Quantum Reservoir Computing Using a Spin-Based Kicked Top System for Classical and Quantum Machine Learning Tasks

A Thesis

submitted to
Indian Institute of Science Education and Research Pune in partial fulfillment of the requirements for the BS-MS Dual Degree Programme

by

Aniruddha Trivedi

Indian Institute of Science Education and Research Pune
Dr. Homi Bhabha Road,
Pashan, Pune 411008, INDIA.

April, 2025

Supervisor: Dr. M.S. Santhanam

Certificate

This is to certify that this dissertation entitled Quantum Reservoir Computing using Quantum Kicked Toptowards the partial fulfilment of the BS-MS dual degree programme at the Indian Institute of Science Education and Research, Pune represents study/work carried out by Aniruddha Trivedi at Indian Institute of Science Education and Research under the supervision of Dr. M.S. Santhanam, Professor, Department of Physics , during the academic year 2024-2025.

Supervisor: Dr. M.S. Santhanam

Aniruddha Trivedi
Roll Number - 20201002

This thesis is dedicated to my friends and family

Declaration

I hereby declare that the matter embodied in the report entitled Quantum Reservoir Computing using Quantum Kicked Top are the results of the work carried out by me at the Department of Physics, Indian Institute of Science Education and Research, Pune, under the supervision of Dr. M.S. Santhanam and the same has not been submitted elsewhere for any other degree. Wherever others contribute, every effort is made to indicate this clearly, with due reference to the literature and acknowledgment of collaborative research and discussions.

Aniruddha Trivedi
Roll Number - 20201002

Supervisor: Dr. M.S. Santhanam

Acknowledgments

I would like to thank my supervisor, Prof. M.S. Santhanam, for constantly guiding me throughout the thesis and supporting me during the entire duration of the thesis, making this journey a very pleasant learning experience. I am very grateful to Nisarg Vyas for his constant guidance and for helping me whenever I faced any difficulties in my work. This study is built upon his work on reservoir computing as well. Discussions with him, Prof. Santhanam, and Dharmesh Yadav helped me immensely in understanding the core concepts in quantum spin systems and reservoir computing.

I would also like to thank Prof. TS Mahesh for agreeing to carry out the experiments in the NMR Lab along with his student Vivek Sabarad. I am really thankful for the time they gave us for this and for ensuring everything was in place even before the experiments.

I acknowledge the KVPY Scholarship awarded to me from the Department of Science and Technology, Government of India, throughout the duration of the BS-MS program.

I want to thank all my teachers and friends at IISER Pune for supporting my passion for Physics and providing an excellent research environment. Lastly, I want to thank my family; without their constant support and love, none of this would have been possible.

Contents

List of Figures

Abstract

Physical reservoir computing is a machine-learning paradigm that aims to harness the natural dynamics of a system to perform a variety of tasks. Quantum reservoir computing is an emerging machine-learning framework that has shown the potential to carry out complex tasks owing to its inherent quantum properties. A quantum reservoir can not only perform on classical data but also can accept quantum inputs for performing quantum tasks, which is not possible on a classical system. A quantum kicked top is a spin system capable of showing signatures of quantum chaos. The inherent non-linearity in this system can be studied and utilized for reservoir computing. In this paper, we have shown the simulation and the encoding scheme for quantum kicked top. A quantum kicked top can also be readily simulated as a system of qubits, which helps experimentally perform the physical reservoir computing. We chose the NMR system for this and also showed reservoir computing in a solid-state adamantane sample using our scheme.

Chapter 1

Introduction

Reservoir computing is a framework that is part of a larger paradigm of neuromorphic and unconventional computing. One of the key motivations behind them is to match the performance and power efficiency of the brain in performing computational tasks [1]. Reservoir computing aims to achieve this by using physical dynamical systems for computational purposes. They are much more power efficient and facilitate fast computation as only the weights at the readout layer need to be trained. In this study, we employ a quantum-kicked top system to perform reservoir computing and note its performance in achieving classical and quantum tasks. All the Python codes used in this thesis are available on GitHub: `https://github.com/AniTrivedi/QKT-Reservoir-Computing`

1.1 Basics of Reservoir Computing

Reservoir computing is derived from several recurrent neural network models [3] and consists of a non-linear reservoir, which maps input to a higher dimensional space, and a readout, trained to analyze patterns from the high-dimensional states in the reservoir [2]. Training in reservoir computing occurs only at readout weights that linearly map the reservoir state to the desired output. We describe this scheme more formally below:

1. Input is fed to the reservoir via an appropriate encoding scheme depending on the task. It can be in the initial state of the reservoir or the evolution parameters. The goal is to have an encoding that has a sufficient impact on the readout depending on the input.

2. The system is then evolved according to its natural dynamics or evolutionary scheme provided. The values of one or more of the reservoir nodes are monitored after every period, which is called time multiplexing. This generates the reservoir state vector corresponding to the input signal.

3. Training is done by creating a reservoir state matrix R, columns of which are the reservoir state vectors corresponding to every input data in the training dataset $\mathbf{x}_{\text{train}}$. Then, via linear regression with corresponding outputs $\mathbf{y}_{\text{train}}$, we obtain a weight matrix W such that

$$W = \mathbf{y}_{\text{train}} R^{-1}. \tag{1.1}$$

4. Having trained the reservoir, we can now get predicted outputs $\tilde{y}_{\text{test}}$ corresponding to unseen testing input data x_{test} by

$$\tilde{y}_{\text{test}} = W r(x_{\text{test}}), \tag{1.2}$$

   where $r(x)$ is reservoir state vector for input x.

5. Next step is to compare the predicted output to the original output data y_{test} and make changes to the reservoir parameters accordingly to minimize the error.

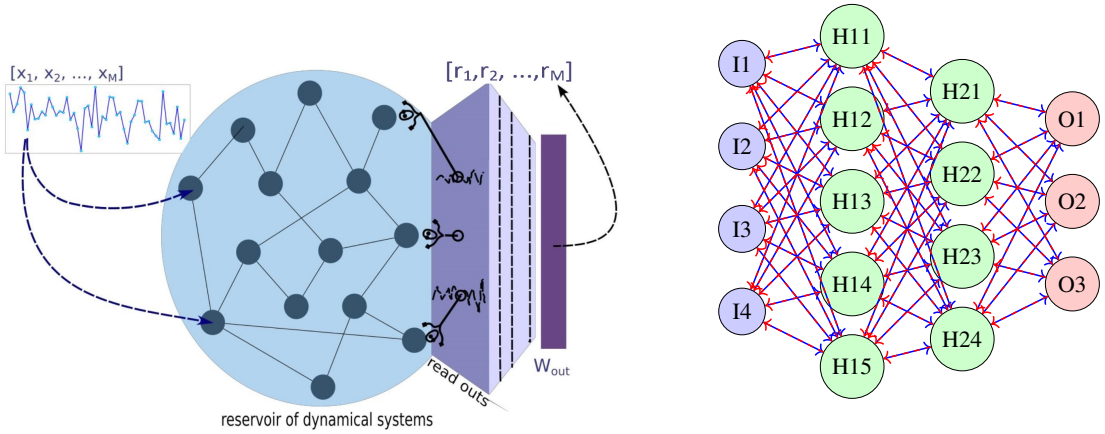
The reservoir can be an artificial network of nodes or a physical dynamical system, which helps in reducing effective computing costs. There are some properties of the reservoir that help to exploit its computational potential:

- **Memory of previous states-** This enables the reservoir to retain some information about earlier inputs in the current state of the reservoir. This helps perform tasks like time series prediction in which the next data point inherently depends on previous data points.

- **Separation of inputs in the reservoir state space-** This helps the linear regression step better map the reservoir states to the desired output and helps to distinguish very close inputs [4].

- **Presence of sufficient non-linearity-** The presence of sufficient non-linearity in readouts helps better capture the dynamics and identify patterns in our data, facilitating better mapping to desired outputs [5].

1.2 Comparison between Reservoir Computing and Neural Networks



(a) Reservoir Computing

(b) Feedforward Neural Network

Figure 1.1: Schematic diagrams of (a) reservoir computing for time series prediction where the connections are fixed and (b), a feedforward neural network schematic, showing the input layer in blue, hidden layers in green, and the output layer in red. The connections are trainable using backpropagation and gradient descent.

While both neural networks and reservoir computing are machine learning paradigms, they have different architectures, training methods, and applications. Schematic illustrations of both is shown in Fig.1.1 We mention below some key differences in both:

1. A neural network is a collection of interconnected artificial neurons (nodes) that process data using weighted connections. On the other hand, reservoir computing is a computational framework that uses a fixed, high-dimensional reservoir of randomly connected nodes or a physical dynamical system where all nodes and connections might not even be identified.

2. In neural networks, the entire network is trained using backpropagation and gradient descent

to optimize the weights. While in reservoir computing, only the output layer is trained using linear regression techniques like ridge regression, where the reservoir remains unchanged.

3. In neural networks, as the weights of all layers are learned and updated using backpropagation, which involves computing gradients and adjusting the network's parameters iteratively, this requires computationally expensive training with large datasets. In reservoir computing, only the output layer is trained, while the connections in the reservoir remain fixed after initialization. This makes the training much faster and computationally efficient. Hence, reservoir computing is much more energy-friendly.

4. Standard feedforward neural networks lack memory unless specialized architectures like Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTMs) are used. However, in reservoir computing, the reservoir naturally acts as a memory system, storing past inputs implicitly without the need for backpropagation through time (BPTT) [6].

5. Reservoir computers can be a good alternative to neural networks where the goal is just to solve some specific tasks as the hyperparameters need not be updated then. This will keep the energy and cost to a minimum.

1.3 Quantum Reservoir Computing

Quantum reservoir computing exploits the natural quantum dynamics of quantum systems as the reservoir. The quantum system lets us get real-time signals directly from exponentially large degrees of freedom [9]. Instead of using quantum neural networks with qubits as neurons, the basis states of these quantum systems serve as nodes of the network [9]. This makes quantum reservoir computing scalable and experimentally realizable [10, 11, 12]. The readout from the quantum reservoir is typically performed by measuring specific quantum observables. The connections from the reservoir to the output layer are trained using classical techniques, similar to classical RC. The goal is to find the optimal linear combination of quantum observables corresponding to the desired output.

Employing quantum mechanical systems for reservoir computing can be advantageous for the following reasons:

- Leveraging non-trivial quantum correlations to improve computational capacity. The presence of entanglement has been shown to improve memory [13].
- Quantum systems allow us to explore a much higher-dimensional Hilbert space than possible classically; this means smaller-sized systems can perform the same tasks as efficiently as larger classical systems[14].
- Any potential speed-ups or improvements in the performance of quantum reservoir computing would be interesting to explore.
- It can serve as a general test bed to understand the computational implications of the quantum properties of the system.

In this study, we investigated the potential of a single quantum system to perform machine learning tasks. This serves as a proof-of-principle demonstration of the learning capabilities of a single dynamical system and establishes the quantum kicked top as a system for potential regular use in unconventional computing. We describe this quantum system in the following section.

Chapter 2

Quantum Kicked Top

2.1 Introduction to Quantum Kicked Top (QKT)

Quantum kicked top (QKT) is a time-dependent spin system whose square of the angular momentum **J** is conserved [15]. Its Hilbert space has finite dimensionality $(2j+1)$ where $j(j+1)$ is the eigenvalue of the $\mathbf{J}^2$ operator. governed by the Hamiltonian

$$H = \hbar\frac{pJ_y}{\tau} + \hbar\frac{\kappa J_z^2}{2j}\left(\sum_{n=-\infty}^{n=\infty}\delta(t-n\tau)\right), \tag{2.1}$$

where J_z, J_y, and J_x are the components of the angular momentum operator along the respective axes, obeying the standard angular momentum commutation relations. j is the size of the spin. The first term describes a constant precession of the top around the y-axis with angular frequency p/τ, and the second term corresponds to non-linear impulsive kicks about the z-axis given at intervals of τ with kick-strength κ, which is also called the chaoticity parameter. The kick-to-kick Unitary Floquet time evolution operator of QKT for one period is given by

$$U = \exp\left(-i\frac{\kappa}{2j}J_z^2\right)\exp\left(-ipJ_y\right). \tag{2.2}$$

So, the evolution of state to n_{th} time-period will be given by the unitary operator U^n.

The directed angular momentum states $|\theta,\phi\rangle$ are of special importance as they are states of min-

imum uncertainty; hence, they are also called coherent states. One such state is $|j,j\rangle$, and all other-directed states can be generated by the application of the unitary rotation operator

$$R(\theta,\phi) = \exp\{i\theta(J_x \sin\phi - J_y \cos\phi)\} \tag{2.3}$$

on $|j,j\rangle$, as shown below

$$|\theta,\phi\rangle = R(\theta,\phi)\,|j,j\rangle\,. \tag{2.4}$$

$1/j$ is the minimum allowed uncertainty in $\mathbf{J}$, which occurs in the case of these coherent states. The classical behavior of a kicked top can be studied in the limit $j \to \infty$ where this uncertainty drops to zero [15]. We first rescale the angular momentum operators as $X_i = J_i/j$ which satisfy $[X_i, X_k] = (1/j)\varepsilon_{ikl}X_l$, then compute the Heisenberg equations for them. We then take the limit of $j \to \infty$ to obtain our classical map of kicked-top

$$\begin{aligned} \left.\begin{matrix} X_{n+1} \\ Y_{n+1} \end{matrix}\right\} &= \begin{cases} \text{Re} \\ \text{Im} \end{cases} (X_n \cos p + Z_n \sin p + iY_n)e^{ik(Z_n \cos p - X_n \sin p)} \\ Z_{n+1} \quad &= -X_n \sin p + Z_n \cos p. \end{aligned} \tag{2.5}$$

By conservation laws

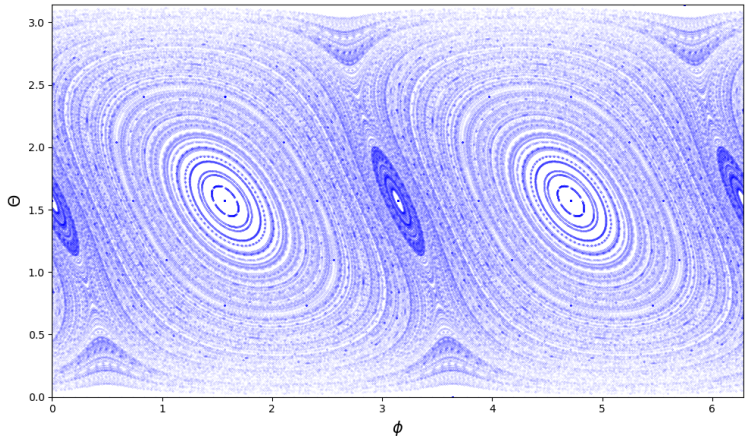
$$\begin{aligned} [\mathbf{J}^2, H] &= 0 \\ [\mathbf{J}^2, U] &= 0, \end{aligned} \tag{2.6}$$

we have $\mathbf{X}^2 = 1$, i.e., they are restricted on a sphere of unit radius, which allows us to rewrite them in spherical coordinates

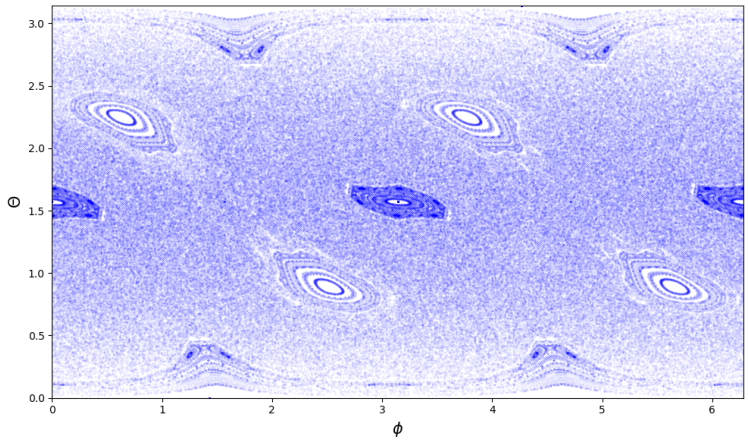
$$X = \sin\theta\cos\phi\,,\; Y = \sin\theta\sin\phi\,,\; Z = \cos\theta\,. \tag{2.7}$$

These angles are classical counterparts of the angles θ and ϕ of the coherent states Eq.2.4. The coherent states become highly localized at these phase space angles, approximating the classical angular momentum state (θ, ϕ) at the limit $j \to \infty$.

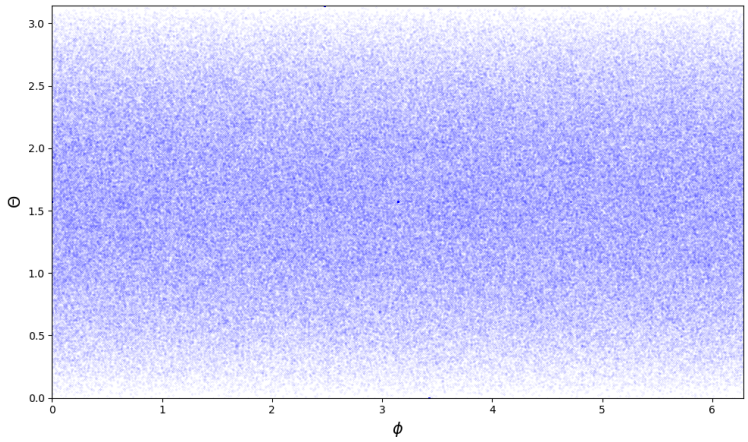
This system shows chaotic dynamics for increasing kick strengths and, hence, is capable of showing sufficient non-linearity to be employed in reservoir computing. The dynamics of the kicked top is completely chaotic for $\kappa > 4.4$ [16]. Phase space dynamics of the kicked top for different κ values are shown in Fig.2.1.



(a)



(b)



(c)

Figure 2.1: Phase space of classical kicked top for values $p = \pi/2$ and (a) $\kappa = 1$, (b) $\kappa = 3$, and (c) $\kappa = 5$ using 1800 initial points which are evolved for 200 time-steps. Notice how the regular regions shrink with increasing kick strength until all that is left is chaotic trajectories for all initial points.

2.2 Simulating The Quantum Kicked Top

The simulation of the dynamics of QKT follows several key steps:

1. **Constructing Angular Momentum Operators**
   In quantum mechanics, for a given spin quantum number j, the system has $n = 2j+1$ states, labeled by the eigenvalues m of J_z, where $m \in \{-j, -j+1, ..., j-1, j\}$. The raising and lowering operators are defined as:

$$J_+|j,m\rangle = \hbar\sqrt{(j-m)(j+m+1)}|j,m+1\rangle, \tag{2.8}$$

$$J_-|j,m\rangle = \hbar\sqrt{(j+m)(j-m+1)}|j,m-1\rangle. \tag{2.9}$$

   We can now construct $n \times n$ matrices corresponding to operators J_+ and J_-, in accordance with Eq.2.8 and Eq.2.9. Using these, the Cartesian components are given by:

$$J_x = \frac{J_+ + J_-}{2}, \quad J_y = \frac{J_+ - J_-}{2i}, \quad J_z = \text{diag}(j, j-1, ..., -j). \tag{2.10}$$

2. **Constructing the Floquet Unitary Operator**
   Now, using the angular momentum operators, we can construct the Floquet Unitary in two parts:

   - A rotation about the y -axis by angle p:

$$U_1 = e^{-ipJ_y} \tag{2.11}$$

   - A non-linear kick along J_z:

$$U_2 = e^{-i\frac{k}{2j}J_z^2} \tag{2.12}$$

   Thus, the full one-step evolution operator is:

$$U = U_1 U_2 = e^{-ipJ_y} e^{-i\frac{k}{2j}J_z^2}. \tag{2.13}$$

   The matrices U_1 and U_2 can be easily generated using matrix exponentiation functions.

3. **Constructing initial state**
   We generally take one of the coherent states $|\theta, \phi\rangle$ as our system's initial state ψ_0. For this

we start with $|j,j\rangle$ state which is given by a vector of dimension $n = 2j+1$ with last entry 1 and all others 0:

$$|j,j\rangle = [0,0,\ldots,0,1]^T. \tag{2.14}$$

Then we construct our desired coherent state $|\theta,\phi\rangle$, using the relation from Eq.2.4. For this, we generate the rotation operator (Eq.2.3) using again the matrix exponentiation functions.

4. **Time Evolution of the System**
   Starting with an initial state vector ψ_0, the evolution follows:

$$|\psi_{t+1}\rangle = U|\psi_t\rangle. \tag{2.15}$$

   This iteration is performed for a given number of time steps to evolve our quantum kicked top.

5. **Computing Expectation Values**
   To analyze the quantum state, we compute expectation values of J_x, J_y, and J_z in the normalized form:

$$\langle J_x\rangle = \frac{\langle\psi|J_x|\psi\rangle}{j}, \quad \langle J_y\rangle = \frac{\langle\psi|J_y|\psi\rangle}{j}, \quad \langle J_z\rangle = \frac{\langle\psi|J_z|\psi\rangle}{j}, \tag{2.16}$$

   where $|\psi\rangle$ is the current state of the system. We store these expectation values at every time step as they describe the system's evolution.

This scheme will help simulate and study quantum phenomena like quantum tunneling and help identify signatures of chaos in quantum systems.

2.3 Simulating The Quantum Kicked Top using a system of interacting qubits

The quantum kicked top can be modeled as a system of interconnected qubits with interaction strength between any two proportional to the kick strength parameter κ [17]. We are interested in this because it also paves the way for experimental implementation. In practice, we might not have a single system of such large spins as required for the quantum kicked top, but creating a network of qubits to simulate a larger spin system is readily possible.

A kicked top of spin size j can be written as a system of $N = 2j$ qubits [18]. The angular momentum operators for the system can be written in terms of Pauli operators:

$$J_\alpha = \frac{1}{2}\sum_{i=1}^{2j}\sigma_\alpha^{(i)}, \quad \alpha \in \{x,y,z\}, \tag{2.17}$$

where $\sigma_\alpha^{(i)}$ denote σ_α acting on i_{th} qubit. Now, we can write QKT Hamiltonian as follows:

$$H = \hbar\frac{p}{2\tau}\sum_{i=1}^{2j}\sigma_y^{(i)} + \hbar\frac{\kappa}{8j}\left(2j + \sum_{\substack{i,k=1\\ i\neq k}}^{2j}\sigma_z^{(i)}\otimes\sigma_z^{(k)}\right)\left(\sum_{n=-\infty}^{\infty}\delta(t-n\tau)\right). \tag{2.18}$$

In this study, we mostly take QKT of spin size $j = 3/2$, which would need three qubits for simulation. Using this in the above equation and simplifying, we now have

$$\begin{aligned} H =& \frac{\hbar\kappa}{12}\left\{3\mathbb{I} + 2(\sigma_z^{(1)}\otimes\sigma_z^{(2)} + \sigma_z^{(1)}\otimes\sigma_z^{(3)} + \sigma_z^{(2)}\otimes\sigma_z^{(3)})\right\}\sum_n\delta(t-n\tau) \\ &+ \frac{\hbar p}{2\tau}(\sigma_y^{(1)} + \sigma_y^{(2)} + \sigma_y^{(3)}). \end{aligned} \tag{2.19}$$

Deriving now the Floquet unitary from this Hamiltonian expression and simplifying, we have

$$U = \exp\left(-i\frac{\kappa}{6}(\sigma_z^{(1)}\otimes\sigma_z^{(2)} + \sigma_z^{(1)}\otimes\sigma_z^{(3)} + \sigma_z^{(2)}\otimes\sigma_z^{(3)})\right)\exp\left(-i\frac{p}{2}(\sigma_y^{(1)} + \sigma_y^{(2)} + \sigma_y^{(3)})\right) \tag{2.20}$$

where we took $\hbar = 1$. In this expression, terms of the form $\exp(-i\frac{p}{2}\sigma_y)$ are simple y-axis rotation operations on the respective qubits by angle p. The other terms of form $\exp(-i\frac{\kappa}{6}\sigma_z\otimes\sigma_z)$ are less obvious. This is actually the form of R_{zz} gate [19]

$$R_{zz}(\theta) = \exp\left(-i\frac{\theta}{2}\sigma_z\otimes\sigma_z\right). \tag{2.21}$$

It can be decomposed in terms of two *CNOT* and one R_z rotation gate as shown in Fig.2.2:

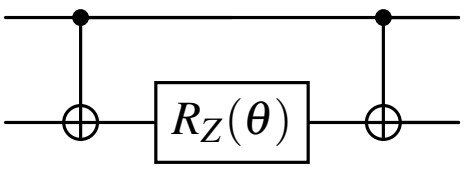


Figure 2.2: Decomposition of R_{zz} gate

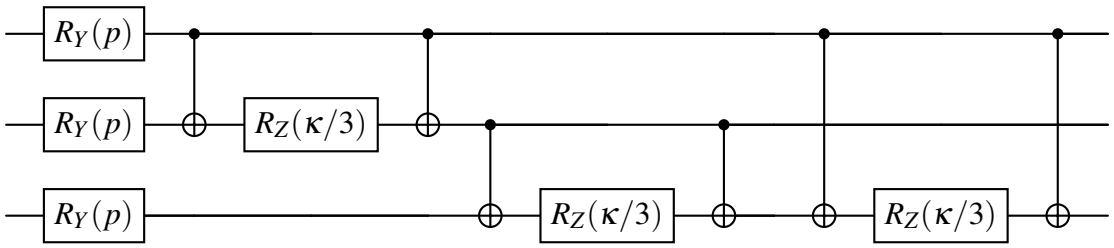


Figure 2.3: Quantum Circuit representation of the Floquet Unitary U for a quantum kicked top simulated using a system of 3 qubits.

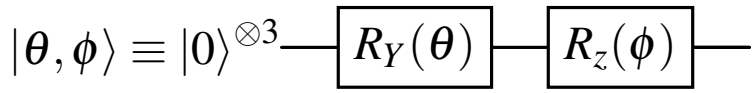


Figure 2.4: Preparation of Initial Coherent State for a quantum kicked top simulated using a system of 3 qubits.

Hence, the action of the unitary in Eq.2.20 can be summarized with the help of the quantum circuit given in Fig.2.3.

To prepare the initial state $|\theta,\phi\rangle$, we start with all qubits in $|0\rangle$ state and apply a rotation of angle θ about the y-axis and rotation of angle ϕ about the z-axis to each qubit (Fig.2.4).

Having prepared the initial state and the Floquet unitary, the evolution of the system would mean repeated application of the set of gates in Fig.2.3 describing the Floquet operator U. The expectation values of desired observables can then be monitored in a manner similar to what was described in the previous section.

This scheme of simulating the quantum kicked top using qubits can be similarly expanded to more than three qubits as well.

2.4 Scheme for Reservoir Computing using Quantum Kicked Top

Once we have a working model of our QKT, we can use it for the purpose of doing reservoir computing. We need to choose the size of the system appropriate for the given task and the kind of inputs we need to provide to the reservoir. We have used QKT of spin size $j = 3/2$ for tasks with classical inputs and spin size $j = 2$ for quantum data. The scheme is as follows:

1. First, the input is encoded in our QKT reservoir. We have done input encoding in the kick-strength parameter κ for most tasks as it provides good non-linearity to the reservoir. Appropriately scaling the inputs and choosing the range of κ for encoding is crucial in this step for optimum results.

2. Once the encoding is done, the system is evolved according to Floquet Unitary U. The expectation value of one of the angular momentum operators is measured after every time-step. We measure the J_x operator's expectation for all the tasks in this paper, which generates the reservoir state corresponding to the input signal.

3. The process of training and testing the QKT reservoir is now the same as any reservoir described in section 1.1

4. After every input, we reset the reservoir again to the initial state before encoding the next one.

Chapter 3

QKT Reservoir Computing with Classical Data

In this chapter, we test the performance of our QKT Reservoir Computer in performing various tasks that demonstrate its ability to handle classical data and its processing. We perform several non-trivial tasks like polynomial regression, prediction, and extrapolation of Lorenz series data, which are considered benchmarks for testing the reservoir's performance [22, 23].

3.1 Regression Task

The purpose of this task is to learn a seven-degree polynomial:

$$f(x) = (x+3)(x+2)(x+1)(x)(x-1)(x-2)(x-3), \tag{3.1}$$

such that reservoir can predict $f(x)\ \forall\ x \in [-3,3]$. This task tests how efficiently input is being processed inside the reservoir, which brings about sufficient non-linear transformation of the input to predict a high-order polynomial.

We used encoding in the kick strength (κ). We scaled our input data to the range $\kappa \in [0, 5.5]$. Reservoir states are obtained from the expectation value of the J_x operator, time multiplexed by running the reservoir for 32 time-steps. That means each reservoir state vector has 32 stored

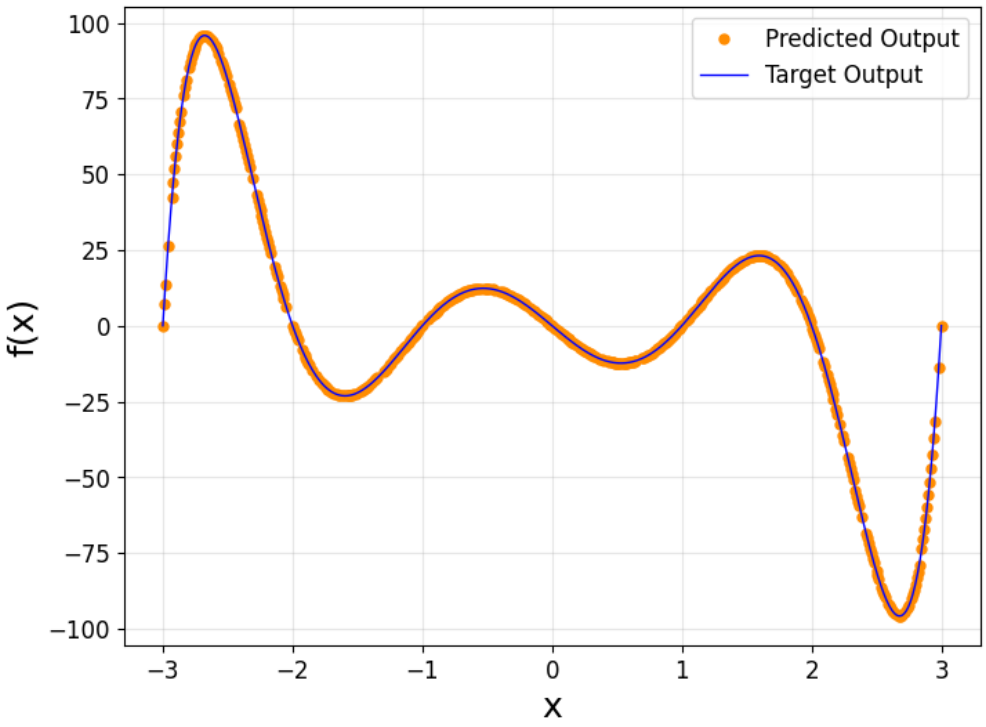


Figure 3.1: Seven-degree polynomial regression results from QKT reservoir ran for 32 time-steps on 450 testing points. It was trained with 150 training data points. This plot has an RMSE of order 10^{-5}.

expectation values generated at subsequent time steps for each input data.

The average root mean square error (RMSE) for the predicted data was obtained in the order of 10^{-6} from training data of length 500 randomly sampled from $x \in [-3, 3]$ from the total data length 600. The remaining is used for testing. Reducing the size of the training to only 25% of the total data (150 points) still keeps the RMSE of the predicted data (450 points) as low as 10^{-5} Fig.3.1.

Input encoding can also be done in the initial state of the reservoir. But in our case, it required expectation values of two operators (Jx and Jy) at every step for each input to give RMSE of comparable order as kick strength encoding, making it more resource-intensive.

We also performed a phase space analysis on the initial state of the kicked top to see how it influences the accuracy of the regression task Fig.3.2. The visual symmetry in this plot is to be noted, which traces back to symmetries present in the kicked-top map.

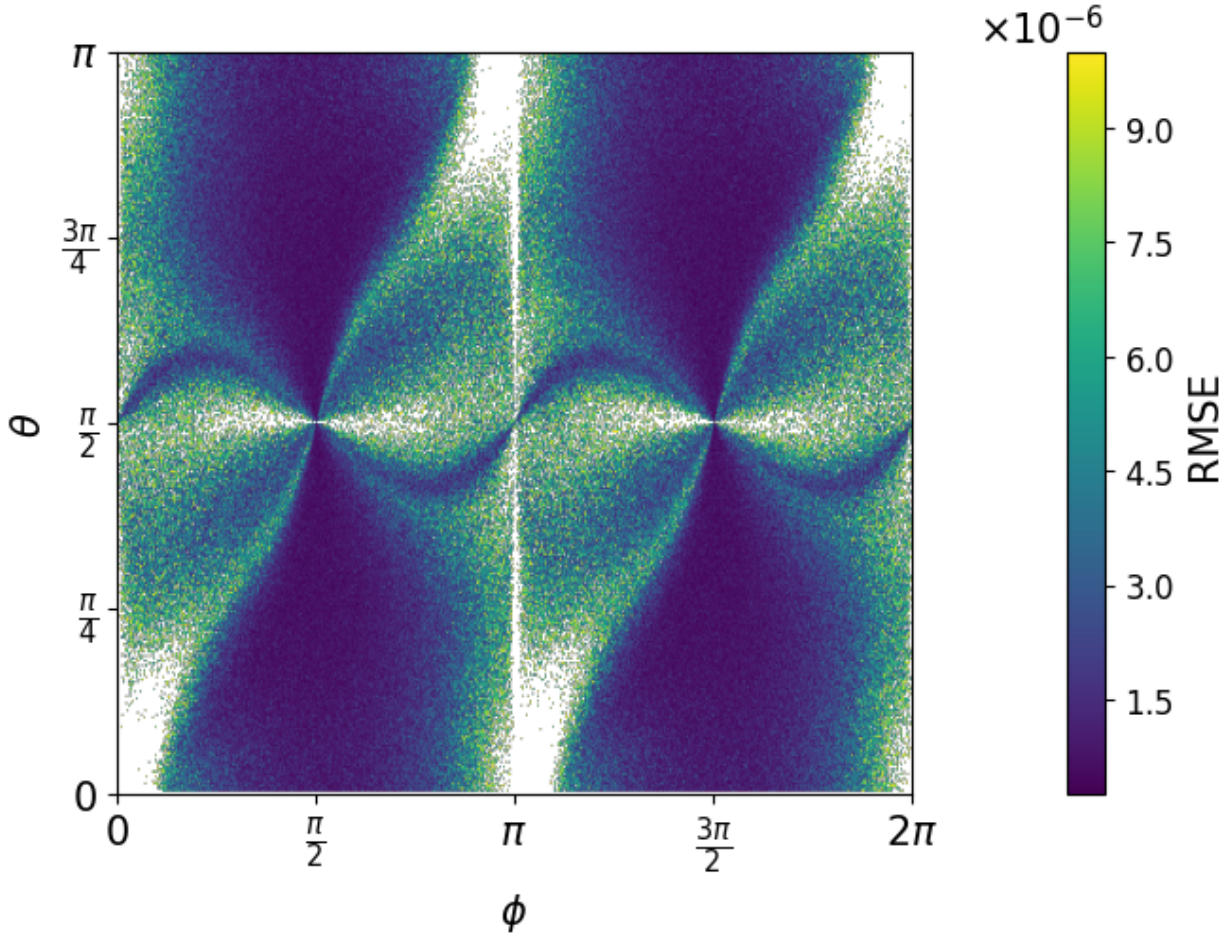


Figure 3.2: Phase space search on the initial state of the QKT for RMSE analysis. This plot corresponds to kick strength encoding in $\kappa \in [0, 5.5]$ with training on 500 data points. In this plot, points that correspond to RMSE value $> 10^{-5}$ (white region) are removed for better readability. Minimum RMSE was obtained at point $(\theta, \phi) \equiv (2.36, 4.89)$.

To understand how the introduction of chaos can change the performance of our tasks, we also did phase space analysis on the initial state with different ranges of kick strength parameters. These ranges corresponds to regular - $\kappa \in [1, 2]$ - Fig.3.3a, boundary - $\kappa \in [3.5, 4.5]$ - Fig.3.3b and complete chaotic - $\kappa \in [6, 7]$ - Fig.3.3c regime of classical kicked top. These figures tell about the effect of performance on changing the reservoir dynamics. More analysis is needed to understand the behavior shown here.

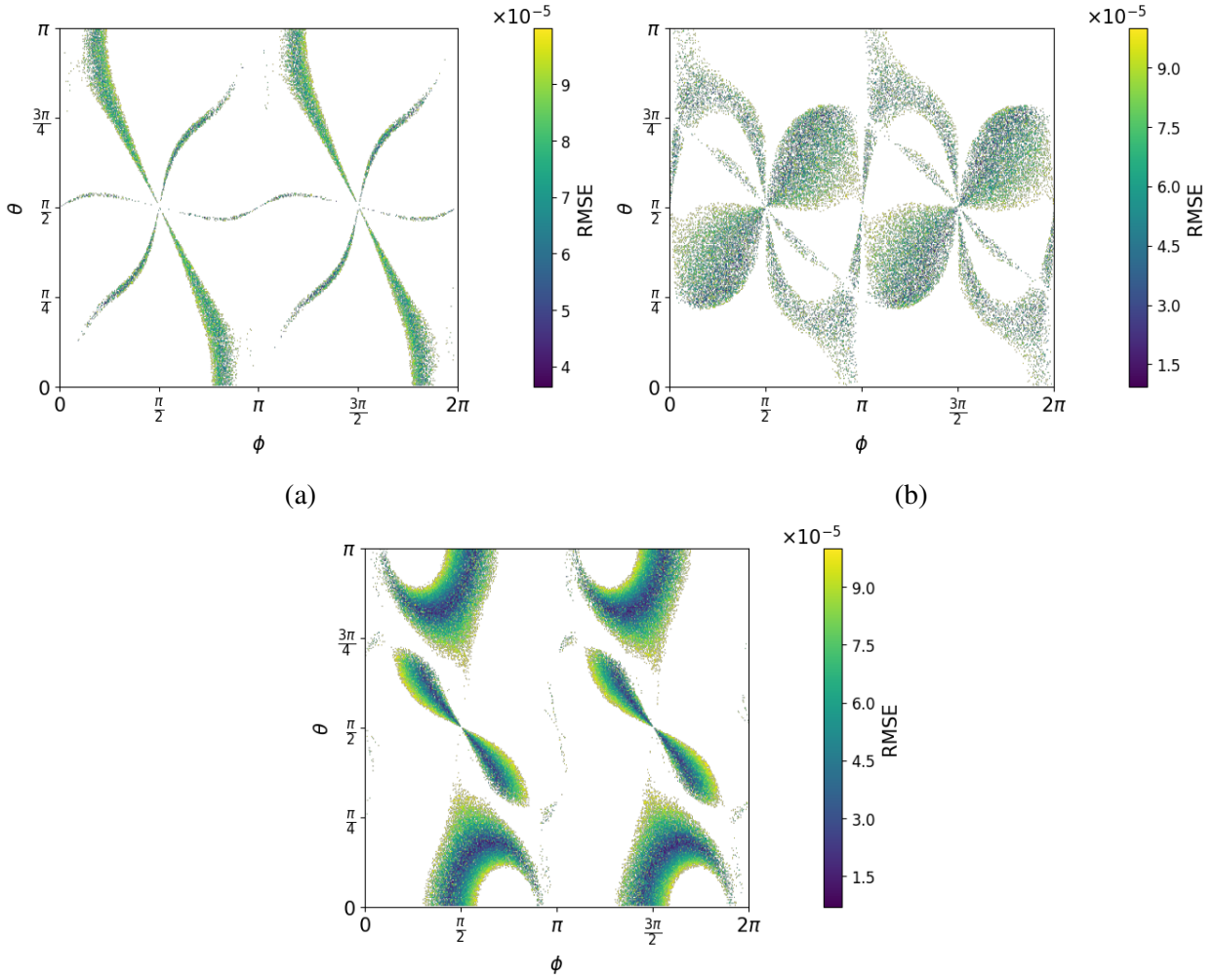


Figure 3.3: Phase space search on the initial state of the QKT for RMSE analysis for varied ranges of input encoding in kick strength parameter: (a) regular ($\kappa \in [1,2]$), (b) boundary ($\kappa \in [3.5,4.5]$) and (c) complete chaotic ($\kappa \in [6,7]$) regime of the classical kicked top. All of them were trained on 500 randomly sampled data points. In these plots, points that correspond to RMSE value $> 10^{-4}$ (white region) are removed for better readability.

3.2 Lorenz Series Prediction

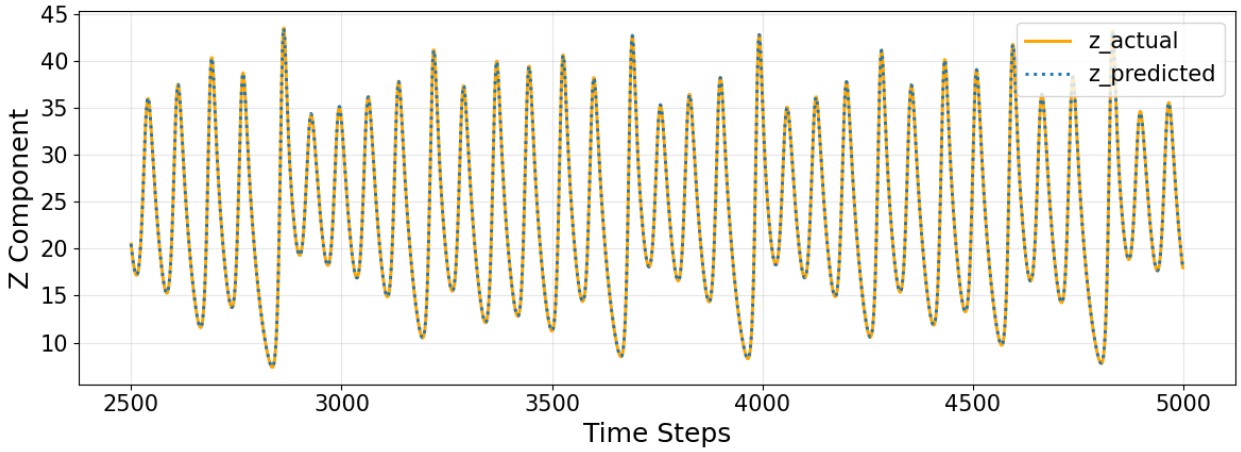


Figure 3.4: Lorenz Series X-Z Prediction task. This plot shows the prediction of the Z component of the Lorenz series given the X component for testing data. Our QKT reservoir has learned the relationship between the two variables, X and Z, of the Lorenz series with the help of training data of length 2500.

The Lorenz attractor is given by the following set of differential equations [20, 21]

$$\frac{dx}{dt} = \sigma(y-x) \tag{3.2}$$

$$\frac{dy}{dt} = x(\rho-z)-y \tag{3.3}$$

$$\frac{dz}{dt} = xy-\beta z \tag{3.4}$$

It shows chaotic behavior for some parameter values, meaning a tiny change in the initial conditions leads to drastically different trajectories. We take $\sigma = 10$, $\rho = 28$ and $\beta = 8/3$. This task aims to predict the z component of the Lorenz series, given the x component consecutively in time.

This task requires memory in the reservoir, i.e., given a reservoir consequently fed an input time-series and evolved for specific time steps after every input without resetting back, it should be able to identify past inputs up to a certain time back. The "memory" of a quantum kicked top refers to the system's ability to retain and exhibit signatures of its past dynamics. We quantify the memory based on how far back it can recall the input with some threshold accuracy. Our study found that the inherent memory of the QKT reservoir is not as much as we require to perform these tasks, so we used artificial memory.

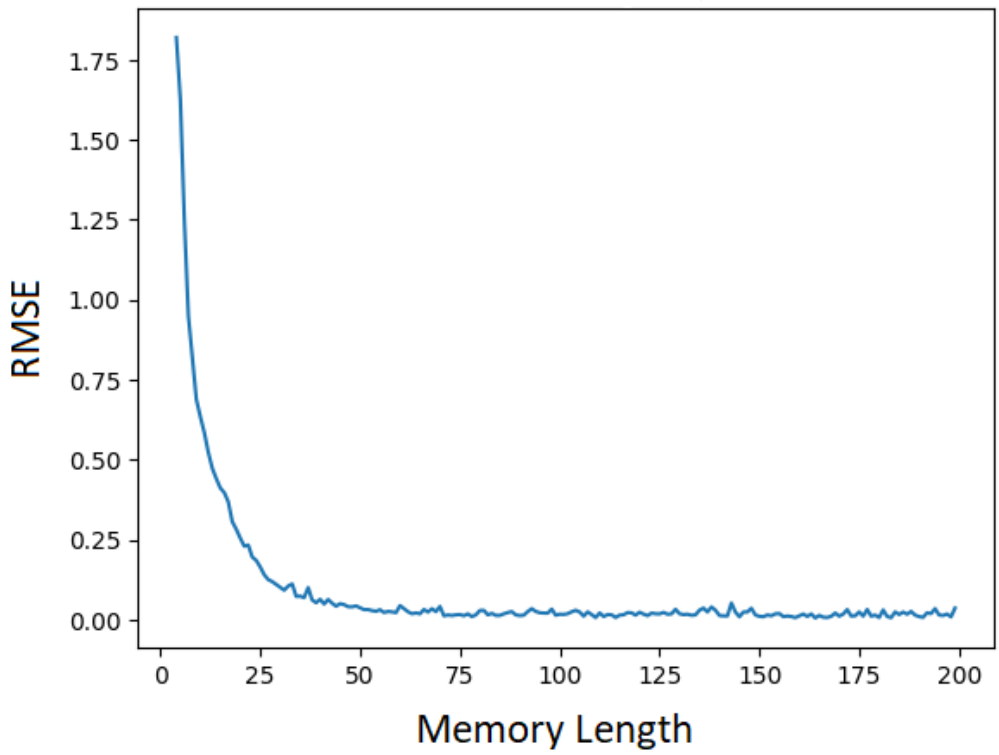


Figure 3.5: Dependence of RMSE error on the length of artificial memory for Lorenz series prediction task.

Artificial memory: For each reservoir state S_n corresponding to the n_{th} input of the given data, previous states corresponding to earlier inputs are also added according to the value of the artificial memory given by d. These previous inputs are reduced by a factor $w_i \in [0,1]$ linearly proportional to their distance from the current input, which gives us the reservoir state for n_{th} input as $r_n = [w_d S_{n-d}, w_{d-1} S_{n-d-1}, \ldots, w_1 S_{n-1}, w_0 S_n]^T$. This scheme of adding memory by hand is described in [24].

For this task, we used an artificial memory of 78 units, and the initial state was $(\theta, \phi) \equiv (2.22, 4.73)$. The encoding was performed using the kick strength parameter in the range $\kappa \in [3.00, 3.33]$, and the reservoir was run for 10 time-steps, recording the Jx expectation values. RMSE value of the order of 10^{-2} is obtained from the predicted data where the training data had a length of 2500 Fig.3.4. An interesting thing is that the RMSE order here is independent of the testing data length once the reservoir is adequately trained.

Fig.3.5 shows the performance of the reservoir with increasing memory. The accuracy of prediction

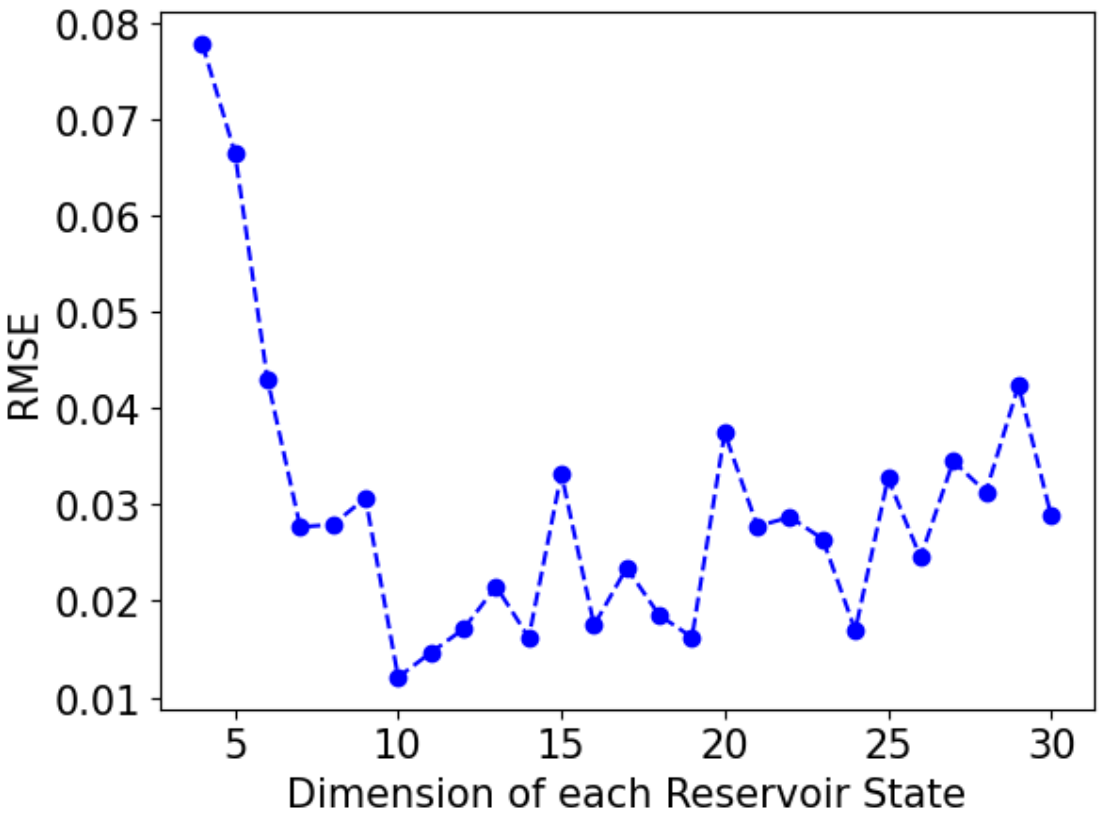


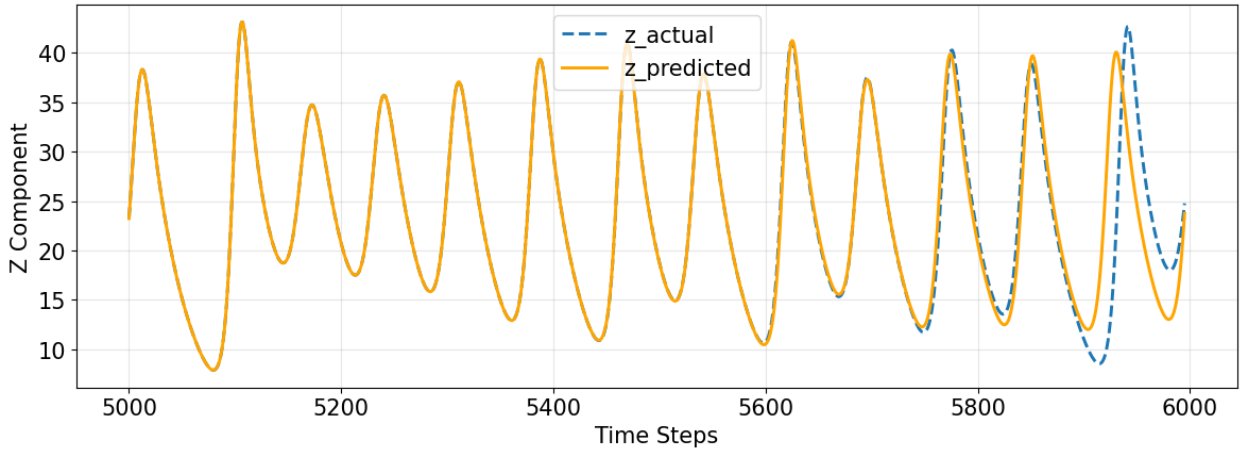
Figure 3.6: RMSE trend for increasing reservoir state dimension corresponding to each input for Lorenz series prediction task

quickly improves with memory, and after about 70 units of memory, the trend is flat-lined on average. Note that the memory requirement also depends on the encoding type, the reservoir states' length, and other variables, so this curve is not general and only represents the dependence.

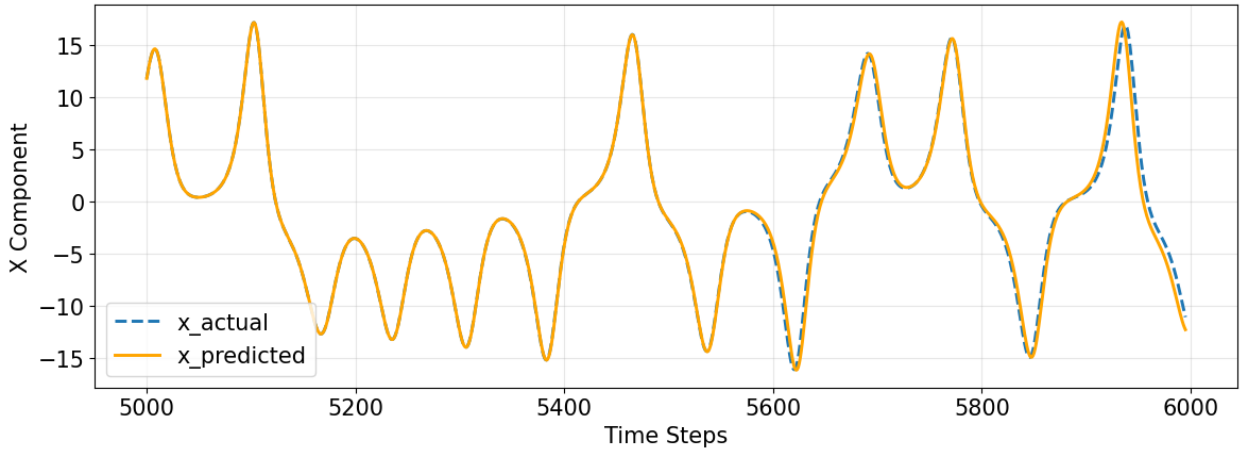
In Fig.3.6, we see how the RMSE changes with a change in the dimension of the reservoir state. We noticed that after time-multiplexing of 5 units, the prediction accuracy does not show any fixed trend and is about the other of $10^{-}2$ on average.

Successful implementation of this task shows the capability of our QKT Reservoir to understand the complex relationship between two interdependent variables of differential equations and, hence, predict one given other.

3.3 Lorenz Series Extrapolation



(a)



(b)

Figure 3.7: Lorenz series extrapolation task where (a) shows the extrapolation of the Z component and (b) shows the extrapolation of the X component by the QKT reservoir.

In this task, we put our reservoir's learnability and predictive power to the ultimate test by trying to extrapolate the evolution of the Lorenz series with time. This means that for input data $x(t)$, the target output is $x(t+1)$. We demonstrate it with the extrapolation of the x and z components.

An artificial memory of 64 units was provided for this task, and the encoding was done in kick-strength in the range $\kappa \in [3.6, 3.8]$. We started with initial state $(\theta, \phi) \equiv (2.36, 4.89)$. The reservoir was run for 137 time-steps, making each reservoir state 137 units long, which consists of J_X expectation values at all the time-steps. Training data of length 5000 was used to train the reservoir with

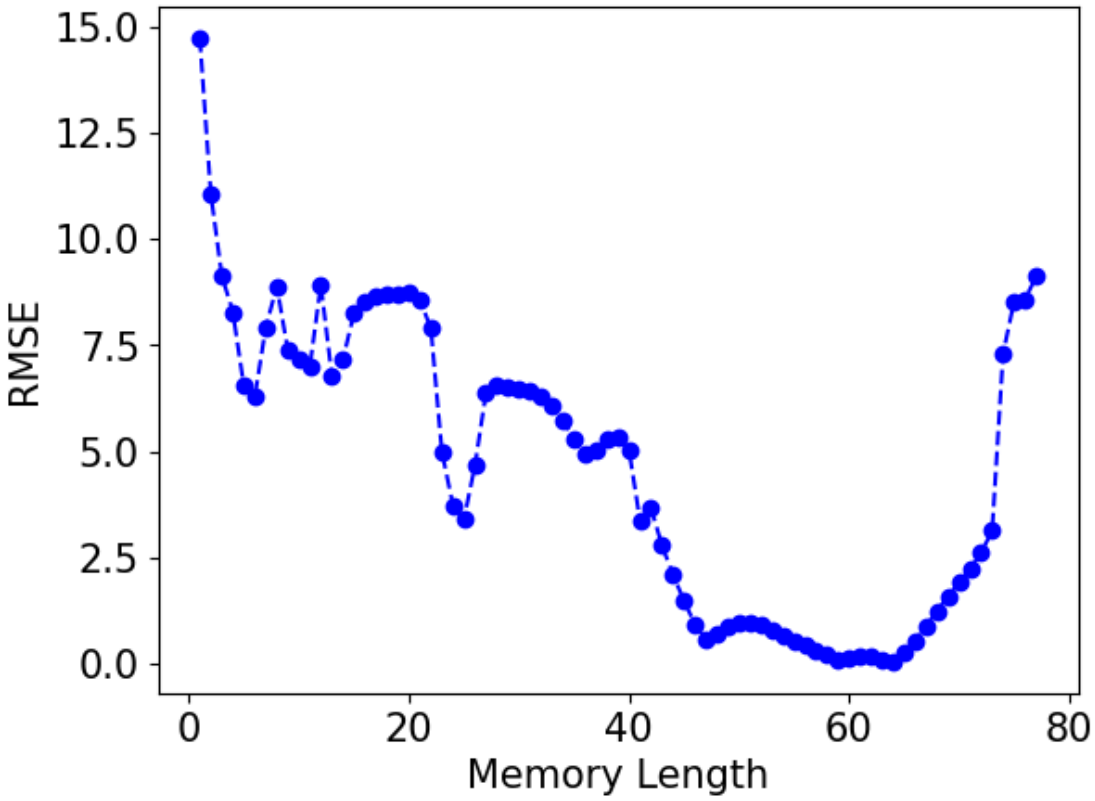


Figure 3.8: Dependence of performance of the QKT Reservoir on the amount of artificial memory provided for extrapolation task of X-component of the Lorenz series. Performance is quantified on RMSE of the first 500 testing data points. Minima is reached at the memory of 64 units.

testing data of length 1000. The reservoir could predict well for length ≈ 800, slowly diverging from the trajectory afterward. Fig.3.7 shows the final result for both the X and Z components.

In Fig.3.8, we plot the relationship between the performance of the reservoir for this task and the amount of artificial memory. We can see that it's not a monotonically decreasing trend; instead, we obtain minima in the middle at 64 units. This suggests that in the Lorenz system, previous data points of only up to a certain number influence the current data point; memory greater than this number would hinder the prediction and thus negatively influence the performance.

The relationship of performance of the QKT reservoir with the dimension of reservoir states for this task Fig.3.9 follows a similar trend as the memory length. There is a sharp minima at 137 units, having RMSE of order 10^{-2}, while all the nearby points have RMSE values of order 10^{-1}

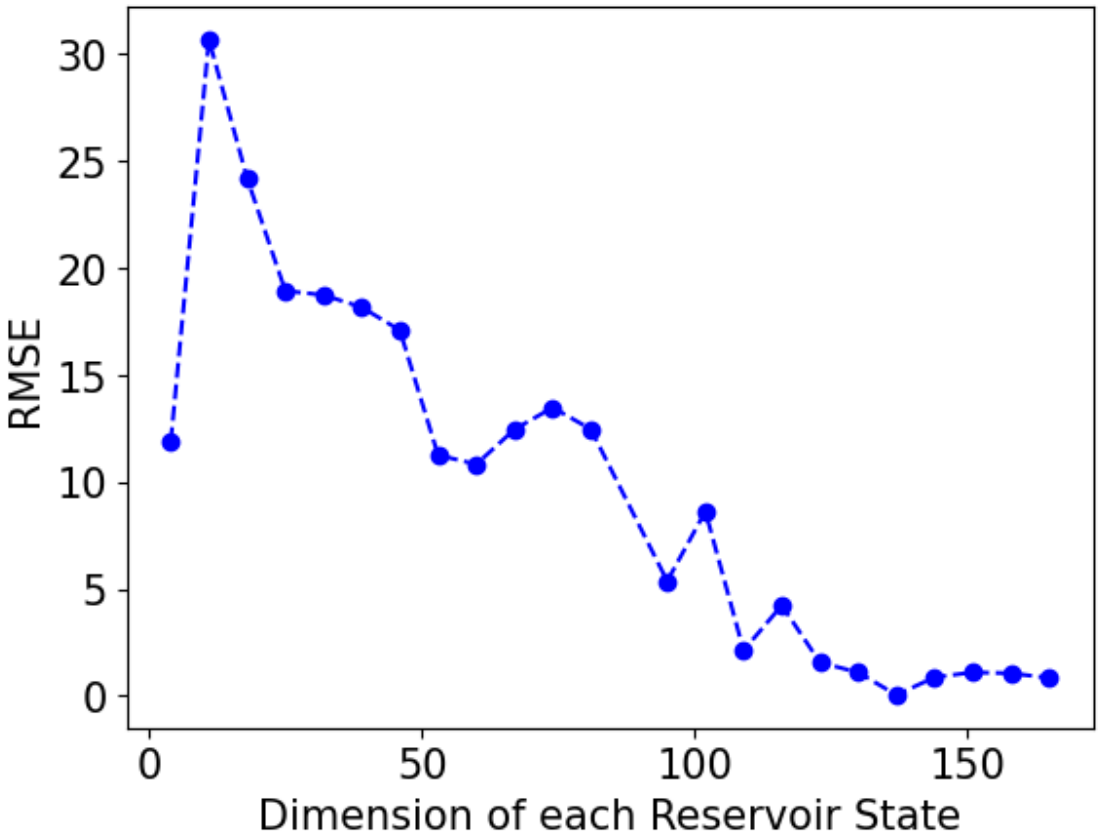


Figure 3.9: Impact of the reservoir state dimension (multiplexing length) on the performance of the Lorenz series extrapolation task. This plot is created for X-component extrapolation, and RMSE is calculated for the first 500 testing data points to quantify the performance. We obtain a minima at length of 137 units.

or much more.

With this task, we have shown the capability of the QKT reservoir computer to learn complex dynamics and identify patterns and relationships of the data points using previous data. It successfully takes advantage of the inherent non-linear transformations of the current input as well as previous ones to predict the next data point of the given series.

Chapter 4

Quantum Tasks with QKT Reservoir Computing

In this chapter, we will study one of the most apparent advantages of Quantum Reservoir Computing in carrying out quantum tasks. A classical reservoir can handle the input data in classical form, implying that to perform any quantum task, we would first need to convert our quantum data in the form of quantum states to classical data. This process is both resource-intensive and time-consuming. But unlike the classical reservoir, a quantum reservoir, like a quantum kicked top, has no such limitation. We can, in principle, encode any of our quantum states in it by simulating the quantum kicked top in a required number of qubits and encoding our quantum state in those qubits.

We have performed two quantum tasks using our QKT Reservoir - Entanglement classification and prediction of Logarithmic-Negativity of a given entangled state. We intend to perform experimental simulations of these tasks using the NMR system. Hence, we generate the quantum states needed for this task using the scheme typically used in NMR experiments to generate them, making it more feasible and easier to carry out the experimental implementation.

4.1 Generating the Quantum State Inputs

In order to perform both tasks, we take the input states to be bipartite mixed quantum states ρ, which are generated by a unitary transformation U_c on the thermal state ρ_0 as shown below,

$$\rho = U_c \rho_0 U_c^{\dagger} \tag{4.1}$$

where

$$\rho_0 = \frac{1}{4}\exp(-\beta H_z), \tag{4.2}$$

$H_z = -\frac{\omega}{2}(\sigma_z^{(1)} + \sigma_z^{(2)})$ and we took inverse temperature $\beta = 1.3$ in units of $\hbar = 1$. ω is the Larmor frequency, which we take as unity for the purpose of generating our states. The unitary operator U_c represents a CNOT-like operation that includes single-qubit rotations and an interaction-driven evolution. In NMR quantum computing, such operations are implemented using radio-frequency (RF) pulses and the intrinsic spin-spin coupling interactions [25].

The unitary operator U is constructed step by step using the following operations:

1. **Single-Qubit Rotation Around Y-axis (P_1)**

$$P_1 = \exp\left(-i\theta\frac{\sigma_y \otimes I}{2}\right) \tag{4.3}$$

   This is a local rotation applied to the first qubit. In NMR, this corresponds to an RF pulse along the Y-axis of the Bloch sphere, which rotates the nuclear spin by the angle θ.

2. **CNOT-like Operation (P_2)**

$$P_2 = \exp\left(-i\frac{\pi}{2}\frac{I \otimes \sigma_x}{2}\right) \tag{4.4}$$

   This is a single-qubit Pauli-X rotation on the second qubit. In NMR, this corresponds to an RF pulse along the X-axis for the second nuclear spin.

3. **Interaction Evolution for CNOT (U_{int})**

$$U_{\text{int}} = \exp\left(-i\frac{H}{2J}\frac{\alpha}{(\pi/2)}\right) \tag{4.5}$$

where the Hamiltonian governing the interaction is given by:

$$H = 2\pi J \frac{\sigma_z \otimes \sigma_z}{4}. \tag{4.6}$$

The system evolves under the Ising-type spin-spin interaction through coupling strength J. In NMR, this corresponds to natural evolution under the J-coupling interaction, which is usually controlled by adjusting the delay time between the pulses.

4. **Rotation Around Y-axis of Second Qubit (P_3)**

$$P_3 = \exp\left(-i\frac{\pi}{2}\frac{I \otimes \sigma_y}{2}\right) \tag{4.7}$$

This is another single-qubit Y rotation applied to the second qubit.

5. **Final Unitary U_c**
The complete unitary is constructed as follows:

$$U_c = (P_3 \cdot U_{\text{int}} \cdot P_2) \cdot P_1 \tag{4.8}$$

$U'(\alpha) = (P_3 \cdot U_{\text{int}} \cdot P_2)$ is effectively a controlled operation resembling a CNOT-like gate and P_1 is an additional single-qubit rotation gate.

The unitary U_c has the form as shown in Fig.4.1.

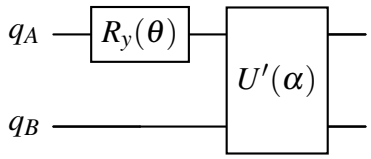


Figure 4.1: Unitary U_c parameterized by θ and α

4.2 Reservoir Computing using Quantum Inputs

Now, encoding these quantum states in the reservoir is where the advantage of using a quantum reservoir becomes obvious. We encode the states to be used as input directly in the initial state of

our quantum kicked top without the need to know the state beforehand. The scheme for performing the quantum tasks using the QKT reservoir is as follows:

1. Use a system of 4 qubits to simulate our quantum kicked top of spin $j = 2$ as described in Section 2.3.
2. The qubits are then initialized in the thermal state $\rho_0 \otimes \rho_0$ Eqs.4.2.
3. Now, by performing the given unitary operation (Eq.4.8) on the two qubits at a time, we obtain our initial state to be $\rho \otimes \rho$, where ρ is a quantum state of two qubits that needs to be worked with, as given in Eq.4.1.
4. To achieve the task at hand, we evolve this state under the Floquet Unitary of the quantum kicked top (Eq.2.20) and follow the same scheme of reservoir computing of Section 2.4, that we have been using throughout the previous chapter.

4.3 Entanglement Classification Task

In this task, we classify whether or not the given state ρ of Eq.4.1 is entangled. The entanglement is determined using Logarithmic-Negativity [26], given by

$$E_N(\rho) = \log_2(2\mathcal{N} + 1), \tag{4.9}$$

where $\mathcal{N}$ is Negativity with respect to one of the subsystem of ρ, say qubit A, and it is defined as,

$$\mathcal{N} = \sum_{\lambda_i < 0} \lambda_i \tag{4.10}$$

where λ_i are the eigenvalues of the matrix ρ^{Γ_A}, which is the partial transpose of ρ with respect to the subsystem A. If $E_N(\rho) > 0$, the state is labeled as entangled. Otherwise, it is labeled as a non-entangled state.

This task of classifying states as entangled or not is a non-trivial problem as determined in [27]. This problem is also tackled in [28] using the quantum kernel method. Here, we show it using QKT reservoir computing.

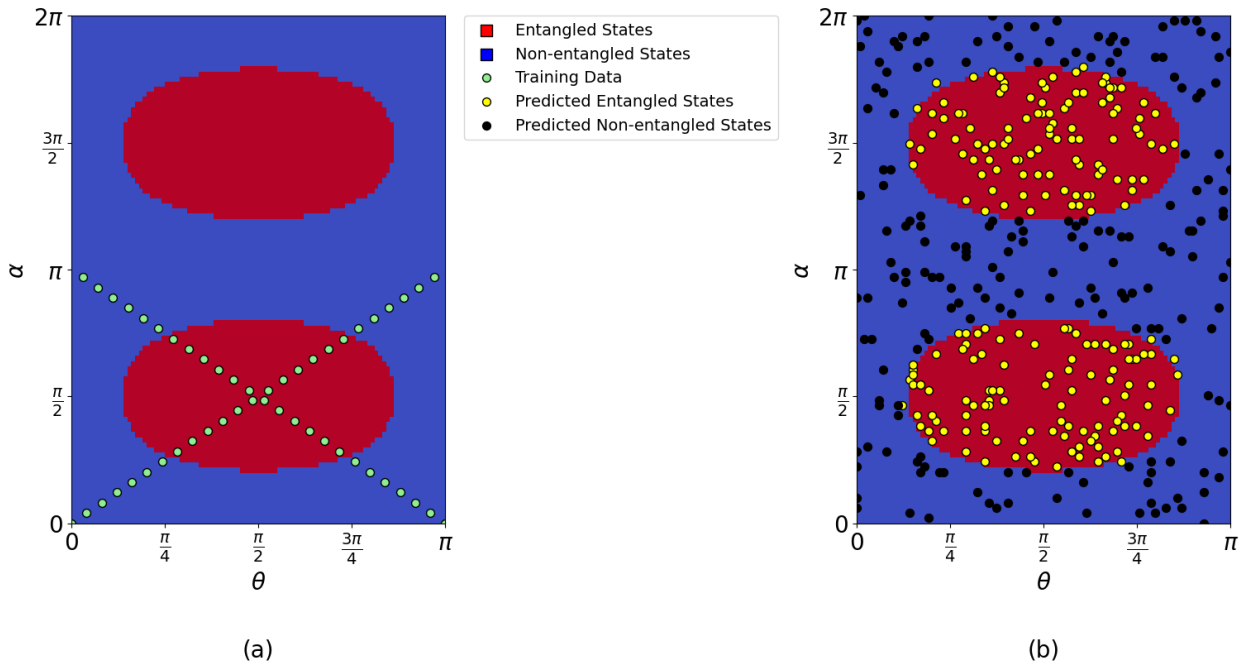


Figure 4.2: Results of entanglement classification task where (a) shows the points used for training data and (b) shows the predicted entanglement values. The red and blue regions in the background correspond to actual labels of entangled and non-entangled states, respectively.

We first initialize the quantum states using the scheme of Section 4.2. We then monitor the J_x expectation for nine time-steps to generate our reservoir state. In training, we then map the reservoir states corresponding to entangled states to 1 and those corresponding to non-entangled states to 0, known from calculating Logarithmic Negativity for each. Now, we check the performance of the reservoir using the unseen testing quantum states. If the output labels corresponding to these states are greater than 0.42 (obtained via optimization), we label those states 1, implying predicted entangled states; otherwise, the states are labeled as 0, meaning predicted non-entangled states. We then compare it with the actual labels of entanglement obtained from Logarithmic Negativity and calculate the accuracy of the classification.

The states ρ are actually generated at a very low temperature (we took $\beta = 1.3$ in Eq.4.2), which is not attainable in a standard NMR lab. It is possible to still classify these states by just performing the same unitaries on the thermal state at room temperature. The reservoir still learns the features from the training data and uses it to determine if the test states would have been entangled or not if they were generated at those lower temperatures. So, in a way, we are classifying the entangling capabilities of the unitary transformation of Eq.4.8 at a different temperature (corresponding to $\beta = 1.3$) rather than the entanglement of a quantum state at the current temperature.

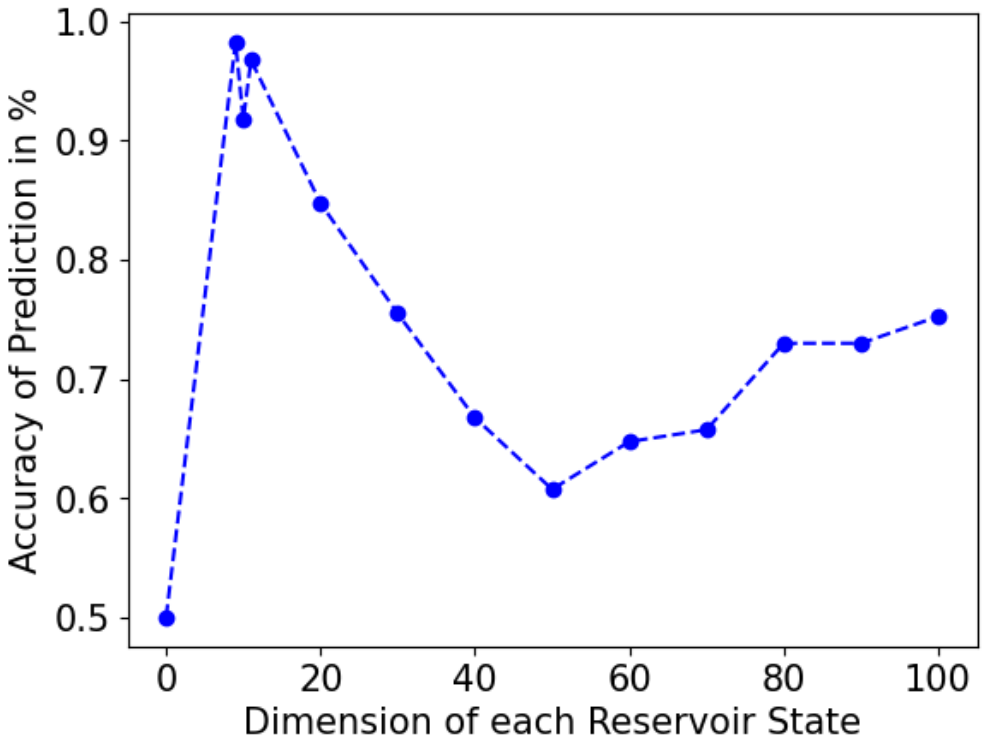


Figure 4.3: Accuracy of prediction of entanglement labels (0 or 1) with the size of the reservoir state.

The set of states to be classified is generated by changing the parameters $\theta \in [0, \pi]$ and $\alpha \in [0, 2\pi]$ of the unitary of Eq.4.8. We generate these states at room temperature ($\beta = 1.6/10^{13}$) but use the entanglement labels from a much lower temperature described by the $\beta = 1.3$ value, as explained in the above paragraph. We also use the value of $\omega = 500$Mhz, which corresponds to the more realistic value of the Zeeman magnetic field in the NMR lab. We used a simple training set of 50 states, as shown in Fig.4.2. As evident from the figure, even though we have provided the training data only from the lower part of the plot, still the reservoir is able to extrapolate this information to the upper half as well and classify the entangled state, which is similar to quantum kernel method [28]. We obtained an average classifying accuracy of 97% on test data of size 400, where half of them were randomly sampled from the set of entangled states and the other half from the non-entangled set.

We also plotted the dependency of classification accuracy on the size of the reservoir state taken Fig.4.3. We obtain a maxima at nine units as accuracy falls on either side of this. Because there are only two possible outputs, 0 or 1, the accuracy of 50 percent would mean that the reservoir is

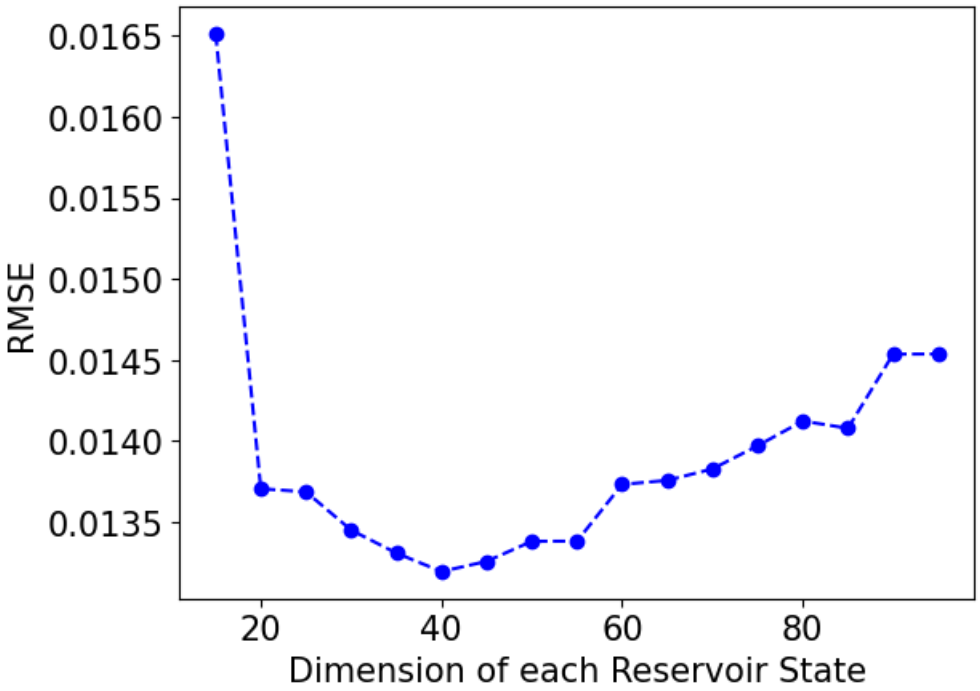


Figure 4.4: Performance of Logarithmic-Negativity prediction task with increasing reservoir state size.

not able to classify at all and is most likely making random guesses.

4.4 Logarithmic-Negativity Prediction

We can use our QKT reservoir computer to help predict various quantities describing a quantum state like Entropy, Purity, Logarithmic-Negativity, etc. We can achieve this in principle without explicitly knowing the quantum state beforehand; only the procedure to generate them would suffice to use them as our quantum inputs in the reservoir, described in Section 4.2.

In this section, we predict the Logarithmic-Negativity values of the entangled quantum states using QKT reservoir computing and check their performance by comparing them with the actual value. Once again, we initialize the quantum state at higher temperatures ($\beta = 1.6/10^{13}$) and take the realistic value of Larmor frequency ω. But still, we take the Logarithmic-Negativity values from the corresponding states generated at lower temperatures ($\beta = 1.3$) and ω equal to unity and use them for training the reservoir. We then evolve the state using QKT Floquet Unitary (Eq.2.20)

for 40 time-steps for optimal results. We keep the kick-strength value κ at 2.8 and the p value of the Floquet Unitary at 3.0. We obtain the average RMSE of order 10^{-2}. This might seem low enough, but values of Logarithmic-Negativity are typically of order 10^{-1} or less. Therefore, the QKT reservoir is only performing decently well. Hopefully, RMSE could be brought down more from further optimization using more sophisticated techniques like Bayesian optimisation of hyperparameters [29]. The relationship between RMSE and the size of the reservoir state is given in Fig.4.4. We obtain a minima somewhere in the middle, so projecting input to an arbitrarily higher dimension is again not beneficial; proper optimization is needed.

Chapter 5

Experimental Scope and Implementation

We intend to simulate our results from Chapters 3 and 4 on a physical NMR system. NMR has proved to be an excellent test-bed to carry out quantum information processing tasks and study quantum correlations [30, 31, 32]. Implementation of Quantum Reservoir Computing using NMR system has already been demonstrated in [12]. The quantum kicked the top of spin $j = 1$ has also been successfully simulated in NMR using a pair of qubits by V. R. Krithika et al. [33] and using three qubits in [34]. We will mostly follow their scheme of simulating the kicked top in NMR and perform reservoir computing on it.

5.1 Simulating QKT in NMR for Reservoir Computing

Simulating QKT in NMR would require us to write its Hamiltonian in the following form

$$H = \frac{\kappa J_z^2}{2j\tau} + pJ_y \sum_{n=-\infty}^{\infty} \delta(t - n\tau), \tag{5.1}$$

where we took $\hbar$ as unity. This form is a bit different from what we have been using. Here, the second term of rotation by angle p about the y-axis involves instantaneous kicks, and the first term describes non-linear evolution between the kicks of interval τ characterized by the chaoticity parameter κ, giving us a non-linear torsion about the z-axis. This Hamiltonian essentially generates the same dynamics as our original Hamiltonian of Eq.2.1 [35], and the corresponding Floquet

unitary is also the same as Eq.2.20.

To perform all the tasks with classical data from Chapter 3, we simulate the QKT of spin-3/2 using a three-qubit NMR system. The suitable molecule for this is dibromofluoromethane which has qubits ^{13}C, ^{1}H and ^{19}F. The sample is dissolved in deuterated acetone. We entirely follow the method described in [34]. The manipulation of the spins can be achieved by using radio frequency (RF) pulses, which are resonant with the corresponding characteristic Larmor frequencies described by the Hamiltonian

$$H_{\mathrm{RF}} = \sum_i \frac{\pi}{2\Delta_i} I_{yi}, \tag{5.2}$$

where Δ_i is the pulse duration given to the i_{th} qubit and I_{yi} is the nuclear spin operator along y-axis. There is also interaction between the qubits and in the weak-coupling limit for three or more qubits, interaction Hamiltonian is given by [36, 37]

$$H_{\mathrm{int}} = \mathscr{J} \sum_{i,j>i} 2\pi I_{zi} I_{zj}, \tag{5.3}$$

where $\mathscr{J}$ is a single effective scalar coupling constant. We hence arrive at the effective NMR Hamiltonian of QKT in the weak-coupling limit as follows:

$$\begin{aligned} H_{\mathrm{NMR}}^{\mathrm{eff}} =& H_{\mathrm{RF}} + H_{\mathrm{int}} \\ =& \sum_i \frac{\pi}{2\Delta_i} I_{yi} + \mathscr{J} \sum_{i,j>i} 2\pi I_{zi} I_{zj}. \end{aligned} \tag{5.4}$$

Comparing with Eq.2.1, the first term of it can be mapped to H_{RF} with the angle $p = \pi/2$, and the second term can be mapped to H_{int} where the kick parameter can be written as [34]

$$\kappa = 2j\pi \mathscr{J} \tau. \tag{5.5}$$

From this relation, we can vary the kick parameter κ by varying the interaction time τ between the qubits. Since $\Delta_i \ll \tau$, we can ignore the H_{int} during the RF pulses and hence obtain the Floquet unitary as

$$\begin{aligned} U_{\mathrm{NMR}} =& U_{\mathrm{int}} U_{\mathrm{RF}} \\ =& \exp(-iH_{\mathrm{int}}\tau) \exp(-iH_{\mathrm{RF}}\Delta) \end{aligned} \tag{5.6}$$

To prepare the initial QKT state, we follow the scheme from [34, 33]. From the thermal state, we

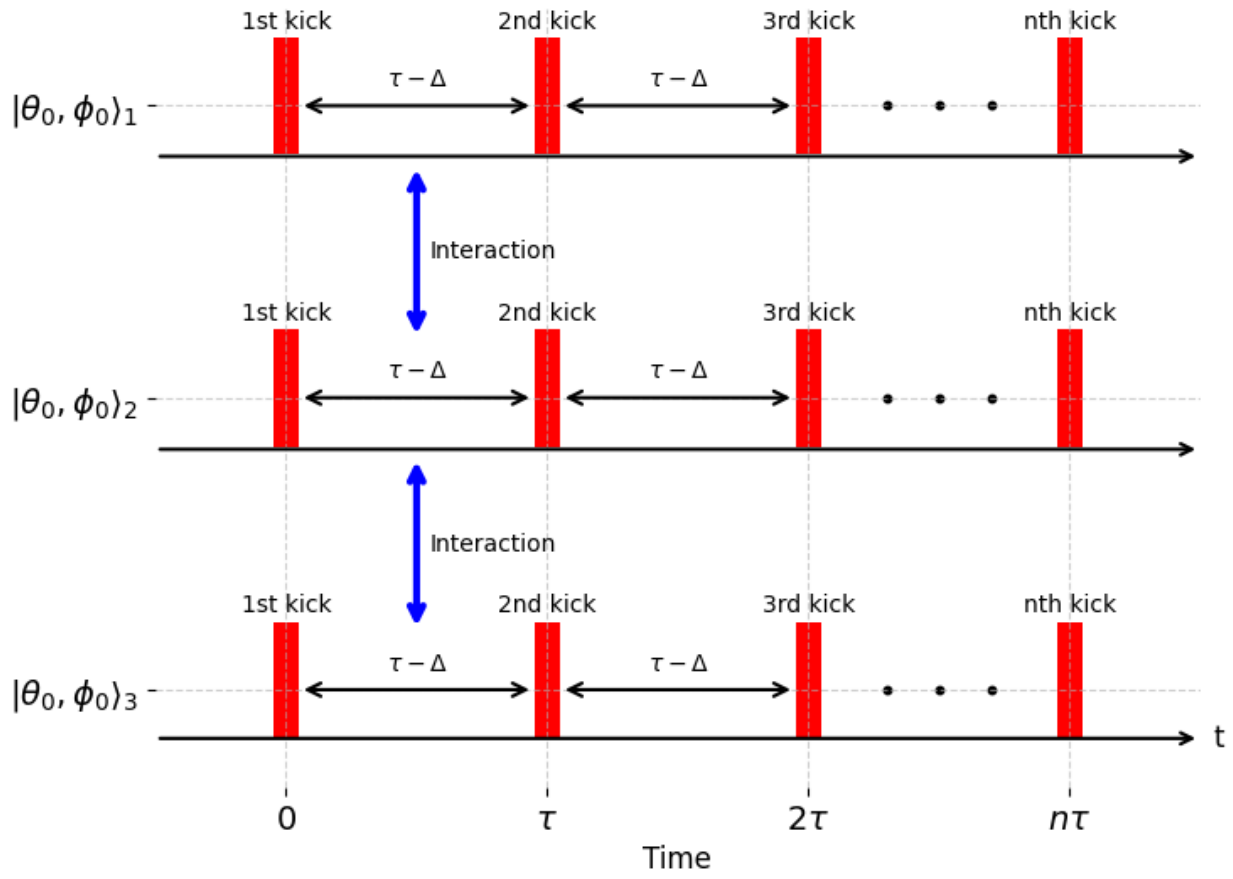


Figure 5.1: Schematic diagram of the RF pulses and non-linear evolution for simulating QKT in NMR using a system of three qubits initialized in coherent state $|\theta_0\phi_0\rangle$. Each kick RF pulse has a width $\Delta \ll \tau$.

first prepare a pseudo-pure state ρ_{PPS} and then transform it via unitary

$$U_{\theta\phi} = \exp(-i\phi \sum_i I_{z_i}) \exp(-i\theta \sum_i I_{y_i}) \tag{5.7}$$

to the coherent state as follows

$$\rho_{\theta\phi} = U_{\theta\phi} \rho_{\text{pps}} U_{\theta\phi}^{\dagger} \tag{5.8}$$

Having prepared the initial state and developed the Floquet operator Eq.5.6, we now apply this unitary evolution operator n-times to study the evolution of our Quantum Kicked Top. The schematic diagram is shown in Fig.5.1. We can now monitor the I_x expectation values at all the time steps, which can be done very easily in the NMR system. Hence, we can perform all of our tasks using

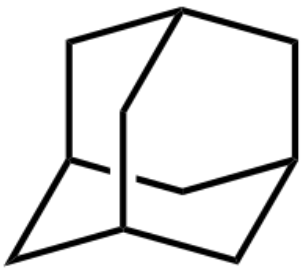


Figure 5.2: Chemical structure of adamantane $(CH)_4(CH_2)_6$ [38]. Each corner of the diagram represents a carbon atom, and hydrogen atom attached to it according to its valency.

this scheme.

The entire scheme can also be very easily used for simulating QKT with four qubits in NMR to perform both the quantum tasks of Chapter 4. We need to find a suitable molecule with four controllable qubits for that.

5.2 Reservoir Computing with Adamantane Sample on NMR System

Adamantane is a polycrystalline organic compound composed of three cyclohexane rings Fig.5.2. It consists of dipolar interactions of nuclear spins of ^{1}H atoms, which corresponds to a 3D spin-coupling network. It is frequently used in solid-state NMR spectroscopy. We got the opportunity to work with this sample, so we used it to perform reservoir computing using the same scheme we used for QKT.

The Hamiltonian of the adamantane system in the interaction picture is the spin-spin interaction [39]

$$\hat{H}_{dd} = \sum_{i<j} d_{ij} \left[2\hat{I}_z^i \hat{I}_z^j - \left(\hat{I}_x^i \hat{I}_x^j + \hat{I}_y^i \hat{I}_y^j \right) \right] \tag{5.9}$$

where d_{ij} are the coupling constants. We use the NMR system to control the spins and perform our scheme of reservoir computing on this sample. The initial state of the system is the thermal state consisting of uncorrelated spins. Between the periods $\tau - \Delta$ of non-linear evolution according to the Hamiltonian $\hat{H}_{dd}$, we gave periodic x-kicks of width Δ corresponding to rotations about x-axis

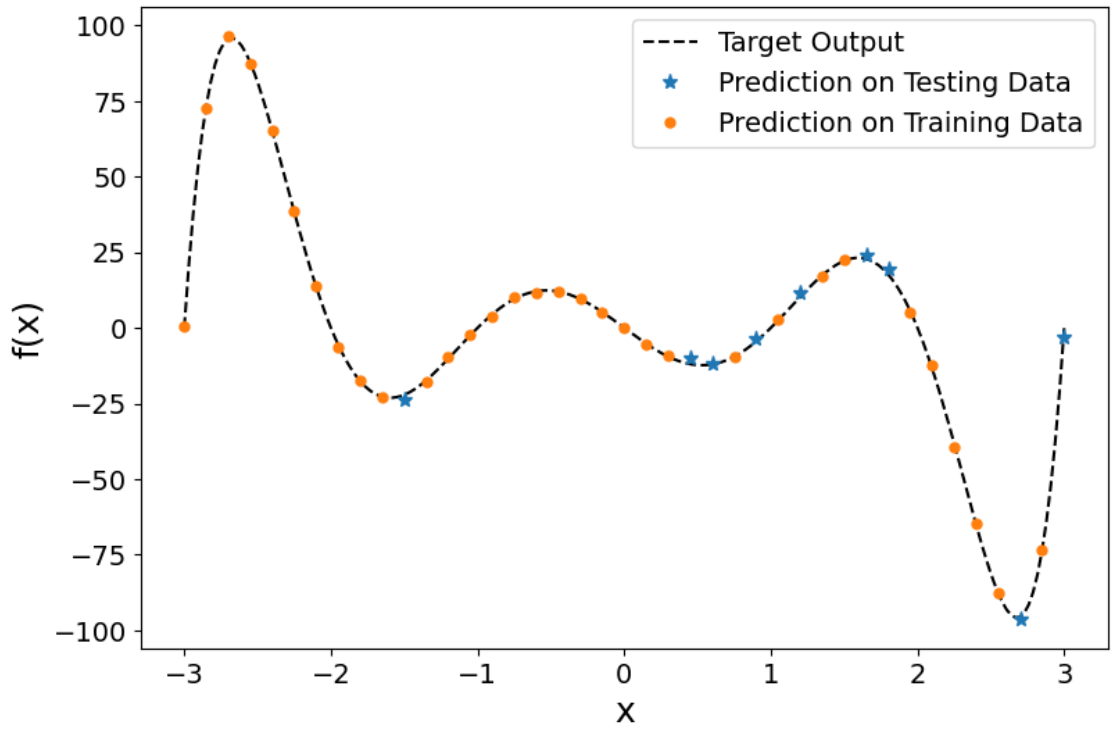


Figure 5.3: Results of Regression task for experimental implementation of adamantane reservoir computing

in the range $[0, \pi/2]$. Input encoding is done in Δ, and $\langle \hat{I}_x \rangle$ is measured 42 times after every period τ to generate the reservoir state.

We performed the earlier simulated regression task using this scheme with a total of 41 data points; 32 points were used for training the reservoir and the remaining for testing. Training data points were fed back to the reservoir to know how well this reservoir learned the training data and how it compares to the new data Fig.5.3. We obtained an RMSE of 0.49 on training data and 1.63 on testing data.

Successful experimental implementation of reservoir computing on NMR shows proof of concept in a real setting for our scheme. Also, it opens the possibility of studying the performance of reservoir computing with an increasing number of interacting qubits present in the adamantane sample owing to the growing cluster size of the correlated spins with interaction time [39].

Chapter 6

Conclusion and Discussion

In this thesis, we have studied the dynamics and properties of quantum kicked top. We took advantage of those to establish QKT as an efficient and practically realizable reservoir for quantum reservoir computing. We showed several ways to simulate the kicked-top dynamics and explored the nuances of each. We devised a straightforward scheme of reservoir computing using QKT and showed good performance in a variety of classical and quantum tasks. We also showed experimental procedures to perform these tasks on a physical NMR system. We used the adamantane sample for experimental implementation and performed the regression task satisfactorily. We are currently working on experimental implementation on the QKT reservoir as well. We intend to show practically at least one task from the classical input tasks and one from the quantum input tasks.

A possible future area of research is to figure out why different hyperparameters of the reservoir can give rise to drastically different performance for a particular task. In Fig.6.1, we have plotted the dynamics of the readouts for different inputs in regression tasks. The plots with $\kappa \in [0, 5.5]$ ((a) and (c)) correspond to RMSE of order 10^{-6} and plots with $\kappa \in [0, 0.5]$ ((b) and (d)), corresponds to RMSE of order 10^{-2}. We kept all the other parameters constant. There is clearly a drastic change in performance across the two ranges of the chaoticity parameter. We explain this difference qualitatively using the plots of Fig.6.1. Notice that the readouts corresponding to different inputs in plot (a) are much more separated and consist of different features than plot (b). Plots (c) and (d) show different frequency components in each readout dynamics.

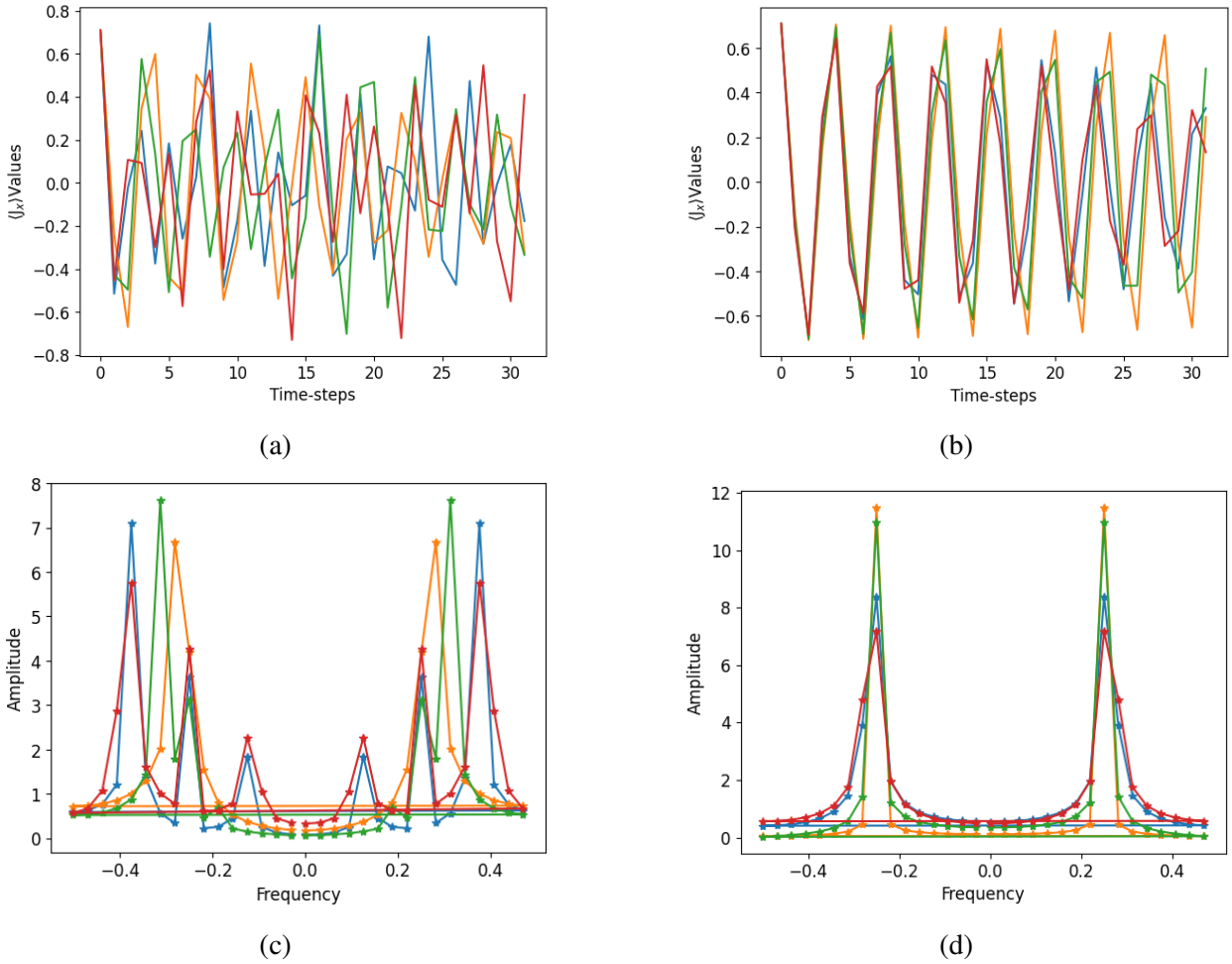


Figure 6.1: Plots (a) and (b) shows QKT reservoir readouts ($\langle J_x \rangle$) for 32 time-steps, for four randomly sampled input (shown in different colors) from the training data for the regression task. (a) corresponds to input encoding in $\kappa \in [0, 5.5]$ and (b) in $\kappa \in [0, 0.5]$. Plots (c) and (d) are Fourier transformations of dynamics in plots (a) and (b), respectively.

Plot (c) has noticeably larger contributions from different frequencies for each input reservoir state than plot (d). More frequencies (meaning more features) and more separability in the dynamics make it easier for linear regression to separate them and map them to the desired output, leading to increased performance.

While this does seem like a conclusive way to find better hyperparameters, it does not always hold. As reservoir dynamics with lesser features and little separability are most likely not to perform the task well, it does not guarantee that dynamics with better separability and contribution from a larger number of frequencies will surely perform satisfactorily. A proper study is needed to find

the best possible way to do hyperparameter optimization, as it is one of the most time-consuming aspects of reservoir computing [40].

Many areas of research need to be explored in the context of reservoir computing, like how to efficiently scale the reservoir to handle more complex data, devising a robust quantitative way to compare different reservoirs as well as other machine learning architectures, and developing a readily available and versatile experimental setup to perform physical reservoir computing. We hope this study will serve as a motivation to consider quantum systems as potential reservoirs and a guide for performing quantum reservoir computing with dynamical spin systems in both simulation and experimental implementation.

Bibliography

[1] Cucchi, M., Abreu, S., Ciccone, G., Brunner, D., & Kleemann, H. (2022). *Hands-on reservoir computing: A tutorial for practical implementation*. Neuromorphic Computing and Engineering, 2(3), 032002.

[2] Tanaka, G., Yamane, T., Héroux, J. B., Nakane, R., Kanazawa, N., Takeda, S., Numata, H., Nakano, D., & Hirose, A. (2019). *Recent advances in physical reservoir computing: A review*. Neural Networks, 115, 100–123.

[3] Jaeger, H. (2001). *The "echo state" approach to analysing and training recurrent neural networks*. DOI: `https://doi.org/10.24406/publica-fhg-291111`

[4] Gaurav, A., Song, X., Manhas, S., Gilra, A., Vasilaki, E., Roy, P., & De Souza, M. M. (2022). *Reservoir Computing for Temporal Data Classification Using a Dynamic Solid Electrolyte ZnO Thin Film Transistor*. Frontiers in Electronics, 3, 869013.

[5] L. Boccato, A. Lopes, R. Attux, and F. J. Von Zuben, *"An extended echo state network using Volterra filtering and principal component analysis,"* Neural Networks 32, 292–302 (2012).

[6] Kong, L. W., Brewer, G. A., & Lai, Y. C. (2024). *Reservoir-computing based associative memory and itinerancy for complex dynamical attractors*. Nature communications, 15(1), 4840.

[7] Gelenbe E 1989 *Random neural networks with negative and positive signals and product form*. solutionNeural Comput. 1 502–10.

[8] Gelenbe E 1993 *Learning in the recurrent random neural network* Neural Comput. 5 154–64.

[9] Fujii, K., & Nakajima, K. (2017). *Harnessing Disordered-Ensemble quantum dynamics for machine learning*. Physical Review Applied, 8(2).

[10] Dasgupta, S., Hamilton, K. E., & Banerjee, A. (2020). *Characterizing the memory capacity of transmon qubit reservoirs*. arXiv (Cornell University) arxiv.2004.08240.

[11] Chen, J., Nurdin, H. I., & Yamamoto, N. (2020). *Temporal information processing on noisy quantum computers*. Physical Review Applied, 14(2).

[12] Negoro, M., Mitarai, K., Fujii, K., Nakajima, K., & Kitagawa, M. (2018). *Machine learning with controllable quantum dynamics of a nuclear spin ensemble in a solid*. arXiv (Cornell University) arxiv.1806.10910.

[13] Götting, N., Lohof, F., & Gies, C. (2023). *Exploring quantumness in quantum reservoir computing*. Physical Review. A/Physical Review, A, 108(5).

[14] Dudas, J., Carles, B., Plouet, E., Mizrahi, F. A., Grollier, J., & Marković, D. (2023). *Quantum reservoir computing implementation on coherently coupled quantum oscillators*. Npj Quantum Information, 9(1), 1-7.

[15] F. Haake, M. Ku´s, and R. Scharf, *Classical and quantum chaos for a kicked top*, Z. Phys. B 65, 381 (1987).

[16] M. Kumari and S. Ghose, *Quantum-classical correspondence in the vicinity of periodic orbits*, Phys. Rev. E 97, 052209 (2018).

[17] Anand, A., Srivastava, S., Gangopadhyay, S., & Ghose, S. (2024). *Simulating quantum chaos on a quantum computer*. Scientific Reports, 14(1), 1-11.

[18] Kumari, M. (2019, August 9). *Quantum-Classical correspondence and entanglement in periodically driven spin systems*. URL: `https://uwspace.uwaterloo.ca/items/9d77a9f3-418b-4075-ac61-f32fc75a79e6`

[19] RZZGate — IBM Quantum Documentation. (n.d.). IBM Quantum Documentation. URL: `https://docs.quantum.ibm.com/api/qiskit/qiskit.circuit.library.RZZGate`

[20] Lorenz, E. N. (1963). *Deterministic Nonperiodic Flow*. Journal of Atmospheric Sciences, 20(2), 130-141.

[21] Guckenheimer, J., & Sparrow, C. (1984). *The Lorenz Equations: Bifurcations, Chaos, and Strange Attractors*. The American Mathematical Monthly, 91(5), 325.

[22] Z. Lu, J. Pathak, B. Hunt, M. Girvan, R. Brockett, and E. Ott, *Reservoir observers: Model-free inference of unmeasured variables in chaotic systems*, Chaos 27, 041102 (2017).

[23] J. Choi and P. Kim, *Reservoir computing based on quenched chaos*, Chaos Solitons Fractals 140, 110131 (2020).

[24] Mandal, S., Sinha, S., & Shrimali, M. D. (2022). *Machine-learning potential of a single pendulum*. Physical Review. E, 105(5).

[25] Jonathan A. Jones (2024). *Controlling NMR spin systems for quantum computation*. Progress in Nuclear Magnetic Resonance Spectroscopy, 140-141, 49-85.

[26] Plenio, M. B. (2005). *Logarithmic Negativity: A Full Entanglement Monotone That is not Convex*. Physical Review Letters, 95(9).

[27] Serrano-Ensástiga, E., & Martin, J. (2023). *Maximum entanglement of mixed symmetric states under unitary transformations*. SciPost Physics, 15(3).

[28] Sabarad, V., & Mahesh, T. S. (2024). *Experimental Machine Learning with Classical and Quantum Data via NMR Quantum Kernels*. arXiv (Cornell University) arxiv.2412.09557.

[29] Yperman, J., & Becker, T. (2016). *Bayesian optimization of hyper-parameters in reservoir computing*. arXiv (Cornell University) arxiv.1611.05193.

[30] Jones, J. A., & Mosca, M. (1998). *Implementation of a quantum algorithm on a nuclear magnetic resonance quantum computer*. The Journal of Chemical Physics, 109(5), 1648–1653.

[31] Chuang, I. L., Gershenfeld, N., & Kubinec, M. (1998). *Experimental implementation of fast quantum searching*. Physical Review Letters, 80(15), 3408–3411.

[32] Mahesh, T. S., Kumar, C. S. S., & Bhosale, U. T. (2017). *Quantum correlations in NMR systems*. In Quantum science and technology (pp. 499–516).

[33] Krithika, V. R., Anjusha, V. S., Bhosale, U. T., & Mahesh, T. S. (2019). *NMR studies of quantum chaos in a two-qubit kicked top*. Physical Review. E, 99(3).

[34] Krithika, V. R., Santhanam, M. S., & Mahesh, T. S. (2023). *NMR investigations of dynamical tunneling in spin systems*. Physical Review. A/Physical Review, A, 108(3).

[35] Sanders, B. C., & Milburn, G. J. (1989). *The effect of measurement on the quantum features of a chaotic system*. Zeitschrift Für Physik B Condensed Matter, 77(3), 497–510.

[36] Levitt, M. (2013). *Spin Dynamics: Basics of Nuclear Magnetic Resonance*. (Wiley, New York, 2013).

[37] Cavanagh, J., Fairbrother, W., Palmer, A., & Skelton, N. (1995). *Protein NMR Spectroscopy: Principles and Practice* (Elsevier, Amsterdam, 1995).

[38] Von R Schleyer, P. (1957). *A SIMPLE PREPARATION OF ADAMANTANE*. Journal of the American Chemical Society, 79(12), 3292.

[39] Gonzalo A. Álvarez et al., *Localization-delocalization transition in the dynamics of dipolar-coupled nuclear spins*.Science 349,846-848(2015).

[40] Picco, E., Jaurigue, L., Lüdge, K., & Massar, S. (2025). *Efficient optimisation of physical reservoir computers using only a delayed input*. Communications Engineering, 4(1), 1-9.