We would first like to thank all reviewers for their thorough assessment of this work, and to apologise for the apparent merge of conclusion and broader impact section; our intention to assess broader impact was to evaluate our algorithm from the perspective of the quantum computing community and potential (near-term) applicability. As pointed out a few times by reviewers, and which we strived to acknowledge ourselves, our model does not compete with classical state of the art models, and likely will not for the next few years, so we felt an indication why our algorithm is "special" in its current form and applicability seemed warranted. We did not intend this section to be the conclusion.

We thoroughly agree that including a comparison against [1] (see reviewer 1) and adding more benchmarks like copying, adding, or semantics would yield further insight. We aimed to strike a balance between "basic" and "higher level" tasks within the given page limit. Our quantum circuit-inspired parametrization *can* parametrize the entire Stiefel manifold; to see this observe that in Fig. 3, the parametrized $\mathbf{R}$ gates are interlaced with entangling gates (the neurons); this means our circuits are a strict superset of existing VQE models, proven to be universal for quantum computation, and hence densely fill $\mathrm{SU}(n)$ in the limit of taking deeper and deeper circuits. We would be happy to add a formal proof of this.

To summarise how the entire computation works, we start with Fig. 4, which depicts a zero-initialized product of initial and hidden state, on which we apply the same cell operation repeatedly; at every step, we supply the cell with varying input, and read out the output (indicated by the double lines, which signal a "classical" control). This cell is depicted in more detail in Fig. 3. The input parametrizes a series of bit flips on the input lanes. This means in order to e.g. input 4 at some step, we first encode it in binary as `100`, which means that the controlled $\mathbf{X}$ gate breaks down to a bit flip on the first lane, and leaves the other states untouched. Right after that, the input lane is in state $|100\rangle$. To transfer this information to the hidden state, a series of quantum neurons is applied. The dots in the circuit diagrams indicate "controlled on", which allows us to deduce which of the individual multi-controlled rotations within the quantum neuron in Fig. 2 apply. After the input stage and work stages, the input lane is reset to $|000\rangle$ to serve as clean slate on which we can write output using further quantum neurons. These neurons entangle the hidden state with the output state; yet measuring the output at the end of a cell collapses the output lanes to a basis state, which can thus be reset for the next cell application. The quantum neurons explained in Sec. 2.2 are fundamentally based on [CGA17]. To emphasise this, we first discuss this neuron and how it works, and include a discussion on how to make it work on superposed inputs using fixed-point amplitude amplification from [Tac+19]. As stated in l.158, we "increas(e) the number of control terms" which is how we extend from an affine to a pseudo-Boolean function, as one of the reviewer summarised.

The QRNN is very much a "quantum-inspired RNN", we wholeheartedly agree with this notion. Yet one crucial feature of parametrized quantum circuits (and in contrast to parametrizations of $SU(d)$ used in uRNN literature) is that the number of parameters grows polynomially in the qubit count, not in the dimension d. As one reviewer mentioned, unitary networks might lack the power to perform more complicated computational tasks; and this is certainly true if one compares, say, a nonlinear map on $\mathbb{R}^d$ with a unitary map on the same space (the latter being much weaker computationally). Yet a unitary map on $n = d$ qubits might be able to offset this disadvantage, as the tensor product $(\mathbb{R}^2)^{\otimes n}$ is an exponentially larger space than $\mathbb{R}^d$. In other words, if we were to simply remove "quantum" from this proposal and treat it as a linear algebra model, then this would be a parametrization of a map on $\mathbb{R}^d$ with a parameter count that grows poly*logarithmic* in d (as compared to polynomial in d for existing uRNN approaches). Thus the expressivity of this model on classical hardware will reach a limit, if we wish to include more and more parameters to increase the capacity of the network. The model thus scales efficiently only on a quantum computer. Nonetheless, the current setup is surprisingly frugal with resources, in that we can simulate small instances on classical hardware with meaningful results; in turn this is promising, as we do not need massive qubit counts on quantum devices either.

Eq. 2 can indeed be inferred from Fig. 1; e.g. if $|x\rangle = |1\rangle$ is a single qubit, the controlled rotations both apply in full; a matrix multiplication of three 4×4 matrices representing the quantum gates yields the pre-measurement state $|1\rangle|\psi\rangle$ where $|\psi\rangle = \cos^2\eta|00\rangle + \sin^2\eta|01\rangle + \cos\eta\sin\eta|01\rangle - \cos\eta\sin\eta|11\rangle$. Postselecting on the middle qubit to be in state $|0\rangle$ yields Eq. 2. The reason why non-superposition inputs like this one do not suffice is because as shown in Fig. 3, e.g. the first quantum neuron in the work space block has a control (the dot) right after a rotation gate; this implies that the control will not generally be a basis state. Overall, the hidden QRNN state will eventually be very entangled, and a superposition of many different basis states, so we have to include this possibility in the analysis.

The "good" transformation we wish to postselect on is a map $|0\rangle \mapsto \cos^{2^{ord}}\theta|0\rangle + \sin^{2^{ord}}|1\rangle =: |x\rangle$, with resulting norm $\||x\rangle\| \geq 1/2^{ord^2-1}$. Thus for $ord = 2$ (which we mostly used) the overhead is capped at 8 rounds of amplitude amplification. A similar argument shows that the postselection overhead during training for the output stage only scales with the width of the output word, independent of the size of the hidden state, which we are thus free to increase.

Further comments: i. The rotation matrix has indeed flipped signs, contrary to convention. ii. Similarly a question of convention, the quantum time evolution has by convention a minus sign in the exponent; yet one can simply absorb this sign in the hermitian $\mathbf{H}$ generating the unitary evolution. iii. In l.134 θ was chosen to indicate an arbitrary parameter, but η would also be valid. iv. We are thankful for all the further typos pointed out.