We thank all reviewers for their thoughtful feedback and suggestions. We are encouraged that all reviewers found the paper well-written and agreed that the technical ideas are important and novel. Based on suggestions from the reviewers, we have further improved the clarity of the paper and included additional details.

**(R1) - Interaction between HiPPO and cost function.** The overall HiPPO-RNN model (Sec. 4) uses the same loss function as the baselines (e.g. cross-entropy for classification, as in pMNIST). Note that the HiPPO-RNN cell in Fig. 2 depicts only the recurrent dynamics; as is convention, each cell also outputs the state $h_t$ for another layer (e.g. softmax) to compute the final prediction. When incorporated in an end-to-end model such as an RNN, the HiPPO module (blue circle in Fig. 2) can simply be viewed as a fixed, non-trainable component – although it has the additional interpretation of memorizing the history of features (in L2 space). Since it only affects feature representation, the HiPPO component does not impact the choice of output space and loss function, which is inherited from the base RNN. These details will be clarified in the next version of the paper, and are also explicit in the submitted supplementary code which will be released.

**(R1) - Reproducibility and experiment details.** Additional space permitted, we have moved much of the architecture details in Appendix F.1 to the main body, including (*) a better diagram of the HiPPO-RNN cell, (*) a discussion of the output layer and loss function, (*) a comparison of network architecture, size, and hyperparameters against all RNN baselines (in summary, they are controlled to be all roughly equal), (*) loss function details for all tasks in Appendix F.

**(R1) - Technical details.** We appreciate R1's close reading of the paper and suggestions for improving the technical presentation. We agree with all details on integrable functions, the Leibniz rule, the Copying task hyperparameter, and the interpretation of coefficients $c(t)$. Thanks for the suggestions and corrections!

**(R1, R2) - Finding the polynomial order $N$.** In the setting of approximating smooth functions, $N$ can be calculated from the explicit error bounds (Sec. 3, Prop. 6). In our end-to-end experiments, $N$ is a hyperparameter analogous to the hidden dimension of standard RNN/LSTM models; we set $N$ equal to the hidden dim. $d$ of the baselines, so that all HiPPO methods have the same number of hidden units and the same (or smaller) model size as the LSTM baseline (line 1094-1104).

**(R2) - HiPPO, RNNs, and universal approximators for dynamical systems.** Similar to universal approximation theorems, HiPPO formalizes the ability of RNNs to approximate functions. HiPPO shows that optimal compression of signals is governed by linear recurrent systems (Thm. 1, 2). Unlike universal approximation theorems, HiPPO shows non-asymptotic error rates on the approximation in terms of memory capacity (Prop. 6). The HiPPO framework is a bridge between RNNs and dynamical systems, and explains how to derive the form of a modern RNN update from first principles.

**(R2) - Directly fitting timeseries in the input space.** HiPPO can be used directly on input features, which is how the basic HiPPO framework is described. For example, the function approximation experiments (Sec. 4.3 and App. F.8) can be interpreted as fitting a long input timeseries with a budget of 256 features. We remark that the presented figures (Fig. 3, 9, 10) are a good visual indicator of the approximation dynamics of HiPPO.

**(R2, R4) - Short-term or more local time series tasks.** While HiPPO was designed to capture long-range dependencies, several of our experiments involve tasks requiring shorter memory: trajectory classification in Sec. 4.2 (length $\approx 100$), chaotic dynamics simulation in App. F.7 (length 15), and the IMDB dataset in App. F.6 (some inputs of length $< 100$).

**(R3) - Experiments.** R3's only concern is that "the paper needs more experiments to better support the proposed model... such as speech processing, NLP, etc." Although we acknowledge R3's suggestion, we note that the submission already builds up from the foundational theory to a comprehensive range of experiments including an NLP application. Indeed, in addition to directly validating the proposed model and online function approximation theory on synthetic experiments, we have included experiments on memory benchmarks, image classification, trajectory classification, timescale robustness, chaotic dynamics prediction, and NLP (the IMDB movie review benchmark); some of these are in Appendix F due to space constraints. We are excited about pursuing further applications in follow-up work.

**(R3) - Baselines.** Although R3 asks about more memory RNN baselines, the memory experiments in the paper include a variety of well-known memory models (including RNNs), chosen for their relevance to each task. Our permuted MNIST results explicitly compare against 15 sequence model baselines (Tab. 4, 5) to establish SoTA; this also establishes a large number of implicit comparisons, e.g., the LMU outperformed 8 other memory RNN models in (Voelker 2019). We ran an additional baseline suggested by R3, the MinimalRNN, on the memory benchmarks (Sec. 4.1). This model failed to solve the Copying task and achieved 89.1 accuracy on the permuted MNIST benchmark, compared to the SoTA 98.3% of our model.

**(R4) - Why LegS is faster than LMU.** The speed difference in Sec. 4.3 is explained by our faster algorithm for LegS. Because of the approximation made by LMU due to its measure (Sec. 2.3 lines 158-161), the LMU transition matrix $A$ (Thm. 1) is not triangular, in contrast to that of LegS (Thm. 2). Numerically stable discretizations (App. B.3) require inverting this matrix, which is efficient in the LegS case (Prop. 4, proved in App. E.2) but not known for the LMU case.

**(R4) - Performance depends on measure and task.** We fully agree with R4's observation that some tasks are more suited to LegS due to the uniform memorization prior. In fact, rather than seeing this as a weakness, we argue that a primary contribution of this paper is introducing the technical framework that exposes these very tradeoffs. This framework allowed us to (1) explain the memory mechanism of existing methods (LMU, LSTM, Fourier Recurrent Unit, etc.) in terms of their approximation measures, (2) introduce new methods that may be more appropriate in different settings such as very long memory and mis-specified timescales, and (3) theoretically analyze and contrast different approaches based on their underlying mechanisms. We thank R4 for bringing up this insightful point!