Author Response: Recovering Bandits #7889

We would like to thank all the reviewers for their detailed and constructive comments. Several reviewers pointed out that the paper would benefit from more discussion of the applicability of our algorithms in recommendation systems. We propose to add a paragraph to the end discussing these practical challenges. We now address each reviewer in turn. All references correspond to the bibliography of the submitted paper.

Reviewer 1:

- The d-step lookahead regret it is the regret in batches of size d and so it is valid for any sort of policy. We can measure the d-step lookahead regret for UCB-Z. However, it may be high since UCB-Z is a greedy strategy which does not look at future states so may play arms which lead to bad reward in the remaining steps in the lookahead.
- We agree that the multiple play regret is more practically relevant, and focused on this in the optimistic planning extension and experiments. However, we believe that first studying the single play regret allows us to gain more insights into the problem. Particularly, we show that in the single play case we can achieve the same rate of regret as the instantaneous case, whereas in the multiple play case we are penalized for not updating the posterior between repeated plays of the same arm. This allows us to conclude that it is the lack of updating that makes the multiple play lookahead case more difficult. We will clarify the benefit of studying the single play case in the final version.
- If we are minimizing regret to some fixed horizon, the optimal dynamic programming solution would indeed depend on this horizon. However, we are interested in anytime algorithms, which do not depend on the horizon. In this case, the oracle is stationary.
- A typical choice of kernel depends on $|z - z'|$ (e.g. Gaussian kernel). By scaling this distance we can interpolate between the case where the rewards are completely unrelated for different z values, and the case with essentially constant reward. Prior knowledge of the smoothness of the functions can be used to tune this scaling factor. Alternatively, it may be possible to adaptively tune it (see e.g. Chapter 5 of [24]). We will discuss this in the additional paragraph.
- For the choice of $z_{\max}$, we first point out that we only require an upper bound on it. We believe that in practice, this is reasonable. As an example for the scale, if we suggest one item per day, the reward from a user not having seen it for 30 days is likely to be the same as if they have not seen it for 60 days, so $z_{\max} = 30$, and 60 is an upper bound. Adaptively choosing $z_{\max}$ is an interesting area for future work. The choice of $z_{\max}$ will be discussed in the extra paragraph.
- In Figure 4, the parameters were sampled uniformly as a method for selecting the parameters of the 'true' recovery curves. We did not use this to influence our choice of kernels for our algorithms. This will be clarified.
- We will correct the issue with some of the references and figures not appearing in the correct place.

Reviewer 2:

- We believe there has been some confusion regarding the definition of γ_T. You are correct that in [28], bounds on γ_T of the form $O(\log(T)^{D+1})$ are given for the squared exponential kernel where D is the dimension of the input space. When we apply this result, our input space is the set of integers $\{0, z_{\max}\}$ which is one dimensional. Hence, $\gamma_T = O(\log(T)^2)$. There is no dependence on d, the depth of the lookahead. We apologize if our notation caused confusion, and will clarify this in the final version.
- The single play d-step lookahead regret of our algorithms are of the same order as the instantaneous regret. At first this may appear surprising, however, it can be explained by noting that in the single play case, we select a sequence of d arms at (approximately) T/d time steps, and since we can only play an arm at most once in a d-step lookahead, the posterior can be updated after each play meaning that we do not lose any information. This will be clarified.
- You are correct that our problem is related to reinforcement learning (as discussed in lines 144–145). However, even if posed as such, it is still challenging since there are no discount factors and the states are never reset.

Reviewer 3:

- Our model captures a wide range of recovery functions, ranging from uncorrelated $f_j(z)$'s to more complex functions. Under mild assumptions most of these prior kernels will be sufficient to represent the true function (if this is relatively smooth). However, we learn faster with good priors, so we suggest selecting the prior based on the smoothness of the functions, which may be known in advance. We will discuss this, and adaptive methods, in the additional paragraph.
- Our approach can be used in the setting with arm-dependent $z_{\max}$ as long as there is an upper bound that holds for all arms. Indeed, we can extend f_j from the arm-dependent value, $z_{\max,j}$, to $z_{\max}$ using the fact that $f_j(z) = f_j(z_{\max,j})$ for $z = z_{\max,j} + 1, \ldots, z_{\max}$ (e.g. in the experiment with logistic recovery curves, in effect $z_{\max,j}$ was arm-dependent so f_j was constant above that value). We will add a discussion of this in the extra paragraph. An interesting area for further work is to develop more principled methods for estimating the $z_{\max,j}$ of each arm within the algorithm.
- There are two ways to formulate skipping rounds. In the first, rounds can be skipped at no cost to the regret. However this may not be practically relevant. Alternatively, skipping a round can incur some regret. In this case, our algorithms can be applied with an additional 'pseudo-arm' with constant negative reward which is played when a round is skipped.
- To map recovering bandits to the setup of [18], we need to define state dynamics matrices. In the notation of [18], A sends z to $z + 1$, and B is $-\infty$, so that the projection onto $[0, z_{\max}]$ is 0. We implemented RogueUCB-Tuned with the parameter η set to the maximal KL-divergence, as in [18]. We will specify this in the final version.