Overall Comments to Program Chair and Reviewers: All the reviewers have praised the paper for the theoretical contributions, and have questions primarily about the empirics. We address these below in detail, but first want to emphasize that *our simulations were chosen not to compare our algorithm to others, but rather, to illustrate our theory* in 2 ways: (i) the ‘small’ examples (2-dimensional, small H) let us control the Q-function and exactly show how ADAMB ‘zooms’ in to regions of higher Q-values, and (ii) the examples show that even in small settings, ADAMB gets optimal regret with *much lower storage and time complexity* than naive discretization.

We do provide more simulations in the appendix (which answer some of the reviewer questions), and we have code implementing ADAMB on a larger problem with a 10-dimensional joint state and action space showing similar insights that we can include. While comparing with other heuristics is clearly important, doing this in a fair empirical way requires extensive engineering and detailed simulations, and also we feel may distract from our main contributions:

- We give the *best available* regret guarantees for model-based RL with continuous state-action spaces and Lipschitz $Q^\star$ functions, and the first with adaptive function approximation, improving on [21,11] with a much simpler algorithm.
- New technique for transition-kernel concentration which bypasses more complex covering arguments over function spaces (see [11, 41, 18]), and a new LP duality argument for adaptive discretization (simplifying [35,36]).
- We provide storage, time, and regret guarantees for our algorithm. Our regret bound is novel vs. prior work with discretization, and as far as we know, we get the *first sublinear dependence on storage and time complexity* as well.

Adaptive vs Fixed Discretization: (**R1, R2, R4**) Empirically, we do show settings where ADAMB performs better than fixed discretization (see Fig 7 in appendix); this gap can be made larger when $Q^\star$ is more peaked (as suggested by our theory), and we can include these in our camera-ready. We also choose the optimal fixed discretization in our experiments, based on knowing the $Q^\star$, while ADAMB has minimal tuning, and needs much less storage. (**R5**) Finally, our theory explains why adaptive and fixed discretization get the same minimax regret bounds (see Corollary D2).

Adaptive Discretization versus Function Approximation (**R3, R5**): Adaptive discretization is a *form of function approximation* (using step-functions as a kernel) – in particular, it is a natural basis for Lipschitz functions. While other function approximation techniques are clearly important in practice, our approach has the following benefits:

- Standard algorithms based on function approximators (linear/polynomial/RBF kernel) use a fixed number of parameters (degree, bandwidth), and have guarantees that depend on some parametric model structure (i.e., how well these functions approximate the Q-function). Our guarantees (also [11,21]) hold uniformly for the larger class of Lipschitz functions, which is suited to many more applications where special parametric model structures may not hold.
- In contrast to standard function approximation techniques (including [11,21]), *our algorithm adaptively chooses number of parameters based on data*. Also, by using a simpler basis (step functions) we get lower storage, runtime.
- Overparametrized models like deep-RL enjoy success in experiments, but require *much* larger storage and training; comparing to ADAMB under resource constraints (i.e., on-policy settings) would be unfair.

Model-Based vs. Model-Free Algorithms: (**R2, R4**) Comparing model-based and model-free algorithms is an important open problem in RL. Folklore says model-based methods outperform model-free ones in practice – however, these comparisons involve very different styles of algorithms. By giving a model-based equivalent for a state-of-the-art model-free method (using adaptive discretization in continuous metric spaces) we can make a direct comparison. Surprisingly, our bounds show ADAMB has *worse* theoretical storage, time, and regret compared to ADAQL from [35] (Table 1). Moreover, simulations show that even in practice, ADAMB *and* ADAQL *have similar performance*, which is a sharp contrast to ϵ-net algorithms (where model-based is much better as folklore suggests).

Our results suggest that Q-learning with adaptive discretization is good for resource-constrained RL in continuous settings, as they learn an efficient discretization of the space subject to the constraints. Moreover, in larger dimensional settings (ambulance problem with ≥ 5 ambulances), ADAMB does better than ADAQL, but also maintains a much larger partition and so needs higher storage. We can include these experiments in the camera ready.

Experimental Results (**R1, R2, R5**): We chose our instances as *proofs of concept* to illustrate the theory (how the algorithm intrinsically partitions the regions across near-optimal parts of the space, in comparison to a fixed discretization) – in particular, using a 2-dimensional problem lets us plot the Q value-estimates and true $Q^\star$ values. Making the algorithm scale to larger problems is an interesting future direction outside the scope of this paper.

(**R5**) The point that ADAMB’s space complexity is monotonically increasing is valid; however, to get sublinear minimax regret in a continuous setting for nonparametric Lipschitz models, the model complexity must grow over episodes. In practice, one would run our algorithm until we run out of space – our experiments show that ADAMB uses resources (storage and computation) much better than a uniform discretization (see Figures 3 and 4 in appendix). We are not aware of any storage-performance lower bounds, so that is also an interesting future direction.

(**R1**) We omitted confidence intervals as they were so small that they were not visible given the figure size. In the camera ready, we will include bigger plots with CIs, and more fine-grained comparisons of the various algorithms.