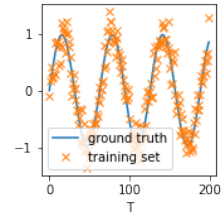
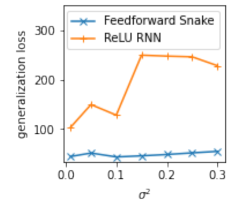
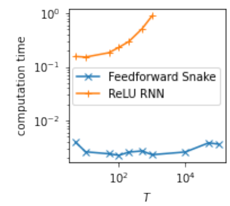
General Comment: We are very grateful to all the reviewers for carefully reading our paper. Their feedback has helped us to improve the paper accordingly. The two messages of this paper are that (1) learning a periodic or semi-periodic function with neural networks is yet unresolved, and we argue that the key to solve this is to focus on the extrapolation properties of neural networks; (2) we proposed to solve this problem with a simple alternative activation function.



(a) Training set, with $\sigma^2 = 0.2$, $T = 200$.



(b) Performance vs σ^2.



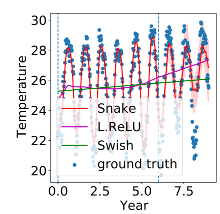
(c) Single update computation time vs. length of the data point.

Figure 1: Comparison of RNN with feedforward neural network with Snake.

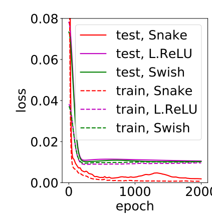
"A significant shortcoming is the lack of comparison with recurrent neural networks" (R2, R4): We respectfully yet strongly disagree with this. It is indeed interesting to see how RNN would perform for these tasks. However, the problem with RNNs is that they implicitly parametrize the data point x by time: $x = x(t)$. It is hence limited to model periodic functions of at most 1d and cannot generalize to a periodic function of arbitrary dimension; e.g., it is not clear how one could define RNN to learn the function $f(x, z) = \sin(x) + \sin(z)$, which is an easy task for feedforward networks with Snake. For this reason, we do not believe our method (with Snake + feedforward) needs to be a competitor to RNN. This being said, we perform a comparison of RNN with Snake with feedforward on a 1d problem. See Figure 1.a for the training set of this task. The simple function we try to model is $y = \sin(0.1x)$, we add a white noise with variance σ^2 to each y, and the model sees a time series of length T. See 1.b for the performance of both models, when $T = 100$, and validated on a noise-free hold-out section from $T = 101$ to 300. We see that the proposed method outperforms RNN significantly. On this task, One major advantage of our method is that it does not need to back-propagate through time (BPTT), which both causes vanishing gradient and prohibitively high computation time during training. In Figure 1.c we plot the average computation time of a single gradient update vs. the length of the time series, we see that, even at smallest $T = 5$, the RNN requires more than 10 times of computation time to update (when both models have a similar number of parameters, about 3000). This is a significant advantage of our method over RNN even for 1d periodic problems. These results will be added to the final version.

Stopping Criterion in Experiments (R4): Our stopping criterion is chosen fairly and reasonably: all experiments are stopped at the time when the training loss of all the methods stop to decrease and becomes a constant, i.e., when the model has converged. The performances of this converged model is not visibly different from the early stopping point for the experiments we considered. For example, the goal of Figure 7 (for the atmospheric experiment) and Figure 14.b (EUR-USD experiment in the appendix) are plotted to order to show that we stopped at the point when the model converged, moreover, neither of our model or the baselines seem to suffer significantly from overfitting, judging from these two figures. Therefore, the comparison is indeed fair and reasonable. In our final version, we will add similar plots for other experiments to clarify the stopping criterion for the experiments.

Comparison to Swish and Leaky-ReLU Seems Necessary (R5): The proposed method indeed outperforms Swish and Leaky-ReLU significantly for tasks in section 6.2 and 6.3. This is because Swish and Leaky-ReLU suffer from the same problem as ReLU, which is guaranteed by our theorem and by the discussion above. Therefore, we did not include them for visual clarity. Their performance on the tasks is now shown in Figure 2. We see that, for Swish and Leaky-ReLU, the learning is hard to mismatched inductive bias, and this leads to their inferior performance. We will add this plot to the appendix in the final version to avoid confusion.



(a) Same as Fig. 6 in the paper.



(b) Training loss and testing loss.

Figure 2: Comparison with Swish etc..

Strength of the claim (R5): 1. **Applicability of the theorem**. We agree that some qualifier is needed in this claim. On the one hand, the specific statement of the theorem applies to $\tanh$ and ReLU, but it is general as discussed in line 64-66. Since the proof is only based on the asymptotic property of activation function when $x \to \pm\infty$, one can prove the same theorem for any continuous activation function that asymptotically converges to a $\tanh$ or ReLU; for example, this would include Swish and Leaky-ReLU (and almost all the other ReLU-based variants), which converge to ReLU; one can follow exactly the same proving procedure to prove a similar theorem for each of these activation functions. On the other hand, we agree that this statement might be too strong, and we will state the above condition clearly for the applicability of our theory in the final version to avoid confusion. 2. **Generality of Snake.** Here, we say that Snake is more "general" in the sense that it is not only capable of approximating a function in the bounded region, but also capable of extrapolating beyond a bounded region (in a periodic way). This claim does need some qualification to clarify, and we will modify this claim in the final version. **Some details. Some Problems in the Appendix** (R1): We will add more details and correct grammatical errors to the Appendix sections.