The empirical methodology is a bit problematic since it does not explore deep learning at all, but rather on the time to realize WY can increase GPU utilization. They instead speed up by using fewer Householder matrices. An article contemporaneous to our submission [18] addresses parallel Householder products with the related CWY decomposition, but attains a $O(d^3)$ bound (see their table 2, serial complexity is $L^3$).

We should soon open-source “neuralsvd.py” from the supplementary material, which we updated to run our main experiments and draw Figure 3. We also updated “README.txt” to contain more details regarding the experimental setup.

We believe there is a misconception here. "FastH retains the same desirable time complexity as the sequential algorithm from [18] while reducing the number of sequential operations" (introduction L31-33). In other words, FastH is $27x$ faster than [18] due to less sequential work, not due to a difference in time complexity.

Thanks for raising this concern. The use of Householder matrices in deep learning has received much attention in previous work, e.g., [6,10,14,16,18]. FastH computes exactly the same as the algorithm used by [6,10,14,16,18]; repeating their experiments with FastH would thus attain the same results, albeit faster. Since we believe [6,10,14,16,18] adequately demonstrate the usefulness of Householder matrices for deep learning, we found the additional value of more such experiments were not that high. Furthermore, there are additional benefits to studying the performance of single operators as opposed to end-to-end deep learning experiments. Time complexity is a more transparent measure to investigate than the validation loss of deep learning models. Such measure shows the benefits of our approach irrespective of the architecture, optimizer, loss function and the many hyperparameters of complex networks.

Novelty: There is very little in the way of a fundamentally new idea. ... . The authors simply adjust the tool to the job.

As evidence against the claimed lack of novelty, we present three articles that would benefit from a $O(dn^2)$ parallel algorithm but did not “simply adjust the tool to the job.” [18] realized the issue with sequential computation and suggested a parallel $O(d^3)$ algorithm. [10] uses the related CWY decomposition for gradient computations, without realizing $WY$ can increase GPU utilization. They instead speed up by using fewer Householder matrices. An article contemporaneous to our submission [https://arxiv.org/abs/2004.08675] addresses parallel Householder products with the related CWY decomposition, but attains a $O(d^3)$ bound (see their table 2, serial complexity is $L^3$).

We would like to ask the authors to provide more details regarding the experimental setup to help reproducibility.