Hybrid quantum-classical reservoir computing for simulating chaotic systems

Filip Wudarski,[1,*] Daniel O'Connor,[2] Shaun Geaney,[2] Ata Akbari Asanjan,[1]
Max Wilson,[1] Elena Strbac,[2] P. Aaron Lott,[1] and Davide Venturelli[1,†]

[1] *USRA Research Institute for Advanced Computer Science (RIACS), CA*
[2] *Standard Chartered Bank, 1 Basinghall Avenue, London, UK*

Forecasting chaotic systems is a notably complex task, which in recent years has been approached with reasonable success using reservoir computing (RC), a recurrent network with fixed random weights (the reservoir) used to extract the spatio-temporal information of the system. This work presents a hybrid quantum reservoir-computing (HQRC) framework, which replaces the reservoir in RC with a quantum circuit. The modular structure and measurement feedback in the circuit are used to encode the complex system dynamics in the reservoir states, from which classical learning is performed to predict future dynamics. The noiseless simulations of HQRC demonstrate valid prediction times comparable to state-of-the-art classical RC models for both the Lorenz63 and double-scroll chaotic paradigmatic systems and adhere to the attractor dynamics long after the forecasts have deviated from the ground truth.

I. INTRODUCTION

In recent years, the rapid advancement of quantum computing (QC) has attracted interest from both academia and industry, as this new computational paradigm carries the potential to transform many disciplines of science and technology. In particular, a promising avenue for utilizing QC subroutines is the data-driven field of machine learning (ML), which is currently known as quantum machine learning (QML) [1–4]. Within the field of QML, hybrid quantum-classical approaches have emerged as a viable candidate to harness the power of both classical and quantum computation effectively, especially in the Noisy-Intermediate Scale Quantum era (NISQ) [5], where qubit numbers are low and the decoherence and error rates are high. These approaches integrate quantum algorithms and classical machine learning techniques to improve the accuracy and efficiency of quantum model training *e.g.* expectation values of multi-qubit observables (*i.e.* exponentially large matrices) are directly estimated through wave function sampling, while other subroutines such as optimization can be carried out on classical devices. Furthermore, hybrid algorithms can benefit from modular and scalable structures, which can either be simulated fully on classical computers, or as a proof-of-concept on NISQ devices.

Most hybrid algorithms are a class of variational quantum algorithms (VQAs) [6] that combine classical optimization routines with quantum hardware to converge towards an (ideally global) optimum of a cost function. Multiple versions of VQAs have been created, tackling various problems ranging from combinatorial optimization problems (quantum approximate optimization algorithm (QAOA) [7, 8]), to quantum chemistry (variational quantum eigensolver (VQE) [9]). All share the property of parameterized quantum circuits that are optimized in order to find the optimum of a cost function [10].

Multiple novel machine learning algorithms have emerged that are ideal candidates to be transferred and run on quantum hardware. In particular, a non-quantum approach called reservoir computing (RC) has gained significant attention recently due to its ability to efficiently approximate complex temporal dynamics in high-dimensional data [11–13]. RC is a paradigm that comprises of a set of recursive neural networks composed of an input layer, an intrinsic complex dynamical system known as a 'reservoir' layer, and a single, trainable readout layer. The main strength of RC lies in its reservoir system, which acts as a rich temporal and spatial feature extractor, and efficiently reduces computational complexity by maintaining fixed reservoir states without the need for backpropagation or weight adjustment within the reservoir network during the training phase. Consequently, only the readout layer undergoes training, resulting in a simplified optimization process that is both computationally efficient and effective in capturing the underlying dynamics of a system. One key aspect of RC is the recurrent topology of the reservoir layer, which fosters the nonlinear mixing and fading memory required for effective temporal pattern recognition. Several examples of reservoir systems have been proposed in literature, including Echo State Networks (ESNs) [11, 14, 15] and Liquid State Machines (LSMs) [16], which apply different activation functions, connectivity rules, and learning methods to achieve varying degrees of performance and robustness.

The application of RC in multiple domains, and its architecture flexibility, has inspired researchers to provide a quantum version of the algorithm, which initially took advantage of disordered dynamics [17]. Subsequently, the quantum reservoir computing framework has been extended [18–29]. In this paper, we aim to provide another perspective on quantum RC approaches. We propose a hybrid quantum reservoir computing (HQRC) algorithmic architecture, that has a scalable modular structure, and can easily be adjusted to accommodate hardware-efficient

* fwudarski@usra.edu
† dventurelli@usra.edu

implementations. It relies on partial state tomography of a parameterized quantum state. Where we restrict the tomographic approach to a handful of measurements (in practice, we use X, Y and Z bases measurements on all qubits) to extract as much dynamical information as possible from quantum circuits, while keeping the protocol cost (time for circuit execution and measurements) efficient. Most importantly, the measurement outcomes are appropriately transformed to form a classical reservoir state that can carry evolution history.

In this paper, we focus on predicting the time series behavior of chaotic systems. We choose the Lorenz63 [30] equations, one of the simplest yet-challenging paradigmatic models that serves as a low-dimensional proxy model for some features of weather dynamics, as the main test-bed for benchmarking algorithm performance. We additionally test the method on another common benchmark: the double-scroll model, to demonstrate that other chaotic systems are well-approximated by the HQRC. We investigate different variations of the HQRC Ansatz, *i.e.* we test impact of different forms of hyperparameters, in particular, different types and number of layers inside the quantum subroutine. This analysis allows us to find setups that are operating on reasonably low-dimensional reservoirs (108 and 271 for Lorenz63 and double-scroll benchmarks, respectively) that yields forecasts that are competitive with classical methods exploiting larger reservoirs. Additionally, our study has revealed that the after hyperparameter tuning, the HQRC algorithm provides accurate long-term reproduction of attractors.

The paper is organized as follows: First, we provide an introduction to standard reservoir computing algorithms, delving into their underlying principles and construction. Subsequently, we shift our attention towards the hybrid quantum model, where we elucidate the integration of quantum principles with classical reservoir computing methods to create quantum-enhanced learning models. Following this, we present a set of simulation results that shed light on the performance, scalability, and robustness of the proposed hybrid model in diverse learning scenarios. These proof-of-concept results offer valuable insights into the efficacy of our quantum-classical approach but also serve as a benchmark to gauge its potential capabilities in comparison with existing classical RC techniques. Finally, we provide concluding remarks, where we outline the implications of our findings for future research, improvement and deployment of our methods.

II. (CLASSICAL) RESERVOIR COMPUTING

Reservoir Computing (RC), a paradigm within Recurrent Neural Networks (RNNs), has emerged as a powerful and efficient computational approach designed for the modeling and prediction of complex, time-dependent data sequences [31, 32]. RC is an umbrella category of recurrent models including approaches such as Echo State Networks (ESN) [14] and Liquid State Machines (LSM) [16]. The primary strength of RC is the fixed, randomly initialized components called the 'input' and 'reservoir' layers, which provide a complex forward graph capable of encoding temporal dependencies and feature correlations [33]. Due to the random and unconventional connections in input and reservoir layers, it is impractical to calculate gradients and perform standard backpropagation, making these layers untrainable [34]. Given an input vector X_t, the reservoir layer maintains the temporal dynamic in the recursively generated state r_t by combining the previous state vector $r_{t-\Delta t}$ and a non-linear function of the input vector and the previous recurrent state

$$r_t = (1-\alpha) r_{t-\Delta t} + \alpha f\left(W_r r_{t-\Delta t} + W_X X_t\right) \tag{1}$$

where W_X and W_r represent the input and reservoir layers, respectively and $\alpha \in (0, 1]$ is referred to as the leak rate, governing the rate of new information leakage into the system. The states of the model are calculated autoregressively and can then be mapped to output data via an output layer called 'readout'. The readout layer is a ridge regression layer mapping the state vectors

$$R_t = (1, r_t, X_t)^T, \tag{2}$$

to the output data (y_t), where the unit element is playing the role of bias. The readout layer defines a linear set of equations for state-output mapping and it is often combined with Tikhonov regularization (widely known as ridge regression) to generalize the performance and prevent overfitting

$$\hat{W}_o = \arg\min_{W_o} \left\{ \|y - R W_o\|_2^2 + \beta \|W_o\|_2^2 \right\}. \tag{3}$$

Where the W_o and $\hat{W}_o$ represent the readout parameter and the optimized version. The y, R and β are the target (training) vectors, state vectors (arranged in a matrix with columns corresponding to different time instances t of y_t and R_t), and regularization parameter, respectively. Equation (3) yields the following vectorized solution

$$\hat{W}_o = (R^T R + \beta I)^{-1} R^T y\,, \tag{4}$$

where I is the identity matrix. The combination of the steps laid out in equations 1 and 4 provides the means for RC to learn and infer dynamical evolution.

III. HYBRID QUANTUM RESERVOIR COMPUTING

In this section, we provide a full description of the proposed HQRC algorithm. The aim of the method is to predict next steps in the evolution of a dynamical system. In particular, for a time-dependent system X_t, we use a fixed number of observations n, at times: $t_1, t_2, \ldots, t_n$ to learn intrinsic relationships in the observed data to obtain new data points for $t_{n+1}, t_{n+2}, \ldots, t_{n+p}$, that ideally match the true dynamics of the system. The algorithm is

composed of intertwined classical and quantum subroutines (see Fig. 1). In the first step, one needs to transform data X_t into a format that can fit the quantum subroutine. The quantum circuit comprises multiple layers that have fixed parameterized structure, though the parameters can (and in reality do) vary between each time step. The layers can be divided into three main categories: i) data encoding, ii) measurement feedback, and iii) random circuit layers. Data encoding layers can be either parameterized or parameter-free. For the former, we use linearly transformed data as input at time t, *i.e.* $Y_t^{(j)} = W_{in}^{(j)} X_t$, where $W_{in}^{(j)}$ is a fixed matrix (usually, but not always, taken to be random as in the case of classical RC) of size $d_{L_j} \times d_{\text{input}}$ (see Appendix A for more details on tested layers). Where d_{L_j} being the number of parameters that the data encoding layer L_j can accommodate (*e.g.* for a system of n qubits, a layer of single qubit X-rotations, will have $d_{L_j} = n$), and d_{input} is the dimension of the input vector, *i.e.*, X_t. To avoid large numerical values, we normalize W_{in} matrices by their largest singular values. In our experiments, we restrict ourselves to layers composed of parameterized R_X, R_Y or R_Z rotations and parameter-free CX gates. In the latter case, one needs to additionally define a graph of qubits upon which the gates act. In general, the graph should also be defined for parameterized gates as well. However, since we use single-qubit rotations, we assume that they act on every qubit in the quantum register. The graph can be selected to match the hardware-specific topology and therefore avoid quantum operations such as SWAP gates. Layers of CX are crucial for introducing entanglement in the circuit, enabling exploration of a larger portion of the Hilbert space. In principle, this gives higher expressibility of the Ansatz.

A similar high-level structure is present in layers ii) and iii). However, instead of taking transformed X_t as their parameters (if the layers utilize parameters), they rely on the measurement induced parameters for type ii) layer and randomly initialized gates for iii) layers. In the case of measurement feedback layers, one can feed the previous iteration measurement vector M_{t-1} or previous reservoir state r_{t-1}. As the size that either of the vectors can differ from the number of parameters in that layer, the practitioner can select which and how many of the components are ultimately utilized (*e.g.* in some experiments we use only single-qubit expectation values *e.g.* $\langle X_i \rangle$ to supplement single-qubit rotation on i-th qubit, *i.e.* $R_Y(\langle X_i \rangle)$). Furthermore, all parameterized layers that are not fixed, come with extra parameter transformation, that follows the feature map encoding approach [35–38]. These additional transformations can introduce non-linearities that are essential for learning complex dynamics.

The prepared quantum circuit is measured according to a selected scheme that defines the reservoir size in the HQRC. In order to optimize measurement efficiency while extracting sufficient amount of information from the circuit, we restrict ourselves to measuring all qubits in X, Y and Z bases. This allows us to form a measurement vector, that combines single-qubit expectation values, *i.e.* $\langle X_1 \rangle, \langle X_2 \rangle, \ldots, \langle X_n \rangle, \langle Y_1 \langle, \ldots, \langle Y_n \rangle, \langle Z_1 \rangle, \ldots, \langle Z_n \rangle$, and multi-qubit correlators featuring the same type of Pauli matrices. The latter operates on user-defined *connectivity* graphs, where vertices are qubits, and edges determine the correlator. For example, for a graph $\mathcal{G} = \{(1,2),(2,3)\}$ one extracts correlators $\langle X_1 X_2 \rangle, \langle X_2 X_3 \rangle$, similarly for Y and Z operators. Therefore, this approach provides many options for the choice of measurement scheme. In our experiments we restricted analysis up to three-body correlators defined on all-to-all connected graphs, which yields a reservoir size of $3N + 3\binom{N}{2} + 3\binom{N}{3}$. Note, that all these correletors can be efficiently calculated from just three measurement bases on the hardware.

The measurement vector is then used to create the reservoir state, denoted as r_t, as follows

$$r_t = (1-\alpha) r_{t-\Delta t} + \alpha g\Big[f_r(W_r \cdot r_{t-\Delta t}) + f_M(W_M \cdot M_t) + f_X(W_X \cdot X_t) \Big] . \tag{5}$$

Here, $\alpha \in [0,1]$ represents the leak rate, a parameter that controls the memory retention within the reservoir. The functions f_r, f_M, and f_X denote activation functions, which can introduce non-linearities into the system dynamics. The matrices W_r, W_M, and W_X correspond to fixed (usually random or identity) matrices associated with the reservoir state r_t, the measurement vector M_t, that implicitly depends on X_t (and previous outcomes, if measurement feedback layer is present), and the input state X_t, respectively. Additionally, we allow a *global* (non-linear if desired) transformation of the current contribution, by a function $g(\cdot)$. This equation encapsulates the critical step where quantum measurements are integrated into the reservoir state, infusing it with valuable information extracted from the quantum domain while allowing for the incorporation of non-linear transformations in the process. It is worth stressing that Eq. 5 gives us sufficient generalization to tune the contribution from each component of that equation such that one can restore the classical reservoir computing Eq. 1, if $f_M \equiv 0$ and $f_r = f_X = id$. Additionally, taking inspiration from classical reservoir computing, we renormalize the weight matrices by their largest singular value. This procedure, even though unnecessary in all cases, stabilizes the method, as the training and prediction steps stay within reasonable boundaries.

The final stages of our classical processing are characterized by operations that are efficient to simulate. At this point, the obtained reservoir states, in conjunction with the input states, are combined to form a vector R_t that plays a pivotal role in the subsequent learning procedure

$$R_t = (1, f_R(r_t), h_X(X_t))^T \ . \tag{6}$$

This learning procedure is governed by the ridge regression method, a well-established technique in machine learning.

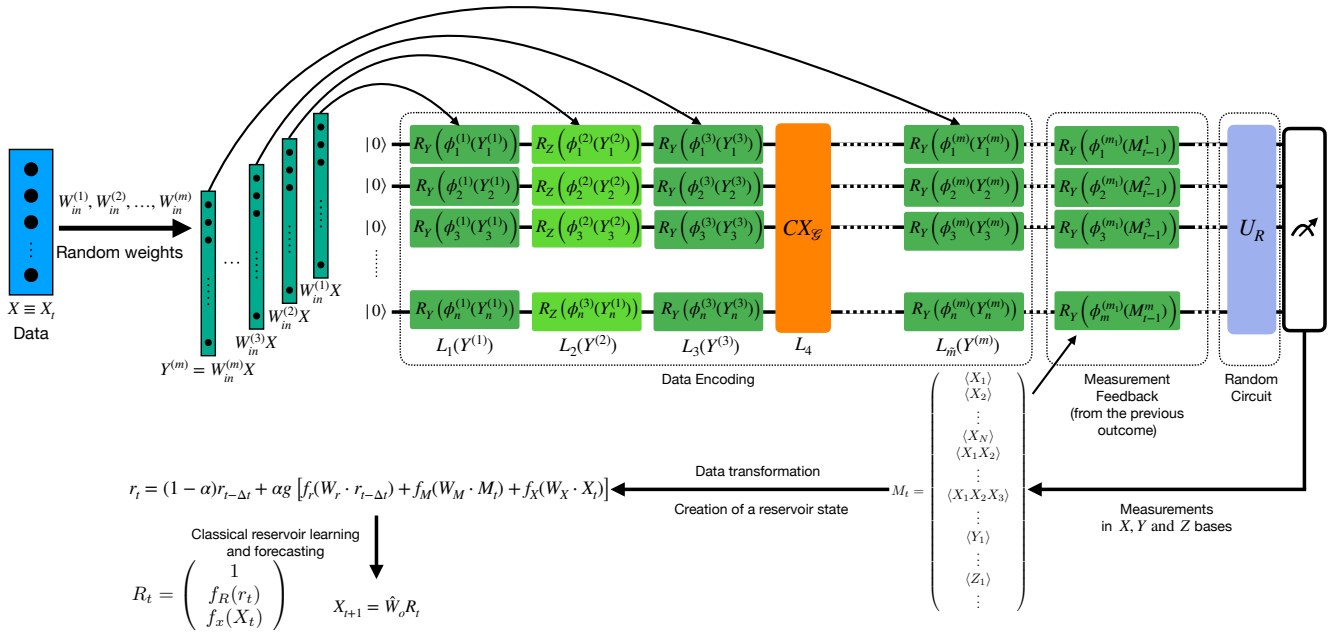

Figure 1. A schematic chart of HQRC approach. The data X_t at time t is transformed by fixed weight matrices $W_{in}^{(j)}$ to create vectors of data encoded parameters $Y^{(j)}$, which are distributed across the quantum circuit in parameterized layers $L_j(Y^{(j)})$ (in the chart we depict single-qubit rotations around Y and Z-axes). The circuit also possesses parameter-free layers (L_4) of CX (CNOT) gates, acting on qubits given by a graph $\mathcal{G}$. Subsequently, the circuit has layers related to measurement feedback that take outcomes from the previous time step $(t-1)$ and encodes them in parameterized gates (here depicted as taking M_{t-1} vector values, but it can also use r_{t-1} as parameters). In both data encoding and measurement feedback layers, one can additionally transform the parameters with feature maps, given by functions $\phi_k^{(j)}$. Finally, a random reservoir layer is applied that takes the form of a network of random gates (*e.g.* single-qubit rotations with random angles followed by a network of CX gates). The circuit is measured in fixed, specified bases (we usually choose X, Y and Z bases) in order to create a measurement vector, that comprises of single-qubit expectation values and multi-qubit correlators defined on a measurement graph, *e.g.*, $\langle X_i\rangle$, $\langle X_iX_j\rangle$, $\langle X_iX_jX_k\rangle$. The outcomes are then combined classically to generate the next reservoir state (Eq. (5)), and are used in the ridge regression procedure to determine $\hat{W}_o$ for future predictions.

It is notable the potential for introducing additional non-linear transformations to the vector before it is utilized in ridge regression in the form of f_R and h_X. The unit value prepended to the R_t vector takes the role of a bias. This affords our model a unique degree of flexibility, enabling it to capture and exploit complex, non-linear relationships within the data, thereby enhancing its capacity for accurate and nuanced learning. Similarly, though focusing on slightly different exploitation of non-linearity, work [13] demonstrated improvements in learning capabilities for classical approaches, if second-order contribution (*i.e.* squared reservoir states) is taken into account.

The introduced method places different stress on the meaning of hybrid quantum algorithms. As it takes inspiration from other techniques such as QAOA, VQE, or Quantum Neural Networks (QNN) [39, 40], it additionally relies more heavily on the incorporation of quantum features from measurements. Instead of focusing on extracting "physics-inspired" information, that is related to a problem to be solved, *e.g.* Hamiltonians in VQEs, it takes various measurements as proxies carrying potentially relevant information. Note that we treat the set of measurement operators as an additional hyperparameter that is arbitrarily chosen, instead of deliberately selected based on the problem. The measurements enable us to combine classical reservoir computing principles with the inherent computational capabilities of quantum systems. The core innovation of our method lies in its ability to map intricate problem spaces onto a high-dimensional Hilbert space constructed within a quantum circuit.

Measurement plays a pivotal role in our method: measurements are efficiently computable within the quantum framework, but they also are introducing non-linearities into the system. This introduction of non-linearity enables us to capture and manipulate complex relationships within the data, which may not be amenable to linear transformations. Furthermore, our method offers the flexibility to further transform these measurements as needed, allowing us to tailor the analysis to the specific characteristics of the problem at hand. This adaptability in measurement transformation ensures that our method can effectively address a wide spectrum of machine learning challenges, from linear to highly non-linear, whilst also maintaining computational efficiency and interpretability.

IV. RESULTS

In this section, we present the results of the HQRC approach applied to chaotic systems. We use the Lorenz63 benchmark as the main test-bed for the algorithm. However, we test our method also on other chaotic systems double-scroll, which is also a three-dimensional problem. Our analysis encapsulates simulations of quantum circuits on a classical computer, where we investigate various setups of hyperparameters.

A. Metrics

As the main metric that describes quality of the network, we use the valid prediction time (VPT) [41, 42]. The VPT is a time instance t, when the deviation between simulated predictions and the ground truth exceed a set threshold with respect to the root mean square error (RMSE)

$$RMSE(t) = \sqrt{\frac{1}{D}\sum_{i=1}^{D}\left(\frac{\tilde{y}_i(t) - y_i(t)}{\sigma_i}\right)^2} \geq \varepsilon, \qquad (7)$$

where $\tilde{y}_i(t), y_i(t)$ are i-th component at time t of predictions and the ground truth, respectively, σ_i is the i-th component of standard deviation of the true data serving as normalization and D is the dimensionality of the problem (*e.g.* for Lorenz63 and double-scroll $D = 3$). In our analysis, we select $\varepsilon = 0.3$, following a systematic review of classical reservoir computing in [42].

In the case of chaotic systems, it is clear that one cannot expect indefinite forecasting. Therefore, an equally important metric for benchmarking these systems is long-term attractor prediction, which means that the system stays in its basins of attractions, while potentially deviating from the correct component-wise predictions. Hence, in our analysis, we investigate the closeness of predicted and ground truth attractors. Additionally, we use Poincaré return map [43] to order all local maxima of the predicted and actual time series (for that we use longer simulations) and order them as $[z_1, z_2, \ldots, z_m]$ (for z component of the Lorenz63 vector, where the subscript denotes m-th maximum of the prediction phase) and plot them against each other $[z_i, z_{i+1}]$.

B. Classical simulations

Since the HQRC provides sufficient flexibility in defining the number of qubits, depth of the circuits as well as the number of components in the measurement vector, which ultimately translate into the size of the reservoir, one may select them such that they can be simulated on a classical computer. In this section, we analyze results that demonstrate that the given framework is capable of providing sufficient expressibility to predict behavior of chaotic systems that are comparable with the state-of-the-art results with classical reservoir computing.

1. *Lorenz63*

Lorenz63 [30] is a standard benchmark for classical RC, as it is well-studied chaotic model. The dynamics of the system is governed by the following set of differential equations

$$\begin{aligned} \frac{dx(t)}{dt} &= 10\,[y(t) - x(t)]\,, \\ \frac{dy(t)}{dt} &= x(t)\,[28 - z(t)] - y(t), \\ \frac{dz(t)}{dt} &= x(t)y(t) - \frac{8z(t)}{3}, \end{aligned} \qquad (8)$$

where we fixed coefficients to match the commonly used value in the literature [13]. These equations are analogous to a simplified weather model of atmospheric convection that experience uniform heating and cooling from below and above, respectively. In our experiments we use the following initial conditions $x(0) = 17.67715816276679$, $y(0) = 12.931379185960404$, and $z(0) = 43.91404334248268$.

In Fig. 2 we present behavior of the best performing setup for the Lorenz63 problem, that we have identified in this study (see Appendices for more setups). The setup is composed of 8-qubits, and the quantum circuit is composed solely of data encoding layers, which means that the recurrence only takes place on the classical processing side. We have identified (see Appendix A) that, in case of Lorenz63 we obtain substantially better results for reservoir Ansätze without the measurement feedback layers and a random circuit, as the addition of the latter may lead to low quality forecasting (see Appendix A). The outcomes are collected from a noiseless simulator without shot-noise, which means, the expectation values have been calculated from the wave function, instead of sampled with a finite number of shots. In Fig. 2 we compile: a) short-time predictions of all three components of Lorenz63 vector, and long-term predictions in panels b) for attractor reconstruction and c) and d) Poincaré return map, respectively. The system comprises of total 7 data encoding layers (panel e)), from which there are 2 parameter-free layers of CX network with underlying graph having connectivity of a ring. Additionally, the remaining 5 layers are single qubit rotations (see Appendix B for further details about hyperparameters). In that setup, since we use X, Y, Z measurements to construct the measurement vector composed of single qubit expectation values and two-qubit correlators between qubits from the fully connected graph, we utilize reservoirs of 108 size only. Despite the small size, for the best hyperparameters, we obtain VPT of 10.68, which is better than the state-of-the-art classical approaches with comparable reservoir sizes (see for comparison [42, 43], in the plot we included our RC simulations that yield better results than one reported in

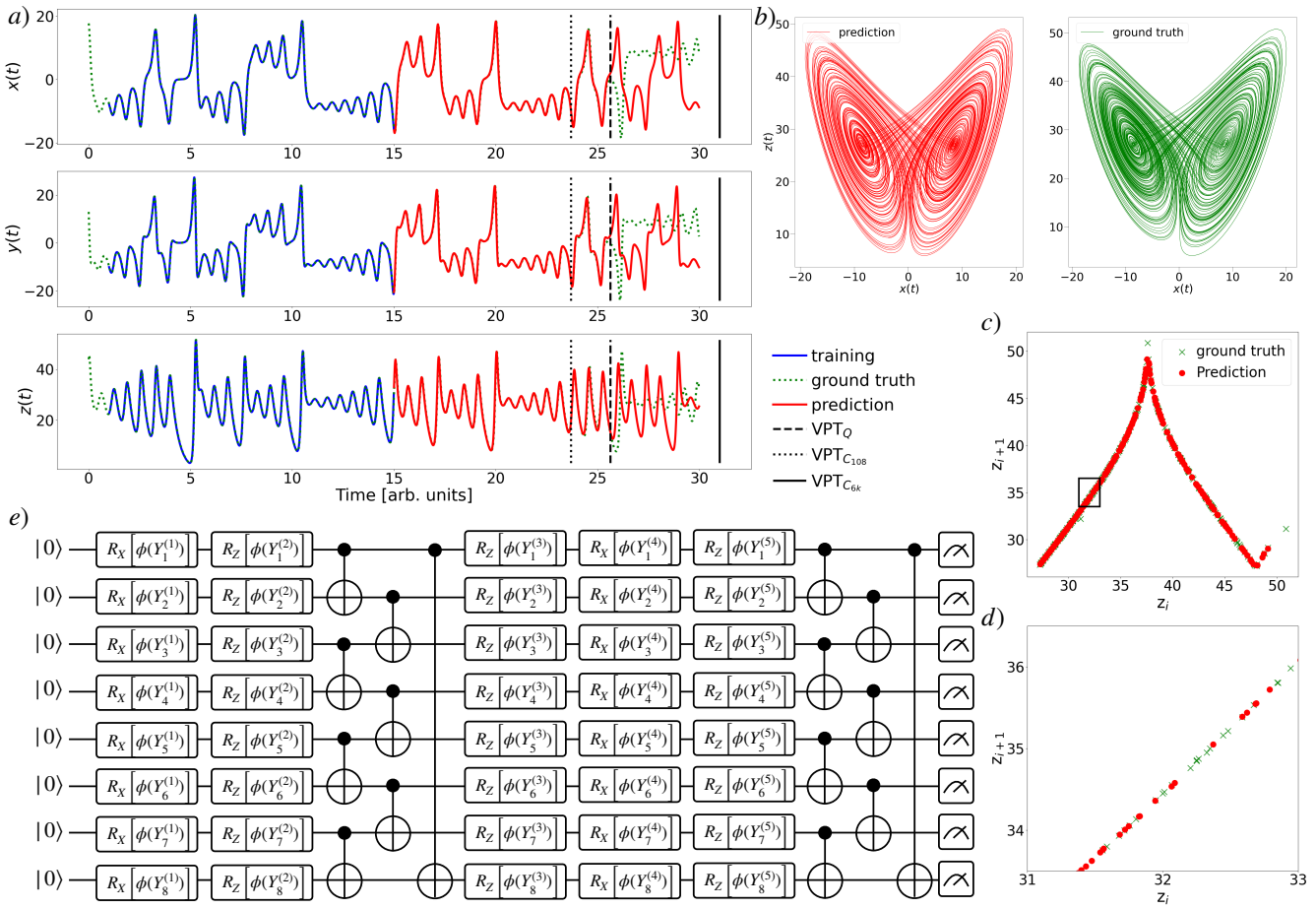

Figure 2. Simulated results of Lorenz63 chaotic system of reservoir computing. a) the HQRC approach simulated with circuit structure depicted in e), where $Y_k^{(j)} = W_{in}^{(j)} X$ is a data encoding vector, transformed with feature function $\phi(x) = \tanh(x)$. We use 1500 training points (with time increment of $dt = 0.01$ in arbitrary units) and 1500 time steps for the prediction phase. The Long-term qualitative correct behavior reconstruction is depicted in panel b) for attractor position, as well as for Poincaré return map in panel c) and zoomed-in version d). This indicates that even component-wise divergence after around 25 time units, the chaotic evolution displays appropriate oscillatory return pattern. We compare the results with VPT values for reservoir size of 6000 from [42] (solid line C_{6k}) and for our own RC for reservoir states of size 108 (dashed line C_{108}). The circuit is measured in X, Y and Z bases in order to extract single-qubit expectation values, and two-qubit correlators for the fully connected graph.

[42] for reservoir size of size 108). However, if larger reservoirs are utilized for classical systems, we observe that it is possible to obtain higher VPT values of ≈16. Note, that the reservoir size is not the only important property of these algorithms. both for classical or quantum ones. However, it is the reservoir size that primarily determines the coat of the most computationally demanding subroutine that scales as $O(Mn^2)$, where n is the reservoir size and M is the number of training steps, which is linked to matrix inversion in the ridge regression [13].

Recognizing that having noiseless simulation with exact expectation values is an idealized scenario, we investigate performance of the algorithm with finite samples, and a certain amount of coherent noise in encoding layers. We inject mispecification noise, as Gaussian noise random variable (leading to over- or under-rotations) centered around ideal value with standard deviation σ. In Fig. 3 we see that restricting to reasonably large number of shots (10,000) already significantly decrease performance. Similar detrimental effects are recorded for coherent noise, even when standard deviation is small. The limited number of shots, not only affect accuracy of the expectation values (effectively making results statistically unstable), but also forces them to be determined with finite decimal precision. The latter is crucial in case of predicting chaotic dynamics, as small deviations in initial conditions (or as in that case, throughout the training process) can lead to higher discrepancies in trajectories.

The presented results provide preliminary evidence to support that the HQRC approach is a viable quantum alternative to the classical RC approaches. In particular, the standard RC approach is sensitive to various hyperparameters (reservoir size, training length, etc. [42]). As the proposed method has multiple hyperparameters to select

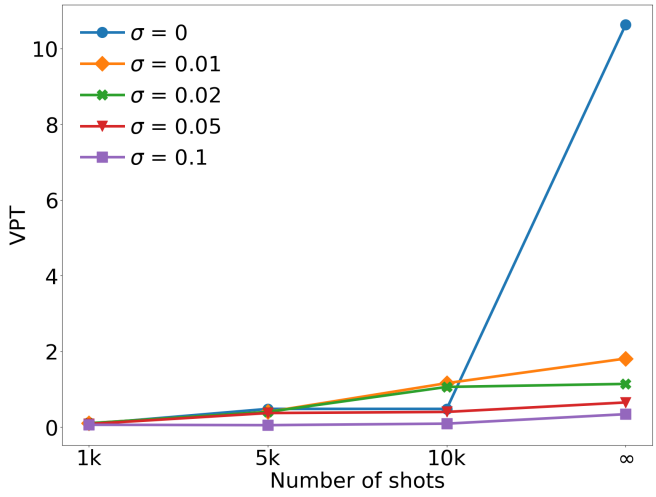

Figure 3. Performance of the HQRC algorithm with quantum circuit given in Fig. 2 e) for varying number of shots (∞ indicates the exact expectation values), for noiseless case $\sigma = 0$ and simulation with extra Gaussian coherent noise in rotation angles with standard deviation σ.

(number and type of layers, type of feature maps, measurement correlators, etc.), we observe a hyperparameter sensitivity as well, however even though the variations between performance can fluctuate, the forecasts rarely diverge from stable solution, which is not the case for the classical RC.

2. *Double-scroll*

Another popular benchmark is based on the dynamics of a double-scroll electronic circuit given by

$$\begin{aligned} \frac{dV_1(t)}{dt} &= \frac{V_1(t)}{R_1} - \frac{\Delta V(t)}{R_2} - 2I_r \sinh\left(\beta \Delta V(t)\right), \\ \frac{dV_2(t)}{dt} &= \frac{\Delta V(t)}{R_2} + 2I_r \sinh\left(\beta \Delta V(t)\right) - I(t), \\ \frac{dI(t)}{dt} &= V_2(t) - R_4 I(t), \end{aligned} \tag{9}$$

in dimensionless form, with $\Delta V(t) = V_1(t) - V_2(t)$. We fixed the parameters to: $R_1 = 1.2, R_2 = 3.44, R_4 = 0.193, \beta = 11.6$ and $I_r = 2.25 \times 10^{-5}$, and we discretize evolution into $dt = 0.25$ increments following [13], and initial conditions as $V_1(0) = 0.37926545$, $V_2(0) = 0.058339$, $I(0) = -0.08167691$.

Inspired by the Ansatz incarnation for the Lorenz63 model (see Appendix A), we performed a restricted search for well-performing hyperparamters. In Fig. 4 we show performance of the HQRC algorithm with 8 qubits and reservoir size of 271. This leads to VPT value of 107.25, that is also competitive with state-of-the-art results [13].

Further details on the hyperparameters and different setups is available in Appendix B.

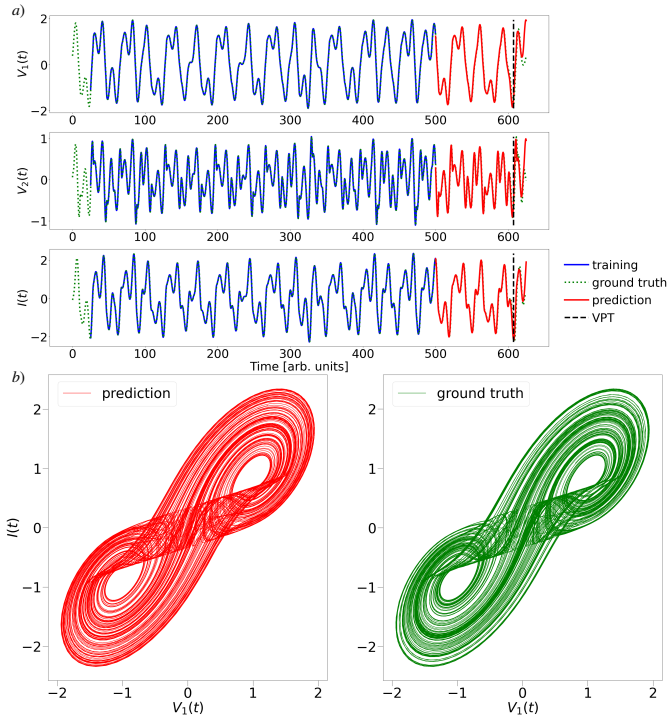

Figure 4. Simulated results for double-scroll system with 8 qubit. In panel a) we see training that overlaps with the ground truth and prediction steps that start to deviate after around 100 time units (VPT is 107.5). In the panel b) we depict reconstruction of long-term behavior based on the attractor.

V. CONCLUSIONS

We have introduced the hybrid quantum reservoir computing (HQRC) model, an extension of classical reservoir computing that introduces additional complexity through the addition of measurements from a modular quantum circuit. These proof-of-concept results demonstrate that the HQRC algorithm is an interesting candidate as a model for short-term forecasting of chaotic time-series, being capable of reconstructing short-term predictions of both Lorenz63 and double-scroll chaotic systems, as well as providing correct long-term attractor behavior. This is despite the reservoir states in HQRC being lower in dimension than in the classical RC benchmarks (*e.g.*in [42] reservoir of size above 1000 yield VPT$>$ 10, while reservoirs of size $\sim$ 100 have VPT$\sim$ 5 in [42] or $\sim$ 8 for our classical RC approach). In particular, the classical RC state-of-the-art approaches rely on two-step optimization, where the additional optimization routine is to determine the most suitable set of hyperparamters. Understanding the proper strategy for hyperparameter tuning, however, requires further development, as the most crucial hyperparameters are related to the quantum Ansatz (*e.g.* selection of number and type of layers can have different impact than fine-tuning regularization or leak rate values - we present partial analysis based on sweeping the parameters in Appendix A)..

We observe that our method as implemented is sensitive to (even coherent) noise and finite number of samples. We believe, that one can overcome these deficiencies with appropriately tailored classical transformations and error mitigation techniques. We leave this for future research.

We finally note that the presented results are based on small system sizes (number of qubits $\neq 10$) which can be easily simulated on a laptop. Therefore, the proposed algorithm, can also serve as a purely classical method. However, the simulation cost of full quantum circuits for the training and prediction phases is substantially more expensive than running state-of-art classical RC at this moment, so we do envision a hardware implementation.

Appendix A: Tested layers

On the high-level, the HQRC enables an arbitrary structure of the used Ansatz. Altough, it is beneficial to tailor it, such that it has sufficient expressibility to differentiate subtle differences in a problem to be solved. However, in most cases, this is *a priori* a complex task, and requires further systematic investigation on case by case basis. Here we present layers that have been tested, and allowed us to restrict the Ansatz to reasonably well-performing architectures. In particular, we have limited our search to repeated layers of general qubit rotations

$$U_3(\alpha,\beta,\gamma) = R_Y(\gamma)R_Z(\beta)R_Y(\alpha), \tag{A1}$$

or

$$U_3'(\alpha,\beta,\gamma) = R_Z(\gamma)R_X(\beta)R_Z(\alpha), \tag{A2}$$

followed by a network of CX gates with control and target qubits defined by a graph. Note, that U_3' exploits two R_Z gates, that in the most hardware architecture are performed virtually without any additional cost. Furthermore, we focused on graphs that are fairly sparse or reflect connectivity of currently available devices (see Fig. 5 e) panel). The main layers that we used are depicted in Fig. 5. In addition to the gate allocation, these layers exploit feature maps in the form of a function $\phi(\cdot)$ transforming rotation angles (see Appendix A 1 for further discussion).

In Fig. 6 we compile statistical results for the Lorenz63 problem, where we test different layer types acting on different number of qubits. The presented results are for noiseless simulations with exact expectation value. The circuits for each layer type are depicted in Fig. 5. Based on the Lorenz63 data, and tested layers we identified L_1 layer as the best performing. Layers containing measurement feedback, demonstrated low VPT values, suggesting that direct incorporation of measurement outcomes fails to benefit the performance. One notices that the used layers exhibit similar structure, and they only differ by the incorporated feature map encoding function, and incorporation of the full graph or either half of the graph.

1. Feature maps

As appropriate selection of feature maps can impact the performance, here we restrict to a handful of choices inspired by earlier works [35, 37, 38], we use ϕ in layers in Fig. 5 to be

1. arccos and arcsin functions to reflect Chebyshev polynomial style, if combined with R_Y rotation, though applied with other rotations as well (one can easily expand the scope of these functions as in [37]),

2. tanh to use standard ML activation function inside the quantum circuit,

3. *Fourier* encoding layers $\mathcal{F}_\beta$, we use fixed matrices of the form βV, where V is a rectangular matrix of size (N, d), where N is the number of qubits and d is input size, with entries $V_{ij} = (i-1)\delta_{i\%3+1,j}$, where $i\%3$ denotes modulo 3, and δ_{ij} is Kronecker delta. For $\beta = 2\pi$ this transformation resembles standard Fourier transform.

4. $\arcsin \circ \tanh$, which is a composition of two functions to avoid having arguments of arcsin beyond its domain of $[-1, 1]$.

Appendix B: Hyperparameters

In Table I we present all hyperparamters that are tunable in the algorithm, with some values that we fix in our simulations.

Appendix C: The role of hyperparamters

It is crucial to understand what role is played by different hyperparamters. Here we collect a number of results testing VPT performance of the algorithm from the main text (see Fig. 2 for more specification).

In Fig. 7 we test the impact of regularization strength for setups with and without noise, given by random Gaussian coherent over- and under-rotations, as well as in terms of shot-noise. Regularization can help stabilize results and prevent overfitting during the training phase. We see that the weaker regularization strength, the better algorithm performs, which is clearly detectable in the noiseless case with exact expectation values. However, once we inject noise, the general performance drops substantially, and even though, in general, lower regularization seems better, we fail to observe "healing" effect of larger values of regularization.

As currently available quantum computers, are not only limited by various imperfections (low fidelities, measurement errors, decoherence, etc.), but also suffer from restricted tuning range. In particular, implementing π

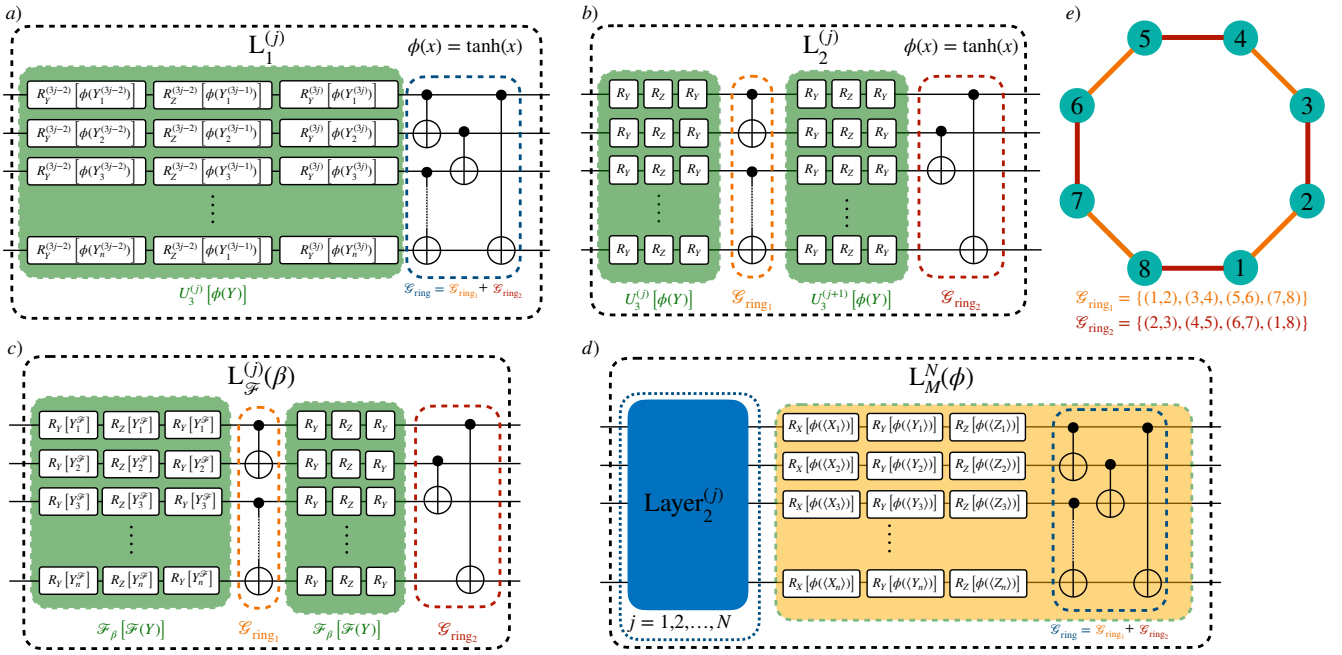

Figure 5. Main layer types that we explored for benchmarking Lorenz63 system. In a) the layer utilizes three single-qubit rotations parameterized by a function ϕ acting on the transformed input $Y^{(j)} = W_{in}^{(j)} X_t$, where the components of the transformed vector are distributed from top to bottom in qubit rotations. Subsequently, we use a network of CX gates that are arranged in a ring graph, first acting according to $\mathcal{G}_{\text{ring}_1}$, then as $\mathcal{G}_{\text{ring}_2}$. An 8-qubit graph partitioning is depicted in e). The second type layer L_2 in b) has similar structure with even layers followed by $\mathcal{G}_{\text{ring}_1}$ and odd by $\mathcal{G}_{\text{ring}_2}$ graph (here two layers are depicted). Panel c) displays a *Fourier* layer (again two layers even and odd lumped together), which instead of having random W_{in} matrices, operates on fixed *Fourier*-like transformations. Finally, in d) we show a layer that has the same structure as L_2, but followed by an extra measurement feedback layer at the end. Note, that single-qubit expectation values of Pauli operators are fed with extra functional transformation ϕ into single-qubit rotations along the same axis as the operator.

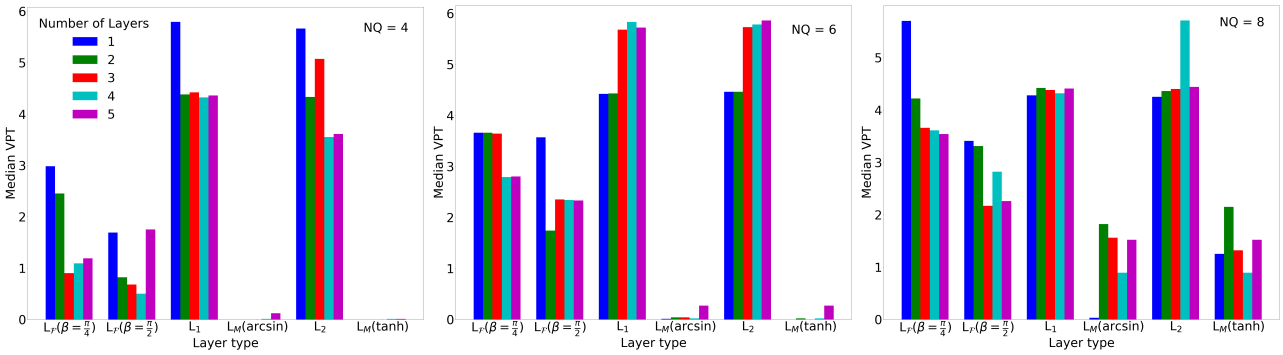

Figure 6. Median values for different number of qubits and layers of a given type. The median is taken over 10 random initialization (different numerical seed) of each setup.

with infinite or high decimal precision is currently prohibited, and one needs to tune rotational angles (on the level of pulse control) with finite decimal precision. Here we compile results of rounding precision, which means that for rounding equal to 2, we will approximate π as 3.14. In Fig. 8 we see that limited tuning freedom is also affecting the performance, which once more is not surprising in case of chaotic systems, where small deviations in initial conditions can cause large discrepancies. In particular, as the network tested here strongly relies on data encoding layer, which with that restriction fails to properly differentiate subtle differences between time steps. In case of noisy simulations with $\sigma \neq 0$, we inject noise after rounding the parameter, which we believe is more realistic scenario, and these two strategies do not commute.

HP	description	value
$\|\psi_{in}\rangle$	initial state	$\|+\rangle$ or $\|0\rangle$
leak rate	$\alpha \in [0, 1]$, controls contribution from the past reservoir state	$\alpha = 0.7$
seed	fixes random matrices for reproduction purposes	2
number of shots	how many times each circuit is measured to approximate expectation values	1,000; 5,000; 10,000 or ∞
f_R	a transformation function for the reservoir states r_t in order to prepare R_t training vector	tanh
f_x	a transformation function for input data X_t in order to prepare R_t training vector	tanh
f_X	a transformation function that acts on a transformed input data in order to create the reservoir state	identity
f_r	a transformation function that acts on the transformed previous reservoir state in order to create the current reservoir state	identity
f_M	a transformation function that acts on a transformed measurement data in order to create the reservoir state	identity
g	a transformation function that acts on contributions from the previous reservoir state, measurement outcomes, and input state in creation of the new reservoir state	identity
training length	how many time steps are used for training required for ridge regression	1500
test length	how many time steps are predicted	1200
prune size (warm up)	how many initial time steps are discarded from the training procedure	100
regularization β	stabilizes matrix inversion by addition of βI in ridge regression	10^{-8}
input type	what is the source of input for parametrized gates	data, measurement feedback, random
gate	gate type in the layer	R_Y, R_Z, R_X, CX
graph	graph of qubits on which the gates act	$\mathcal{G}_{\text{ring}}$
ϕ	feature map function that transforms input parameters	tanh
Pauli basis	which measurement basis is selected	X, Y and Z
graph	which qubits constitute the measurement correlators	all-to-all connectivity

Table I. A list of hyperparameters that specify the algorithm's setup. Here we focus on parameters related to the classical part of the algorithm. Value column refers to what are common choices in our experiments.

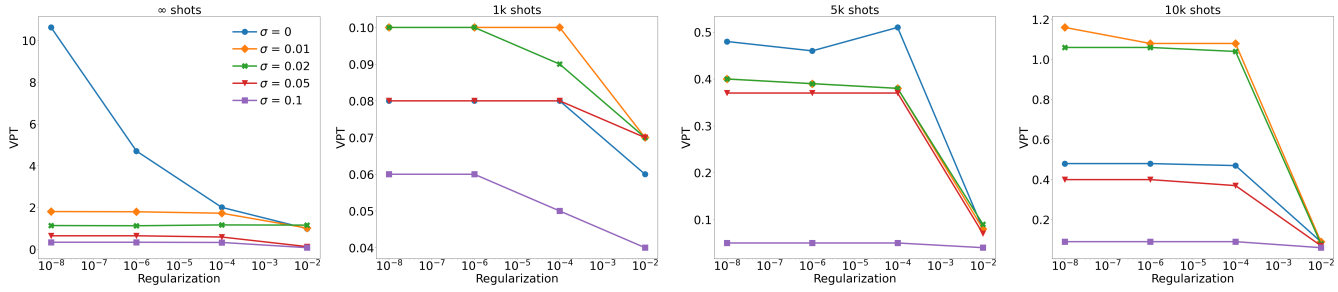

Figure 7. Investigation of the role of regularization strength for setup from Fig. 2 e) with 1500 training steps, 100 prune length and $\alpha = 0.7$. Each panel corresponds to different number of shots, and we report results for varying level of coherent noise strength parameterized by standard deviation of Gaussian noise σ.

[1] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd, "Quantum machine learning," Nature **549**, 195–202 (2017).
[2] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione, "An introduction to quantum machine learning," Contemporary Physics **56**, 172–185 (2014).
[3] Vedran Dunjko, Jacob M. Taylor, and Hans J. Briegel, "Quantum-enhanced machine learning," Physical Review

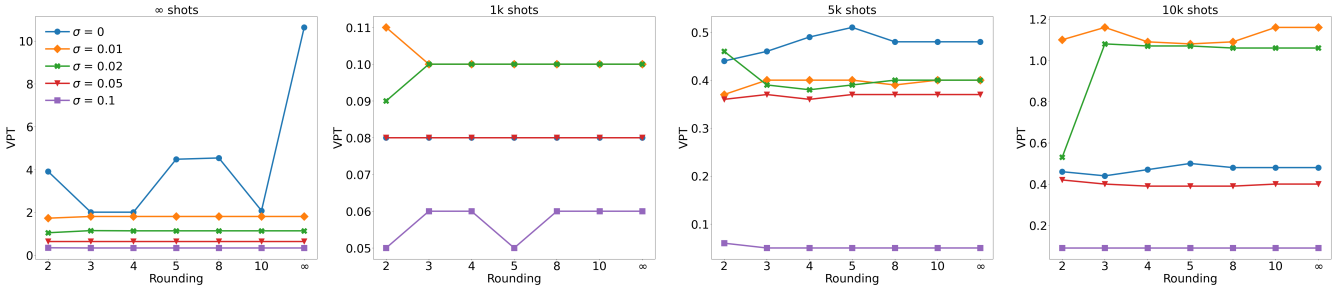

Figure 8. Effect of rounding the parameters in the encoding layers for the Ansatz given by Fig. 2 e) with 1500 training steps, 100 prune length and $\alpha = 0.7$. Different panels display results for different number of shots. We color code varying level of coherent noise with noise strength quantified by standard deviation of this Gaussian noise σ.

Letters **117** (2016), 10.1103/physrevlett.117.130501.

[4] Patrick Rebentrost, Maria Schuld, Leonard Wossnig, Francesco Petruccione, and Seth Lloyd, "Quantum gradient descent and newton's method for constrained polynomial optimization," (2018), arXiv:1612.01789 [quant-ph].

[5] John Preskill, "Quantum computing in the NISQ era and beyond," Quantum **2**, 79 (2018).

[6] María Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and Patrick J. Coles, "Variational quantum algorithms," Nature Reviews Physics **3**, 625 – 644 (2020).

[7] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann, "A quantum approximate optimization algorithm," (2014), arXiv:1411.4028 [quant-ph].

[8] Stuart Hadfield, Zhihui Wang, Bryan O'Gorman, Eleanor Rieffel, Davide Venturelli, and Rupak Biswas, "From the quantum approximate optimization algorithm to a quantum alternating operator ansatz," Algorithms **12**, 34 (2019).

[9] Alberto Peruzzo, Jarrod R. McClean, Peter J. Shadbolt, Man-Hong Yung, Xiaoqi Zhou, Peter J. Love, Alán Aspuru-Guzik, and Jeremy Lloyd O'Brien, "A variational eigenvalue solver on a photonic quantum processor," Nature Communications **5** (2013).

[10] Marcello Benedetti, Erika Lloyd, Stefan Sack, and Mattia Fiorentini, "Parameterized quantum circuits as machine learning models," Quantum Science and Technology **4**, 043001 (2019).

[11] Mantas Lukoševičius, "A practical guide to applying echo state networks," in *Neural Networks* (2012).

[12] Matteo Cucchi, Steven Abreu, Giuseppe Ciccone, Daniel Brunner, and Hans Kleemann, "Hands-on reservoir computing: a tutorial for practical implementation," Neuromorphic Computing and Engineering **2**, 032002 (2022).

[13] Daniel J. Gauthier, Erik Bollt, Aaron Griffith, and Wendson A. S. Barbosa, "Next generation reservoir computing," Nature Communications **12** (2021), 10.1038/s41467-021-25801-2.

[14] Herbert Jaeger, "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note," Bonn, Germany: German National Research Center for Information Technology GMD Technical Report **148**, 13 (2001).

[15] Herbert Jaeger, "Echo state network," Scholarpedia **2**, 2330 (2007).

[16] Wolfgang Maass, Thomas Natschläger, and Henry Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," Neural computation **14**, 2531–2560 (2002).

[17] Keisuke Fujii and Kohei Nakajima, "Harnessing disordered-ensemble quantum dynamics for machine learning," Physical Review Applied **8**, 024030 (2017).

[18] Pere Mujal, Rodrigo Martínez-Peña, Johannes Nokkala, Jorge García-Beni, Gian Luca Giorgi, Miguel C Soriano, and Roberta Zambrini, "Opportunities in quantum reservoir computing and extreme learning machines," Advanced Quantum Technologies **4**, 2100027 (2021).

[19] Pere Mujal, Rodrigo Martínez-Peña, Gian Luca Giorgi, Miguel C Soriano, and Roberta Zambrini, "Time-series quantum reservoir computing with weak and projective measurements," npj Quantum Information **9**, 16 (2023).

[20] Rodrigo Araiza Bravo, Khadijeh Najafi, Xun Gao, and Susanne F Yelin, "Quantum reservoir computing using arrays of rydberg atoms," PRX Quantum **3**, 030325 (2022).

[21] Philipp Pfeffer, Florian Heyder, and Jörg Schumacher, "Hybrid quantum-classical reservoir computing of thermal convection flow," (2022), arXiv:2204.13951 [quant-ph].

[22] Philipp Pfeffer, Florian Heyder, and Jörg Schumacher, "Reduced-order modeling of two-dimensional turbulent rayleigh-bénard flow by hybrid quantum-classical reservoir computing," (2023), arXiv:2307.03053 [physics.flu-dyn].

[23] Danijela Marković and Julie Grollier, "Quantum neuromorphic computing," Applied physics letters **117** (2020).

[24] LCG Govia, GJ Ribeill, GE Rowlands, HK Krovi, and TA Ohki, "Quantum reservoir computing with a single nonlinear oscillator," Physical Review Research **3**, 013077 (2021).

[25] LCG Govia, GJ Ribeill, GE Rowlands, and TA Ohki, "Nonlinear input transformations are ubiquitous in quantum reservoir computing," Neuromorphic Computing and Engineering **2**, 014008 (2022).

[26] Kohei Nakajima, Keisuke Fujii, Makoto Negoro, Kosuke Mitarai, and Masahiro Kitagawa, "Boosting computational power through spatial multiplexing in quantum reservoir computing," Physical Review Applied **11**, 034021 (2019).

[27] Daniel Fry, Amol Deshmukh, Samuel Yen-Chi Chen, Vladimir Rastunkov, and Vanio Markov, "Optimizing quantum noise-induced reservoir computing for nonlinear and chaotic time series prediction," arXiv preprint arXiv:2303.05488 (2023).

[28] Tomoyuki Kubota, Yudai Suzuki, Shumpei Kobayashi, Quoc Hoan Tran, Naoki Yamamoto, and Kohei Nakajima,

"Quantum noise-induced reservoir computing," arXiv preprint arXiv:2207.07924 (2022).
[29] Yudai Suzuki, Qi Gao, Ken C Pradel, Kenji Yasuoka, and Naoki Yamamoto, "Natural quantum reservoir computing for temporal information processing," Scientific reports **12**, 1353 (2022).
[30] Edward N Lorenz, "Deterministic nonperiodic flow," Journal of atmospheric sciences **20**, 130–141 (1963).
[31] Benjamin Schrauwen, David Verstraeten, and Jan Van Campenhout, "An overview of reservoir computing: theory, applications and implementations," in *Proceedings of the 15th european symposium on artificial neural networks. p. 471-482 2007* (2007) pp. 471–482.
[32] David Verstraeten, Benjamin Schrauwen, Michiel d'Haene, and Dirk Stroobandt, "An experimental unification of reservoir computing methods," Neural networks **20**, 391–403 (2007).
[33] Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott, "Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach," Physical review letters **120**, 024102 (2018).
[34] Matthias Freiberger, Peter Bienstman, and Joni Dambre, "A training algorithm for networks of high-variability reservoirs," Scientific reports **10**, 14451 (2020).
[35] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, "Quantum circuit learning," Physical Review A **98** (2018), 10.1103/physreva.98.032309.
[36] Javier Gil Vidal and Dirk Oliver Theis, "Input redundancy for parameterized quantum circuits," (2020), arXiv:1901.11434 [quant-ph].
[37] Oleksandr Kyriienko, Annie E. Paine, and Vincent E. Elfving, "Solving nonlinear differential equations with differentiable quantum circuits," Physical Review A **103** (2021), 10.1103/physreva.103.052416.
[38] Maria Schuld and Nathan Killoran, "Quantum machine learning in feature hilbert spaces," Physical Review Letters **122** (2019), 10.1103/physrevlett.122.040504.
[39] Quynh T. Nguyen, Louis Schatzki, Paolo Braccia, Michael Ragone, Patrick J. Coles, Frédéric Sauvage, Martín Larocca, and María Cerezo, "Theory for equivariant quantum neural networks," ArXiv **abs/2210.08566** (2022).
[40] Steve Abel, Juan Carlos Criado, and Michael Spannowsky, "Completely quantum neural networks," ArXiv **abs/2202.11727** (2022).
[41] Pantelis R. Vlachas, Jaideep Pathak, Brian R. Hunt, Themistoklis P. Sapsis, Michelle Girvan, Edward Ott, and Petros Koumoutsakos, "Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics," (2020), arXiv:1910.05266 [eess.SP].
[42] Jason A. Platt, Stephen G. Penny, Timothy A. Smith, Tse-Chun Chen, and Henry D. I. Abarbanel, "A systematic exploration of reservoir computing for forecasting complex spatiotemporal dynamics," (2022), arXiv:2201.08910 [cs.NE].
[43] Jaideep Pathak, Zhixin Lu, Brian R. Hunt, Michelle Girvan, and Edward Ott, "Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data," Chaos: An Interdisciplinary Journal of Nonlinear Science **27** (2017), 10.1063/1.5010300.