We thank the reviewers for thoughtful comments! We are encouraged you found our overall contribution of using neural ODEs for continuous-time RL well-motivated, novel, and well-explained. Below we respond to specific questions:

Reviewer 1. 1) Scalability/experiments with a larger state-space. We agree there are more interesting domains with larger state-spaces, and acknowledge that currently neural ODEs can be tough to scale. We emphasize that our main contribution is to introduce neural ODEs in a model-based RL framework for SMDPs: our experiments help us understand why neural ODEs work but other baselines fail, show benefits on standard RL benchmarks (e.g. Mujoco), and demonstrate opportunities for learning from irregular measurements and optimizing measurements. As neural ODE solvers improve in efficiency, we can expect these core contributions to carry over to larger domains.

2) Show a failure case of transferring to environments with different time schedules. Thank you for asking about limitations! We have an example in Table 6 in Supplement D.1: in some cases, (e.g. $\tau = 1$ and 2 on the HIV environment), there is a huge gap between all model-based policies and the model-free policy (oracle). This is because the model hardly sees transitions where $\tau = 1$ or 2 during training, causing poor performance when transferring to such mostly unseen time discretizations. As with any learning algorithm, one has to be careful of extrapolation. We detail additional limitations of optimizing time schedules (different from transferring to different time settings) in Supplement A.1.

3) Off-policy setting. Our training on three simpler domains (including the HIV task), learning world model schema, can be viewed as off-policy, in the sense that the model is trained on a single batch of irregularly sampled data from the environment and then agent is only trained with fictional data generated by the model and not the true environment. In the fully batch setting (learn from only one fixed set of observational data), our work can still apply but will be limited, as all batch methods are, in terms of how much it can extrapolate. However, since our training procedure for dynamics models can be used on any batch of data, we can utilize any start-of-the-art model-based off-policy RL algorithm (e.g., [1]) for planning. This is also an important point regarding the ethics and broader impact: we will emphasize this in the final paper.

Reviewer 2. 1) The necessity of using neural ODEs vs. regular ODEs. If we understand the question, the point is why apply RL using neural ODEs vs. ODEs in general. If we knew that a system was characterized by a particular ODE, then we could absolutely use RL to learn the parameters of that ODE. However, in many cases, e.g. healthcare contexts, we do not, and the neural ODE offers a very flexible way to define the system dynamics as an ODE-IVP problem/provide the inductive bias that the system can be modeled with an ODE without explicitly knowing its parameterization. (Apologies if we did not understand the question correctly.)

2) The role of latent variables z and neural ODEs. The latent variable z is from the *learned* dynamics model, *not* the environment, so it is always available to the agent. Using the learned dynamics models for planning (e.g., Dyna-style policy optimization or model predictive control), the agent can develop a good policy with less real data by relying mostly on simulating from the model; neural ODEs now let us do this for continuous-time environments.

Another note re z: if the state is fully observable, z is just a compression of s. If we only receive partial observations (i.e. not Markov), z will include the hidden dynamics which summarize the past (see lines 84-87 in the main paper).

Reviewer 3. 1) More environment steps for Swimmer. Great point! We extended Swimmer to 450k steps below. The policy developed with the Latent-ODE is still better than the one from Δt-RNN, and learns those policies faster. We could not finish extended runs for the other Mujoco in time for the rebuttal, but will do so for the final paper.

[1] Yu, Tianhe, et al. "MOPO: Model-based Offline Policy Optimization." arXiv preprint arXiv:2005.13239 (2020).

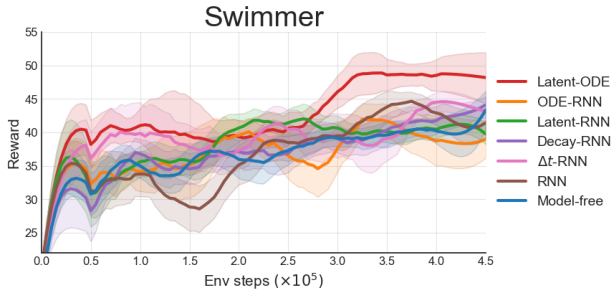

Figure 1: The learning curve for the swimmer task.