Entanglement classification of Gaussian and non-Gaussian states with quantum reservoir computing

Master's thesis
University of Turku
Theoretical Physics
2026
B.Sc. Johannes Annala
Supervisors:
Prof. M. Paternostro
Adj. Prof. J. Nokkala
Examiners:
Prof. J. Piilo
Adj. Prof. J. Nokkala

The originality of this thesis has been checked in accordance with the University of Turku quality assurance system using Turnitin Originality Check service.

UNIVERSITY OF TURKU
Department of Physics and Astronomy

Annala, Johannes Entanglement classification of Gaussian and non-Gaussian states with quantum reservoir computing

Master's thesis, 59 pp.
Theoretical Physics
May 2026

Reservoir computing is a neural network framework capable of nonlinear information processing. The ambiguous nature of the information processing unit provides a fascinating opportunity of employing physical systems as reservoirs. This has also spiked an interest in the research and development of quantum systems. As quantum computing still faces many problems such as scalability and the loss of coherence, quantum reservoir computing offers an alternative way of conducting intricate quantum tasks. These results are not limited to quantum tasks, as the evidence in quantum reservoir computing suggest a possible advantage over classical algorithms in classical nonlinear tasks.

This work provides a comprehensive inspection of the theory behind quantum reservoir computing and an entanglement classification task conducted on the reservoir. This includes the review of tools for the measurement of entanglement, Gaussian and non-Gaussian nature of systems, machine learning and quantum computing. The theoretical review is accompanied by results of classical reservoir computing and a greater research on the capabilities of different quantum substrates used in quantum reservoir computing.

The main part of this work is to build a working quantum reservoir computing system containing a fermionic reservoir. The task of the reservoir is to learn to classify quantum entanglement of two-mode bosonic inputs. The bosonic inputs used in this work are squeezed thermal states, photon added squeezed vacuum states, photon subtracted squeezed vacuum states and number states. This way Gaussian and non-Gaussian entanglement recognition is tested. Reservoirs of sizes 2, 3 and 4 modes are used and different connectivities inside the reservoir networks are studied. Even though the entanglement is assessed for different classes of states, the training is conducted with only squeezed thermal states.

Keywords: quantum reservoir computing, entanglement classification, quantum computing, machine learning, Gaussian, non-Gaussian

Contents

Introduction

The emergence of convenient travel options and the internet has made the world around us smaller and smaller. As a result, trends rise and some possibilities previously impossible have become available to a large quantity of people. In the end not only are the people affected but also the environment around us. This offers an opportunity to collect enormous amounts of data to study the new behavior. As data is usually cheap and abundant, one is left with the problem of turning the data into something valuable, knowledge. Machine learning offers a plethora of tools for interpreting the data and producing meaningful conclusions. This has produced an era where different disciplines are implementing various artificial intelligence tools in every problem imaginable [1–3]. With the promise of exponential speedup over classical counterparts, an increasing interest has risen in utilizing quantum computing protocols to further enhance machine learning algorithms [4, 5].

Embedding quantum mechanics in computing schemes is not a straightforward task, and many challenges in generalized quantum computing still exist [6, 7]. These challenges include but are not limited to the loss of coherence in quantum systems and the difficulty of building a quantum computer with a sufficient amount of computing power. This has provoked an exploration for alternative ways at harnessing quantum systems in the search of computing power. Reservoir computing is a machine learning framework where nonlinear information can be processed through a "black box" or an oracle, also in this case called the reservoir. Quantum reservoir computing takes this structure a step further by replacing the information processing unit by a quantum reservoir. The ambiguous nature of the reservoir offers a more generalized way of operating quantum systems as a part of computing routines. Like all real physical systems, quantum reservoirs are inherently open systems. Therefore the study of different quantum substrates is crucial for the development in quantum reservoir computing.

The aim of this work is to build and test the entanglement classification capabilities of quantum reservoir computing systems. Entanglement is introduced to the system by Gaussian and non-Gaussian two-mode inputs. An approach resembling the systems introduced in this work has previously been explored in literature [8]. This work takes the matter a step further and also analyzes the performance of other quantum reservoirs implemented in a similar setting. The new substrates are created by either changing the amount of modes present in the reservoir or by varying the connectivity inside the mode network.

This thesis gives a comprehensive overview of all the relevant theory and dynamics related to reservoir computing systems and the tools used in the simulations. These include topics like the physics of entanglement, Gaussianity, non-Gaussianity, machine learning, neural networks and quantum computing. Also a brief demonstration of the capabilities of classical reservoir computing is demonstrated. As familiarity with nonlinear systems is established, the quantum reservoir computing system of this work is introduced and the entanglement classification capabilities of different quantum substrates is evaluated. In the end, a throughout review of the results is presented. All the code for the classical [9] and quantum [10] reservoir computing simulations are publicly available on my personal Github page.

1 Underlying theory of reservoir computing

This section introduces all underlying theory of the quantum reservoir computing simulation later introduced in section 3. The subjects progress in an intuitive fashion building on the previous topics from entanglement detection and Gaussian states to machine learning and quantum computing. For the sake of clarity, operators will be represented with bolded characters in this section.

1.1 Dynamics of quantum systems

1.1.1 Separability and entanglement

All real classical or quantum systems are inherently open systems. With the right tools, one can construct a system that minimizes the effects of open system dynamics. In some scenarios the effects are negligible or even of use. For any classical system one can determine the parameters governing it's physics, assuming one has the means to measure those parameters. For a quantum system one can only determine the probability of finding it's parameters in some state. Therefore, to understand the evolution of a quantum system, knowledge about the parameter's probability distributions is needed.

A state vector $|\psi\rangle \in \mathcal{H}$ can be presented in the position basis $\langle x|\psi\rangle = \psi(x)$, where the single-mode Hilbert space $\mathcal{H}$ is spanned by the infinite-dimensional number-state basis $\{|j\rangle\}_{j=0}^{\infty}$, the Fock-state basis. Now, the state vector has a statistical interpretation $A = |\psi(x)|^2$, which can in theory be integrated $\int_a^b A dx$ to find the probability of finding the state between the points a and b. Generally speaking, a quantum state can be described with a density operator $\boldsymbol{\rho} = \sum_k p_k \, |\psi_k\rangle\langle\psi_k|$, where p_k is the probability of finding the state in $|\psi_k\rangle$. The density operator is positive and has $\mathrm{Tr}[\boldsymbol{\rho}] = 1$. Pure states $\boldsymbol{\rho} = |\psi\rangle\langle\psi|$ describing the complete nature of a quantum system have $\boldsymbol{\rho}^2 = \boldsymbol{\rho}$, and statistical ensembles of states or mixed states have $\boldsymbol{\rho}^2 < \boldsymbol{\rho}$.

Quantum systems may be divided into subsystems to study them separately and their effect on the combined dynamics. A separable quantum state can be written with a tensor product. For a bipartite system $\mathcal{H}_{AB} = \mathcal{H}_A \otimes \mathcal{H}_B$ one can now write $|\psi\rangle_{AB} = |\psi\rangle_A \otimes |\psi\rangle_B$. Physically this means that the state can be prepared by operating on the subsystems A and B separately, i.e. with local operations. This preparation may be communicated classically between the subsystems, and therefore all correlations between them may be deemed classical [11].

The correlations between quantum systems can be classical or quantum in nature. With the latter a quantum system is said to be entangled. Let's define a bipartite state with orthonormal number basis $|j\rangle = |j_A\rangle \otimes |j_B\rangle = |j_A j_B\rangle$ and write out $\boldsymbol{\rho}$ with matrix elements:

$$\boldsymbol{\rho}_{AB} = \sum_{j_A,k_A=1}^{d_A} \sum_{j_B,k_B=1}^{d_B} \langle j_A j_B|\boldsymbol{\rho}|k_A k_B\rangle \, |j_A\rangle\langle k_A| \otimes |j_B\rangle\langle k_B| \tag{1}$$

Here $d_A = \dim\mathcal{H}_A$ and $d_B = \dim\mathcal{H}_B$ are the dimensions of respective Hilbert spaces. To study the dynamics of a subsystem of a composite system, one needs to calculate the partial trace of the full system. The partial trace is calculated over the discarded subsystems. For the bipartite system this is a simple task.

$$\begin{aligned}
\boldsymbol{\rho}_A &= \mathrm{Tr}_B[\boldsymbol{\rho}_{AB}] \\
&= \mathbb{I}_A \otimes \mathrm{Tr}_B\Big(\sum_{j_A,k_A=1}^{d_A} \sum_{j_B,k_B=1}^{d_B} \langle j_A j_B|\boldsymbol{\rho}|k_A k_B\rangle \, |j_A\rangle\langle k_A| \otimes |j_B\rangle\langle k_B|\Big) \\
&= \sum_{j_A,k_A=1}^{d_A} \sum_{j_B,k_B=1}^{d_B} \langle j_A j_B|\boldsymbol{\rho}|k_A k_B\rangle \, |j_A\rangle\langle k_A| \otimes \mathrm{Tr}_B[|j_B\rangle\langle k_B|] \\
&= \sum_{j_A,k_A=1}^{d_A} \sum_{j_B=1}^{d_B} \langle j_A j_B|\boldsymbol{\rho}|k_A j_B\rangle \, |j_A\rangle\langle k_A|
\end{aligned} \tag{2}$$

Similarly, the partial trace can be used to study the other parts of the system $\boldsymbol{\rho}_B = \mathrm{Tr}_A[\boldsymbol{\rho}_{AB}]$.[12]

The amount of entanglement in a system can vary between no entanglement and fully entangled. Depending on the system, the variation can be measured with

logarithmic negativity. This requires one to calculate the trace norm of the partial transpose of a state. To understand the partial transpose operation and it's effect on the density matrix, let's again consider the bipartite system with matrix elements from Eq. (1). The partial transpose on the system A can now be calculated

$$\begin{aligned}
\boldsymbol{\rho}^{\top_A} &= \boldsymbol{T}_A(\boldsymbol{\rho}) \\
&= \boldsymbol{T} \otimes \mathbb{I}_B(\sum_{j_A,k_A=1}^{d_A} \sum_{j_B,k_B=1}^{d_B} \langle j_A j_B|\boldsymbol{\rho}|k_A k_B\rangle \, |j_A\rangle\langle k_A| \otimes |j_B\rangle\langle k_B|) \\
&= \sum_{j_A,k_A=1}^{d_A} \sum_{j_B,k_B=1}^{d_B} \langle j_A j_B|\boldsymbol{\rho}|k_A k_B\rangle \, \boldsymbol{T}(|j_A\rangle\langle k_A|) \otimes |j_B\rangle\langle k_B| \\
&= \sum_{j_A,k_A=1}^{d_A} \sum_{j_B,k_B=1}^{d_B} \langle j_A j_B|\boldsymbol{\rho}|k_A j_B\rangle \, |k_A\rangle\langle j_A| \otimes |j_B\rangle\langle k_B| \,,
\end{aligned}$$

where $\boldsymbol{T}$ is the transpose operator. Similarly, one can take the partial transpose on the system B. The effect of a partial transpose may also be written as the change of matrix elements with formulas:

$$\boldsymbol{\rho}^{\top_A}((k_A-1)d_B+j_B,(j_A-1)d_B+k_B) = \boldsymbol{\rho}((j_A-1)d_B+j_B,(k_A-1)d_B+k_B)$$
$$\boldsymbol{\rho}^{\top_B}((j_A-1)d_B+k_B,(k_A-1)d_B+j_B) = \boldsymbol{\rho}((j_A-1)d_B+j_B,(k_A-1)d_B+k_B)$$

The partial trace as well as the partial transpose are applicable for bigger composite systems. The trace norm

$$||\boldsymbol{A}||_{\mathrm{Tr}} = \mathrm{Tr}\left[\sqrt{\boldsymbol{A}^\dagger \boldsymbol{A}}\right] \overset{A=A^\dagger}{=} \sum_i |\lambda_i|, \tag{3}$$

is calculated from the partially transposed state $||\boldsymbol{\rho}^{\top_A}||_{\mathrm{Tr}}$. Here λ_i are the eigenvalues of the operator in question. The last part of Eq. (3) is true for any Hermitian operator $\boldsymbol{A}$. The choice which system is partially transposed is usually dependent on the situation.[13]

As stated previously, $\boldsymbol{\rho}$ is a positive operator and has $\mathrm{Tr}[\boldsymbol{\rho}] = 1$. All of it's eigenvalues are therefore positive. For a density operator, the nonpositivity would imply a negative probability for some states making the system nonphysical. In

the most simple terms, negativity is a measure of how much an operator fails to be positive. Like the density operator, the trace of a partially transposed system is also $\mathrm{Tr}[\boldsymbol{\rho}^{\top_A}] = 1$, but it's eigenvalues might be negative. One can therefore calculate the negativity $\mathcal{N}$ of $\boldsymbol{\rho}$ in a few different ways. The measure of entanglement, logarithmic negativity $E_{\mathcal{N}}$ has a connection to negativity

$$\mathcal{N}(\boldsymbol{\rho}) = \frac{||\boldsymbol{\rho}^{\top_A}||_{\mathrm{Tr}} - 1}{2} = \frac{\sum_i |\lambda_i| - 1}{2} = |\sum_{\lambda_i<0} \lambda_i|$$

$$E_{\mathcal{N}}(\boldsymbol{\rho}) = \log_2(||\boldsymbol{\rho}^{\top_A}||_{\mathrm{Tr}}) = \log_2(2\mathcal{N}(\boldsymbol{\rho}) + 1). \tag{4}$$

If the negativity of a state is found to be $\mathcal{N}(\boldsymbol{\rho}) > 0$, the state is known to be entangled. For separable systems negativity goes to 0.[14, 15]

If a composite system can be presented as

$$\boldsymbol{\rho} = \sum_k w_k \boldsymbol{\rho}'_k \otimes \boldsymbol{\rho}''_k,$$

where w_k are positive weights and $\boldsymbol{\rho}'_k, \boldsymbol{\rho}''_k$ are density matrices of subsystems, the state is known to be separable. In other words, the partial transpose of a density matrix is positive. Logarithmic negativity is a good indicator of entanglement, albeit not a perfect one. The Peres's criterion gives a criterion for detecting entanglement with logarithmic negativity in composite systems having dimensionality $d \leq 6$ [14]. This raises the question: how can one classify entangled composite systems that have dimensionality greater than 6?

Entangled systems that pass the positive partial transpose (PPT) test are called bound entangled states (BE). To better understand bound entanglement, let us introduce quantum state distillability or purification. Distillable quantum states or free entangled states (FE) are mixed entangled states, that by some local operations and classical communication are able to be transformed into pure entangled states, or in other terms purified [16–18]. In this way multiple "weakly" entangled states can be converted to a few "strongly" entangled states [19]. All pure quantum states

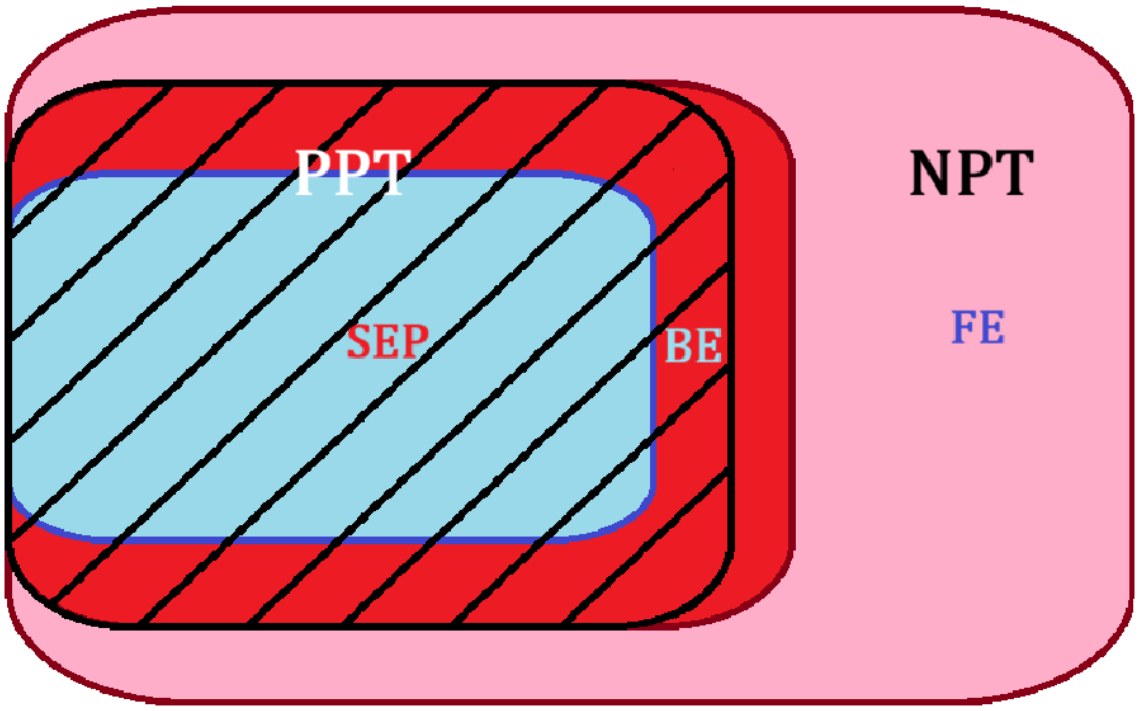

Figure 1. The distinction between separable, bound entangled and free entangled states may not always be clear. For the most part the, PPT test is a good indicator of the distinction between bound entangled and free entangled states.

evolve into mixed states as time passes. Therefore, distillation is a powerful property and is used in quantum information as a way to create maximally entangled states and to pass information over noisy channels [20].

Undistillable quantum states are known as bound entangled. As the Peres's criterion suggests [21], bound entangled states do not exist for composite systems with dimensionality $d \leq 6$. As entangled PPT states are known to be bound entangled, a question remains if there exist also nonpositive partial transpose (NPT) bound entangled states. The distinction between separable, BE and FE states with PPT and NPT indicators is presented in Fig. 1. Moreover, detecting the entanglement of bound entangled states can be difficult and requires more intricate tools [22, 23]. Bound entangled states show that logarithmic negativity is a good but not robust indicator of the entanglement of a given state. In this work all the quantum states are distillable and therefore logarithmic negativity is used as an entanglement estimator.

1.1.2 Evolution of quantum states

Closed systems are real only in theory and on paper. Even with this caveat, it's useful to understand their evolution so that the additional dynamics introduced by open systems can be minimized. In a closed system one experiences unitary dynamics, where processes are reversible and information is not lost. The evolution of a state $|\psi(t)\rangle$ from time t_0 to t is given by a unitary operator $\boldsymbol{U}(t,t_0)\,|\psi(t_0\rangle = e^{-i\boldsymbol{H}t/\hbar}\,|\psi(t_0\rangle$, where $\boldsymbol{H}$ is the Hamiltonian describing the system's dynamics. An open system experiences non-unitary dynamics. This happens when the system interacts with it's environment. In theory, one could still describe open system dynamics in the same way one does with closed systems by declaring a combined closed system of the system and the environment. In practice this becomes extremely challenging as the interaction Hamiltonian often becomes too complicated. With non-unitary dynamics, as processes become irreversible, information may be lost.

The dynamics of open systems may be described with master equations. Master equations are equations of motions of density matrices, of which the Lindblad form in the most general conformation is

$$\frac{d}{dt}\boldsymbol{\rho} = -\frac{i}{\hbar}[\boldsymbol{H},\boldsymbol{\rho}] + \sum_i \gamma(\boldsymbol{L}_i\boldsymbol{\rho}\boldsymbol{L}_i^\dagger - \frac{1}{2}\boldsymbol{L}_i^\dagger\boldsymbol{L}_i\boldsymbol{\rho} - \frac{1}{2}\boldsymbol{\rho}\boldsymbol{L}_i^\dagger\boldsymbol{L}_i). \tag{5}$$

Here γ is the dissipation coefficient and $\boldsymbol{L}_i$ jump operators. The first term on the right hand side of Eq. (5) accounts for the Hermitian dynamics of the system. The second term describes the jumps between energy levels and the last two terms represent the dissipation of energy. The Lindblad master equation can also be used to describe Hermitian dynamics by discarding the non-Hermitian terms. This is also known as the von Neumann equation.[24]

The dynamics of open systems may further be classified as Markovian or non-Markovian. As the system interacts with it's environment, energy flows out of the system. This means a loss of information and coherence. For systems with Markovian dynamics the loss is permanent and the information is lost. On the

other hand, non-Markovian systems define a quantum memory of the information lost to the environment. After a while, the information stored in the environment may flow back to the system allowing a regain of the quantum state coherence.[25]

1.2 Gaussian and non-Gaussian states

A continuous variable quantum system can be represented by bosonic modes, i.e. quantum harmonic oscillators. Here the single-mode Hilbert space $\mathcal{H}$ is spanned by the number-state basis $\{|n\rangle\}_{n=0}^{\infty}$ composed of the eigenstates of the number operator $\boldsymbol{n} = \boldsymbol{a}^\dagger \boldsymbol{a}$:

$$\boldsymbol{n}\,|n\rangle = n\,|n\rangle\,,$$

where $\boldsymbol{a}$ and $\boldsymbol{a}^\dagger$ are the annihilation and creation operators of the mode. The actions of single mode field operators on the number state are

$$\boldsymbol{a}\,|n\rangle = \sqrt{n}\,|n-1\rangle\,, \qquad \boldsymbol{a}^\dagger\,|n\rangle = \sqrt{n+1}\,|n+1\rangle\,.$$

With the field operators and vacuum state one can create an infinite amount of energy-states

$$\begin{aligned} |1\rangle &= \boldsymbol{a}^\dagger\,|0\rangle \\ |2\rangle &= \frac{\boldsymbol{a}^\dagger}{\sqrt{2}}\,|1\rangle = \frac{(\boldsymbol{a}^\dagger)^2}{\sqrt{2}}\,|0\rangle \\ &\cdots \\ |n\rangle &= \frac{(\boldsymbol{a}^\dagger)^n}{\sqrt{n!}}\,|0\rangle \end{aligned}$$

The vacuum state $|0\rangle$ is the lowest energy state. Bosonic modes N are associated with the Hilbert space $\mathcal{H}^{\otimes N}$ and field operators $\{\boldsymbol{a}_k, \boldsymbol{a}_k^\dagger\}_{k=1}^{N}$. The bosonic field operators satisfy the commutation relations

$$[\boldsymbol{a}_i, \boldsymbol{a}_j^\dagger] = \delta_{ij}, \qquad [\boldsymbol{a}_i, \boldsymbol{a}_j] = [\boldsymbol{a}_i^\dagger, \boldsymbol{a}_j^\dagger] = 0,$$

where δ_{ij} is the Kronecker delta.

The field operators can be redefined as position- and momentum-like quadrature operators for each mode through cartesian decomposition

$$\boldsymbol{q}_k = \frac{\boldsymbol{a}_k + \boldsymbol{a}_k^\dagger}{2\kappa} \tag{6}$$

$$\boldsymbol{p}_k = \frac{\boldsymbol{a}_k - \boldsymbol{a}_k^\dagger}{2i\kappa}. \tag{7}$$

The quadratures satisfy commutation relations

$$[\boldsymbol{q}_k, \boldsymbol{p}_l] = \frac{i}{2\kappa^2}\delta_{kl}.$$

Here $\kappa \in \mathbb{R}$ and for canonical position and momentum operators we will use $\kappa = \frac{1}{\sqrt{2}}$. One can generalize the quadratures found in Eqs. (6) and (7) to find the rotated quadratures

$$\begin{aligned}\boldsymbol{x}_k(\phi) &= \frac{\boldsymbol{a}_k e^{i\phi} + \boldsymbol{a}_k^\dagger e^{-i\phi}}{\sqrt{2}} \\ &= \cos(\phi)\boldsymbol{q}_k + \sin(\phi)\boldsymbol{p}_k,\end{aligned}$$

that satisfy the commutation relations

$$[\boldsymbol{x}_k(\phi), \boldsymbol{x}_l(\varphi)] = i\delta_{kl}\sin(\varphi - \phi).$$

The quadratures in Eqs. (6) and (7) may also be arranged in a vector form $\boldsymbol{r} = (\boldsymbol{q}_1, \boldsymbol{p}_1, \ldots, \boldsymbol{q}_n, \boldsymbol{p}_n)^\top$ and one can find commutation relations

$$[\boldsymbol{r}_k, \boldsymbol{r}_l] = i\Omega_{kl}. \tag{8}$$

Here Ω is a symplectic matrix defined as

$$\Omega = \bigoplus_{k+1}^{n} \omega, \quad \omega = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

The vector may also be organized in a different way $\boldsymbol{R} = (\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n, \boldsymbol{p}_1 \ldots, \boldsymbol{p}_n)^\top$, and a commutation relation similar to the one in Eq. (8) can be found

$$[\boldsymbol{R}_k, \boldsymbol{R}_l] = iJ_{kl}.$$

The matrix J here is also a symplectic matrix [11, 26, 27]

$$J = \begin{pmatrix} 0 & \mathbb{I}_n \\ -\mathbb{I}_n & 0 \end{pmatrix}.$$

In short, symplectic transformations are linear transformations that keep the symplectic structure and properties. Matrix S is symplectic if and only if it satisfies

$$S\Omega S^\top = \Omega. \tag{9}$$

Now after the symplectic transformation $\boldsymbol{r'} = S\boldsymbol{r}$ the commutation relation in Eq. (8) remains unchanged $[\boldsymbol{r}'_k, \boldsymbol{r}'_l] = i\Omega_{kl}$. Real symplectic matrices form the group $\mathrm{SP}(2N, \mathbb{R})$. For each S one is able to find a unitary operator $\boldsymbol{U}$ suct that

$$\boldsymbol{r'} = S\boldsymbol{r} = \boldsymbol{U}^\dagger r\, \boldsymbol{U},$$

where $\boldsymbol{U}$ is arbitrary up to a phase factor.[28]

Any density operator $\boldsymbol{\rho}$ has an equivalent representation with a quasiprobability distribution over the phase space. If we introduce the Weyl operator

$$\boldsymbol{D}(\xi) = e^{i\boldsymbol{r}^\top \Omega \xi} \tag{10}$$

where $\xi \in \mathbb{R}^{2n}$, the Wigner characteristic function is

$$\chi(\xi) = \mathrm{Tr}[\boldsymbol{\rho}\boldsymbol{D}(\xi)].$$

With a Fourier transform one is able to find the Wigner function

$$W(r) = \int_{\mathbb{R}^{2n}} \frac{d^{2n}\xi}{(2\pi^2)^{2n}} e^{i\boldsymbol{r}^\top \Omega \xi} \chi(\xi). \tag{11}$$

The Wigner function is a quasiprobability distribution function normalized to 1. The continuous variables $r \in \mathbb{R}^{2n}$ are eigenstates of operators $\boldsymbol{r}$, which span a real symplectic space, the phase space. As a quasiprobability distribution, the Wigner function may assume negative values. The negativity of these values is an effect of non-classical, i.e. quantum behavior in the phase space.[27]

The quantities best characterizing the Wigner representations are it's statistical moments. Let's introduce a random variable X. The expected value of X^s is called the s^{th} moment of the random variable X

$$m_s = E[X^s].$$

Central moments are computed with respect to the first moment, mean

$$\mu_s = E[(X - E[X])^s].$$

Different moments have different statistical interpretations, but we are only interested on the first moment and the second, third and fourth order central moments, mean, variance, skewness and kurtosis. These moments can be written out in terms of expectation values of powers of the random variable X

$$\begin{aligned}
m_1 &= E[X] \\
\mu_2 &= E[X^2] - E[X]^2 \\
\mu_3 &= E[X^3] - 3E[X]E[X^2] - 2E[X]^3 \\
\mu_4 &= E[X^4] - 4E[X]E[X^3] + 6E[X^2]E[X]^2 - 3E[X]^4.
\end{aligned}$$

The moments mean, variance, kurtosis and skewness will be addressed as the first, second, third and fourth order moments moving on.

The states for which the Wigner function describing them is a Gaussian distribution are called Gaussian states. Gaussian states are completely described by the first and second moments of the quadrature operators $\boldsymbol{r}$. These moments are called the displacement vector $\bar{r} = \langle \boldsymbol{r} \rangle$ and the covariance matrix σ. For a given state $\boldsymbol{\rho}$, the covariance matrix has elements

$$\sigma_{kl} = \frac{1}{2} \langle \{ \boldsymbol{r}_k, \boldsymbol{r}_l \} \rangle - \langle \boldsymbol{r}_k \rangle \langle \boldsymbol{r}_l \rangle \tag{12}$$

and fulfills the equation $\sigma + \frac{i}{2}\Omega \geq 0$. The diagonal terms of σ are the variances and the off-diagonal covariances of quadratures. The Wigner function of a Gaussian state

in Eq. (11) can be expressed in terms of the displacement vector and the covariance matrix [26]

$$W(r) = \frac{1}{(2\pi)^n \sqrt{\det(\sigma)}} e^{-\frac{1}{2}(r-\bar{r})^\top \sigma^{-1}(r-\bar{r})}.$$

The simplest Gaussian state is the vacuum state $|0\rangle$, which covariance matrix in the canonical case $\kappa = \frac{1}{\sqrt{2}}$ is diagonal $\frac{1}{2}\mathbb{I}$. Therefore, the position and momentum operators have noise variances $\sigma(\boldsymbol{q}) = \sigma(\boldsymbol{p}) = \frac{1}{2}$, which is the minimum variance symmetrically reached by position and momentum. This is also called vacuum noise or quantum shot noise.[27]

Gaussian states stay Gaussian through the actions of Gaussian operations. Unitary Gaussian operations are generated from at most quadratic Hamiltonians of field operators $\boldsymbol{a}_i, \boldsymbol{a}_i^\dagger$. In terms of the statistical moments $\bar{r}$ and σ, the transformation through a Gaussian unitary is characterized as

$$\bar{r} \to S\bar{r} + d$$

$$\sigma \to S\sigma S^\top.$$

Here $d \in \mathbb{R}^{2n}$ is a vector of displacements and S a symplectic matrix satisfying Eq. (9). We find that every Gaussian unitary corresponds to a symplectic matrix and displacement vector characterized by $\bar{r}$ and σ. The usefulness of this result in numerical simulations is further discussed in section 3.2.

Different Gaussian states may be produced through different Gaussian transformations. The displacement operator

$$\boldsymbol{D}(\alpha) = e^{\alpha \boldsymbol{a} - \alpha^* \boldsymbol{a}^\dagger},$$

a complex version of the Weyl operator in Eq. (10), can be used to displace the vacuum state to create coherent states $|\alpha\rangle = \boldsymbol{D}(\alpha)\,|0\rangle$. Here α is a complex amplitude. The displacement operator keeps the covariance matrix σ the of vacuum untouched and changes only the displacement vector $\bar{r}$. In other terms, in the phase space the

shape of the state's Wigner function is kept unchanged and only the mean value is "moved". On the other hand, the squeezing operator

$$\boldsymbol{S}_1(\alpha) = e^{\frac{1}{2}(\alpha(\boldsymbol{a}^\dagger)^2 - \alpha^* \boldsymbol{a}^2)} \tag{13}$$

keeps the displacement vector intact and only effects the covariance matrix. In the phase space the effect of squeezing can be seen as the distribution is squeezed on one axis and elongated on the other. The distribution may also be rotated.

Along with the vacuum state, thermal states are another important group of Gaussian states. A bosonic state is called thermal when it maximizes it's von Neumann entropy $S = -\mathrm{Tr}[\boldsymbol{\rho} \ln \boldsymbol{\rho}]$. With a thermal state $\boldsymbol{\rho}_{\mathrm{th}}$ described by a diagonal covariance matrix D_{th} and a unitary transformation associated with some symplectic matrix S, one is able to obtain every Gaussian state. Therefore, all Gaussian states can be decomposed into thermal states and thermal states may be considered as the fundamental building blocks of Gaussian states. Presented in the number basis the single mode thermal state may be written as

$$\boldsymbol{\rho}_{\mathrm{th}}(\bar{n}) = \sum_n \frac{\bar{n}^n}{(1+\bar{n})^{n+1}} \left|n\right\rangle\!\left\langle n\right|,$$

where $\bar{n}$ is the mean photon number of the thermal state. [11, 26]

Gaussian states and transformations may also be considered in a composite system. The two-mode squeezing operator is similar to the single mode operator

$$\boldsymbol{S}_2(\alpha) = e^{\alpha \boldsymbol{a}_1^\dagger \boldsymbol{a}_2^\dagger - \alpha^* \boldsymbol{a}_1 \boldsymbol{a}_2}.$$

Here $\boldsymbol{a}_i, \boldsymbol{a}_i^\dagger$ are the field operators of the respective systems. Now the two-mode squeezed thermal (ST) state can be written as

$$\begin{aligned} \boldsymbol{\rho}_{\mathrm{ST}} &= \boldsymbol{S}_2(\alpha) \boldsymbol{\rho}_{\mathrm{th}}(\bar{n}) \otimes \boldsymbol{\rho}_{\mathrm{th}}(\bar{n}) \boldsymbol{S}_2^\dagger(\alpha) \\ &= \boldsymbol{S}_2(\alpha) \sum_{n_1,n_2} \left(\frac{1}{1+\bar{n}}\right)^2 \left(\frac{\bar{n}}{1+\bar{n}}\right)^{n_1+n_2} \left|n_1 n_2\right\rangle\!\left\langle n_1 n_2\right| \boldsymbol{S}_2^\dagger(\alpha) \end{aligned} \tag{14}$$

When one operates a Gaussian state with a non-Gaussian transformation, one is usually left with a non-Gaussian state. Let's consider a two-mode squeezed vacuum

$$\boldsymbol{\rho}_{\text{sq-vac}} = \boldsymbol{S}_2(\alpha)\,|00\rangle\langle 00|\,\boldsymbol{S}_2^\dagger(\alpha).$$

The squeezed vacuum state is a Gaussian state satisfying $\text{Tr}[\boldsymbol{\rho}_{\text{sq-vac}}] = 1$. One can create a two-mode photon added squeezed vacuum (PASV) state by operating the squeezed vacuum with field operators

$$\begin{aligned}
\boldsymbol{\rho}_{\text{PASV}} &= \boldsymbol{a}_1^\dagger \boldsymbol{a}_2^\dagger \boldsymbol{S}_2(\alpha)\,|00\rangle\langle 00|\,\boldsymbol{S}_2^\dagger(\alpha)\boldsymbol{a}_2\boldsymbol{a}_1 \qquad (15)\\
&= \boldsymbol{S}_2(\alpha)\boldsymbol{S}_2^\dagger(\alpha)\boldsymbol{a}_1^\dagger\boldsymbol{S}_2(\alpha)\boldsymbol{S}_2^\dagger(\alpha)\boldsymbol{a}_2^\dagger\boldsymbol{S}_2(\alpha)\,|00\rangle\langle 00|\,\boldsymbol{S}_2^\dagger(\alpha)\boldsymbol{a}_2\boldsymbol{S}_2(\alpha)\boldsymbol{S}_2^\dagger(\alpha)\boldsymbol{a}_1\boldsymbol{S}_2(\alpha)\boldsymbol{S}_2^\dagger(\alpha)\\
&= \boldsymbol{S}_2(\alpha)(\boldsymbol{a}_1^\dagger\cosh|\alpha| + \boldsymbol{a}_2 e^{-i\theta}\sinh|\alpha|)(\boldsymbol{a}_2^\dagger\cosh|\alpha| + \boldsymbol{a}_1 e^{-i\theta}\sinh|\alpha|)\,|00\rangle\langle 00|\times\\
&\quad(\boldsymbol{a}_2\cosh|\alpha| + \boldsymbol{a}_1^\dagger e^{i\theta}\sinh|\alpha|)(\boldsymbol{a}_1\cosh|\alpha| + \boldsymbol{a}_2^\dagger e^{i\theta}\sinh|\alpha|)\boldsymbol{S}_2^\dagger(\alpha)\\
&= \boldsymbol{S}_2(\alpha)(\cosh^2|\alpha|\,|11\rangle + e^{-i\theta}\cosh|\alpha|\sinh|\alpha|\,|00\rangle)\times\\
&\quad(\cosh^2|\alpha|\,\langle 11| + e^{i\theta}\cosh|\alpha|\sinh|\alpha|\,\langle 00|)\boldsymbol{S}_2^\dagger(\alpha)\\
&= \boldsymbol{S}_2(\alpha)\nu'(\nu'\,|11\rangle\langle 11| + \nu''\,|00\rangle\langle 00| + \nu'''\,|00\rangle\langle 11| + (\nu''')^*\,|11\rangle\langle 00|)\boldsymbol{S}_2^\dagger(\alpha),
\end{aligned}$$

where $\nu' = \cosh^2|\alpha|$, $\nu'' = \sinh^2|\alpha|$ and $\nu''' = e^{-i\theta}\cosh|\alpha|\sinh|\alpha|$. In this form the state is no longer Gaussian nor is it real. To make it real one needs to normalize it to $\text{Tr}[\boldsymbol{\rho}_{\text{NASV}}/N_{\text{PASV}}] = 1$ with a normalization factor. The most convenient way for computing this normalization is to divide the state with it's trace. With similar actions one is able to find the two-mode photon subtracted squeezed vacuum (PSSV) state

$$\boldsymbol{\rho}_{\text{PSSV}} = \frac{1}{N_{\text{PSSV}}}\boldsymbol{a}_1\,\boldsymbol{a}_2\boldsymbol{S}_2(\alpha)\,|00\rangle\langle 00|\,\boldsymbol{S}_2^\dagger(\alpha)\boldsymbol{a}_2^\dagger\boldsymbol{a}_1^\dagger. \qquad (16)$$

An important distinction between Gaussian and non-Gaussian states is, that one is generally not able to describe non-Gaussian states with the first two moments. For this task one would need higher order moments like skewness and kurtosis, but other more sophisticated methods also exist [29].

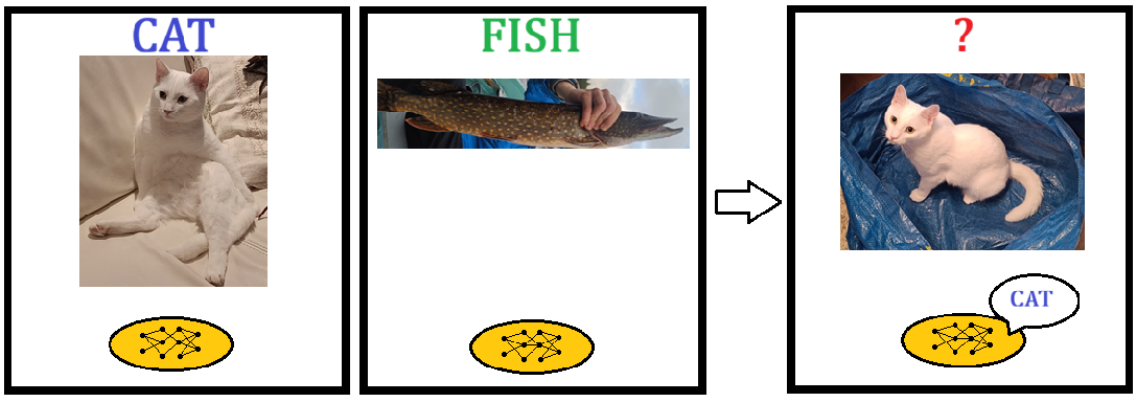

Figure 2. A supervised learning model learns to label unseen data with pre-labeled training data. In a classification task new unseen inputs are labeled according to the training labels. The handsome cat is called Valle.

Subjecting a Gaussian state to non-Gaussian operations is not the only way to find non-Gaussian states. As an example let's consider a two-qubit state

$$|\psi_q\rangle = c_0 |00\rangle + c_1 |11\rangle , \tag{17}$$

where c_0 and c_1 are complex coefficients. Any continuous-variable quantum system that is not Gaussian is in fact non-Gaussian. Generally, where classical bits adopt discrete values 0 or 1, qubits can be found in states $|0\rangle$, $|1\rangle$ or a superposition of them $|\psi_s\rangle = \alpha |0\rangle + \beta |1\rangle$ where $\alpha, \beta \in \mathbb{C}$. Measuring $|\psi_s\rangle$ will project the state into $|0\rangle$ or $|1\rangle$ with respective probabilities $|\alpha|^2$ and $|\beta|^2$.

1.3 Machine learning, neural networks and reservoir computing

Developing a system which requires specific detailed skills or knowledge might be too difficult or expensive to create manually. Machine learning (ML) aims to program a computer to optimize a performance criterion using example data or past experience. In general, different predictive models generalize training data for new data the system has not seen before by minimizing some cost function. Different ML algorithms exist which perform better depending on the task.

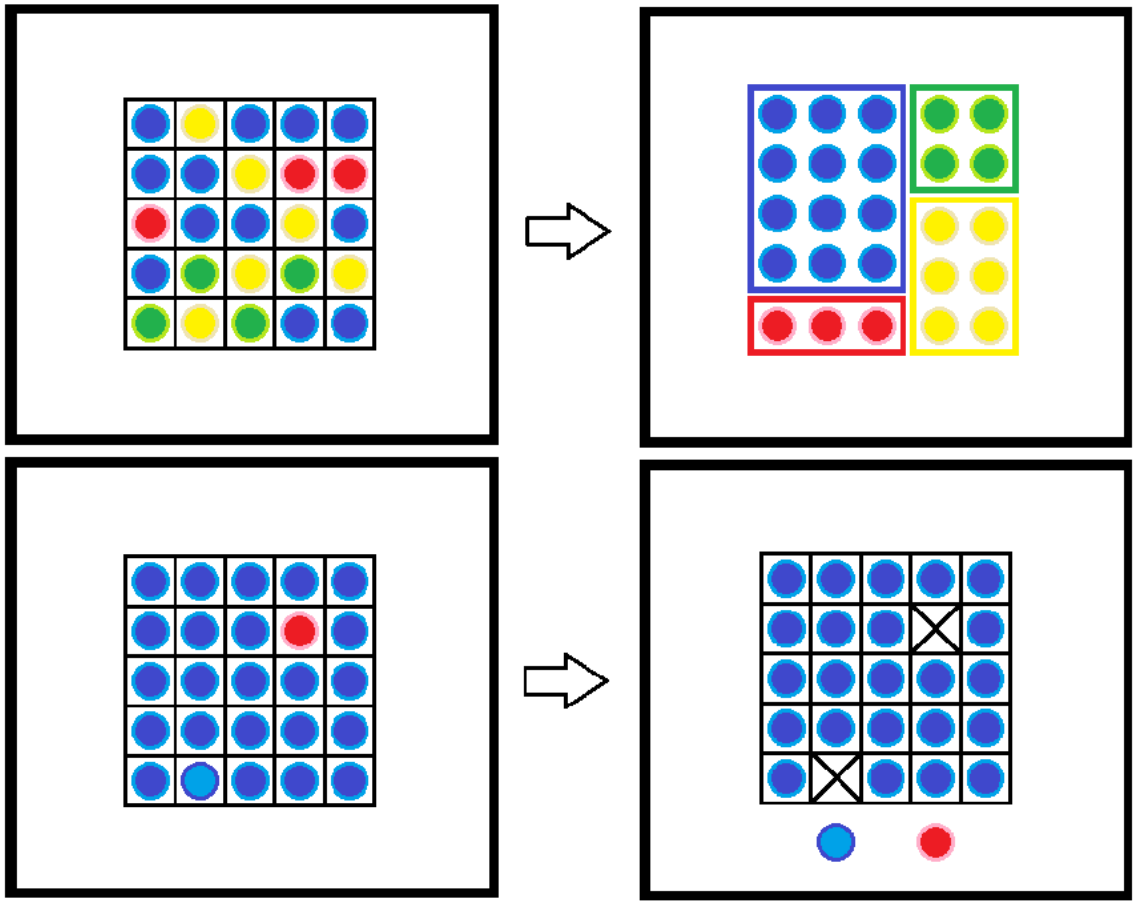

Figure 3. In unsupervised learning an algorithm tries to find meaningful connections in the training data without any pre-existing knowledge. Examples of unsupervised learning algorithms are clustering presented on top and outlier detection presented on the bottom.

Learning can broadly be split into supervised, unsupervised and reinforcement learning. Some algorithms behave in a way characteristic to more than one these groups, but for the sake of clarity they will be addressed separately briefly. In supervised learning a system is trained with pre-labeled data. The task of the ML algorithm is to find patterns and create a mathematical model to label unforeseen data. Supervised learning includes different classification and regression models. The difference being, that classification maps inputs into predefined labels and regression into numerical values. A thematic illustration of supervised learning is presented in Fig. 2.[30]

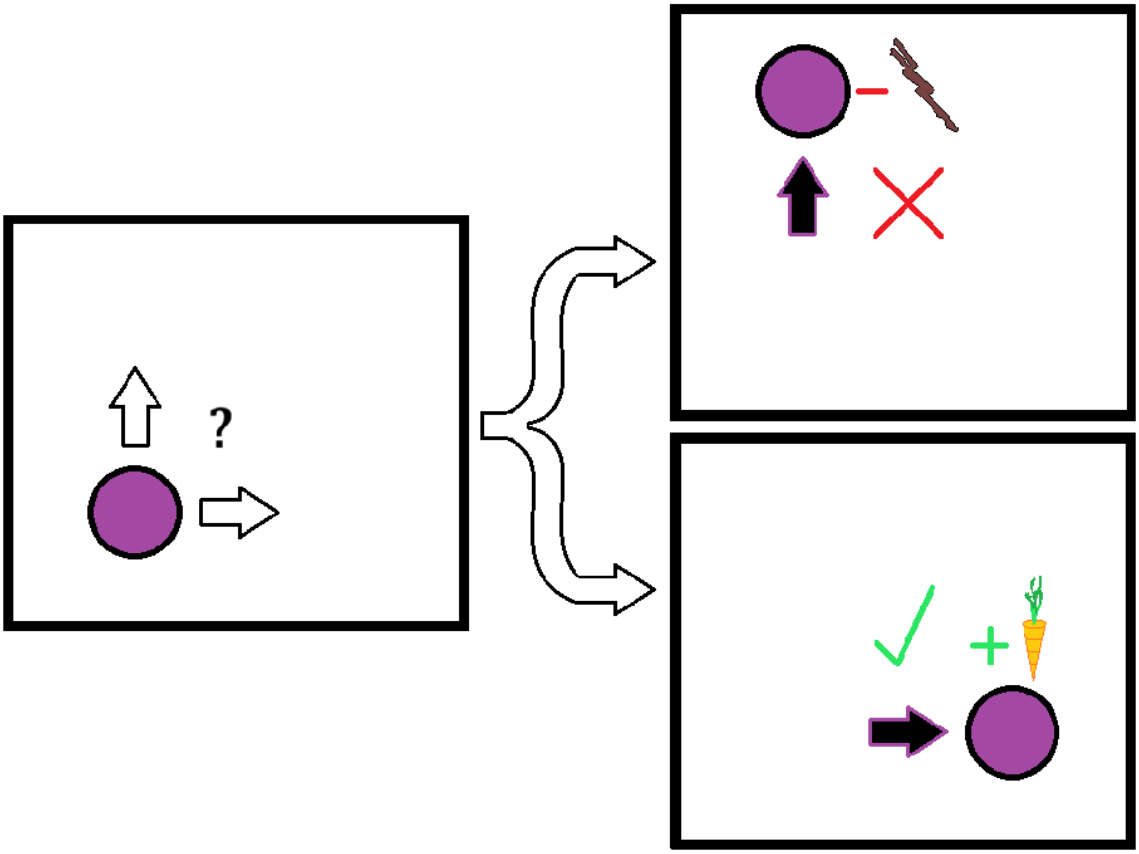

Figure 4. In reinforcement learning an agent learns through trial and error. Correct actions are rewarded reinforcing the behavior.

In unsupervised learning the algorithm is left without any predetermined labels. Through similarities in the training data set, the model is trying to end up with meaningful conclusions [31]. This way unforeseen or even unknown information may be harvested. Different unsupervised learning algorithms are illustrated in Fig. 3. On the other hand, in reinforcement learning an agent is let to interact with an environment. The agent observes it's environment and performs actions. Depending on the action the agent is rewarded or punished. Through trial and error the agent learns an optimal policy to maximize the rewards. This way certain behavior is reinforced and certain behavior diminished. The reinforcement learning process is illustrated in Fig. 4.

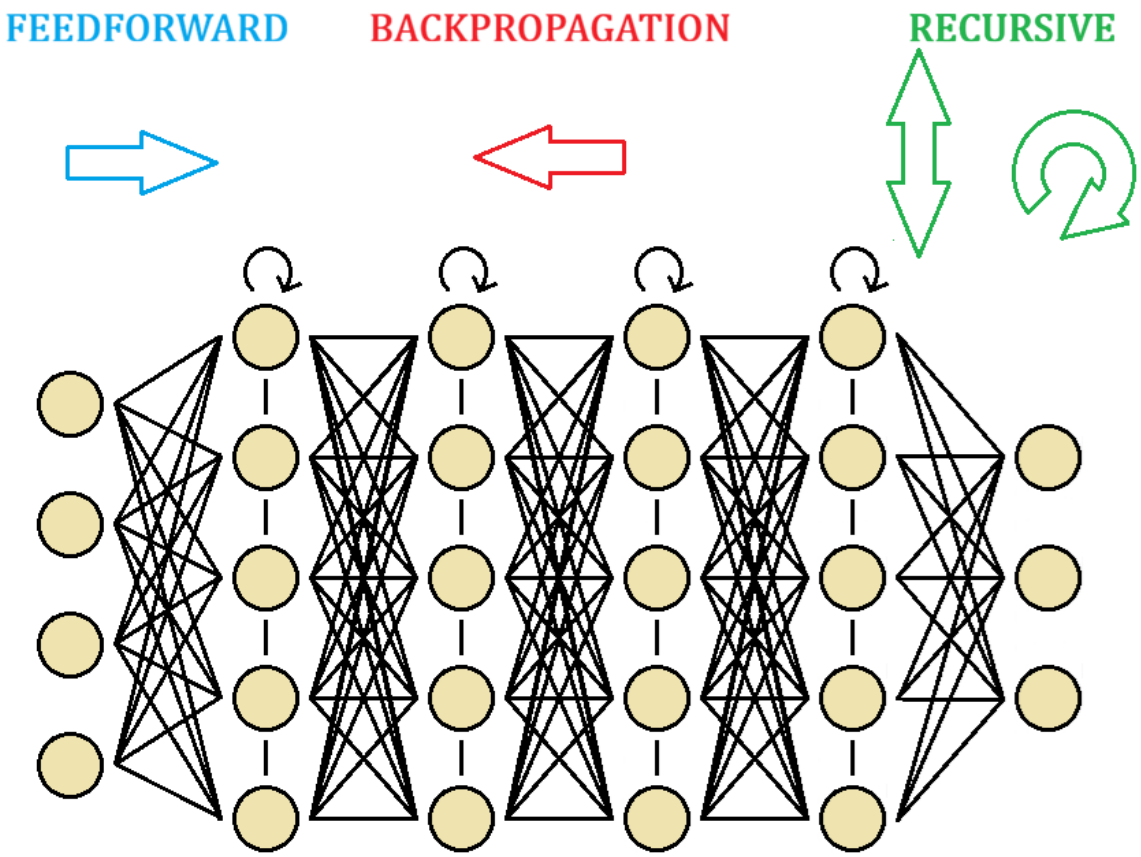

Figure 5. In a feedforward neural network the flow of information is most often presented from left to right. This way the input layer is the leftmost and the output layer the rightmost layer. In backpropagation the information flows from the output layer to the input layer reversing the process. Recursive models allow more complicated connections.

The human brain performs well in complex tasks like vision, motor control and language understanding, but lacks in performance in tasks like mathematical factorization. This is due to the neuron structure of our brains which uses high connectivity and parallelism. Artificial neural networks, or in short neural networks (NN) try to mimic this behavior. Like the biological counterpart the artificial neural networks thrive in similar tasks. In NN information is processed through connected binary indicators called neurons. A perceptron or neural network consists of layers of neurons. More specifically, an input layer, an output layer and an arbitrary amount of hidden layers. A model without hidden layers is called a single-layer perceptron, and

with hidden layers a multilayer perceptron. Learning happens through fine tuning of the internal weights between the neurons.

Depending on the network information may flow in different directions. In a feedforward network information flows only in one direction, from input to output. Additionally, the connections are made only between different layers of neurons. Backpropagation turns this around and allows the information to move from "end to start" reversing the process. Recursive models allow the information to flow inside the neuron layer, between neuron layers and even from a neuron to itself. In Fig. 5 the flow of information inside the neural network is visualized. Better connectivity inside the network usually results in better results. This comes with the cost of more complicated models and greater training times. Complicated models don't always outdo their simpler counterparts. After an "expensive" training session one might end up with sub-optimal behavior. Nevertheless, the possibilities recurrent neural networks (RNN) offer have made them a worthwhile field of study.

Echo state networks (ESN) are a type of supervised temporal RNN, where the adjusting of weights or training happens only on the output layer. The recurrent network is created with random connections and random weights. The temporal aspect means that as inputs are fed at different times, the network is able to keep a fading memory of past inputs [32]. This is also called the echo state property. If a vector x_k describes the state of the network at timestep k, the evolution of the network for input s_{k+1} can be described by

$$x_{k+1} = f(x_k, s_{k+1}),$$

where $f(\cdot)$ is some activation function. As one is only interested in the readout of the network, the model can be though as a "black box" system where the information processing happens in a reservoir. In other words, ESNs are a part of a bigger group of reservoir computing (RC) systems. For the substrate of reservoir computing, one is able to consider different physical systems. In this work this property is further

considered when quantum reservoir computing (QRC) is discussed.

When the readouts of an ESN are linear and feedforward, one can write the readout layer as

$$Y = XW^{\text{out}}, \tag{18}$$

where Y are the target output values, W^{out} the optimal trained weights and X measured output values. For the task of finding the optimal weights one can use a few different ways. The most universal and stable way to do the training is ridge regression [32]

$$W^{\text{out}} = X^{\top}(XX^{\top} + \lambda\mathbb{I})^{-1}Y^{\text{target}}. \tag{19}$$

Here λ is the regularization strength. Ridge regression is a linear regression model that minimizes the mean squared error on the training data. It uses a regularization coefficient that penalizes overfitted data making the model more stable. The training may also be conducted using

$$W^{\text{out}} = X^{+}Y^{\text{target}}, \tag{20}$$

where $X^{+} = (X^{\top}X)^{-1}X^{\top}$ is the Moore-Penrose pseudoinverse. It can be calculated when X has full column rank [32, 33]. To reduce training times, instead of training one large system, one can choose to train several smaller systems. This spatial multiplexing might be a viable option if a larger memory is not a requirement for training.

RC thrives in problems requiring nonlinear information processing. These tasks include but are not limited to time-series prediction, signal processing and different dynamical temporal tasks. For their simplicity, effectiveness and low cost of training, RC models are a great choice for many tasks. On the other hand if one is faced with a task where a fading memory is a liability, one should opt for some different model like transformers [34]. These models are a lot heavier and more expensive to train, but they excel in tasks like natural language processing. In general, transformers

are a relatively new machine learning framework in the research of RNNs [35]. They have brought advanced generalized chatbots such as Ghat-GPT for people to use in their everyday lives. These large language models have revolutionized different tasks with AI assistants, but also carry the unsettling possibility of making some people obsolete as they get replaced by the very same man-made helpers.

1.4 Quantum computing

To fully utilize the opportunities quantum computing offers, it's important to understand the properties and the line between classical and quantum computing. Classical computers are "jack of all trade" machines capable of performing a plethora of tasks. Different parts of the computer are specialized in performing different assignments for a smooth user experience. Information is stored in bits taking values 0 or 1, which on the smallest scale are controlled by capacitors and electrical signals. With the proper tools the computing power of a classical computer can be increased to great lengths. These high performance computing (HPC) environments have been a great asset for academic work and theoretical simulations where traditional computers lack power [36].

Quantum computing (QC) utilizes different physical phenomenon to gain great computing power. In theory, QC offers drastic advantages over it's classical counterpart with it's sheer processing power but also the ability to solve more sophisticated problems. In the center of qubit-based QC is entanglement and superposition. Where a classical system composed of n bits can take 2^n different values one at a time, a quantum system with n qubits is able to represent the same 2^n values as a superposition.

$$\text{Classical bits: } 001\ldots10$$

$$\text{Qubits: } |\psi\rangle = \alpha_1 |000\ldots00\rangle + \alpha_2 |000\ldots01\rangle + \ldots + \alpha_{2^n} |111\ldots11\rangle$$

where $\alpha_i \in \mathbb{C}$ and $\sum_i |\alpha_i|^2 = 1$. Importantly, with qubit-based QC one is able

to perform operations on 2^n states simultaneously offering exponential speedup on classical counterparts. The most studied QC systems are gate-based systems, which have been realized on different quantum substrates [6]. Gate based quantum algorithms may be represented as quantum circuits, where a set of qubits are operated by logical quantum gates. For a system with n qubits, these logical gates are $2^n \times 2^n$ unitary operations. Fundamentally, many aspects of classical computing are different for QC. As the measurement of any quantum system changes or destroys it, the readout of a quantum computer is more intricate [37, 38]. Utilizing a quantum memory also has it's own challenges [39, 40].

The increase in classical computing power with HPC has enabled the simulation of many complicated physics phenomenon previously too heavy for computations. As QC is really difficult to implement on a real quantum computer, one could wonder if a quantum computer could also be simulated on a classical computer. To some extend this is possible with the elements belonging to the Clifford group [41]. Nevertheless, classical computers are unable to produce the true complicated nature of quantum correlations. This property is also called the magic of quantum states [42, 43]. Magic is a measurable quantity and to achieve the quantum advantage in computing one needs magic [44, 45]. Even though the quantum logical gates belonging to the Clifford group are able to be simulated on classical computers, they are also used in fault-tolerant quantum computing when stabilizer code is implemented in quantum error correction (QEC) [46]. In QEC many noisy imperfect physical qubits are combined to produce fault-tolerant logical qubits.

An age old question in classical computational complexity theory exists between the sets of problems solvable in deterministic polynomial time **P** and nondeterministic polynomial time **NP**. Whether or not the sets are equal $\mathbf{P} = \mathbf{NP}$, or as more commonly is believed $\mathbf{P} \subset \mathbf{NP}$. These sets can be further generalized in polynomial

hierarchy (PH) [47] with complexity classes

$$\Sigma_{i+1}^{\mathbf{P}} = \mathbf{NP}(\Sigma_i^{\mathbf{P}})$$
$$\Pi_{i+1}^{\mathbf{P}} = \mathbf{coNP}(\Sigma_i^{\mathbf{P}})$$
$$\Delta_{i+1}^{\mathbf{P}} = \mathbf{P}(\Sigma_i^{\mathbf{P}}).$$

Here **coNP** is the set whose complements are in **NP**, $\Sigma_0^{\mathbf{P}} = \Pi_0^{\mathbf{P}} = \Delta_0^{\mathbf{P}} = \mathbf{P}$, $\Sigma_1^{\mathbf{P}} = \mathbf{NP}$ and $\Pi_1^{\mathbf{P}} = \mathbf{coNP}$. PH follows an inclusion structure

$$\Sigma_k^{\mathbf{P}} \cup \Pi_k^{\mathbf{P}} \subseteq \Delta_{k+1}^{\mathbf{P}} \subseteq \Sigma_{k+1}^{\mathbf{P}} \cap \Pi_{k+1}^{\mathbf{P}} \tag{21}$$

which is true for all $k \geq 0$. Simply put, higher levels of the hierarchy are created with an increasing amount of conditions or proofs C, where entries in the set fulfill these conditions for some $\exists x_i$ or $\forall y_j$ as $C(x_i, y_j)$. The conditions are substantiated by a deterministic polynomial time verifier. Therefore with an increasing level in the hierarchy one is left with sets containing more restrictions. In other words these sets are more powerful in nature with a more complex structure. It is still unknown whether the inclusions in the hierarchy are proper and if

$$\begin{aligned} &\Sigma_k^{\mathbf{P}} \neq \Sigma_{k+1}^{\mathbf{P}}, \text{ for all } \ k \geq 0 \\ &\Sigma_k^{\mathbf{P}} \neq \Pi_k^{\mathbf{P}}, \text{ for all } \ k \geq 1 \\ &\Delta_k^{\mathbf{P}} \neq \Sigma_k^{\mathbf{P}} \cap \Pi_k^{\mathbf{P}}, \text{ for all } \ k \geq 1. \end{aligned} \tag{22}$$

are actually true. If the inclusions in Eq. (21) are not proper or the conditions in Eqs. (22) met meaning the levels of the hierarchy are equal, one will find the collapse of the polynomial hierarchy. In a nutshell the polynomial hierarchy collapses to the level where the conditions are not met, simplifying the complex logical structure. Importantly if $\mathbf{NP} = \mathbf{coNP}$ is proven be true PH would collapse to level 1, and if $\mathbf{P} = \mathbf{NP}$ the structure would collapse all the way to level 0. However this is widely believed to be untrue.

Analogies to PH structures can also be found in quantum computing with quantum polynomial hierarchy (QPH). Although similar, defining and building such a structure for the quantum case is a lot more complicated. Therefore different degrees of quantumness in the structures varies [48, 49]. There also exist several different definitions of QPH. The levels in the purely quantum polynomial hierarchy are substantiated with quantum proofs. Conditions are for quantum states $\boldsymbol{\rho}$ and the verifiers are quantum in nature. An infinite amount of quantum states having infinite dimensionality makes the formulation of the polynomial hierarchy a formidable task. Similarly to the collapse of PH, the collapse of QPH is also widely studied. Simply put, a collapse of the QPH at some level would imply that any additional quantumness introduced would not result in an increase in computing power. Indeed results show and imply a collapse of QPH to a certain level [48–50]. Moreover, if PH = QPH were proven to be true, quantum protocols would be considered obsolete.

Physical qubits are fragile systems prone to errors. Similarly to other physical systems they suffer from the same difficulties as any other real system with their innate property of being open systems. The challenges in QC [6, 51] centralize in the difficulty of scaling. In other words it's really difficult to create working quantum computers with a sufficient amount of computing power. In literature [52] the amount of physical qubits needed for factorizing a 2000-bit number in a reasonable time using the Shor's factorization algorithm [53] has been calculated. Depending if one accounts for the physical qubits needed for quantum QEC protocols, one will need an amount of qubits of the order $10^7 \sim 10^8$. This assumes imperfect fidelities of the physical qubits. Here 2000 being a number of interest, as the standard RSA encryption is based on factorizing a 2048-bit number.

Many problems in gate-based QC such as the difficulty in scaling has spiked an interest in other more unconventional ways to harness and utilize computing power. Moreover combining classical and quantum protocols for information pro-

cessing. Dissecting a computing task and considering the nature of each step offers an interesting point of view to QC [54]. In a fully quantum QC task a quantum input would be injected to some quantum substrate and a quantum task would be performed on the substrate. An example of this could be a state or a set of states $\boldsymbol{\rho}_i$ fed into a qubit-based quantum computer for some gate-based quantum algorithm. On the other hand the input could also be a scalar value making it classical in nature. The combinations are not limited to a single task with different aspects. In variational quantum algorithms (VQA) [55] parameters used in quantum routines are optimized using classical means. This way subroutines are used to make a more robust quantum algorithm.

QRC offers a robust alternative for QC. As with all RC, one is mainly interested in the readout of the reservoir, and the internal happenings are rarely meaningful. This makes QRC inherently more fault-tolerant, as the exact quantum state of the system is not important. Protocols like QEC are therefore less prominent and in some cases the noise of the system can be learned or even used as an advantage [54, 56]. Another great advantage of QRC is the relatively small size of reservoirs needed for meaningful results. QRC can be realized with gate-based systems [57], but also more generalized implementations of quantum systems are commonly used [58, 59]. QRC systems thrive in similar problems as classical RC systems [33] and are also able to solve more sophisticated problems like entanglement detection [8, 60]. This work studies these entanglement classification systems.

2 Classical reservoir computing

To better understand the properties of QRC, let's for now consider an example of the classical counterpart ESN and it's properties. Let there be a reservoir of N neurons, and a vector x describing the states of the neurons. With inputs s the evolution of

the reservoir can be presented with an activation function of the network

$$x_{k+1} = \tanh(\varepsilon W x_k + b + \iota v s_{k+1}). \tag{23}$$

Here ε and ι are the feedback gain and input gain controlling the relative importance between the previous state and the input. x_k is the state of the reservoir at time k and W, b, v are a $N \times N$ matrix and $N \times 1$ vectors of uniformly chosen random values between $[-1, 1]$. Additionally, W is scaled to have a spectral radius of 1, meaning that the greatest absolute value of it's eigenvalues does not exceed 1. In short, W describes the internal connections of the reservoir, v the input strength and b is a constant vector. To facilitate reservoir computing, an upper bound on the feedback gain ε should be imposed [33].

The ESN is able to process nonlinear data through it's nonlinear internal connections. The properties of an ESN may be tested with a suitable benchmark task, Nonlinear autoregressive moving average (NARMA). NARMA is able to test the reservoir's ability to produce nonlinear functions and to recover inputs from the past [33]. For a uniformly distributed input $s \in [-1, 1]$, NARMA is defined as

$$y_{k+1} = \alpha_N y_k + \beta_N y_k \sum_{\tau=0}^{n-1} y_{k-\tau} + \gamma_N s_{k-N+1} s_k + \delta_N, \tag{24}$$

where $\alpha_N, \beta_N, \gamma_N$ and δ_N are real values and y_k is the value of NARMA at time step k. The degree of NARMA dictates the length of memory of inputs n needed to produce correct outputs. With normalized mean squared error (NMSE), one can measure how the reservoir manages the benchmark task

$$\mathrm{NMSE}(\bar{o}, o) = \frac{\sum_i (\bar{o}_i - o_i)^2}{\sum_i \bar{o}_i^2}. \tag{25}$$

Here $\bar{o}_k$ is the true value of NARMA y_k and o_k the output of the reservoir computing system. The values of NMSE range from $[0, 1]$, where 0 corresponds to a perfect match and 1 a complete failure.

Let's consider an ESN with an activation function from Eq. (23). We'll study the impact of spatial multiplexing on a system with 4 modes. Additionally the

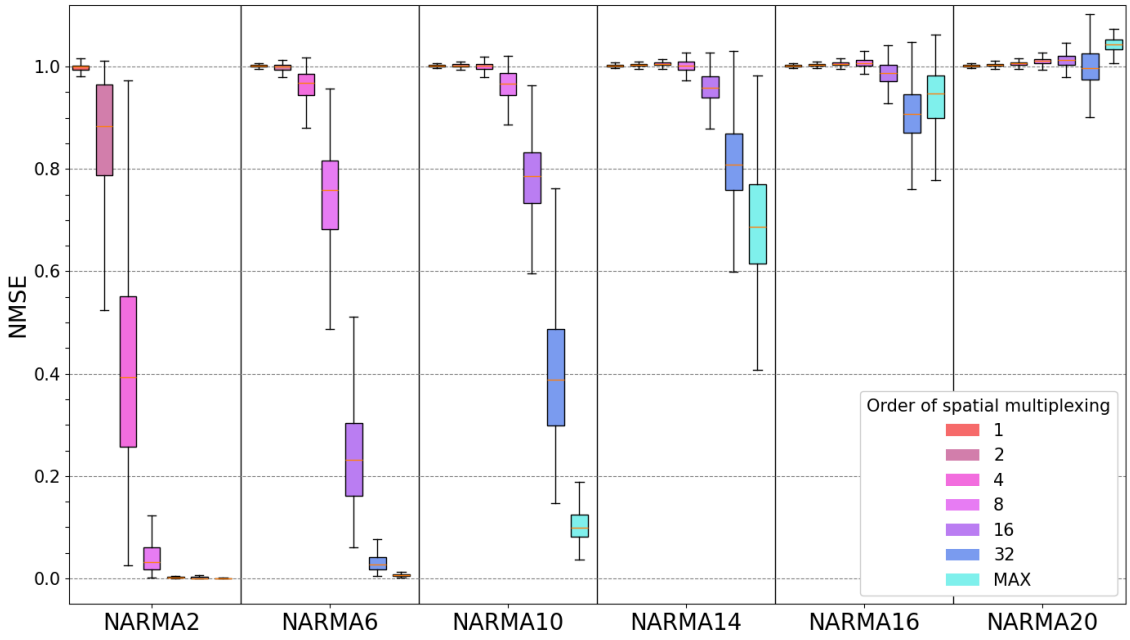

Figure 6. A 4-mode ESN with different orders of spatial multiplexing compared to a single 128-mode component ESN MAX. All the networks are tested with different orders of NARMA and evaluated with NMSE. Final results are averaged over 500 systems. The performance of the ESNs eventually drop for greater orders of NARMA.

results will be compared to a single system with 128 modes. We'll set the feedback gain as $\varepsilon = 0{,}9$ for systems with spatial multiplexing and $\varepsilon = 1{,}0$ for a single bigger component. An input gain of $\iota = 0{,}05$ will be used across all systems. The properties of the systems will be tested with the degrees $2, 6, 10, 14, 16, 20$ of the NARMA task described in Eq. (24). The real values $\alpha_N, \beta_N, \gamma_N$ and δ_N will be set as $0{,}3, 0{,}05, 0{,}375$ and 0 respectively. The inputs $s \in [-1, 1]$ will be fed into the reservoir in the preparation, training and testing phases. The amount of inputs used in these phases are $1000, 3000$ and 5000 respectively. The preparation phase is used to remove the dependence of the initial state of reservoir on the output. In this phase no readouts are collected. In the training phase the readouts are collected into matrix X where X_{ij} is the j^{th} readout at timestep i. A unit column is appended to X to account for a bias weight term $W^\top = (w_1, w_2, \ldots, w_0)$. The training of weights is conducted using the pseudoinverse described in Eq. (20). Finally, in the

testing phase with the readouts X and trained weights W^{out} one is able produce the reservoir output Y as in Eq. (18), and find out the performance of the task with NMSE described in Eq. (25). All results will be averaged over 500 systems.

In Fig. 6 the comparison of performance in different orders of NARMA task for different orders of spatial multiplexing ESN networks and a single component network is shown. The systems with spatial multiplexing are indicated with numbers, and the single component as MAX. From the figure it's apparent that the systems with greater spatial multiplexing perform better in the task. Eventually, as the degree of NARMA grows and the tasks become more difficult, the performance of all models deteriorate. This is due to the lack of sufficient nonlinear memory in the systems. Some discrepancies were noticed during the training. Firstly, some models were found to be instable, not being able to converge. This produced NMSE values far greater than 1. The problem wasn't universal and mainly affected the system with spatial multiplexing 32. Outlier values are not shown in the plots. Secondly, for the last tasks the performance of the big component model performs worse than the spatially multiplexed counterpart. Some instability in the model can therefore also be assumed for greater orders of NARMA. The choice of parameters ε and ι was conducted via grid search. A different choice for the hyperparameters might have resulted in more stable results.

3 Quantum reservoir computing

In this section the quantum reservoir computing simulation of this work is thoroughly explained. The master equation of the system is unveiled, internal reservoir dynamics are explained and the process of input state creation is presented. Some discussion on the heaviness of the simulation and an alternative way of conducting the simulations for Gaussian states is introduced.

3.1 Entanglement classifier

Quantum reservoir dynamics

With the nonlinear information processing capability, QRCs are a great alternative for certain demanding classes of quantum tasks. The scope of this work is to build a working QRC system for classifying the entanglement of certain classes of quantum states. The QRC system in question consists of a reservoir of an arbitrary amount of fermionic modes and a set of two-mode bosonic inputs. The system takes inspiration from literature [8], and in this work different configurations of the quantum reservoir are tested. The whole system can be described with a combined density matrix and a master equation

$$\begin{aligned}\dot{\rho} = &-\frac{i}{\hbar}[H_{\mathrm{R}}, \rho] + \frac{\gamma}{2\hbar}\sum_j \mathcal{L}(b_j) + \frac{P}{2\hbar}\sum_j \mathcal{L}(b_j^\dagger) \\ &+ \frac{1}{\hbar}\sum_{k,j} f_k(t) W_j^{\mathrm{in}} \left([a_k\rho, b_j^\dagger] + [b_j, \rho a_k^\dagger]\right) \\ &+ \frac{\eta}{2\gamma\hbar}\sum_k f_k(t)\mathcal{L}(a_k).\end{aligned} \tag{26}$$

The unitary Hamiltonian of the reservoir, the Fermi-Hubbard Hamiltonian is

$$H_{\mathrm{R}} = \sum_{ij} J_{ij}(b_i b_j^\dagger + b_j^\dagger b_i),$$

where $J_{ij} \in [-\gamma, \gamma]$ are random hopping amplitudes. The combined density matrix of the inputs and reservoir is presented with ρ and a_k and b_j are the respective field operators. Parameter γ is the decay rate, $P = 0{,}1\gamma$ the incoherent pumping strength and $\eta = \sum_j (W_j^{\mathrm{in}})^2$. The last two lines of Eq. (26) account for the cascaded formalism [61]. The functions $f_k(t)$ indicate which mode of the input is interacting with the reservoir. In our case of two-mode inputs, when the first mode interacts with the reservoir $f_1(t) = 1$ and $f_2(t) = 0$ for $t_1 < t < t_1 + \tau$ and when the second mode interacts $f_1(t) = 0$ and $f_2(t) = 1$ for $t_1 + \tau < t < t_1 + 2\tau$. $f_{1,2} = 0$ for any other times. Parameter t_1 describes the time the reservoir takes to reach a steady state

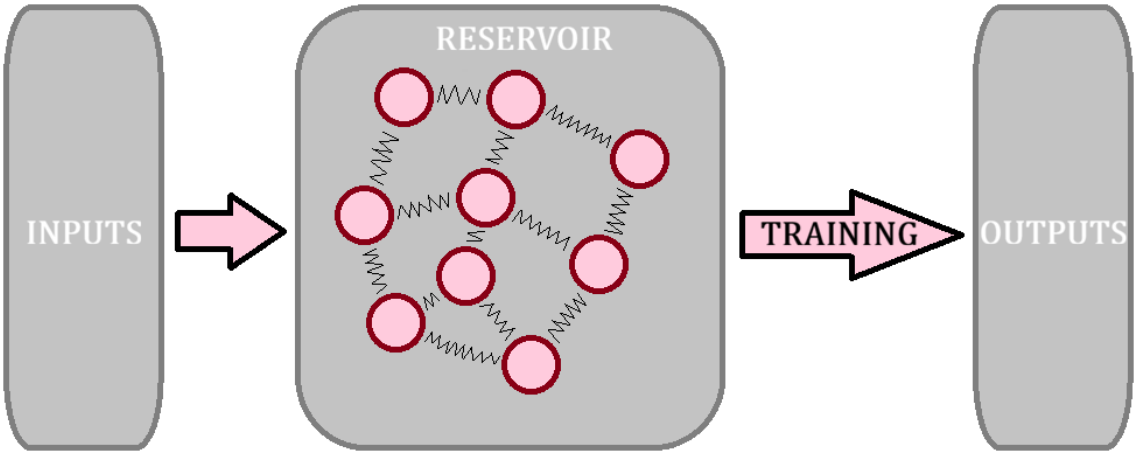

Figure 7. As inputs are fed in the reservoir one at a time, the couplings change the state of the reservoir. The change varies depending on the entanglement of the input. These minor differences are able to be learned in training from the measured reservoir occupation numbers.

and $\tau = \frac{\hbar}{\gamma}$. The input weight matrix $W^{\text{in}} \in [0, \gamma]$ is randomly uniformly distributed. Lastly, the Lindblad operator reads as

$$\mathcal{L}(x) = 2x\rho x^\dagger - x^\dagger x\rho - \rho x^\dagger x.$$

An illustration of the QRC system is depicted in Fig. 7. In this work we take $\hbar = 1$ and $\gamma = 1$.

The desired output, the correct classification of entanglement of input states is found with

$$Y_j^{\text{out}} = \sum_j W_{ij}^{\text{out}} n_j.$$

Here Y^{out} is a 2D vector indicating the entanglement of a given input state, W^{out} is the optimal output weight matrix and $n_j = \langle b_j^\dagger b_j \rangle = \text{Tr}[\rho b_j^\dagger b_j]$ the occupation numbers of the reservoir fermionic sites measured at $t_1 + 2\tau$ after each input. For each input the true nature of entanglement is calculated using logarithmic negativity $E_{\mathcal{N}}(\rho_{\text{in}})$ from Eq. (4) or decided upon state creation. If the state is entangled $Y_i^{\text{true}} = (1, 0)$ and $Y_i^{\text{true}} = (0, 1)$ otherwise.

Each input is fed into the reservoir by letting it interact with the reservoir sepa-

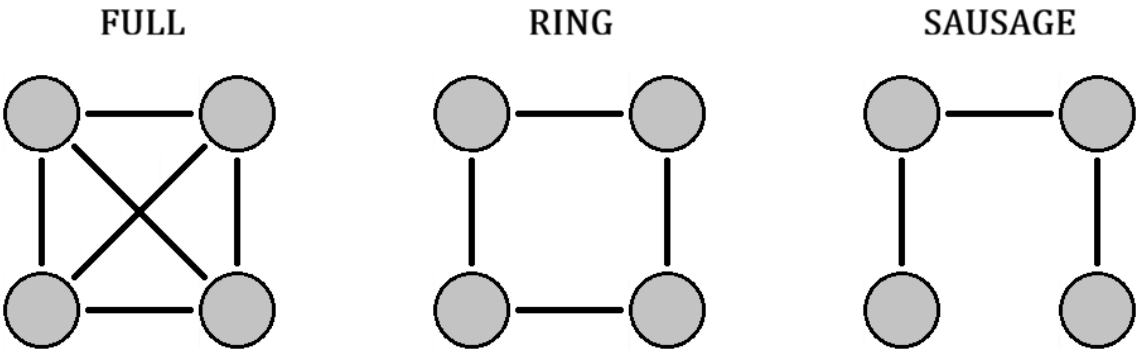

Figure 8. In this work, reservoir networks are classified as fully connected (FULL), a connected ring (RING) or a connected sausage (SAUSAGE).

rately. The reservoir is initialized to the vacuum state and the training of the model is conducted using ST states in Eq. (14). With the measured occupation numbers n_j and true values of entanglement, the output weights are optimized using ridge regression described in Eq. (19). The learning process of updating the weights is relatively simple:

$$W^{\text{out}}_{i,t+1} = W^{\text{out}}_{i,t} - \upsilon \frac{\partial E}{\partial W_i} \tag{27}$$

Here E is the error or cost function and υ the learning rate. The error function is simply

$$E = \frac{1}{2}(Y^{\text{out}}_{\text{true}} - Y^{\text{out}}_{\text{pred}})^2 + \frac{\lambda}{2}||W^{\text{out}}||^2_2$$
$$E = \frac{1}{2}(Y^{\text{out}}_{\text{true}} - \sum_j W_j n_j)^2 + \frac{\lambda}{2}\sum_j W_j^2,$$

which makes the differential in Eq. (27)

$$\frac{\partial E}{\partial W_i} = -n_i(Y^{\text{out}}_{\text{true}} - \sum_j W_j n_j) + \lambda W_i$$

Here λ is the regularization strength.

In this work different substrates for reservoir computing are tested. Reservoir sizes of 2, 3 and 4 fermionic modes are assessed and different configurations of network connectivity are studied. In Fig. 8 a fully connected, a connected ring and a sausage network are visualized with 4 modes. It's important to emphasize that

for reservoir sizes of 2 and 3 some classes of connectivities overlap creating same networks. In this case the systems are labeled as the most connected network the classes allow.

Even though the training is conducted using only a single class of states, the entanglement classification capability expands to also other classes of states. Therefore during the testing phase, the model is evaluated with ST, PASV, PSSV and number or NUM states found in Eqs. (14)–(17) respectively. This way the recognition of both Gaussian and non-Gaussian entanglement may be tested. During testing, if the first element of Y_i^{out} is greater than the second, the model predicts the state to be entangled and with the second value greater the opposite.

Used input states

To achieve a varying amount of entanglement in each class of states, all states are created randomly with the parameters introduced in [8]. For ST states the squeezing parameter is written as $\alpha = |\alpha|e^{i\theta}$ where $|\alpha| = s\text{sin}\phi$ and the average thermal occupation number as $\bar{n} = s^2\text{cos}^2\phi$. Parameters θ, s and ϕ are uniformly distributed from $[0, 2\pi]$, $[0.8, 0.95]$ and $[0.5-\pi/10, 0.5+\pi/10]$ respectively. ST states are classified to entangled and separable by evaluating the logarithmic negativity of a given state truncated to 2 decimals. If logarithmic negativity is still $E_{\mathcal{N}}(\rho_{\text{in}}) > 0$ after the truncation, the state is classified as entangled. This way roughly an equal amount of entangled and separable states is achieved. With PASV and PSSV states the squeezing parameter is once again $\alpha = |\alpha|e^{i\theta}$. PASV states are created by uniformly choosing $|\alpha| \in [0.1, 0.25]$ and $\theta \in [0, 2\pi]$, and PSSV states with uniformly choosing $|\alpha| \in [0.8, 0.95]$ and $\theta \in [0, 2\pi]$. Entangled PASV and PSSV states are created according to Eqs. (15) and (16). Separable counterparts are created similarly

assuming only classical correlations

$$\rho_{\text{PASV-sep}} = \frac{1}{N_{\text{PASV-sep}}} a^\dagger S_1(\alpha_1) \left|0\right\rangle\!\left\langle 0\right| S_1^\dagger(\alpha_1) a \otimes a^\dagger S_1(\alpha_2) \left|0\right\rangle\!\left\langle 0\right| S_1^\dagger(\alpha_2) a$$
$$\rho_{\text{PSSV-sep}} = \frac{1}{N_{\text{PSSV-sep}}} a S_1(\alpha_1) \left|0\right\rangle\!\left\langle 0\right| S_1{}^\dagger(\alpha_1) a^\dagger \otimes a S_1(\alpha_2) \left|0\right\rangle\!\left\langle 0\right| S_1^\dagger(\alpha_2) a^\dagger.$$

Here $N_{\text{PASV-sep}}$ and $N_{\text{PSSV-sep}}$ are normalization coefficients and $S_1(\alpha)$ the single mode squeezing operator from Eq. (13). The hyperparameter ranges for creating PASV and PSSV states are chosen in a way that their average occupation numbers are similar to the training ST states [8]. Entangled NUM states are created similarly to Eq. (17). Similar in a sense that the created NUM states mimic qubit states, but actually have the same degrees of freedom as any other input state making them more general number states. The parameterization of states is done with $c_0 = \sin\theta$ and $c_1 = \cos\theta e^{i\phi}$, which are sampled uniformly on a Bloch sphere. In our case the uniform sampling refers to a uniform sample of input state entanglement values. This is achieved by uniformly sampling a random variable $X \in [0, 1]$ and calculating $\theta = 0.5 \cdot \arcsin(x)$, and uniformly sampling $\phi = [0, 2\pi]$. For separable states, roughly an equal amount of $|00\rangle$ and $|11\rangle$ states are used.

Numerical simulation

In the simulation everything revolves around a combined density matrix of the input and the reservoir. For the sake of clarity, let's briefly explain what happens to the reservoir over one interaction with an input state. To begin, we create the combined density matrix of an input and the reservoir $\rho = \rho_{\text{in}} \otimes \rho_{\text{res}}$. We then numerically simulate the dynamics described by the master equation in Eq. (26) with the 4^{th} order Runge-Kutta method (RK4). The Runge-Kutta method is a numerical iterative method for solving initial value problems. Therefore to calculate the value of a function at the next timestep, one needs an initial value y_0 and to choose a step size h. If $\frac{dt}{dy} y(t) = f(t, y)$, one can numerically calculate

$$
\begin{aligned}
k_1 &= f(t_n, y_n) \\
k_2 &= f(t_n + 0.5h, y_n + 0.5hk_1) \\
k_3 &= f(t_n + 0.5h, y_n + 0.5hk_2) \\
k_4 &= f(t_n + h, y_n + hk_3) \\
y_{n+1} &= y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\
t_{n+1} &= t_n + h
\end{aligned}
$$

In our case, to simulate the system for the time of one input interaction 2τ, we will calculate 4 rounds of RK4 making the step size $h = 2\tau/4 = 0.5$. This is done to ensure sufficient interaction times for both input modes. After the interaction period is finished, the input is eliminated from the density matrix $\rho_{\mathrm{R}} = \mathrm{Tr}_{\mathrm{in}}[\rho]$ similarly to Eq. (2). Finally the average occupation numbers n_j are collected and the cycle starts over again. This process is presented in the **Metacode** below.

QRC simulation for Entanglement classifier

QRC_system : Initialized QRC system

train_inputs : Initialized 200 train ST states

test_inputs : Initialized 200 test ST, PASV, PSSV and NUM inputs

for input **in** train_inputs:

⇒Numerically simulate the interaction of an input and the QRC_system

⇒Measure reservoir outputs n_j

Train the output weights W^{out} with n_j and Ridge regression

for class_of_states **in** test_inputs:

⇒**for** input **in** class_of_states:

→Numerically simulate the interaction of an input and the QRC_system

→Measure reservoir outputs n_j

⇒Predict the entanglement of input states with n_j and evaluate model performance

For smooth functions, RK4 has a truncation error dependent on the step size $\mathcal{O}(h^4)$ [62]. This means that halving h and doubling the numerical calculations results roughly in an error 16 times smaller. As the two-mode input mode interacts with the reservoir one mode at a time, the interaction can be formalized with a piecewise function. To achieve errors similar to smooth functions, RK4 steps should not be taken over the non-smooth part of the piecewise function. This way each individual step of RK4 can be thought of an approximation of a smooth function. The dynamics of the system in Eq. (26) require, that the input modes don't interact with the reservoir in the beginning and end of the interaction and in between the switch of modes. Therefore some discontinuous parts are included in the calculated RK4 steps. The actual numerical error produced by RK4 is therefore most likely greater than $\mathcal{O}(h^4)$.

The classical and quantum reservoir computing simulations in this work are run on python on a basic personal laptop. They are mainly built with basic python functions and science and mathematics based python packages like Numpy and Scipy. This is due to the specific dynamics needed for the system like cascaded formalism. Dynamics like this were not conveniently available in common quantum based packages. Having to build every tool has made it so that the majority of time spent for this work was spent on programming. Especially finding convenient ways to optimize heavy simulations. This is further discussed in section 3.2.

3.2 Computational complexity of the simulation

As the simulation revolves around the combined density matrix, changing the parameters effecting it's size can also make a great impact on simulation times. Before discussing the size of the density matrix, let's first take a look at the amount of matrix multiplications in the simulation. This way we can get an idea of the heaviness of the simulation. Let's consider the master equation describing the system

dynamics in Eq. (26). If we represent every operator with a matrix, which is the size of the combined density matrix we need to perform

$$24 \cdot (\text{reservoir size}) + 6$$

matrix multiplications. With a reservoir of 4 modes this means 102 multiplications. Now, in our case we numerically simulate the system with RK4, which has to calculate the value of the function 4 times. Additionally we take 4 steps of RK4 to simulate the interaction of the input and reservoir. Therefore, without any optimizations we need to make $4 \cdot 4 \cdot 102 = 1632$ matrix multiplications to simulate just one interaction of an input and a 4-mode reservoir. This amount can be drastically decreased with some algebra. For this let's move the terms around in the master equation.

$$\begin{aligned}
\dot{\rho} =& -iH_{\mathrm{R}}\rho + i\rho H_{\mathrm{R}} \\
&+ \frac{\gamma}{2}\sum_j (2b_j\rho b_j^\dagger - b_j^\dagger b_j\rho - \rho b_j^\dagger b_j) \\
&+ \frac{P}{2}\sum_j (2b_j^\dagger\rho b_j - b_j b_j^\dagger\rho - \rho b_j b_j^\dagger) \\
&+ \sum_{kj} f_k(t)W_j^{in}(a_k\rho b_j^\dagger - b_j^\dagger a_k\rho + b_j\rho a_k^\dagger - \rho a_k^\dagger b_j) \\
&+ \frac{\eta}{2\gamma}\sum_k f_k(t)(2a_k\rho a_k^\dagger - a_k^\dagger a_k\rho - \rho a_k^\dagger)
\end{aligned}$$

$$\begin{aligned}
\dot{\rho} =& \sum_{jk}((-iH_{\mathrm{R}} - \frac{\gamma}{2}b_j^\dagger b_j - \frac{P}{2}b_j b_j^\dagger - \frac{\eta}{2\gamma}f_k(t)a_k^\dagger a_k - f_k(t)W_j^{in}b_j^\dagger a_k)\rho \\
&+ \rho(iH_{\mathrm{R}} - \frac{\gamma}{2}b_j^\dagger b_j - \frac{P}{2}b_j b_j^\dagger - \frac{\eta}{2\gamma}f_k(t)a_k^\dagger a_k - f_k(t)W_j^{in}a_k^\dagger b_j) \\
&+ (\gamma b_j + f_k(t)W_j^{in}a_k)\rho b_j^\dagger + Pb_j^\dagger\rho b_j + f_k(t)(W_j^{in}b_j + \frac{\eta}{\gamma}a_k)\rho a_k^\dagger \\
\\
=& \sum_{jk}(A\rho + \rho B + C\rho b_j^\dagger + Pb_j^\dagger\rho b_j + D\rho a_k^\dagger a_k)
\end{aligned}$$

Using the Einstein summation convention

$$A = -iH_{\mathrm{R}} - \frac{\gamma}{2} b_j^\dagger b_j - \frac{P}{2} b_j b_j^\dagger - \frac{\eta}{\gamma} f_k(t) a_k^\dagger a_k - f_k(t) W_j^{in} b_j^\dagger a_k$$
$$B = iH_{\mathrm{R}} - \frac{\gamma}{2} b_j^\dagger b_j - \frac{P}{2} b_j b_j^\dagger - \frac{\eta}{\gamma} f_k(t) a_k^\dagger a_k - f_k(t) W_j^{in} a_k^\dagger b_j$$
$$C = \gamma b_j + f_k(t) W_j^{in} a_k$$
$$D = f_k(t)(W_j^{in} b_j + \frac{\eta}{\gamma} a_k).$$

With this modification we are able to bunch up several unchanging operators and essentially don't need any additional matrix multiplications. Therefore for one round of calculations one needs to only make

$$4 \cdot (\text{reservoir size}) + 4$$

matrix multiplications. The last 4 multiplications are only needed when an input is interacting with the reservoir. With 4 modes and 4 rounds of RK4, the total amount is 320 (the actual amount is even a bit smaller due to calculations of non-interactions), which is a significant decrease.

The true benefit of decreasing the amount of multiplications becomes evident when the size of the matrices is considered. In this work the dimensions of a two-mode input density matrix are effected by the chosen Fock-space truncation. For every added energy-level the dimensions grow quadratically. On the other hand adding a single mode to the reservoir grows the dimensions of the reservoir exponentially. The size of the density matrix may therefore be presented as

$$(\text{Fock space truncation})^2 \cdot 2^{(\text{reservoir size})}.$$

In this work the Fock space truncated to 5 degrees of freedom. For a reservoir of 4 modes this gives us a combined density matrix with dimensions 400×400. This makes the simulation incredibly heavy and the need for small optimizations is crucial. With this setup running 200 inputs through the reservoir took on average

around 20-25 minutes. Decreasing the amount of reservoir modes decreased this time to around 2 minutes for 3 modes and to 20 seconds for 2 modes. Some of the operators used in the simulation are sparse matrices. In this work the sparsity is not utilized in any way nor was the possible optimizations regarding it studied.

Gaussian states can be simulated in the Hilbert space or in the phase space. The dynamics of an equation of motion of a density matrix may be translated to an equation motion of a covariance matrix. Let's consider a multipartite system described by a master equation in Eq. (5). One could consider a reservoir and inputs with respective field operators a_i, b_j such a system. From the master equation, one can distinguish the effective Hamiltonian and jump terms. Now, in the Heisenberg picture one is able to find the equations of motions of field operators, the Langevin equations

$$\dot{a}_i = i[H_{\text{eff}}, a_i] + \text{noise}$$
$$\dot{b}_j = i[H_{\text{eff}}, b_j] + \text{noise}.$$

The terms resulting from the effective Hamiltonian account for the deterministic part and from the jump terms the stochastic part, the noise. As described in section 1.2, the Langevin equations can be rewritten with quadratures from Eqs. (6) and (7) resulting in a set of dimensionless position and momentum equations of motions. The new equations can be conveniently rewritten in an equation

$$\dot{u} = Au + h,$$

where $u = (q_0, p_0, q_1, p_1, \ldots)^\top$ is a vector of quadratures. Matrix A accounts for all the deterministic parts of the equations and vector h for the stochastic terms. With u one is able to construct the covariance matrix in Eq. (12) of the system. Furthermore, with the covariance matrix σ and the deterministic and stochastic terms of the dynamics one can build the equation of motion of the covariance matrix

$$\dot{\sigma} = A\sigma + \sigma A^\top + D. \tag{28}$$

Here D is a diagonal matrix of the stochastic dynamics. [60, 63]

Let's consider the QRC system in this work from Eq. (26) simulated in the phase space with Eq. (28). Similarly to the combined density matrix, the dimensions of the covariance matrix describing the state of the system are also effected by the setup of our system. The big difference here being, that one doesn't have to care about Fock space truncation as we are studying the dynamics in phase space. For every mode we add either as an input mode or as a reservoir mode the dimensions of the respective covariance matrix grows by $2n$. The state of a QRC system of two-mode input and a reservoir of 4 modes can therefore be presented with a covariance matrix $\sigma_{12\times12}$. This is possible as before any interaction there exist no correlation between the inputs and the reservoir and the initial combined covariance matrix is

$$\boldsymbol{\sigma}(0) = \begin{pmatrix} \boldsymbol{\sigma}_{\text{in}}(0) & 0 \\ 0 & \boldsymbol{\sigma}_R(0) \end{pmatrix}.$$

With the parameters described above performing the calculations in the phase space seems to be a lot more convenient numerically than in the Hilbert space. In this work the input states studied are Gaussian and non-Gaussian. Therefore the use of covariance matrices was deemed inadequate for all purposes, and a more general approach was chosen with density matrices. Some effort was put into finding a convenient way to adapt the non-Gaussian elements to the covariance matrix simulations but no good fit was found.

4 Entanglement classification results

In this work the training of each system is conducted with 200 input states. Likewise, each class of states is tested with 200 input states. During the start of testing of a new class of states, the reservoir is returned to the state right after training. This is done to ensure similar initial conditions for all testing experiments. In total six different substrates or models are tested: 4-mode fully connected, connected ring

and connected sausage reservoirs, 3-mode fully connected and sausage reservoirs and 2-mode fully connected reservoirs. In the upcoming tables and figures these models will be labeled respectively as '4-mode FULL', '4-mode RING', '4-mode SAUSAGE', '3-mode FULL', '3-mode SAUSAGE' and '2-mode FULL'. The final results are averaged over 10 different systems.

The entanglement classification rates for all models and input types are presented in tables I–VI. A classification rate of 1.0 equals a perfect entanglement classification, where a classification rate of 0.5 results from a complete failure. A fluctuation margin of one standard deviation is presented along the results. The results are further visualized in Figs. 9–14 with histograms, and the classification capabilities for entangled and separable states in figures Figs. 15–20 with confusion matrices.

Table I. 4-mode FULL

Input class	Classification rate
ST	0.921±0.043
PASV	0.666±0.101
PSSV	0.781±0.113
QUBIT	0.702±0.052

Table II. 4-mode RING

Input class	Classification rate
ST	0.921±0.055
PASV	0.669±0.073
PSSV	0.837±0.137
QUBIT	0.714±0.049

Table III. 4-mode SAUSAGE

Input class	Classification rate
ST	0.916±0.029
PASV	0.629±0.102
PSSV	0.890±0.095
QUBIT	0.714±0.046

Table IV. 3-mode FULL

Input class	Classification rate
ST	0.858±0.060
PASV	0.67±0.070
PSSV	0.958±0.038
QUBIT	0.715±0.022

Table V. 3-mode SAUSAGE

Input class	Classification rate
ST	0.829±0.076
PASV	0.648±0.024
PSSV	0.978±0.029
QUBIT	0.711±0.037

Table VI. 2-mode FULL

Input class	Classification rate
ST	0.816±0.087
PASV	0.635±0.077
PSSV	0.978±0.038
QUBIT	0.735±0.034

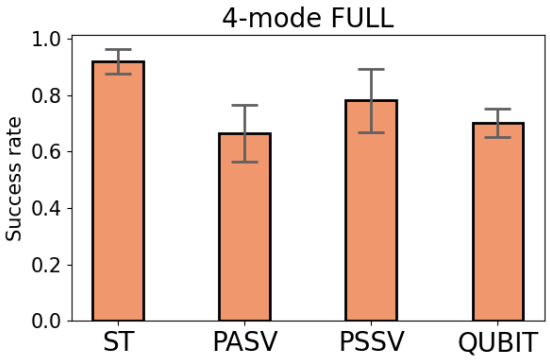

Figure 9. Classification results.

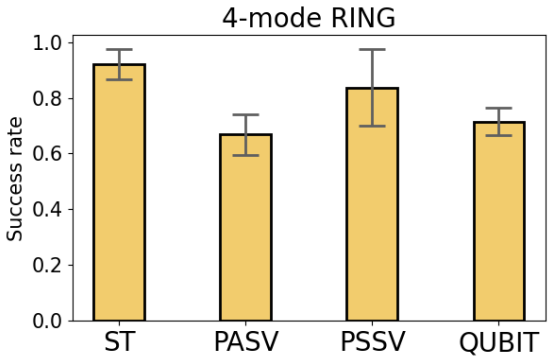

Figure 10. Classification results.

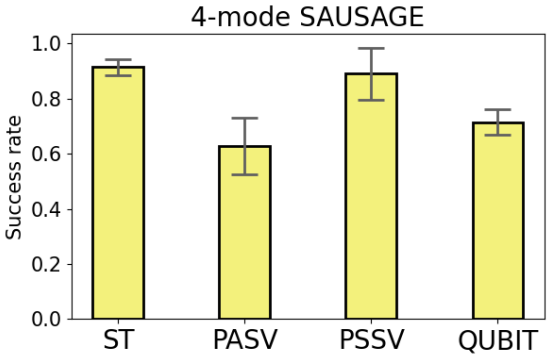

Figure 11. Classification results.

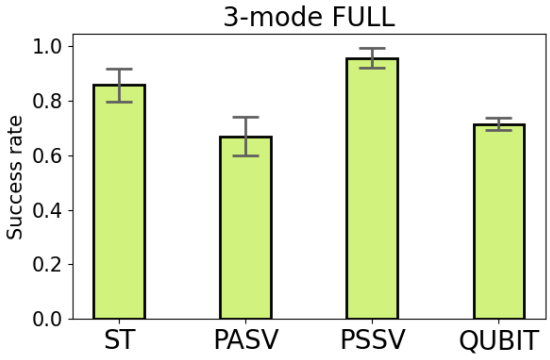

Figure 12. Classification results.

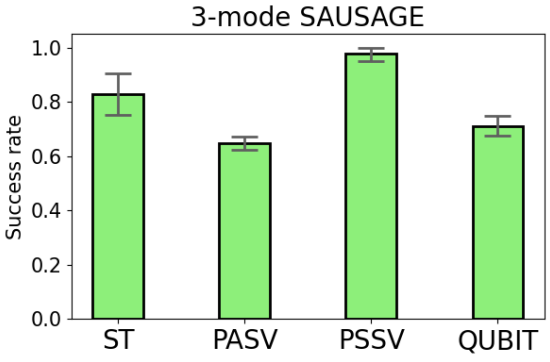

Figure 13. Classification results.

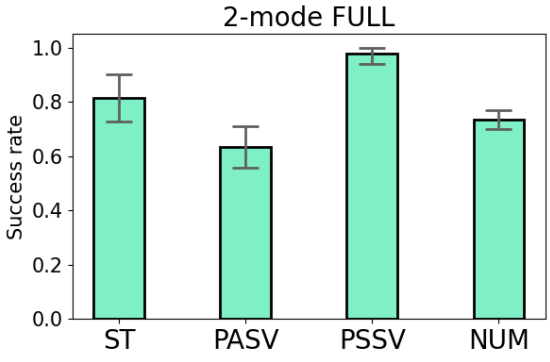

Figure 14. Classification results.

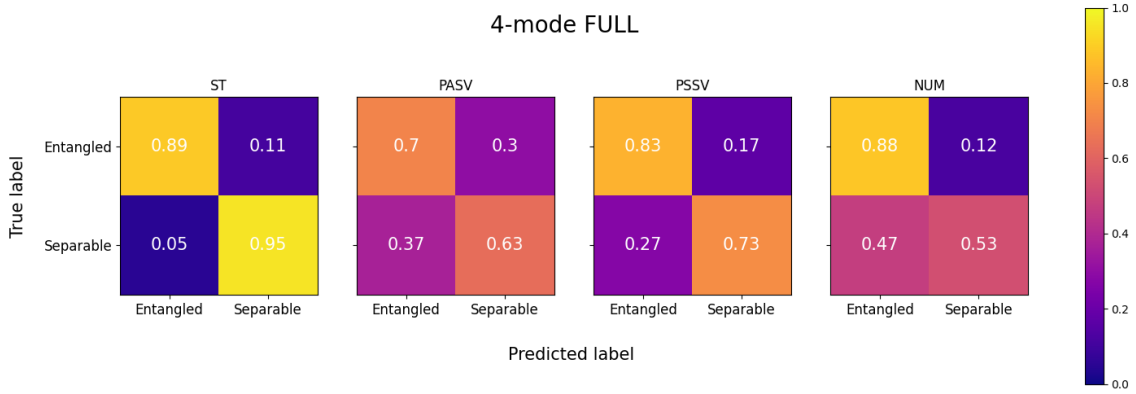

Figure 15. Confusion matrices of 4-mode FULL classification results.

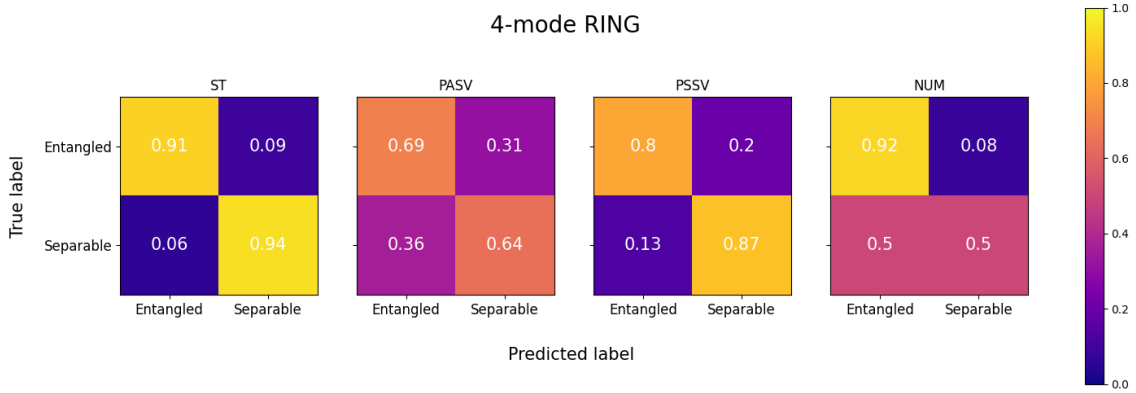

Figure 16. Confusion matrices of 4-mode RING classification results.

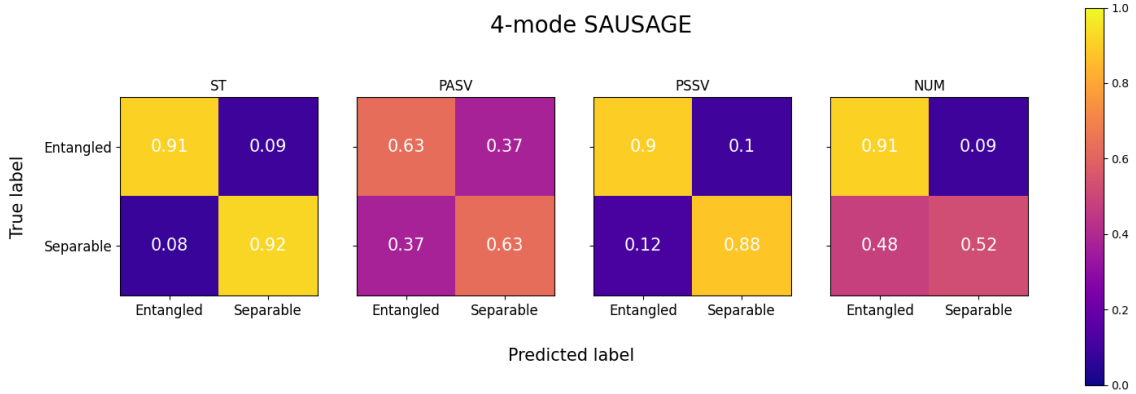

Figure 17. Confusion matrices of 4-mode SAUSAGE classification results.

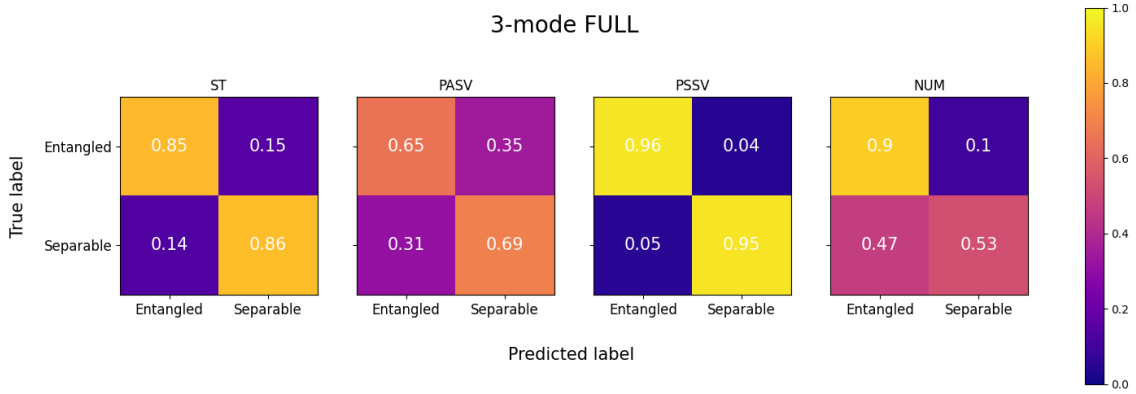

Figure 18. Confusion matrices of 3-mode FULL classification results.

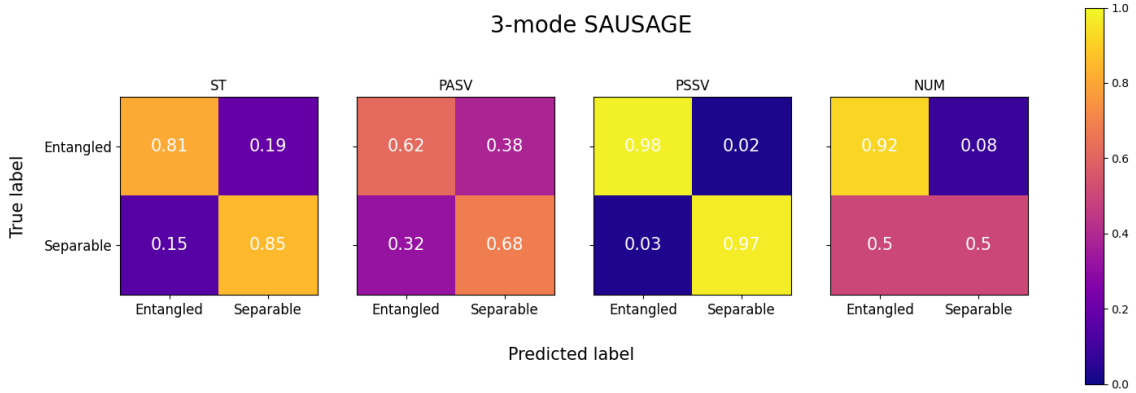

Figure 19. Confusion matrices of 3-mode SAUSAGE classification results.

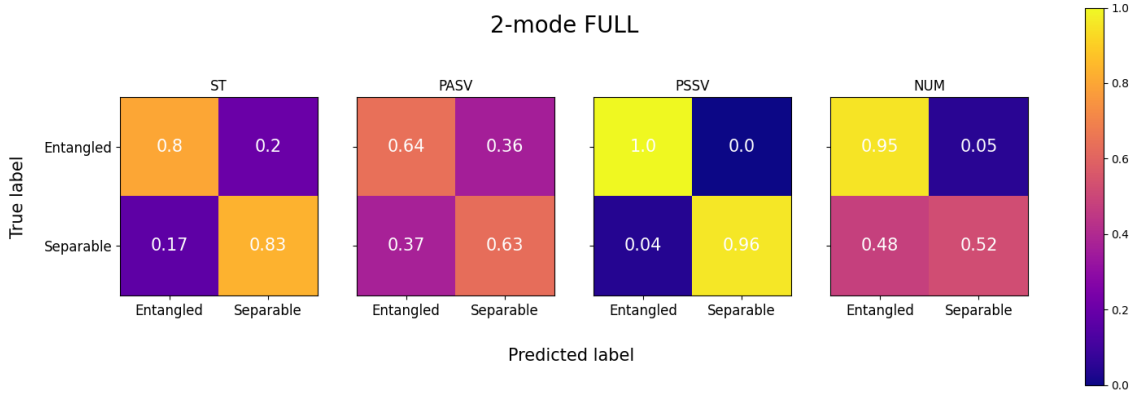

Figure 20. Confusion matrices of 2-mode FULL classification results.

The results regarding different classes of input states vary a lot. For ST states, the more complicated models perform better with less fluctuation in the results. The fall in performance is more or less expected as the simpler models presumably have a lesser ability for nonlinear information processing. In general all models seem to have a slight edge upon the classification of separable states as separable over the classification of entangled states into entangled.

The average classification of PASV states is poor across all models. From the results no clear advantage for any model is apparent, but a decreasing trend in the fluctuation margins for simpler models could be suggested. The results are a bit surprising as the created PASV states employ similar average occupation numbers as the training ST states. Taking a look at the raw classification data reveals another interesting notion. The 4-mode models sometimes achieve classification rates of over 0.8, but are also more prone to producing a total failure.

The PSSV states serve perhaps the most surprising results. The entanglement classification rate increases and the confidence interval decreases as the models get simpler. Surprisingly, the simpler models are able to classify the PSSV states even more robustly than the training ST states. There also exists some interesting behavior in the classification process. All the models recognize the entangled states as separable and separable states as entangled. Therefore, all the results are actually inverted and a classification rate of 0.05 is here interpreted as 0.95. Similarly to PASV states the PSSV states were created to have the average occupation numbers close to those of the training ST states. Therefore the difference in classification rates is really surprising.

Finally, the NUM states offer yet again another unique result to the classification problem. Taking a look at the confusion matrices, the classification of entangled states into entangled is decent across all models. On the other hand all models fail completely in the classification of separable states into separable. There seems

to be no outliers in this behavior in the raw data, and some small changes seem to be a result of random fluctuations. For the NUM state there exists an equal amount of two different kinds of separable states, $|00\rangle\langle 00|$ and $|11\rangle\langle 11|$. As the simulation shuffles the input states and no later record is kept on them other than the entanglement value. It is unclear if the complete failure is apparent for both types of separable input states, or if only one of them causes this behavior.

Across all models and all classes of input states similar trends with the confidence intervals can be noticed. The recognition of ST entanglement can be though of as the task the models were trained to do. For the ST states, the models with 4-mode reservoirs seem to be more robust in the classification resulting in smaller confidence intervals. This is more or less expected as these models have a greater capability for nonlinear information processing. For all the other classes of input states, PASV, PSSV and NUM the classification task changes to a more general recognition of entanglement. For these states, the smaller more simpler reservoirs seem to perform the classification more robustly. This doesn't necessarily mean a better overall performance in classification but smaller confidence intervals.

The classification results of the 4-mode fully connected reservoir in this work can be compared to the results of a similar system from literature [8]. The entanglement classification results in the literature are better across all input states. In addition to this the confidence intervals are smaller. This raises the question: why were the results in this work so much worse than in the literature? The discussion and comparison of the results can be extrapolated to also other models.

As discussed in section 3.2, the simulation is quite heavy with big matrices and multiple matrix multiplications. This likely introduced some chaotic behavior in the simulations which presumably affected the final results for all models. Some steps were taken to counteract this chaotic behavior. In the first generation of the simulations an escalating floating point error was noticed for every round of

simulation. This made the system lose coherence and the simulation to diverge. A floating point truncation to 13 decimals was found to fix this, and a truncation to 12 decimals was used for the final simulations. This made the 3- and 2-mode models stable resulting in on average smaller confidence intervals. For 4-mode models some chaotic behavior was still present. Sometimes during training after some amount of input states had been introduced, the reservoir began once again diverging resulting in failed training. These systems had to be reset by reinitializing them and training them with a new set of inputs. After a system had been successfully trained, no divergent behavior was present during the testing of ST states. On the other hand this wasn't true for PASV and PSSV testing. A system showing no loss of coherence during the training and testing of ST states could still lose coherence with PASV or PSSV states. Interestingly the divergent behavior sometimes manifested only with PASV or PSSV states but not both. Also in these cases a system had to be reset. The NUM states didn't cause any additional divergence. In the end it was unclear if some initial parameter in the reservoir or when creating the input states or both caused the loss of coherence in the system. The effect of minimizing the non-smooth parts of the piecewise function discussed in section 3.1 to decrease the error of RK4 was not studied

It is suspected that the chaotic behavior caused the surprising classification results discussed previously. This is also likely the reason for the difference in the results from this work and the literature. As mentioned previously, better than average results were also present for the 4-mode models. These results were unfortunately accompanied with awful results. This once again raises a question if one could "nit pick" systems for the general performance based on the entanglement classification capability of a certain class of states. This was briefly studied with Spearman correlation to see if there were any meaningful correlations between the results in classification across all models. In conclusion, there were not any mean-

ingful correlations. This was evident as the small correlations and anticorrelations seemed to be randomized between different models. Taking a closer look at the raw data confirmed these results.

5 Conclusions and final words

Previous efforts in entanglement classification of reservoir computing systems have put less emphasis on the study of different reservoir substrates [8, 60, 64]. Therefore, no overview or benchmark of different frameworks have been presented in the literature. Some recent work has been done on different substrates for optical systems without memory [65], and the aim of this work was to extend this study to fermionic systems.

In the thesis we began our journey by discussing the differences of separable and entangled quantum states. Moreover tools were given for the detection of entanglement of different classes of states. The evolution of open quantum systems was presented before a comprehensive description of Gaussian state formality was given. This included the description of states in both Hilbert space and the phase space. Examples of different Gaussian and non-Gaussian states were given which were later used in the simulations. Basics of machine learning were addresses with brief explanations on supervised, unsupervised and reinforcement learning. These explanations were accompanied by illustrative figures of the learning processes. Also the fundamentals of reservoir computing were explored with neural networks and the concept of a fading memory.

The differences of classical and quantum computing were considered. Similarities and dissimilarities from hardware to polynomial hierarchies were found. The exponential computing speedup of quantum computers was remarked requiring not only the entanglement and superposition of quantum states, but also a more stronger property called magic. Struggles in quantum computing were discussed such as the

difficulty in scaling and the loss of coherence of quantum systems. The combinations of classical and quantum protocols in quantum computing were addresses such as variational quantum algorithms. Also the main topic of the thesis quantum reservoir computing was properly introduced.

The capabilities of classical reservoir computing were demonstrated with echo state networks. Nonlinear information processing and the ability to recall inputs from the past were tested with different degrees of nonlinear autoregressive moving average and the tasks were evaluated with normalized mean squared error. 4-mode networks with different degrees of spatial multiplexing and a fully connected network were tested. The models with a greater degree of spatial multiplexing were seen to perform better in the more difficult tasks alongside the fully connected network. The performance of all models was seen to deteriorate with the most difficult tasks.

When the foundation for quantum reservoir computing was set, the system studied in this thesis was introduced. Dynamics of the input, the reservoir and the interaction of input and reservoir were given with a single master equation of the combined density matrix. The process of entangled and separable input state creation was thoroughly explained. This included the choice of hyperparameters for two-mode squezeed thermal states, two-mode photon added squeezed vacuum states, two-mode photon subtracted squeezed vacuum states and qubit states. Different methods were used to achieve roughly an equal amount of separable and entangled states. Different quantum substrates were introduced for 2-mode, 3-mode and 4-mode reservoirs. The simulation used to run the system dynamics were also explained with a **Metacode** example. Along the simulation the numerical side with 4^{th}-order Runge-Kutta method was presented.

The systems were trained with only two-mode squeezed thermal states, but tested with 4 different classes of states. The entanglement classification results were surprising for almost all classes of input states. For some classes the classification was

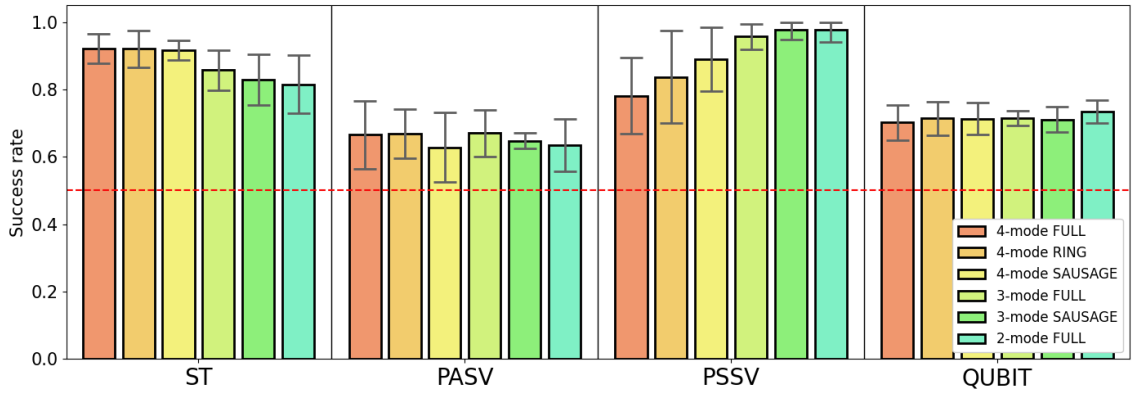

Figure 21. Side-by-side comparison of entanglement classification results of the 2-, 3- and 4-mode reservoirs with connectivities presented in Fig. 8. Fluctuation margins of one standard deviation are presented along all histograms, and a red line describing the complete failure of the classification task is shown. The final results are averaged over 10 realizations of the systems.

closer to a total failure than a perfect score. The results from Figs.9–14 are collected in Fig. 21 for a convenient overview of all the outcomes. Perhaps the only expected behavior was seen with squeezed thermal states, where the classification fidelity dropped along simpler models. No clear advantage in system size or complexity was apparent in the testing of photon added states and a gain in performance was seen for photon subtracted states. This gain was indeed so great, that the classification fidelity was the best for all classes of input states across all models. Similarly to the photon added states, number states didn't exhibit any reasonable differences in computing power for any model. Instead a total failure in the classification of separable states into separable was noticed.

Some reasoning for the surprising results was discussed. As all the models are relatively small in size, no big disadvantage for any case in a greater size or complexity was expected. On the contrary the more complex models were expected to outperform the simpler ones. Some comparable results for the simulations can be found in the literature, where far better and robust outcomes were established. No original code for the simulations was found, so a clear comparison of the underlying

dynamics could not been made. In the end a majority of the differences were thought to be a results of somewhat chaotic dynamics. On the other hand a different choice of some hyperparameters could also have resulted in more robust results.

A Action of squeezing on creation and destruction operators

We define the two-mode squeezing operator as

$$\begin{aligned} S(\alpha) &= \exp\{\alpha a_1^\dagger a_2^\dagger - \alpha^* a_1 a_2\}, \quad \alpha \in \mathbb{C} \\ A &= \alpha^* a_1 a_2 - \alpha a_1^\dagger a_2^\dagger \\ S(\alpha)^\dagger &= e^A. \end{aligned}$$

With the use of general equality, one can define for operators B and C

$$e^C B e^{-C} = \sum_{k=0}^{\infty} \frac{1}{k!} [C, [C, \ldots, [C, B] \ldots]].$$

The commutators of A and field operators can now be written as

$$\begin{aligned} [A, a_{1,2}] &= \alpha a_{2,1}^\dagger, \quad [A, [A, a_{1,2}]] = [A, \alpha a_{2,1}^\dagger] = |\alpha|^2 a_{1,2} \\ [A, a_{1,2}^\dagger] &= \alpha^* a_{2,1}, \quad [A, [A, a_{1,2}]] = [A, \alpha^* a_{2,1}] = |\alpha|^2 a_{1,2}^\dagger. \end{aligned}$$

For k commutators one can prove with induction

$$[A, [A, \ldots, [A, a_{1,2}] \ldots]] = \begin{cases} \alpha |\alpha|^{k-1} a_{2,1}^\dagger, & \text{for odd } k \\ |\alpha|^k a_{1,2}, & \text{for even } k \end{cases}$$

$$[A, [A, \ldots, [A, a_{1,2}^\dagger] \ldots]] = \begin{cases} \alpha^* |\alpha|^{k-1} a_{2,1}, & \text{for odd } k \\ |\alpha|^k a_{1,2}^\dagger, & \text{for even } k \end{cases}$$

Therefore, the action of two-mode squeezing on creation and destruction operators

$$\begin{aligned} S(\alpha)^\dagger a_{1,2} S(\alpha) &= e^A a_{1,2} e^{-A} \\ &= a_{1,2} \sum_{k=0}^{\infty} \frac{|\alpha|^{2k}}{(2k)!} + a_{2,1}^\dagger \frac{\alpha}{|\alpha|} \sum_{k=0}^{\infty} \frac{|\alpha|^{2k+1}}{(2k+1)!} \\ &= a_{1,2} \cosh|\alpha| + a_{2,1}^\dagger e^{i\theta} \sinh|\alpha| \end{aligned}$$

$$\begin{aligned} S(\alpha)^\dagger a_{1,2}^\dagger S(\alpha) &= e^A a_{1,2}^\dagger e^{-A} \\ &= a_{1,2}^\dagger \sum_{k=0}^{\infty} \frac{|\alpha|^{2k}}{(2k)!} + a_{2,1} \frac{\alpha^*}{|\alpha|} \sum_{k=0}^{\infty} \frac{|\alpha|^{2k+1}}{(2k+1)!} \\ &= a_{1,2}^\dagger \cosh|\alpha| + a_{2,1} e^{-i\theta} \sinh|\alpha|. \end{aligned}$$

B Use of AI in the thesis

In this thesis, large language models like Ghat-GPT were used in the assistance of Python and Latex code writing. Small parts of the Python code were optimized in the search for faster simulation times. All of the AI-produced code was thoroughly tested and later edited for better readability in the program and to better match the relevant physics. In some cases, the AI-suggested code was not able to produce the correct physics. The assistance in Latex was used to produce a more satisfactory layout of the thesis.

References

[1] Sawyer D Campbell, Ronald P Jenkins, Philip J O'Connor, and Douglas Werner. The explosion of artificial intelligence in antennas and propagation: How deep learning is advancing our state of the art. *IEEE Antennas and Propagation Magazine*, 63(3):16–27, 2020.

[2] Derick Almeida, Wim Naudé, and Tiago Sequeira. Artificial intelligence and the discovery of new ideas: Is an economic growth explosion imminent? 2024.

[3] William MacAskill and Fin Moorhouse. Preparing for the intelligence explosion. *arXiv preprint arXiv:2506.14863*, 2025.

[4] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 549(7671):195–202, 2017.

[5] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. An introduction to quantum machine learning. *Contemporary Physics*, 56(2):172–185, 2015.

[6] Carmen G Almudever, Lingling Lao, Xiang Fu, Nader Khammassi, Imran Ashraf, Dan Iorga, Savvas Varsamopoulos, Christopher Eichler, Andreas Wallraff, Lotte Geck, et al. The engineering challenges in quantum computing. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, pages 836–845. IEEE, 2017.

[7] Marco Cerezo, Guillaume Verdon, Hsin-Yuan Huang, Lukasz Cincio, and Patrick J Coles. Challenges and opportunities in quantum machine learning. *Nature computational science*, 2(9):567–576, 2022.

[8] Sanjib Ghosh, Andrzej Opala, Michał Matuszewski, Tomasz Paterek, and Timothy CH Liew. Quantum reservoir processing. *npj Quantum Information*, 5(1):35, 2019.

[9] Code for the classical simulations. https://github.com/JohannesAnnala/EchoStateNetwork-and-NARMA.git. Accessed: 2026-04-14.

[10] Code for the quantum simulations. https://github.com/JohannesAnnala/Entanglement-Classification-with-QRC.git. Accessed: 2026-04-14.

[11] Alessandro Ferraro, Stefano Olivares, and Matteo GA Paris. Gaussian states in continuous variable quantum information. *arXiv preprint quant-ph/0503237*, 2005.

[12] Jonas Maziero. Computing partial traces and reduced density matrices. *International Journal of Modern Physics C*, 28(01):1750005, 2017.

[13] Jonas Maziero. Computing partial transposes and related entanglement functions. *Brazilian Journal of Physics*, 46(6):605–611, 2016.

[14] Asher Peres. Separability criterion for density matrices. *Physical Review Letters*, 77(8):1413, 1996.

[15] Guifré Vidal and Reinhard F Werner. Computable measure of entanglement. *Physical Review A*, 65(3):032314, 2002.

[16] Lieven Clarisse. The distillability problem revisited. *arXiv preprint quant-ph/0510035*, 2005.

[17] Wolfgang Dür, J Ignacio Cirac, and Rolf Tarrach. Separability and distillability of multiparticle quantum systems. *Physical review letters*, 83(17):3562, 1999.

[18] Indrani Chattopadhyay and Debasis Sarkar. Npt bound entanglement-the problem revisited. *arXiv preprint quant-ph/0609050*, 2006.

[19] Beatrix C Hiesmayr, Christopher Popp, and Tobias C Sutter. Bipartite bound entanglement. *arXiv preprint arXiv:2406.13491*, 2024.

[20] Xiao Yuan, Bartosz Regula, Ryuji Takagi, and Mile Gu. Virtual quantum resource distillation. *Physical Review Letters*, 132(5):050203, 2024.

[21] Maciej Lewenstein, Dagmar Bruß, Juan Ignacio Cirac, Barbara Kraus, Marek Kuś, Jan Samsonowicz, Anna Sanpera, and Rolf Tarrach. Separability and distillability in composite quantum systems-a primer. *Journal of Modern Optics*, 47(14-15):2481–2499, 2000.

[22] MA Jafarizadeh, N Behzadi, and Y Akbari. Entanglement witnesses and characterizing entanglement properties of some ppt states. *The European Physical Journal D*, 55(1):197–203, 2009.

[23] Saronath Halder, Manik Banik, and Sibasish Ghosh. Family of bound entangled states on the boundary of the peres set. *Physical Review A*, 99(6):062329, 2019.

[24] Daniel Manzano. A short introduction to the lindblad master equation. *Aip advances*, 10(2), 2020.

[25] Heinz-Peter Breuer, Elsi-Mari Laine, Jyrki Piilo, and Bassano Vacchini. Colloquium: Non-markovian dynamics in open quantum systems. *Reviews of Modern Physics*, 88(2):021002, 2016.

[26] Jonatan Bohr Brask. Gaussian states and operations–a quick reference. *arXiv preprint arXiv:2102.05748*, 2021.

[27] Christian Weedbrook, Stefano Pirandola, Raúl García-Patrón, Nicolas J Cerf, Timothy C Ralph, Jeffrey H Shapiro, and Seth Lloyd. Gaussian quantum information. *Reviews of Modern Physics*, 84(2):621–669, 2012.

[28] Arvind, Biswadeb Dutta, N Mukunda, and R Simon. The real symplectic groups in quantum mechanics and optics. *Pramana*, 45(6):471–497, 1995.

[29] Marco G Genoni and Matteo GA Paris. Quantifying non-gaussianity for quantum information. *Physical Review A—Atomic, Molecular, and Optical Physics*, 82(5):052341, 2010.

[30] Vladimir Nasteski. An overview of the supervised machine learning methods. *Horizons. b*, 4(51-62):56, 2017.

[31] Samreen Naeem, Aqib Ali, Sania Anam, and Muhammad Munawar Ahmed. An unsupervised machine learning algorithms: Comprehensive review. *International Journal of Computing and Digital Systems*, 2023.

[32] Mantas Lukoševičius. A practical guide to applying echo state networks. In *Neural Networks: Tricks of the Trade: Second Edition*, pages 659–686. Springer, 2012.

[33] Johannes Nokkala, Rodrigo Martínez-Peña, Roberta Zambrini, and Miguel C Soriano. High-performance reservoir computing with fluctuations in linear networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33(6):2664–2675, 2021.

[34] Richard E Turner. An introduction to transformers. *arXiv preprint arXiv:2304.10557*, 2023.

[35] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. *Advances in neural information processing systems*, 32, 2019.

[36] Top500 website. https://top500.org/. Accessed: 2026-02-12.

[37] Philip Walther, Kevin J Resch, Terry Rudolph, Emmanuel Schenck, Harald Weinfurter, Vlatko Vedral, Markus Aspelmeyer, and Anton Zeilinger. Experimental one-way quantum computing. *Nature*, 434(7030):169–176, 2005.

[38] Jeremy L O'brien. Optical quantum computing. *Science*, 318(5856):1567–1570, 2007.

[39] Göran Lindblad. A general no-cloning theorem. *Letters in Mathematical Physics*, 47(2):189–196, 1999.

[40] Alexander I Lvovsky, Barry C Sanders, and Wolfgang Tittel. Optical quantum memory. *Nature photonics*, 3(12):706–714, 2009.

[41] Jiří Tolar. On clifford groups in quantum computing. In *Journal of Physics: Conference Series*, volume 1071, page 012022. IOP Publishing, 2018.

[42] Mark Howard, Joel Wallman, Victor Veitch, and Joseph Emerson. Contextuality supplies the 'magic'for quantum computation. *Nature*, 510(7505):351–355, 2014.

[43] Zi-Wen Liu and Andreas Winter. Many-body quantum magic. *PRX Quantum*, 3(2):020333, 2022.

[44] Xin Wang, Mark M Wilde, and Yuan Su. Quantifying the magic of quantum channels. *New Journal of Physics*, 21(10):103002, 2019.

[45] Victor Veitch, SA Hamed Mousavian, Daniel Gottesman, and Joseph Emerson. The resource theory of stabilizer quantum computation. *New Journal of Physics*, 16(1):013009, 2014.

[46] Daniel Gottesman. *Stabilizer codes and quantum error correction*. California Institute of Technology, 1997.

[47] Larry J Stockmeyer. The polynomial-time hierarchy. *Theoretical computer science*, 3(1):1–22, 1976.

[48] Sabee Grewal and Justin Yirka. The entangled quantum polynomial hierarchy collapses. *arXiv preprint arXiv:2401.01453*, 2024.

[49] Avantika Agarwal, Sevag Gharibian, Venkata Koppula, and Dorian Rudolph. Quantum polynomial hierarchies: Karp-lipton, error reduction, and lower bounds (2024). *arXiv preprint arXiv:2401.01633*, 2024.

[50] Michael J Bremner, Richard Jozsa, and Dan J Shepherd. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 467(2126):459–472, 2011.

[51] Nathalie P De Leon, Kohei M Itoh, Dohun Kim, Karan K Mehta, Tracy E Northup, Hanhee Paik, BS Palmer, Nitin Samarth, Sorawis Sangtawesin, and David W Steuerman. Materials challenges and opportunities for quantum computing hardware. *Science*, 372(6539):eabb2823, 2021.

[52] Austin G Fowler, Matteo Mariantoni, John M Martinis, and Andrew N Cleland. Surface codes: Towards practical large-scale quantum computation. *Physical Review A—Atomic, Molecular, and Optical Physics*, 86(3):032324, 2012.

[53] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[54] 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(8):2100027, 2021.

[55] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.

[56] Laia Domingo, G Carlo, and Florentino Borondo. Taking advantage of noise in quantum reservoir computing. *Scientific Reports*, 13(1):8790, 2023.

[57] Yudai Suzuki, Qi Gao, Ken C Pradel, Kenji Yasuoka, and Naoki Yamamoto. Natural quantum reservoir computing for temporal information processing. *Scientific reports*, 12(1):1353, 2022.

[58] Gerasimos Angelatos, Saeed A Khan, and Hakan E Türeci. Reservoir computing approach to quantum state measurement. *Physical Review X*, 11(4):041062, 2021.

[59] LCG Govia, GJ Ribeill, GE Rowlands, HK Krovi, and TA Ohki. Quantum reservoir computing with a single nonlinear oscillator. *Physical Review Research*, 3(1):013077, 2021.

[60] Tanjung Krisnanda, Tomasz Paterek, Mauro Paternostro, and Timothy CH Liew. Quantum neuromorphic approach to efficient sensing of gravity-induced entanglement. *Physical Review D*, 107(8):086014, 2023.

[61] JC López Carreño and FP Laussy. Excitation with quantum light. i. exciting a harmonic oscillator. *Physical Review A*, 94(6):063825, 2016.

[62] Marc Nico Spijker. Error propagation in runge-kutta methods. *Applied numerical mathematics*, 22(1-3):309–325, 1996.

[63] Tanjung Krisnanda, Guo Yao Tham, Mauro Paternostro, and Tomasz Paterek. Observable quantum entanglement due to gravity. *npj Quantum Information*, 6(1):12, 2020.

[64] Sanjib Ghosh, Kohei Nakajima, Tanjung Krisnanda, Keisuke Fujii, and Timothy CH Liew. Quantum neuromorphic computing with reservoir computing networks. *Advanced Quantum Technologies*, 4(9):2100053, 2021.

[65] Danilo Zia, Luca Innocenti, Giorgio Minati, Salvatore Lorenzo, Alessia Suprano, Rosario Di Bartolo, Nicolò Spagnolo, Taira Giordani, Valeria Cimini, G Massimo Palma, et al. Quantum reservoir computing for photonic entanglement witnessing. *Science Advances*, 11(50):eady7987, 2025.