1]. Additionally, we discuss the Dynamic TCP acknowledgment problem within our framework. This problem has not been considered in [1].

References

[1] Purohit, M., Svitkina, Z., and Kumar, R. *Improving online algorithms via ML predictions*, Advances in Neural Information Processing Systems, 9661–9670, 2018.