universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Feedback-driven photonic reservoir computing

verfasst von | submitted by

Sina Thier B.Sc.

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2026

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 066 876
Studienrichtung lt. Studienblatt | Degree programme as it appears on the student record sheet:	Masterstudium Physics
Betreut von | Supervisor:	Univ.-Prof. Dipl.-Ing. Dr. Philip Walther

Acknowledgments

First, I want to thank Prof. Philip Walther for giving me the opportunity to be a part of his research group. I have learned so much while working on this project, both academically and personally, and I am so grateful for the experience.

I especially want to thank Iris Agresti for all the support and guidance throughout and most of all, for believing in me! Your encouragement really made a difference and I truly look forward to continuing to work with you.

A big thank you also to all my colleagues for their moral support, the many helpful discussions, and all the help in the lab!

Thank you to my parents for always supporting me in everything I do and for encouraging me every step of the way.

And to the people who were there for me throughout all of this: thank you for keeping me sane. Your support means more than I can express, and I will never forget it.

Abstract

Conventional computing architectures face significant challenges in energy efficiency due to the physical separation of processing and memory units. As modern large-scale machine learning models demand increasingly high energy consumption, neuromorphic approaches that emulate the brain's structure offer a compelling alternative. Within this domain, reservoir computing provides a potentially efficient framework for processing temporal data by mapping inputs into a high-dimensional space through a fixed dynamical system. In this thesis, we explore an experimental implementation of reservoir computing based on photonic quantum memristors. Building on previous designs, our framework introduces an injection feedback mechanism. This amounts to injecting measured reservoir outputs back into the device, significantly enhancing the system's memory capacity and nonlinearity. To further increase the model's expressivity and the dimension of the feature space, we employ spatial multiplexing, utilizing multiple independent memristor units operating in parallel. We evaluate the performance across several benchmark tasks, including the short-term memory and the nonlinear auto-regressive moving average task. Furthermore, we present numerical simulations for the autonomous forecasting of a chaotic Lorenz system. Results demonstrate that the injection mechanism enables the system to outperform previous models by maintaining high accuracy over longer temporal delays.

Zusammenfassung

Herkömmliche Rechnerarchitekturen stehen aufgrund der physischen Trennung von Rechen- und Speichereinheiten vor erheblichen Herausforderungen hinsichtlich Energieeffizienz. Da moderne, komplexe Modelle des maschinellen Lernens einen immer höheren Energieverbrauch erfordern, bieten neuromorphe Ansätze, die die Struktur des Gehirns nachahmen, eine Alternative. In diesem Bereich bietet Reservoir Computing ein effizientes Framework für die Verarbeitung zeitlicher Daten, indem Eingaben über ein festes dynamisches System in einen hochdimensionalen Raum abgebildet werden. In dieser Arbeit untersuchen wir eine experimentelle Umsetzung von Reservoir Computing basierend auf photonischen Quantenmemristoren. Aufbauend auf früheren Designs führt unser Framework einen Injektions-Rückkopplungsmechanismus ein. Dabei werden gemessene Ausgangssiganle des Reservoirs wieder in den Apparat eingespeist, wodurch die Speicherkapazität und die Nichtlinearität des Systems erheblich verbessert werden. Um die Ausdruckskraft des Modells und die Dimension des Merkmalsraums weiter zu erhöhen, setzen wir Spatial Multiplexing ein, wobei mehrere unabhängige Memristor-Einheiten parallel betrieben werden. Wir bewerten die Leistung anhand mehrerer Benchmark-Aufgaben, darunter die Kurzzeitgedächtnis- und die NARMA-Aufgabe. Darüber hinaus präsentieren wir wir numerische Simulationen zur autonomen Vorhersage eines chaotischen Lorenz-Systems. Die Ergebnisse zeigen, dass der Rückkoppplungsmechanismus es dem System ermöglicht, frühere Modelle zu übertreffen, indem es über längere zeitliche Verzögerungen hinweg eine hohe Genauigkeit beibehält.

Contents

Introduction

The conventional computing systems that we use in our everyday lives are based on a von Neumann architecture, in which processing and memory units are physically separated. This implies data must constantly be transferred between them, consuming both time and energy. This approach is becoming increasingly problematic for modern large AI models and has fueled research interest in alternative computing architectures that physically co-locate processing and memory units [1] [2].

One promising alternative computation strategy is neuromorphic computing which seeks to emulate the structure and function of biological neural networks to increase energy efficiency and performance [3]. Thus, rather than executing sequential instructions on a fixed architecture, neuromorphic systems process information through parallel networks of artificial neurons that adapt based on experience.

Reservoir computing (RC) presents a particularly appealing neuromorphic framework because it is equivalent to recurrent neural networks, while drastically reducing their training cost. In RC, a dynamical system, called the reservoir, maps temporal inputs into a high-dimensional space. Only a simple linear readout layer is trained, making the approach computationally cheap. This has proven beneficial for processing time series, forecasting, and speech recognition [4]. A variety of physical systems can serve as reservoirs, but memristor-based ones are particularly promising, as they can provide the necessary nonlinear, history-dependent dynamics [5].

Indeed, the memristor is a device whose response depends not only on the present but also on its input history. Therefore, it can provide both information processing and memory. Numerous memristor-based neuromorphic applications have been explored, including convolutional neural networks [6], spiking neural networks [7], and reservoir computing [8] [9].

An open question is whether extending this paradigm to the quantum regime can yield further advantages, either in terms of the richness of the dynamics or in the scaling of computational resources [10]. Photonic platforms offer one promising route towards neuromorphic computing [11] [12], including (quantum) reservoir computing [13] [10]. Notable examples include quantum reservoir computing on continuous variable [14] and orbital angular momentum [15] [16] photonic platforms. In this framework, photonic integrated circuits are a pivotal tool, as they can stack multiple photonic components on a single chip, enabling compact, high-performance optical systems as well as precise manipulation of photonic qubits [17].

More specifically in the realm of photonic neuromorphic architectures, a keystone could be represented by the photonic quantum memristor, which, as demonstrated

by Spagnolo et al. [18], implements memristive dynamics in the evolution of single photons. Most recently, a first example of quantum reservoir computing based on a quantum memristor was introduced [19], which exploited the device's memory and nonlinear response to perform several benchmark tasks involving time-series predictions. Despite the successful application of this photonic model to the aforementioned tasks, the full capabilities of quantum memristor-based models remain to be explored.

In this thesis, a novel quantum reservoir computing model based on the photonic quantum memristor is equipped with two feedback mechanisms that enhance its memory and nonlinearity in its response. Furthermore, we exploit multi-photon inputs and significantly enlarge the dimension of the considered reservoir. In particular, to enhance memory and nonlinearity, we re-inject previously measured reservoir outputs into the device. Additionally, we enlarge the reservoir's output dimension by using parallel units through spatial multiplexing, in which multiple independent memristor units process information in parallel.

The thesis is structured as follows. Chapter 1 introduces the theoretical background, covering reservoir computing, the photonic quantum memristor, and the basis of our experimental platform. Chapter 2 reviews the previous implementation of reservoir computing based on photonic quantum memristors and details the addition of the injection feedback dynamic to the design as well as the full reservoir computing framework used in this work. Furthermore, the experimental implementation is described, including the single photon source, the integrated photonic circuit, and the measurement procedure. Chapter 3 presents the results for different benchmark tasks and analyzes the performance of the different memory dynamics. To conclude, a final discussion and an outlook on future work are presented.

1 Background

In this chapter, we introduce the theoretical concepts that underpin the work of the present thesis. First, we give an overview of machine learning, specifically the reservoir computing framework, and how these concepts relate to neuromorphic computing. In this thesis, the photonic quantum memristor serves as the building block of our reservoir; hence, we describe its working principle. Furthermore, we discuss how the photonic quantum memristor can be implemented experimentally using integrated photonics and describe the basis of our experimental platform, which is an integrated circuit with a universal architecture. Finally, the process of spontaneous parametric down-conversion, on which our single photon source is based, is defined.

1.1 Machine learning and reservoir computing

Today, the world generates vast amounts of data at an unprecedented scale. To process this data, artificial intelligence (AI) and more specifically machine learning (ML) has emerged as the dominant paradigm. Artificial neural networks (ANNs) in particular are very popular for their computational capabilities and represent the core of many modern ML models like deep learning. However, the continued scaling of these techniques faces fundamental constraints. Conventional computing systems are based on the von Neumann architecture, where processing and memory units are separated. In complex machine learning algorithms, this architecture not only limits performance, as data has to be constantly moved between the units, but also consumes vast amounts of energy. As a result, the power requirements for training large-scale models have raised serious environmental concerns, as data centers now account for a significant share of global energy consumption [20].

In contrast, the human brain, which originally inspired ANNs, is highly efficient at processing information. Therefore, neuromorphic computing has emerged as a promising approach to address these challenges [21]. Neuromorphic systems aim to emulate the brain's morphology by co-locating memory and computation to enable parallel, event-driven processing. Within this domain, reservoir computing provides a particularly compelling framework for processing temporal data by exploiting the complex dynamics of a fixed, high-dimensional system for computation, as explained in the following sections.

Artificial neural networks

ML is concerned with building systems that learn from data from experience and thereby improve their performance on a task without explicit programming. Given a dataset, a machine learning model adjusts its internal parameters to capture patterns that allow it to generalize to new, unseen data. One can

differentiate between three machine learning strategies: supervised, unsupervised, and reinforcement learning. In supervised learning, which is also the one considered in this work, the algorithm learns from a labeled dataset consisting of correct input-output pairs. The goal is to find a mapping function that accurately predicts the correct label for a new input, correctly generalizing the operation beyond the given examples.

ANNs are machine learning models inspired by biological neural networks, but unlike neuromorphic models, they do not aim to replicate the biological mechanisms of the brain. An ANN is organized into layers of interconnected units, where each one computes a weighted sum of its inputs and passes the results through a nonlinear activation function. In a feedforward network, information travels in one direction from an input layer through one or more hidden layers to an output layer. The network learns the specific task by iteratively optimizing each weight to

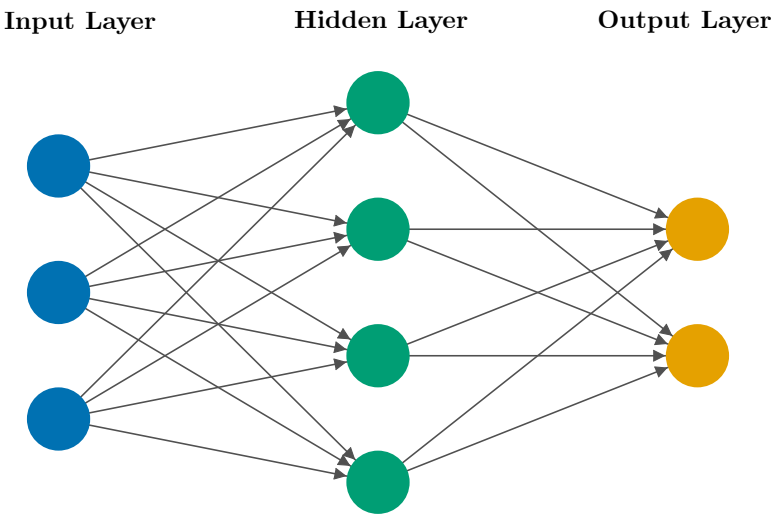


Figure 1.1 / Schematic representation of a simple feedforward neural network. The network consists of an input layer, a single hidden layer, and an output layer. The connections between the nodes are assigned weights and are directed toward the output layer.

minimize a cost function. This training procedure is most commonly done using the backpropagation technique [22]: starting from the output layer, the gradients of the cost function are computed with respect to each weight. The results are then used to update the weight using gradient descent. While feedforward neural networks are very effective for static pattern recognition tasks such as image classification, they are not naturally suited to sequential or time-series data because they treat each input independently and cannot retain information across time steps.

Recurrent neural networks (RNNs) address this limitation by introducing cyclical connections that allow information to persist across time, making them suitable for learning temporal tasks such as natural language processing or time-series prediction. However, in practice, this makes training RNNs computationally

expensive as the backpropagation algorithm must consider all previous time steps.

Reservoir computing

RC is a framework that was initially developed with the motivation to reduce the computational cost of training RNNs. More recently, it has gained popularity as an architecture for neuromorphic computing, as it can mimic the way the brain processes information by embedding neurons in a complex network.

RC was developed independently as echo state networks by Jaeger [23] and as liquid state machines by Maas et al. [24]. Instead of actively updating connections, as in RNN, a fixed dynamical system or a static recurrent structure, called the reservoir, is employed. Through this complex system, the input data is mapped into a higher-dimensional space, from which it can be effectively post-processed using simple machine learning algorithms, such as linear regression, which is the only part of model requiring training. This separation between the (untrained) reservoir and the (trained) readout simplifies the overall process and reduces its computational cost compared to fully trained ANNs. In particular, the number of connections that need to be trained is equal to the number of nodes in the output layer of an ANN.

Figure 1.2 shows the general layout of an RC system. Since the reservoir itself is fixed, it can be seen as a "black box" that, in general, can be any physical system that exhibits sufficiently rich, nonlinear dynamics. Physical reservoirs have the advantage of an inherent input-output mapping that does not require any calculations. This can be beneficial for both speed and energy efficiency [25].

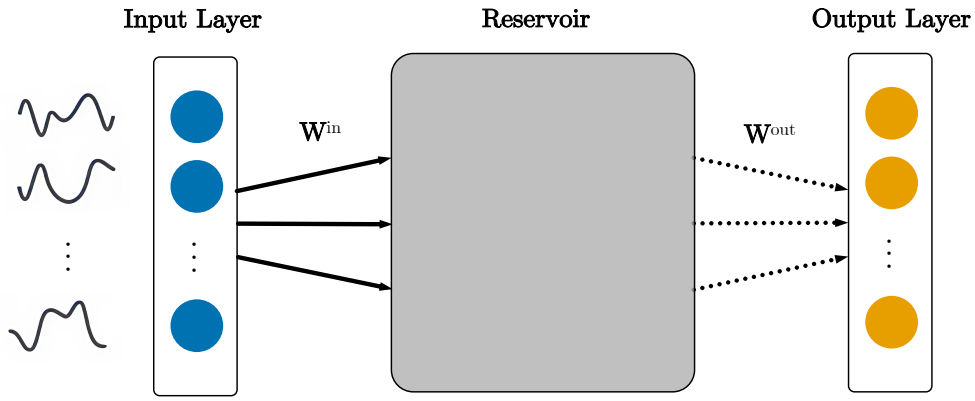


Figure 1.2 / Layout of a reservoir computing system. The system consists of three parts: First, the data is encoded in the input layer. The reservoir can be any dynamical, nonlinear system that maps the input into a higher-dimensional space. The weights of the output layer $\mathbf{W}^{out}$ are the only part of the network that is trained, while the input weights $\mathbf{W}^{in}$ and the reservoir remain fixed.

For a given RC system to function as a useful reservoir, it should exhibit some key properties [26]:

Echo-state property. Most importantly, the echo-state property dictates that the reservoir state should be determined solely by the history of the input signal. In other words, given a certain input sequence, the reservoir state must converge to the same trajectory regardless of the initial condition, such that the response is reproducible.

Fading memory. The reservoir must retain information from past inputs to process temporal patterns. However, the influence should decay as they lie further in the past, i.e., the reservoir should gradually "forget" past inputs. This ensures that the system is not perpetually influenced by distant inputs and is most sensitive to recent ones.

High-dimensionality and nonlinearity. A reservoir's ability to solve complex tasks depends on its capacity to map input into a high-dimensional feature space. Indeed, the goal is that previously nonlinearly separable inputs become linearly separable, allowing a simple readout layer to separate the data. Furthermore, the reservoir must provide sufficient nonlinearity to solve complex tasks.

Physical RC has demonstrated competitive performance on a wide range of platforms, exploiting nonlinear phenomena in mechanical, electronic, optical, and biological systems [25]. Among them, photonics has emerged as a promising platform for physical RC due to the unique advantages of light, like parallelism, high bandwidth, and the potential for superior energy efficiency [13] [27]. Furthermore, the development of photonic integrated circuits (PICs) has enabled dense, reconfigurable networks of optical elements with enhanced scalability features.

Another possibility in physical reservoir computing is the use of a non-classical reservoir, to increase the dimension of the output space. Indeed, a classical reservoir made up of N physical nodes generates an N-dimensional feature space. In contrast, a quantum reservoir of n qubits operates in a 2^n-dimensional Hilbert space. This exponential scaling can grant high computational performance without a proportional increase in physical resources. Furthermore, quantum effects have the potential to enrich the dynamics of the system. A quantum reservoir would also be able to deal with quantum native tasks.

1.2 Photonic quantum memristor

Our reservoir computing framework is based on the photonic quantum memristor. Memristors are an ideal building block for neuromorphic systems, serving as artificial synapses. Here, we will briefly introduce memristive devices and present the experimental realization of a photonic quantum memristor designed by [18].

Classical memristor and neuromorphic applications

The memristor was first proposed by Chua [28] in 1971 as the fourth fundamental electrical component, alongside the resistor, capacitor, and inductor. As its name suggests, it functions as a resistor whose resistivity depends on its internal state history, thereby retaining a memory of past states.

In later work Chua and Kang [29] generalized the idea of the memristor to a specific class of dynamical systems called *memristive devices*. These systems are defined by the following characteristic equations:

$$\dot{x} = f(x, u, t), y = g(x, u, t)u, \tag{1.3}$$

where u and y are the input and output of the system and x is a state variable of the system. All variables depend on time t and $f(\cdot)$ and $g(\cdot)$ are arbitrary functions. The central characteristic of memristive devices is that their output depends on both the current input and the device's history. A key feature of this input-output relationship is its inherent nonlinearity. When driven by a periodic input, memristive devices exhibit a characteristic pinched hysteresis loop in their input-output (e.g. voltage-current) plane (see Fig. 1.4). The curve always crosses itself and the enclosed area shrinks as the driving frequency increases, eventually collapsing into a straight line for high frequencies. This response distinguishes a memristive device from a simple linear resistor.

In 2008, Strukov et al. [30] implemented a memristive device based on a titanium dioxide semiconductor junction. This research sparked enormous interest in the potential applications of memristors based on their unique properties. The inherent memory and nonlinear dynamics of memristors have enabled diverse applications across computing and electronics. Memristive devices can be used as a non-volatile memory as they can retain data even after power is removed. Thus, they can offer lower power consumption compared to traditional flash memory [31] [32]. Their analog resistance modulation has been exploited for programmable analog circuits and reconfigurable logic systems [33].

Neuromorphic computing represents another compelling use case for memristive devices. Their behavior closely resembles that of biological neurons, making them ideal candidates for implementing artificial synapses in neuromorphic hardware [34]. Recent advances have demonstrated memristor-based implementations of convolutional neural networks [6], spiking neural networks [7], and reservoir computing [8] [9].

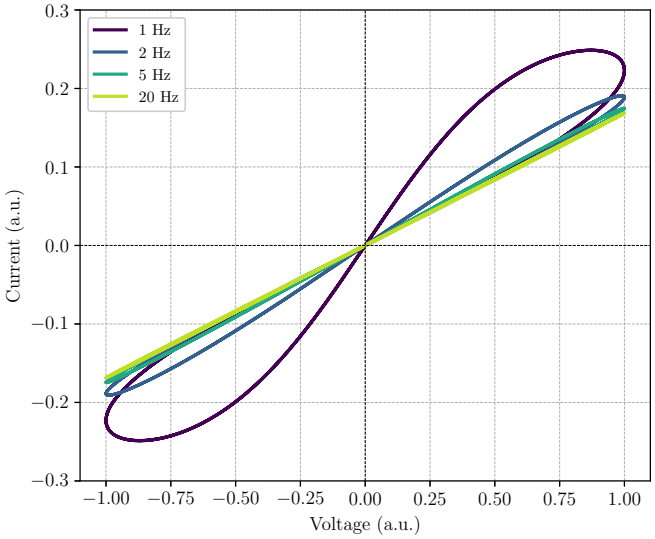


Figure 1.4 / Typical memristor hysteresis loop. Characteristic pinched hysteresis loop of the memristor described in [30] driven with a periodic input signal. The curve always crosses at the origin and shrinks to a line as the frequency of the input signal increases.

Photonic quantum memristor

While classical memristors have demonstrated remarkable capabilities in electronic systems, extending memristive behaviors to the quantum regime opens new possibilities for quantum information processing and quantum machine learning. The implementation of a memristive device using quantum hardware was first proposed in [35] [36] for superconducting circuits. The possibility of realizing a quantum memristor on photonic platforms was explored in [37]. In 2022, Spagnolo et al. [18] expanded on the concept and demonstrated the first experimental implementation of a photonic quantum memristor.

A quantum memristor should (1) exhibit memristive behavior according to Eqs. 1.3 and (2) preserve the coherence of the quantum state. The challenge lies in combining these two requirements. Closed quantum systems have an intrinsically linear evolution, therefore achieving nonlinear behaviors would require interactions with an environment. However, this is in stark contrast with preserving coherence. This apparent conundrum was solved by Spagnolo et al. [18], where it was shown that a beam splitter with tunable reflectivity R, whose value depends on the measurement carried out on one of its output modes complies with the definition of memristive device. The functioning principle of this device is explained in more detail in the next section.

Quantum memristor dynamics

As mentioned previously, the quantum memristor consists in a tunable beam splitter, with two input and two output ports, as is shown in Fig. 1.5. Let us

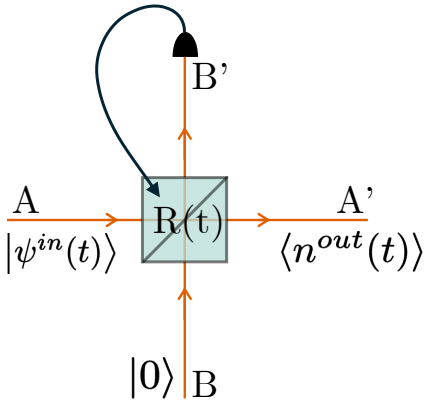


Figure 1.5 / Photonic quantum memristor scheme. The device consists of a tunable beam splitter with reflectivity R that is updated through a feedback loop depending on the expected value of one of the outputs.

indicate the two input modes as A and B and the two output ones as A' and B'. The input state is injected into one of the arms; one of the output is measured and the result is used to change the reflectivity according to a feedback rule. The quantum state entering the beam splitter at time t is given by:

$$\left|\psi^{tot}(t)\right\rangle = \left|\psi^{in}(t)\right\rangle_A \otimes |0\rangle_B = (\alpha(t)|0\rangle_A + \beta(t)|1\rangle_A) \otimes |0\rangle_B, \tag{1.6}$$

where $\alpha(t)$ and $\beta(t)$ are complex coefficients satisfying $|\alpha(t)|^2 + |\beta(t)|^2 = 1$ and $|0\rangle_X$ and $|1\rangle_X$ represent the vacuum and single-photon state in an optical mode $X = A, B$. The input is introduced to mode A, while mode B is unused. The expectation value of the photon-number operator $\hat{N}_A = |1\rangle\langle 1|_A$ in the input state given by 1.6 yields

$$\left\langle n^A(t)\right\rangle = \left\langle \psi^{tot}(t) \,\middle|\, \hat{N}_A \,\middle|\, \psi^{tot}(t)\right\rangle = |\beta(t)|^2. \tag{1.7}$$

This corresponds to the mean photon number at the input of the beam splitter, denoted as $\langle n^{in}(t)\rangle \equiv |\beta(t)|^2$. The beam splitter transformation is given by

$$U_{BS} = \begin{bmatrix} \sqrt{T} & i\sqrt{R} \\ i\sqrt{R} & \sqrt{T} \end{bmatrix}, \tag{1.8}$$

where the transmission coefficient is given by $T = 1 - R$. By applying Eq. 1.8, the states in Eq. 1.6 are transformed as follows:

$$\begin{aligned} |00\rangle &\Rightarrow |00\rangle \\ |10\rangle &\Rightarrow \sqrt{T}|10\rangle + i\sqrt{R}|01\rangle, \end{aligned} \tag{1.9}$$

which results in the output state

$$\left|\psi^{out}(t)\right\rangle = \alpha(t)|00\rangle_{A'B'} + \beta(t)\sqrt{T}|10\rangle_{A'B'} + i\beta(t)\sqrt{R}|01\rangle_{A'B'}. \tag{1.10}$$

The mean photon number measured in mode A' is used as the beam splitter's output. With the photon-number operator $\hat{N}_{A'} = |1\rangle\langle 1|_{A'}$, it is calculated as

$$\left\langle n^{out}(t)\right\rangle \equiv \left\langle n^{A'}(t)\right\rangle = \left\langle \psi^{out}(t)\,\middle|\,\hat{N}_{A'}\,\middle|\,\psi^{out}(t)\right\rangle = [1 - R(t)]\left\langle n^{in}(t)\right\rangle. \tag{1.11}$$

The update rule for the reflectivity $R(t)$ is chosen to be

$$\dot{R}(t) = \left\langle n^{in}(t)\right\rangle - 0.5. \tag{1.12}$$

Together, Eqs. 1.11 and 1.12 satisfy Eqs. 1.3, defining a memristive device with state variable $R(t)$. It is easily obtained by integrating Eq. 1.12:

$$R(t) = 0.5 + \frac{1}{\tau_R}\int_{t-\tau_R}^{t} [\left\langle n^{in}(T)\right\rangle - 0.5]\,\mathrm{d}T\,, \tag{1.13}$$

where τ_R is the integration window width. At time t, only inputs inside the integration window $[t - \tau_R, t]$ are considered in the update of the reflectivity. As is shown in Fig. 1.14, when the input number of photons is varied periodically with $\langle n^{in}\rangle = \sin^2(\pi/T_{osc}t)$, the output produces a pinched hysteresis curve that reduces to a line for high frequencies, as is characteristic of a memristive device.

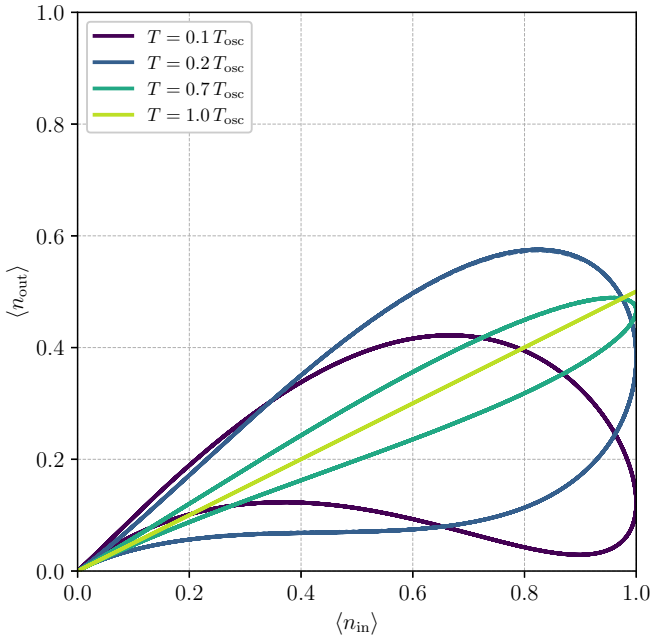


Figure 1.14 / Photonic quantum memristor hysteresis loop. The input number of photons is varied in time as $\langle n^{in}\rangle = \sin^2(\pi/T_{osc}t)$ and the reflectivity $R(t)$ is updated according to Eq. 1.13. The curve is pinched at the origin and linear in the high-frequency regime, characteristic of a memristive device.

Quantum coherence

To show that the photonic quantum memristor also fulfills the second requirement of preserving the coherence of the quantum state, let us first compute the density

matrix of the state after the beam splitter transformation (Eq. 1.10):

$$\begin{aligned}\rho^{out}_{A'B'} &= \left|\psi^{out}\right\rangle\!\left\langle\psi^{out}\right| \\ &= |\alpha|^2|00\rangle\langle 00| + \alpha\beta^*\sqrt{T}|00\rangle\langle 10| - i\alpha\beta^*\sqrt{R}|00\rangle\langle 01| \\ &\quad + \alpha^*\beta\sqrt{T}|10\rangle\langle 00| + |\beta|^2T|10\rangle\langle 10| - i|\beta|^2\sqrt{RT}|10\rangle\langle 01| \\ &\quad + i\alpha^*\beta\sqrt{R}|01\rangle\langle 00| + i|\beta|^2\sqrt{RT}|01\rangle\langle 10| + |\beta|^2R|01\rangle\langle 01|\end{aligned} \tag{1.15}$$

where the indices have been omitted. Mode B' functions as the feedback port while the user is only able to access output port A'. Thus, we consider only the partial trace of the density matrix, describing the sub-system at mode A':

$$\begin{aligned}\rho^{out}_{A'} &= \mathrm{Tr}_{B'}(\rho^{out}_{A'B'}) \\ &= |\alpha|^2|0\rangle\langle 0|_{A'} + \alpha\beta^*\sqrt{T}|0\rangle\langle 1|_{A'} + \alpha^*\beta\sqrt{T}|1\rangle\langle 0|_{A'} \\ &\quad + |\beta|^2T|1\rangle\langle 1| + |\beta|^2R|0\rangle\langle 0|,\end{aligned} \tag{1.16}$$

or in matrix form with basis $|0\rangle_{A'}, |1\rangle_{A'}$:

$$\rho^{out}_{A'} = \begin{bmatrix} |\alpha|^2 + |\beta|^2R & \alpha^*\beta\sqrt{T} \\ \alpha\beta^*\sqrt{T} & |\beta|^2T \end{bmatrix}. \tag{1.17}$$

The purity of a quantum state is defined as follows:

$$\mathrm{Tr}(\rho^{out^2}_{A'}(t)) = 1 - 2|\beta(t)|^4R(1 - R(t)). \tag{1.18}$$

It is possible to see that the purity is equal to 1/2 (fully mixed state), only if $|\beta|^2 = 1, R = 0.5$. In all other cases, the purity is larger which means that the device is capable of preserving quantum information.

1.3 Integrated photonics

Integrated optics offers several practical advantages. Indeed, it provides excellent stability compared to free-space optical setups, as the optical components are fixed and therefore are insensitive to mechanical vibrations in the environment. Furthermore, integrated platforms enable precise control of optical phases and are highly scalable [17]. The following sections introduce the basic building blocks of photonic integrated circuits and their fabrication. Furthermore, we describe universal photonic processors (UPPs) that will be used in our experiment and how the photonic quantum memristor is implemented on integrated photonic platforms.

Waveguides and fabrication

The fundamental component of any PIC is the optical waveguide, which functions by confining light through total internal reflection. This is achieved by using a core material with a higher refractive index surrounded by a cladding material with a lower index. Light entering the core reflects repeatedly at the boundary, guiding it along the structure.

By bringing two waveguides close together, one can realize a directional coupler, which can serve as the integrated counterpart to beam splitters. It consists of two closely spaced parallel waveguides over a coupling length. When the evanescent fields of the two waveguides overlap, light is transferred periodically between the cores, with the amount depending on the waveguide design and the phase/wavelength of the incident light.

Waveguides can be fabricated by femtosecond laser micromachining [38] [39]. A tightly focused, femtosecond pulse is focused in a transparent medium, e.g., silica. The nonlinear absorption in the material produces a permanent, localized increase in the refractive index. By moving the laser beam across the substrate, one can write lines of higher refractive index, which can create single-mode waveguides.

Universal photonic processors

In our experiment, we implement the full photonic reservoir layout, including the photonic quantum memristor, on a six-mode UPP.

A UPP can physically implement any unitary input/output transformation in the Fock basis. In a UPP, a unitary matrix is decomposed into a specific sequence of passive elements (beam splitters and phase shifters) that are implemented on an integrated circuit. The fundamental unit cell of a UPP is the Mach-Zehnder interferometer (MZI). In integrated photonics, an MZI is realized by two balanced directional couplers acting as the two beam splitters and a phase shifter in one of the arms. This structure is equivalent to a tunable beam splitter, where the internal phase controls its reflectivity. With an additional phase shifter in one of

the output paths of the interferometer, a MZI realizes a unitary transformation given by:

$$\begin{aligned} U_{\mathrm{MZI}}(\theta,\phi) &= \frac{1}{2}\begin{bmatrix} e^{i\phi} & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}\begin{bmatrix} e^{i\theta} & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix} \\ &= e^{i(\frac{\theta}{2}+\frac{\pi}{2})}\begin{bmatrix} e^{i\phi}\sin(\theta/2) & \cos(\theta/2) \\ e^{i\phi}\cos(\theta/2) & -\sin(\theta/2) \end{bmatrix}. \end{aligned} \tag{1.19}$$

If also the phase difference between the input ports can be controlled, any arbitrary 2×2 matrix in $U(2)$ can be implemented. Therefore, a reconfigurable MZI represents the unit cell of a UPP. Reck et al. [40] extended the design to larger Hilbert spaces, finding the minimal combination of MZIs required to implement an $N \times N$ unitary and showed that this amounts to a triangular mesh. Later, Clements et al. [41] showed an equivalent, symmetric, and more compact configuration. Both topologies are shown in Fig. 1.20. An N port UPP requires $N(N-1)/2$

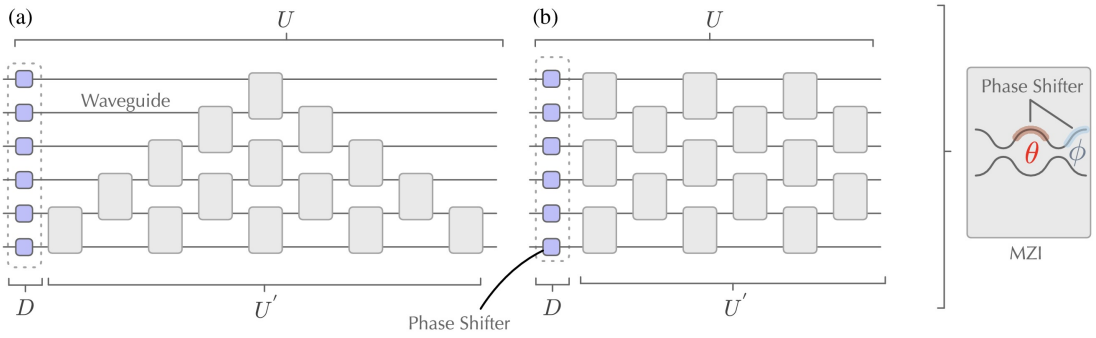


Figure 1.20 / UPP layouts using reconfigurable MZIs. (a) Reck encoding [40]. (b) Clements encoding [41]. Both configurations require 15 unit cells to realize all $U(6)$ unitary matrices. Fig. from [42].

unit cells described by Eq. 1.19.

In this work, we use a six-mode UPP in the Clements configuration, consisting of 15 unit cells, as described in more detail in Chapter 2.2.

Integrated photonic quantum memristor

For implementing a photonic quantum memristor using integrated optics, it is most convenient to realize the tunable beam splitter as an MZI. It consists of two balanced beam splitters with a phase shifter in between, as shown in Fig. 1.21a. For a balanced beam splitter $R = T = 1/2$, this is, up to a global phase, equivalent to a beam splitter transformation given by Eq. 1.8 with reflectivity $R = \cos^2(\theta/2)$. Therefore, the MZI can be used as a tunable beam splitter, with the reflectivity adjusted by tuning the phase shift θ.

In Spagnolo et al. [18], the photonic quantum memristor is realized on a photonic integrated circuit fabricated by femtosecond-laser micromachining on an alumino-

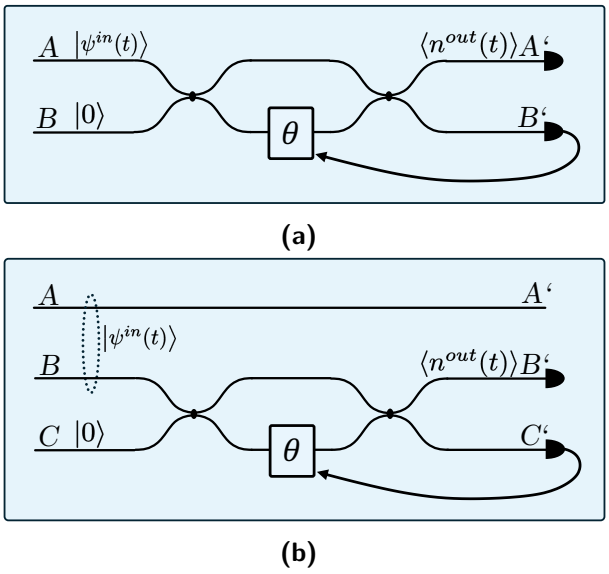


Figure 1.21 / Quantum memristor implementation using a Mach-Zehnder interferometer. (a) The lines refer to the modes in the integrated circuit and the crossings represent balanced directional couplers. The reflectivity is set via the phase shifter θ. (b) Dual-rail encoding scheme with an additional optical mode.

borosilicate glass substrate. The optical phase shifters are realized by thin metal layers that heat up and change the material's refractive index.

In equation 1.6, the qubit is encoded in a superposition of the vacuum state $|0\rangle$ and the one-photon state $|1\rangle$ (single-rail encoding). However, depending on the type of source and considered application, it can be more practical to encode qubits in a superposition of two paths (dual-rail encoding) in experiments. The equivalent dual-rail description of the two states is given by

$$|0\rangle \rightarrow |10\rangle, |1\rangle \rightarrow |01\rangle. \tag{1.22}$$

Instead of defining $|0\rangle$ as the absence and $|1\rangle$ as the presence of the photon in a single mode, an additional mode is introduced with the photon occupying either one mode or the other. In this representation, the input state is written as:

$$\left|\psi^{tot}\right\rangle = (\alpha(t)|10\rangle_{AB} + \beta(t)|01\rangle_{AB}) \otimes |0\rangle_{C}, \tag{1.23}$$

with the added mode as shown in Fig. 1.21b. The vacuum state corresponds to the photon being in mode A, while the state $|1\rangle$ corresponds to the photon being in mode B. Compared to Eq. 1.6, only the way in which the state is encoded is different. The output of the device is now given by mode B' instead of A' and C' acts as the feedback mode. Let us note that, both the states $|10\rangle_{A'B'}$ and $|00\rangle_{A'B'}$ correspond to a vacuum state in said feedback mode.

1.4 Spontaneous parametric down-conversion

In our experiment, the single-photon source is based on the spontaneous parametric down-conversion (SPDC) process. This method is probabilistic, generating correlated photon pairs via a nonlinear process.

Specifically, SPDC takes place in a medium with a second order susceptibility $\chi^{(2)}$ where a pump photon is annihilated and to lower-energy photons, referred to as signal and idler, are created. The process is "spontaneous" in the sense that it is initiated by vacuum fluctuations of the electromagnetic field. Depending on the polarization of input and output photons, SPDC is categorized in different types. The source in our experiment is based on type-II SPDC, in which the polarizations of signal and idler photons are perpendicular.

Energy conservation requires that

$$\omega_p = \omega_s + \omega_i, \tag{1.24}$$

and momentum conservation imposes the phase-matching condition:

$$\mathbf{k}_p = \mathbf{k}_s + \mathbf{k}_i, \tag{1.25}$$

where ω is the respective frequency and $\mathbf{k}$ is the respective wave vector of pump, signal and idler photon. In general, the phase-matching condition is not automatically satisfied, giving rise to a phase mismatch $\Delta\mathbf{k} = \mathbf{k}_p - \mathbf{k}_s + \mathbf{k}_i$, which impacts conversion efficiency.

Phase-matching can be achieved via birefringence where the polarization-dependent refractive index is exploited. However, this imposes strict constraints on crystal orientation and operating temperature. A more flexible approach, also used in the present work, is quasi phase-matching. Here, the sign of the nonlinear coefficient is periodically inverted along the propagation direction. This is achieved in ferroelectric crystals such as potassium titanyl phosphate via periodic poling where a strong electric field reverses the orientation of the domains during fabrication. The domain inversion resets the phase of the nonlinear polarization before it can destructively interfere with the field, so that the signal accumulates along the crystal.

2 Reservoir computing with photonic quantum memristors

This chapter presents a previous implementation of reservoir computing based on the photonic quantum memristor, on which we build for the original work in this thesis. The improved memristor design and the full reservoir computing framework will be described in detail, along with the experimental apparatus and measurement procedure.

2.1 Reservoir computing framework

The quantum memristor is the core element of our RC framework. Previous implementations of RC with quantum memristors [19] relied solely on updating the reflectivity of the memristor for memory and nonlinearity. Here, we extend this approach by introducing an injection mechanism that make following inputs depend on previous measurement outcomes, thereby enhancing memory capacity.

Original design

In the original memristor design [19], depicted in Fig. 2.3(a), the total state at the input of the memristor is given by:

$$\left|\psi^{tot}(t)\right\rangle = (\alpha(t)|10\rangle + \beta(t)|01\rangle)_{AB} \otimes |0\rangle_C = \left|\psi^{in}(t)\right\rangle_{AB} \otimes |0\rangle_C \tag{2.1}$$

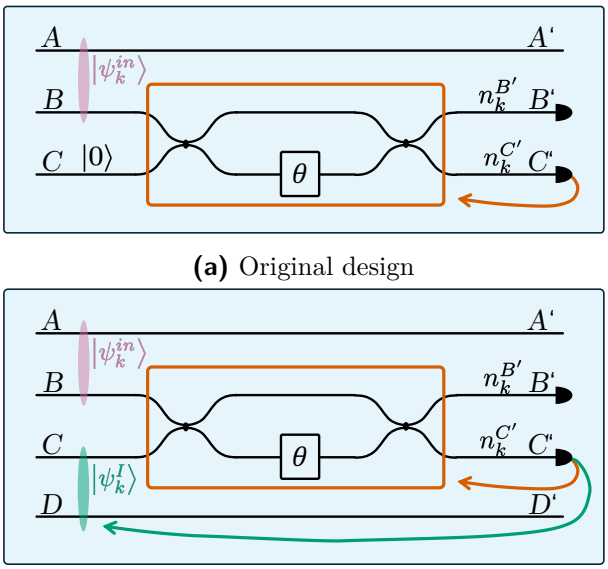
where $\alpha(t)$ and $\beta(t)$ are complex coefficients satisfying $|\alpha(t)|^2 + |\beta(t)|^2 = 1$. The input data is encoded in modes A and B, while mode C remains unused. The photon-number expectation value of the photon number operator $\hat{n}_B = |1\rangle\langle 1|_B$ corresponding to the mean number of photons in the input of mode B yields

$$\left\langle \hat{n}^B(t) \right\rangle = \left\langle \psi^{tot}(t) \,\middle|\, \hat{n}^B \,\middle|\, \psi^{tot}(t) \right\rangle = |\beta(t)|^2. \tag{2.2}$$

The reflectivity R is updated using the average of the previous k_R inputs in mode B inside the integration window $\tau_R = k_R \Delta t$.

In [19], reservoir computing with a photonic quantum memristor was first experimentally implemented. It was shown that a single device with a feedback loop that updates based on past measurement outcomes can successfully perform machine learning tasks. The results for several time series prediction tasks are shown in Fig. 2.4, comparing the performance with a device without a feedback loop. For these tasks, the feedback rule is chosen as $R_k = R_{k-1} + \frac{\langle n^B_{k-1}\rangle - R_{t-1}}{m_d}$, where m_d defines how fast the memory of past outputs will decay.

While this demonstrates the architecture's potential, it is limited in its memory capacity and the type of tasks it can carry out. Furthermore, the performance could



(a) Original design

(b) Layout with input injection

Figure 2.3 / Schematic of the photonic quantum memristor. (a) The original design proposed in [18], where the input state is dual-rail encoded in the two upper modes and mode C is unused. The reflectivity is updated with the measurement outcome. (b) A fourth optical mode is added to allow for encoding a state with the previous inputs of the device in the lower two modes.

be improved by constructing a reservoir of multiple memristor units, expanding the feature space.

Injection protocol

To improve memory capacity, we modify the original protocol by using mode C for reinjecting previous outputs and adding mode D to enable dual-rail encoding of both the input state and injection feedback. The new layout is shown in Fig. 2.3(b). The state entering the memristor at time step k becomes:

$$\left|\psi^{tot}(t)\right\rangle = (\alpha_k(t)|10\rangle + \beta_k(t)|01\rangle)_{AB} \otimes (\gamma_k|01\rangle + \lambda_k|10\rangle)_{CD} \tag{2.5}$$

where $\alpha_k, \beta_k, \gamma_k$ and λ_k are complex coefficients satisfying $|\alpha_k|^2 + |\beta_k|^2 = 1$ and $|\gamma_k|^2 + |\lambda_k|^2 = 1$. The state is expressed as the tensor product of the original memristor input state in modes A and B and the state containing the history of the device in modes C and D. With $\left\langle n^X(t_k)\right\rangle \equiv n_k^X$, the equations for the mean photon number at the output can be written as:

$$\begin{aligned} n_k^{A'} &= 1 - n_k^B, \\ n_k^{B'} &= R_k n_k^C + T_k n_k^B, \\ n_k^{C'} &= R_k n_k^B + T_k n_k^C, \\ n_k^{D'} &= 1 - n_k^C \end{aligned} \tag{2.6}$$

where R_k is the reflectivity at time step k and $T_k = 1 - R_k$. The reflectivity depends on the previous k_R inputs. The average number of photons in mode

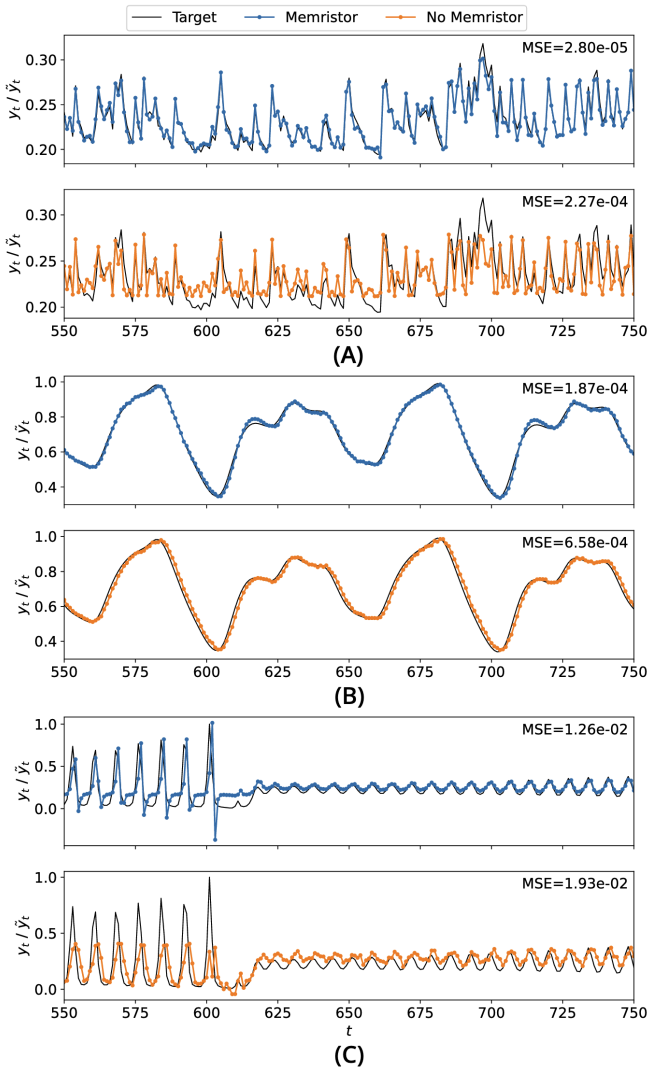


Figure 2.4 / Experimental results of prediction tasks with and without memristor. As shown in [19], the memristor can predict time series with high accuracy. Without the memristor feedback loop, the performance is worse.

C' will be used for injection feedback. Analogous to k_R, we define k_I with the integration window $\tau_I = k_I \Delta t$. k_I is the number of previous outputs from C' that will be used to calculate the injection input n_k^C. In general, any arbitrary function acting on previous inputs and outputs can be used to update the reflectivity and injection, provided the result lies in the allowed range $[0, 1]$ for R and n^C.

We note that by adding this additional feedback mechanism, the device does not meet the formal definition of a memristor given by Eq. 1.3. Nevertheless, this design significantly extends the system's memory, as shown in Chapter 3. To simplify the discussion, we will continue referring to the device as a photonic quantum memristor throughout the rest of the thesis, even though it is not strictly one.

Spatial multiplexing

To expand the output space to which our inputs are mapped and thus enhance the expressivity of the model, we can resort to parallel units operating independently. Therefore, we will consider a reservoir consisting of N independent units operating in parallel or in sequence. Each unit follows the dynamics described in the previous section, but with unique parameters to ensure distinct behavior.

We define a vector $\mathbf{n}_k^M$ containing the mean photon numbers of all N devices in an arbitrary mode $M = A, A', B, \ldots$ at time step k with the elements $\left(\mathbf{n}_k^M\right)_j \equiv \left\langle n_{k;j}^M \right\rangle$, where the index j refers to the j-th device. Generalizing from Eq. 2.6, the full reservoir output at each time step can be written as:

$$\begin{aligned} \mathbf{n}_k^{A'} &= \mathbf{1}^N - \mathbf{n}_k^B, \\ \mathbf{n}_k^{B'} &= \mathbf{R}_k \odot \mathbf{I}_k + \mathbf{T}_k \odot \mathbf{n}_k^B, \\ \mathbf{n}_k^{C'} &= \mathbf{R}_k \odot \mathbf{n}_k^B + \mathbf{T}_k \odot \mathbf{I}_k, \\ \mathbf{n}_k^{D'} &= \mathbf{1}^N - \mathbf{I}_k \end{aligned} \tag{2.7}$$

where $\odot$ denotes the Hadamard product and $\mathbf{1}^N$ is an N-dimensional vector of ones. The terms $\mathbf{R}_k$ and $\mathbf{I}_k$ contain the values of the reflectivity and injection terms for all devices at time step k. The feedback terms are obtained by applying a function to the corresponding feedback vectors and can be written as:

$$\mathbf{R}_k = \mathbf{h}_R(\mathbf{F}_{k,k_R}^R), \mathbf{I}_k = \mathbf{h}_I(\mathbf{F}_{k,k_I}^I) \tag{2.8}$$

with

$$\mathbf{h}_X(\mathbf{F}_{k,k_X}^X) = \begin{bmatrix} h_{X;1}(\mathbf{f}_{k;1}^X) \\ h_{X;2}(\mathbf{f}_{k;2}^X) \\ \vdots \\ h_{X;N}(\mathbf{f}_{k;N}^X) \end{bmatrix}_{N\times 1} \tag{2.9}$$

where $\mathbf{h}_X$ applies a function $h_{X;j}(\cdot)$ applies a function to the feedback vector $\mathbf{f}_{k;j}^X$ of each device for feedback mechanism ($X = R, I$). This function can differ across units and, as mentioned in the previous section, it is generally arbitrary. The specific functions that are used in this implementation are shown in Table 2.10. The feedback vector contains information about the device's past states within the integration window k_X. That is, $\mathbf{f}_{k;j}^R$ depends on the previous k_R mean photon numbers at the input mode B and $\mathbf{f}_{k;j}^I$ depends on the previous k_I mean photon numbers at input mode C.

The feedback vectors depend on past states within the integration windows k_R and k_I:

$$\begin{aligned} \mathbf{f}_{k;j}^R &= [(\mathbf{M}_j^B)_1 n_{k-1;j}^B, ..., (\mathbf{M}_j^B)_{k_R} n_{k-k_R;j}^B]^T, \\ \mathbf{f}_{k;j}^I &= [(\mathbf{M}_j^{C'})_1 n_{k-1;j}^{C'}, ..., (\mathbf{M}_j^{C'})_{k_I} n_{k-k_I;j}^{C'}]^T. \end{aligned} \tag{2.11}$$

Here, $\mathbf{M}_j^B$ and $\mathbf{M}_j^{C'}$ are random masks with values between zero and one that weight past terms, aiding optimization by giving each device a distinct output.

Case	Analytical Expression	Description
$h^{a_n}_{X;j}$	$X_{k;j} = \left(\frac{1}{k_X}\sum_{i=1}^{k_X}\left(f^X_{k;j}\right)_i\right)^n$	Average value of the feedback vector raised to the power of $n > 0$.
$h^{b_n}_{X;j}$	$X_{k;j} = \frac{1}{k_X}\sum_{i=1}^{k_X}\left(f^X_{k;j}\right)^n_i$	Average value of the feature vector elements raised to the power of $n > 0$.
$h^{c}_{X;j}$	$X_{k;j} = \frac{1}{k_X}\sum_{i=1}^{k_X}\left[1 - \left(f^X_{k;j}\right)_i\right]$	Average value of the complementary of the feature vector.

Table 2.10 / Feedback functions. Functions for the reflectivity ($X = R$) and injection ($X = I$) update mechanisms. The functions are applied to the elements of the feedback vector $\mathbf{f}^X_{k;j}$, i.e. the mean photon numbers inside the integration window multiplied by a mask.

Based on these considerations, we can identify three different memory dynamics that we can exploit:

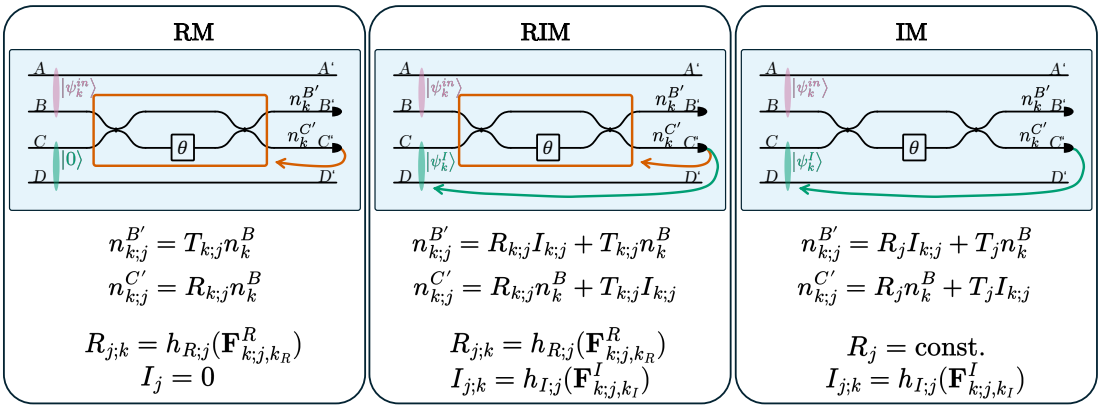


Figure 2.12 / Overview of the three different memory dynamics that can be employed on a single device. The memory is provided by either only the reflectivity update (RM), only the injection in the input (IM), or a combination of both (RIM). The equations for the output are given by Eq. 2.7.

Reflectivity Memory (RM) Dynamics. The memory is exclusively provided by the reflectivity update. The injection mechanism is turned off with $I_{k;j} = 0 \forall j$. By choosing $h^{a_1}_{R;j}$ as the feedback function, the dynamics from the original memristor design are recovered except for the random masks that are used in this framework. The reservoir retains only information about the previous k_R inputs, and the device has no memory of earlier inputs. It follows that while the echo state property is satisfied, this dynamic does not exhibit a fading memory.

Injection Memory (IM) Dynamics. The memory is exclusively provided by the injection update. Each device is assigned a random reflectivity that remains

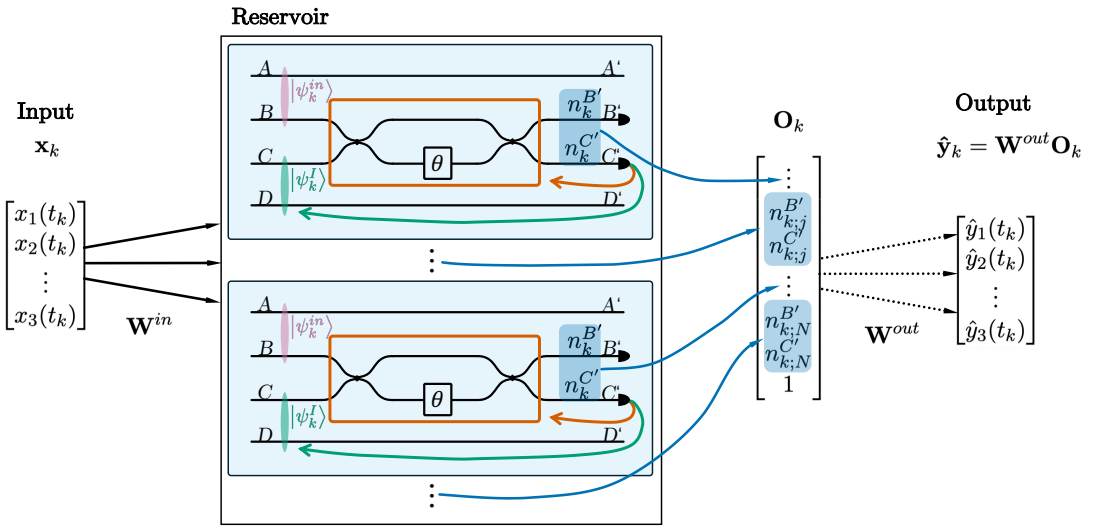


Figure 2.13 / Full reservoir computing scheme using spatial multiplexing. The reservoir consists of N photonic quantum memristors. At time step k, the input $\mathbf{x}_k$ is introduced into the reservoir using a random layer $\mathbf{W}^{in}$. Each device encoded the input in optical modes A and B and the injection term is introduced in C and D. The feature vector $\mathbf{O}_k$ is constructed with the mean photon number measured in output modes B' and C'. The output layer $\mathbf{W}^{out}$ is trained using ridge regression.

constant. This dynamic provides more memory than RM, since each previous output injected also contains information about the past and the input. However, this contribution fades with time as the terms are multiplied by the reflectivity and transmission coefficients at each time step. Hence, a unit with IM dynamics is expected to exhibit fading memory behavior.

Reflectivity and Injection Memory (RIM) Dynamics. This dynamic combines RM and IM dynamics, so the memory is given by both reflectivity and injection update.

An overview of the dynamics along with the corresponding equations given by Eq. 2.7 is shown in Fig. 2.12. Each reservoir unit can employ different memory dynamics and feedback functions, ensuring uniqueness via random masks and reflectivities.

Fig. 2.1 shows the full reservoir computing scheme with spatial multiplexing of N units. At time step k, an input data point $\mathbf{x}_k \in \mathbb{R}^{d_x}$ is introduced into the reservoir. In general, the input is multiplied by an input layer matrix $\mathbf{W}^{in} \in \mathbb{R}^{N\times d_x}$ so that $\mathbf{n}_k^B = \mathbf{W}^{in}\mathbf{x}_k$. This matrix can either be an identity or a fully random matrix, generated from uniformly distributed numbers between zero and one and properly scaled such that the input in each device is normalized. The output of the reservoir follows from Eqs. 2.7 and from it the observation feature vector $\mathbf{O}_k$ is constructed:

$$\mathbf{O}_k = [n_{k;1}^{B'}, n_{k;1}^{C'}, \ldots, n_{k;N}^{B'}, n_{k;N}^{C'}, 1]^T \in \mathbb{R}^{2N+1} \tag{2.14}$$

which includes the mean photon numbers at output modes B' and C' plus a

constant bias term. The final result is obtained by

$$\hat{\mathbf{y}}_k = \mathbf{W}^{out}\mathbf{O}_k \tag{2.15}$$

where $\mathbf{W}^{out}$ is the output layer matrix whose parameters are the only ones that are trained in this process. We use ridge regression to optimize $\mathbf{W}^{out}$, which is a regularized linear regression. The training data set consists of N_{train} time steps. The output layer can be expressed as:

$$\mathbf{W}^{out} = \mathbf{y}^{tgt}\mathbf{O}_{total}^{T}(\mathbf{O}_{total}\mathbf{O}_{total}^{T} - \alpha\mathbf{I})^{-1} \tag{2.16}$$

where $\mathbf{O}_{total} \in \mathbb{R}^{(2N+1)\times N_{train}}$, $\mathbf{I}$ is the identity matrix and α is the ridge regularization parameter.

2.2 Experimental implementation

The full experimental setup is shown in Fig 2.17. The core part is a photonic processor which physically realizes a single reservoir neural unit. To increase the output space of the reservoir, spatial multiplexing is achieved by operating units with different initializations sequentially. The input of the reservoir is a two-photon Fock state, where the photons are generated by SPDC. The detection is performed with single-photon detectors at each output mode of the processor. In the following sections, the experimental apparatus consisting of source, processor and detection along with the measurement procedure is described in detail.

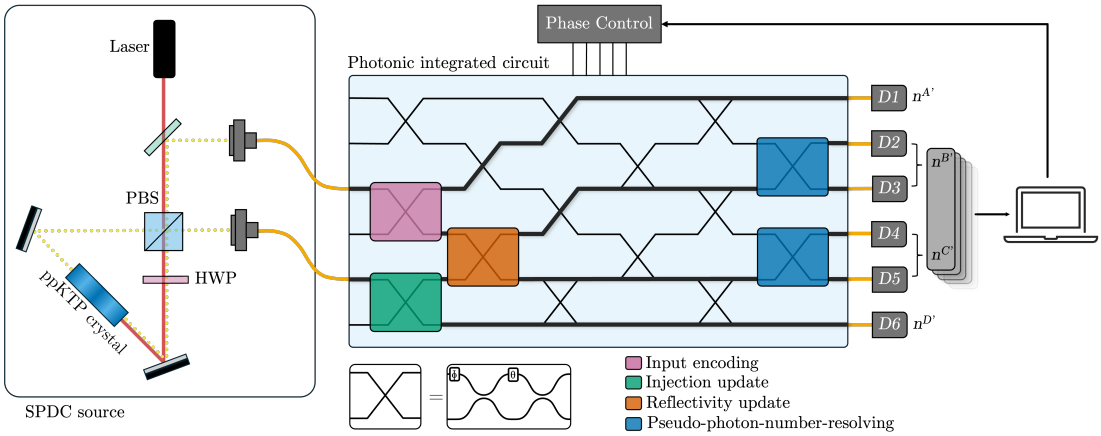


Figure 2.17 / Full experimental implementation. The apparatus consists of a single photon source, the universal photonic processor, single photon detectors and the classical linear regression model. The photons are generated by a SPDC source with a periodically poled potassium titanyl phosphate (ppKTP) crystal pumped by a 775 nm laser. The two generated 1550 nm photons are collected and sent to two input modes of the processor via single-mode fibers. Each crossing in the chip layout represents one MZI. One of the MZIs at the input encodes the input data point on the first photon and another two the injection and memristor reflectivity according to the feedback dynamic. The other MZIs are set such that each mode is sent the corresponding output of the chip, with modes B and C featuring two output modes to perform pseudo-number-resolving detection. The reflectivity/injection feedback is calculated and set by the chip's electronic phase control. This is repeated for each reservoir unit after which the measured values for each time step are used to train a linear regression network.

Single-photon source

The single photons constituting the input of the protocol are generated via type-II SPDC in a Sagnac loop configuration (see Fig. 2.17) pumped at 775 nm and generating degenerate photons at 1550 nm. The employed nonlinear crystal is a ppKTP crystal, which generates photons in a collinear way. Since we do not require entanglement for our experiment, the crystal is pumped only in one direction within the interferometer. Let us note that such a crystal is of type-II,

implying that the generated photons have orthogonal polarizations. The photons are collected into single-mode fibers, which route them to the input modes of the photonic processor.

Since the feedback, as well as the feature vector used to train the reservoir, consists solely of the measured mean photon numbers, this framework does not rely on quantum interference, and the photons do not have to be indistinguishable for this experiment.

Photonic integrated circuit

We implement our RC framework on a six-mode UPP, the device is capable of implementing any 6×6 unitary transformation between input/output states in the Fock basis [43]. It is made up of only linear optical elements, i.e., beam splitters and phase shifters that comprise fifteen tunable MZIs in the arrangement described in [41]. The device is fabricated on a borosilicate glass substrate through femtosecond laser writing [44]. The tunability of the MZIs is achieved with thermo-optic phase shifters, implemented through gold microheaters deposited on the surface of the chip. Applying a current to a microheater resistively heats it, locally changing the refractive index of the glass and thereby introducing a phase shift. Each MZI features one internal phase shifter θ, which sets the splitting ratio of the MZI, and an external phase shifter ϕ, which sets the relative phase between the two arms. The phase shifters are calibrated to accurately map applied electrical current to the resulting optical phase, and to account for the nonlinear dependence of the phase on the current as well as the thermal crosstalk between neighboring phase shifters.

The quantum reservoir computing (QRC) framework shown in Fig. 2.3 is implemented using the MZIs on the chip by setting the internal phases as shown in Fig. 2.18. The photons enter modes 3 and 5 of the chip, where the MZIs path-encode

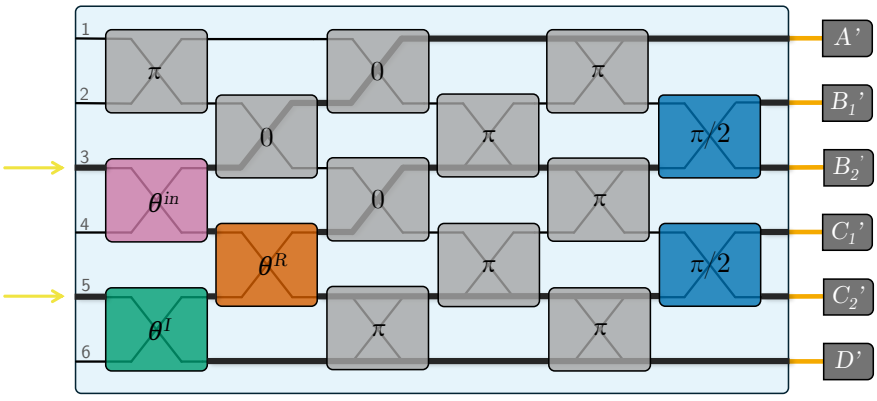


Figure 2.18 / Internal phases of the MZIs on the 6-mode UPP. To implement the QRC framework from Fig. 2.3, three of the MZIs have to be updated, the rest of the internal phases remain constant and guide the photons to the output modes. Output modes B' and C' are both split into two modes to realize pseudo-photon-number-resolving detection. All external phases are set to zero.

the input data point x_k in the state $|\psi_k^{in}\rangle = \sqrt{1-x_k}|10\rangle_{AB} + \sqrt{x_k}|01\rangle_{AB}$ and the injection feedback I_k in $\left|\psi_k^I\right\rangle = \sqrt{I_k}|10\rangle_{CD} + \sqrt{1-I_k}|01\rangle_{CD}$. Another MZI acts as the memristor MZI, which is updated with the reflectivity feedback R_k. Specifically, the internal phases at each time step k are set as:

$$\begin{aligned} \theta_k^{in} &= 2\arccos\left(\sqrt{x_k}\right) \\ \theta_k^{I} &= 2\arcsin\left(\sqrt{I_k}\right) \\ \theta_k^{R} &= 2\arccos\left(\sqrt{R_k}\right). \end{aligned} \tag{2.19}$$

The rest of the phases remain constant at each time step with the subsequent MZIs being set to be either fully transmittive ($\theta = \pi$) or reflective ($\theta = 0$) to route the photons to the correct output modes. Since both photons can end up in modes B' or C', the MZIs at these modes is set as a 50/50 beam splitter to perform pseudo-number-resolving detection in order to obtain the correct number of photons. This will be described in more detail in the following section. The external phases are not needed for this application and are set to zero.

Detection and feedback

Each of the six output ports is coupled to a single-mode fiber that send the photons to the eight superconducting nanowire single-photon detectors (SNSPDs). When a photon is detected by a SNSPD, it creates an electric pulse that is first amplified and then sent to a time-tagger for processing. We are considering the two-fold coincidences between photon pairs that are detected by any two detectors inside a 1 ns time window. We integrate the coincidence counts over a few seconds, which are then used to compute the mean photon number in each mode. Modes B' and C' each have two detectors to realize photon-number-resolving detection: if there are two photons in the mode, the 50:50 beam splitter probabilistically splits them up implying that both detectors of the mode produce a click. On the contrary, when the photons are in the same mode, only one of the detectors will click and no photon is detected. Because the ratio between both cases is 50%, we can multiply the number of events in the former case by two to approximate the mean photon number in modes B' and C'. Specifically, the mean photon numbers are calculated as follows:

$$\begin{aligned} n^{A'} &= \sum_X \frac{cc_{A'-X}}{cc_{tot}} \\ n^{B'} &= \sum_X \frac{cc_{B'_1-X}}{cc_{tot}} + \sum_X \frac{cc_{B'_2-X}}{cc_{tot}} + 2\,\frac{cc_{B'_1-B'_2}}{cc_{tot}} \\ n^{C'} &= \sum_X \frac{cc_{C'_1-X}}{cc_{tot}} + \sum_X \frac{cc_{C'_2-X}}{cc_{tot}} + 2\,\frac{cc_{C'_1-C'_2}}{cc_{tot}} \\ n^{D'} &= \sum_X \frac{cc_{D'-X}}{cc_{tot}}\,, \end{aligned} \tag{2.20}$$

where the sums are calculated over all detectors $X \in \{A', B'_1, B'_2, C'_1, C'_2, D'\}$. The feedback vectors from Eq. 2.11 are then constructed by multiplying the reflectivity

mask M_j^B of the current unit with index j with the previous inputs in mode B, specified by k_R, and the injection mask $M_j^{C'}$ with the measured outputs of mode C', specified by k_I. The feedback functions $h_{R;j}(\cdot)$ and $h_{I;j}(\cdot)$ are then applied to the feedback vector to calculate R_{k+1} and I_{k+1}, which are then used to update the phases in Eq. 2.19 in the next step. After repeating the procedure for the whole input series, the linear network is trained. A small portion of the data is discarded to ensure the reservoir has reached a state where the output does not depend on the initial conditions. For the remaining data, the feature vector of each time step k is constructed from the measured mean photon numbers in modes B' and C' (Eq. 2.14). The data is split into a training and test set. The data of the training set is used to train the parameters of the output layer matrix W^{out} with ridge regression (Eq. 2.16).

In particular, the whole procedure is as follows:

Input: Time series data $\mathbf{x} \in \mathbb{R}^{N_{total}}$
Parameters: N units, k_R, k_I, N_{warmup}, N_{train}, N_{test}, α (ridge parameter)

// Initialize Reservoir Units
for $j = 1$ to N **do**
  Initialize unit U_j with:
    Feedback mode $\in \{\text{RM}, \text{IM}, \text{RIM}\}$
    Random reflectivity R_j
    Feedback lengths k_R, k_I
    Random masks $\mathbf{M}_j^B, \mathbf{M}_j^C$
    Feedback functions $h_{R;j}(\cdot)$, $h_{I;j}(\cdot)$
end for

// Sequential Reservoir Operation
for $j = 1$ to N **do** ▷ For each reservoir unit
  Compute initial MZI phase $\theta_0^R = 2\arccos\left(\sqrt{R_j}\right)$
  Initialize injection $I_0 = 0$
  for $k = 1$ to N_{steps} **do** ▷ For each time step
    // Update Feedback
    if $k > 1$ **then**
      if IM or RIM **then**
        $\tau = \min(k_I, k)$
        $\mathbf{f}_k^I = \mathbf{M}_j^C[1:\tau] \odot \mathbf{n}_{j,[k-\tau:k-1]}^{C'}$
        $I_k = h_{I;j}(\mathbf{f}_k^I)$
      end if
      if RM or RIM **then**
        $\tau = \min(k_R, k)$
        $\mathbf{f}_k^R = \mathbf{M}_j^B[1:\tau] \odot \mathbf{x}_{[k-\tau:k-1]}$
        $R_k = h_{R;j}(\mathbf{f}_k^R)$
      end if
    end if
    // Set Phase Shifters

Calculate phases ▷ see Eqs. 2.19
Convert phases to electrical currents
Apply to chip phase shifters
// Wait and Measure
Wait for phase shifters to heat up (∼1s)
Measure coincidence counts over integration time (∼ 3s)
Calculate mean photon numbers ▷ see Eqs. 2.20
Store result
end for
// Construct Feature Vector
$\mathbf{O}_k = [\langle n_{k;1}^{B'} \rangle, \langle n_{k;1}^{C'} \rangle, \ldots, \langle n_{k;N}^{B'} \rangle, \langle n_{k;N}^{C'} \rangle, 1]^T \in \mathbb{R}^{2N+1}$
end for

// Discard Warmup Period
Discard first N_{warmup} time steps to remove transient effects

// Train Output Layer
Construct training matrix:
$\mathbf{O}_{total} = [\mathbf{O}_{N_{warmup}+1}, \ldots, \mathbf{O}_{N_{warmup}+N_{train}}] \in \mathbb{R}^{(2N+1)\times N_{train}}$
Construct target matrix according to task:
$\mathbf{y}^{tgt} = [y^{tgt}_{N_{warmup}+1}, \ldots, y^{tgt}_{N_{warmup}+N_{train}}]$
Compute output weights via Ridge regression:
$\mathbf{W}^{out} = \mathbf{y}^{tgt}\mathbf{O}_{total}^T(\mathbf{O}_{total}\mathbf{O}_{total}^T + \alpha\mathbf{I})^{-1}$

// Test and Evaluate
for $k = N_{warmup} + N_{train} + 1$ to N_{total} **do**
$\hat{\mathbf{y}}_k = \mathbf{W}^{out}\mathbf{O}_k$
end for

Using this process, we obtain the results for two different benchmark machine learning tasks. Their performance will be discussed in the next chapter.

3 Results

In the following sections, we describe the benchmark machine learning tasks used for evaluating the performance of our reservoir computing scheme and present the results. To quantify the performance of each task, we define the correlation coefficient as follows:

$$L_{\text{task}} = \frac{\text{cov}^2(\mathbf{y}^{\text{tgt}}, \hat{\mathbf{y}})}{\sigma^2(\mathbf{y}^{\text{tgt}}) \cdot \sigma^2(\hat{\mathbf{y}})}\,, \tag{3.1}$$

where $\mathbf{y}^{\text{tgt}}$ is the target time series of the corresponding task. The functions $\text{cov}^2(\cdot,\cdot)$ and $\sigma^2(\cdot)$ refer to the square of the covariance and standard deviation, respectively. A correlation coefficient of $L_{\text{task}} = 1$ means the target was predicted perfectly by the model, whereas $L_{\text{task}} = 0$ implies that there is no correlation between the predicted result and the target.

We analyze the performance of two benchmark tasks. The short-term memory (STM) task, which characterizes the system's linear memory, is performed for three reservoirs with units following RM, IM or RIM dynamics. In the nonlinear auto-regressive moving average (NARMA) task, we evaluate the system's nonlinear memory and compare the performance of a RM reservoir to that of a reservoir utilizing the injection feedback mechanism containing both IM and RIM units. Finally, we discuss the possibility of forecasting the dynamics of a Lorenz system with our reservoir by presenting simulated data.

3.1 Short-term memory task

In the STM task, the reservoir is fed an input sequence of uniformly distributed random numbers in the range $[0, 1]$, and the output layer is trained to generate a sequence of past inputs with a specific delay. This task evaluates the system's linear memory. In our experiment, the input time series x_k is a sequence of uniformly distributed random numbers in the range $[0, 1]$. The target is given by the input sequence delayed by k_d time steps as $y_k^{\text{tgt}} = x_{k-k_d}$ [45]. With larger values of k_d, the system requires a longer linear memory to achieve good performance. For this task, the correlation coefficient is denoted as $L_{\text{STM}}(k_d)$.

We use $N = 20$ parallel units to perform the STM task. All units follow either RM, IM, or RIM dynamics (see Section 2.1), and we evaluate the performance of different integration windows, k_R and k_I. In the case of RM dynamics, the feedback function $h^{a_1}_{R;j}\forall j$ is used, which corresponds closest to the previous quantum memristor implementation without input injection [18] [19]. The IM dynamics employs $h^{a_1}_{I;j}\forall j$ and the RIM dynamics use both $h^{a_1}_{R;j}\forall j$ and $h^{a_1}_{I;j}\forall j$ for this task. For this task, $N_{warmup} = 20$, $N_{train} = 180$ and $N_{test} = 100$ data points are used for training and testing. The ridge regression parameter (see Eq. 2.16) is set to $\alpha = 10^{-5}$ for experimental data and $\alpha = 10^{-7}$ for theoretical data. The experimental error,

arising primarily from the finite sample size of the measurements, is estimated by sampling from a Poisson distribution centered on the measured probabilities and repeating the training and testing steps. The error is then calculated as the standard deviation of 20 correlation coefficients calculated from data sampled from the distribution.

The results for each feedback dynamics are shown in Fig. 3.2. For the RM reservoir, results are similar for experimental and theoretical data. It shows a high performance of $L_{\mathrm{STM}} \approx 1$ only for the trivial case with no delay, where $k_d = 0$. For delays in the range $1 \leq k_d \leq k_R$, the correlation coefficient plateaus around $L_{\mathrm{STM}} \approx 0.75$. For larger delays, $L_{\mathrm{STM}} \approx 0$, meaning the system fails to learn the delayed input sequence. This is expected because the reservoir retains only the last k_R input values.

The IM reservoir achieves a significantly better performance than the RM reservoir for both experimental and theoretical data. A high performance of $L_{\mathrm{STM}} \approx 1$ is retained for longer delays, and the performance declines more gradually than in the RM reservoir, where the decline is abrupt. Additionally, greater values of k_I result in a longer linear memory. The IM reservoir performs better than the RM reservoir because the injection dynamic not only introduces information from the last k_I outputs but also from all previous outputs. While also outperforming the RM reservoir, the experimental data shows a significantly lower performance than the theoretical data, with a higher discrepancy for larger values of k_I. This is due to the measured outputs having experimental errors, primarily from finite statistics. When injecting the previous outputs into the next time step, an error is introduced into the reservoir and propagates to the newly measured outputs.

The RIM reservoir's performance is slightly worse than that of the IM reservoir. The correlation coefficient drops below 1 with $k_d > 0$. This result is expected, as the STM task measures the system's linear memory. The combination of RM and IM dynamics improves the system's nonlinear memory but reduces its linear memory, thereby reducing performance on this task. This can be understood by considering the reservoir output shown in Fig. 2.12; for the RIM dynamic, both the reflectivity term and the injection term depend on the device's history, multiplying them increases the nonlinear memory of the system. Thus, for this task, the IM reservoir outperforms the RIM and RM reservoirs, highlighting the importance of tuning the reservoir to the task requirements. The following section discusses a more complex task that requires nonlinearities.

3.2 Nonlinear auto-regressive moving average task

This task evaluates the system's ability to learn a nonlinear function of a delayed time series. An input sequence x_k of uniformly distributed random numbers in the range $[0, 0.02]$ is introduced to the reservoir. The NARMA-n series is given by

[46]:

$$z_{k+1} = 0.3z_{k-1} + 0.05z_k \sum_{i=0}^{n-1} z_{k-i} + 1.5x_k x_{k-(n-1)} + 0.1, \tag{3.3}$$

where n is the order of the series, setting the maximum delay. The function is quadratic in both the input and the reservoir variable. The target of each series corresponds to the $y^{\text{tgt}} = z_k$. The correlation coefficient is denoted as $L_{\text{NARMA}}(n)$.

In the previous task, all N units of the reservoir followed the same memory dynamics, reflectivity functions, and injection functions. To achieve richer dynamics that are beneficial for more complex tasks like NARMA, we use a RIM-IM reservoir composed of a mixed set of devices. The reservoir consists of:

- 25 devices employing IM dynamics with feedback function $h^{a_1}_{I;j}$ with $k_I = 5 \forall j$.
- 25 devices employing RIM dynamics with feedback functions $h^{c}_{R;j}$ and $h^{a_1}_{I;j}$ with $k_R = k_I = 5 \forall j$.

The RM reservoir contains 50 units following RM dynamics with $h^{a_1}_{R;j} \forall j$ and $k_R = 5$. We use $N_{\text{warmup}} = 30$, $N_{\text{train}} = 250$ and $N_{\text{test}} = 120$ data points. The parameters for the Ridge regression analysis and the calculation of the experimental error are the same as in the previous task. In the experiment, the RIM-IM reservoir outperforms the RM reservoir for lower-order time series and performs similarly for higher-order time series. For the RM reservoir, the correlation coefficients calculated from experimental data match those from theoretical data. The RIM-IM reservoir shows a much larger discrepancy between experimental and theoretical performance, again due to finite statistics, as in the STM task.

3.3 Lorenz system forecasting

For this task, the experiment has not yet been carried out. Here, we present numerical results with simulated noise intended to reproduce the conditions in our experiment, showing how the reservoir will perform on a more sophisticated task. To emulate the experimental conditions, which, as described in the previous sections, are primarily limited by the finite measurement statistics, we sample values from a Poisson distribution centered around the theoretical reservoir outputs calculated in the simulation.

This task aims to forecast the chaotic dynamics of a Lorenz system, a simplified model for atmospheric convection. The behavior is given by three coupled differential equations [47]:

$$\begin{aligned} \dot{x_1} &= 10(x_2 - x_1), \\ \dot{x_2} &= x_1(28 - x_3) - x_2 \\ \dot{x_3} &= x_1 x_2 - \frac{8}{3} x_3 \end{aligned} \tag{3.5}$$

The system exhibits deterministic chaos, i.e., it is highly sensitive to perturbations in its initial conditions, making predictions difficult. The input is given by the

solution of Eq. 3.5. For the input $\mathbf{x}_k$ at time step k, the target is defined as

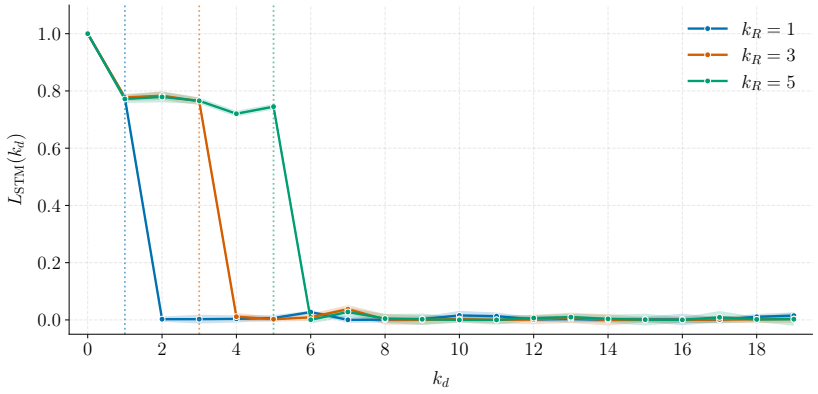
$$\mathbf{y}_k^{tgt} = \mathbf{x}_{k+1} - \mathbf{x}_k, \tag{3.6}$$

which is the difference between the current step and the subsequent one. The training phase is similar to the previous tasks, whereas during the testing phase, the reservoir evolves autonomously. Specifically, the testing phase begins by feeding the first data point of the test set into the system. In the next step, the reservoir's prediction is used to predict the following step.

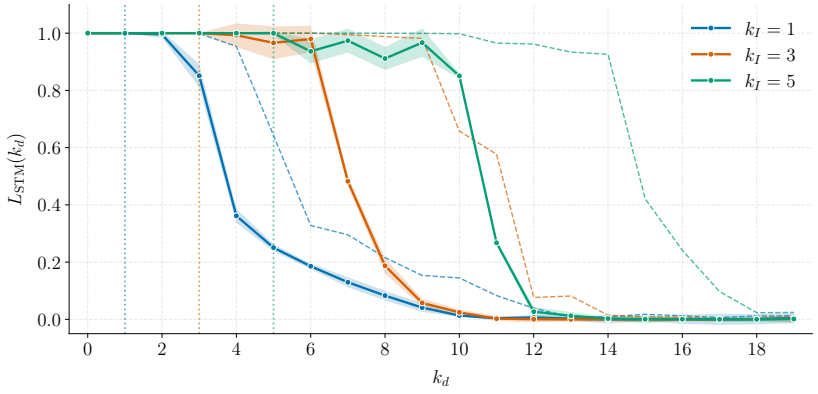
For this task, the reservoir consists of:

- 20 devices employing IM dynamics with feedback function $h_{I;j}^{a_2}$ with $k_I = 2 \forall j$.
- 20 devices employing RIM dynamics with feedback functions $h_{R;j}^{a_1}$ and $h_{I;j}^{a_1}$ with $k_R = k_I = 2 \forall j$.
- 20 devices employing RIM dynamics with feedback functions $h_{R;j}^{a_2}$ and $h_{I;j}^{a_1}$ with $k_R = k_I = 2 \forall j$.
- 20 devices employing RIM dynamics with feedback functions $h_{R;j}^{a_2}$ and $h_{I;j}^{a_2}$ with $k_R = k_I = 2 \forall j$.

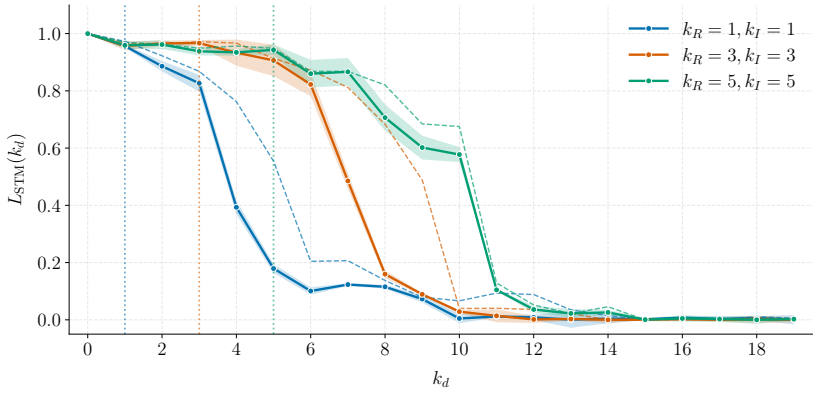
We use $N_{\text{warmup}} = 10$, $N_{\text{train}} = 500$ and $N_{\text{test}} = 300$ data points. The three dimensional input is multiplied by a random input layer matrix $\mathbf{W}^{in} \in \mathbb{R}^{N \times 3}$, resulting in a linear combination of the three coordinates for each reservoir unit. Each coordinate is initialized with a value in the range $[-0.5, 0.5]$ and the input data is normalized to lie within the range $[0, 1]$ so that it can be introduced in the photonic reservoir. The results of the simulation of this task are presented in Fig. 3.7, showing that the memristor based reservoir has the potential to be used in complex tasks.



(a) RM dynamics



(b) IM dynamics



(c) RIM dynamics

Figure 3.2 / Short-term memory task performance for different feedback dynamics and integration windows. Experimental (solid line) and theoretical (dashed line) performance is shown for reservoirs with units employing either (a) RM, (b) IM, or (c) RIM feedback dynamics. The vertical line is where the maximum delay k_d is equal to the integration window length k_X with $X = R, I$.

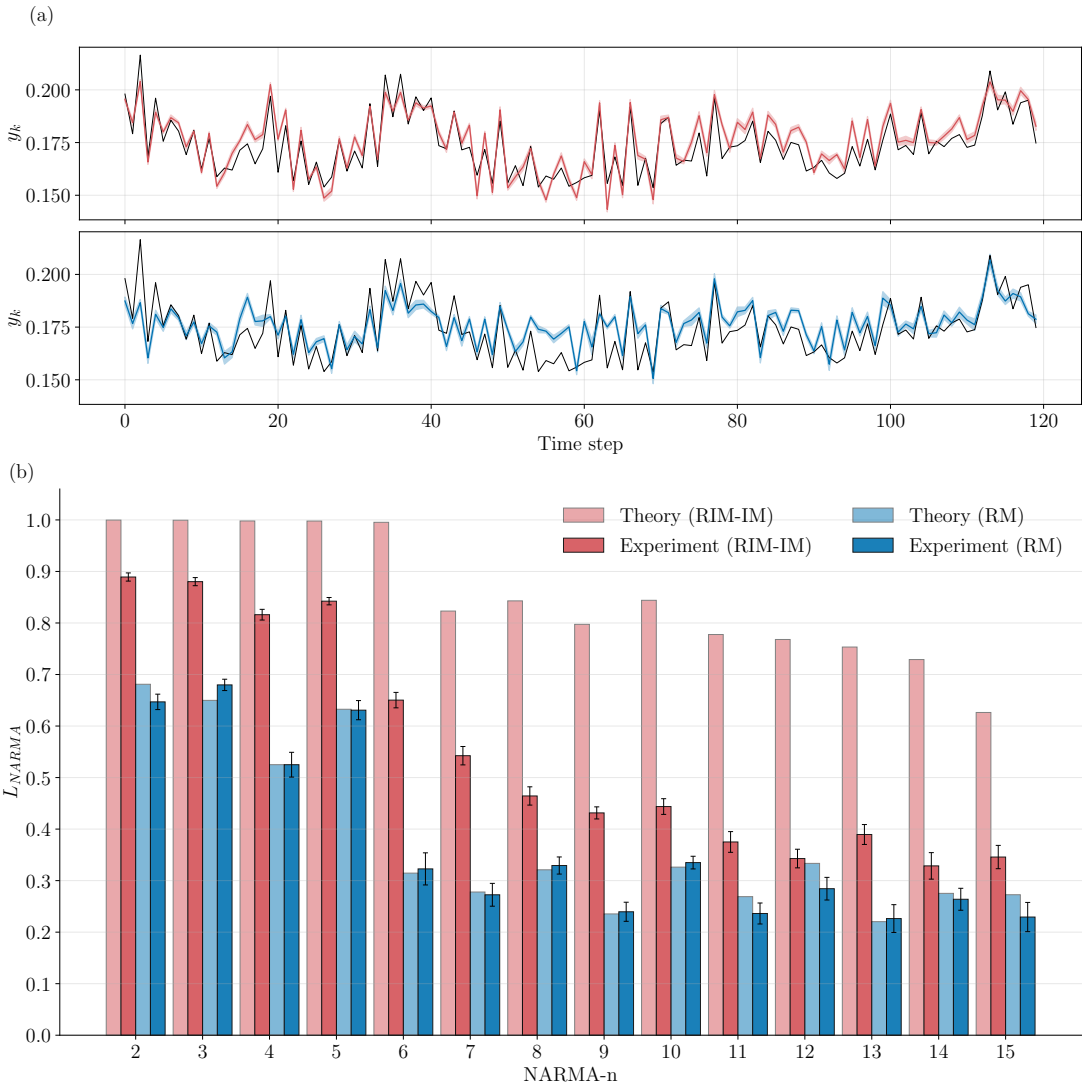


Figure 3.4 / Performance of predicting NARMA-n time series. We compare the performance of a reservoir employing RM feedback dynamics with that of a reservoir combining both IM and RIM dynamics. (a) Prediction of target series for RIM-IM reservoir (top) and RM reservoir (bottom) for the NARMA-5 task during the test phase. (b) Experimental and theoretical correlation coefficient for both reservoirs. Both reservoirs employ $N = 50$ units.

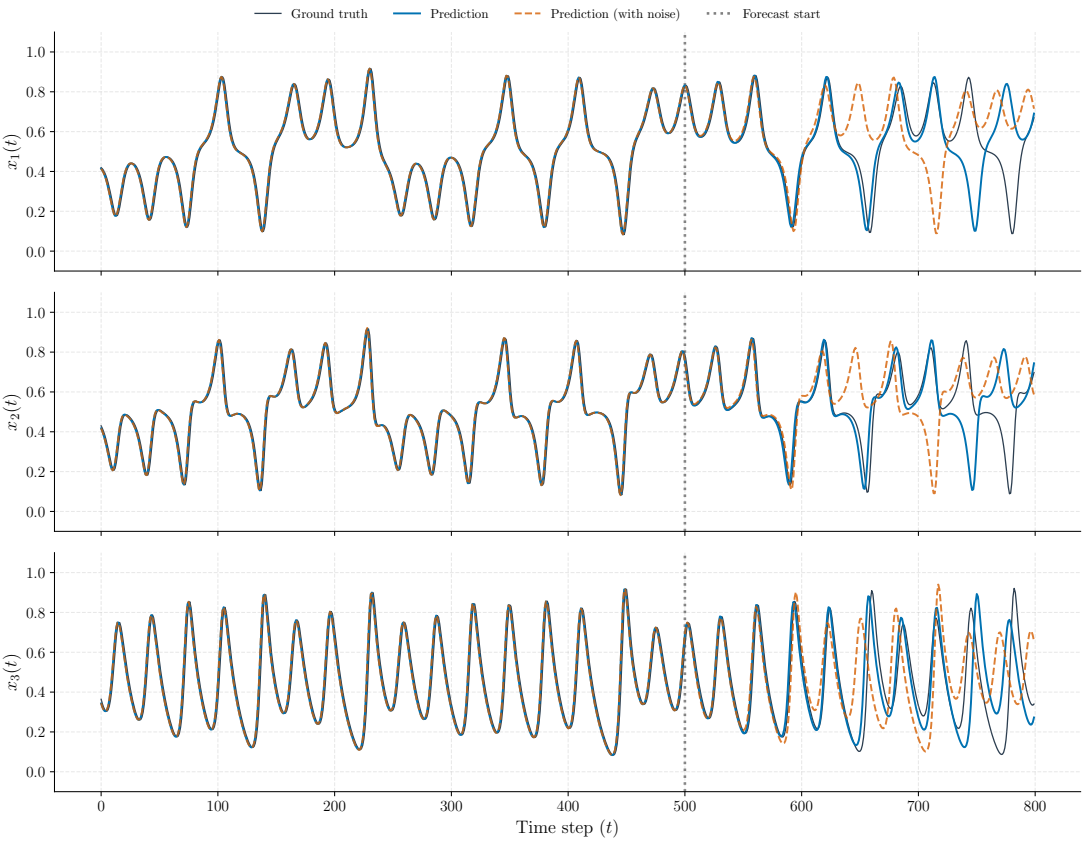


Figure 3.7 / Simulated forecasting result of a Lorenz system. The reservoir consists of $N = 80$ units. The plot shows the test and training phase of the three coordinates for data with and without added noise, as well as the true trajectory of the system.

Conclusion

In this thesis, we presented an experimental implementation of reservoir computing based on photonic quantum memristors, building on the device introduced by Spagnolo et al. [18] and the previous neuromorphic computing framework of Selimović et al. [19]. Our design goes beyond earlier work by introducing an additional feedback mechanism that injects previous reservoir outputs into the device, extending the system's memory capacity and nonlinear dynamics. Additionally, we expanded the system's output space by employing multiple parallel units.

We characterized three different memory dynamics: Reflectivity Memory (RM), Injection Memory (IM), and the combination, Reflectivity and Injection Memory. We demonstrated their performance through two benchmark machine learning tasks. In the STM task, the IM reservoir substantially outperformed the RM reservoir by maintaining a high correlation coefficient over longer delays, a direct consequence of the fading memory introduced through the injection feedback. The RIM reservoir showed a slight reduction in linear memory for this task. In the NARMA task, a mixed RIM-IM reservoir outperformed a purely RM reservoir for lower-order time series. We also identified a gap between experimental and theoretical performance for reservoirs with IM or RIM dynamics, which we attribute primarily to finite measurement statistics. Because the injection feedback propagates measured outputs into subsequent time steps, experimental noise accumulates over the integration window.

These results demonstrate that augmenting the memristive dynamics by injecting previous outputs is an experimentally feasible strategy for enhancing the memory capacity of reservoir computing systems based on photonic quantum memristors. Furthermore, because the framework is modular, the choice of memory dynamics, feedback functions, integration windows, and the number of units can be tuned to match the requirements of a given task, offering greater flexibility than the single-memristor design.

Based on this work, one can further explore the idea of a photonic quantum memristor-based reservoir, e.g., by coupling multiple devices. We also note that this framework does not exploit any non-classical properties of the quantum photonic platform since the input photons are distinguishable. Future work could explore whether the framework can be improved to leverage the exponentially large Hilbert space that motivates quantum reservoir computing in the first place.

References

Back-references to the pages where the publication was cited are given by •.

[1] Daniele Ielmini and H.-S. Philip Wong. In-memory computing with resistive switching devices. *Nature Electronics*, **2018**.
DOI: 10.1038/s41928-018-0092-2 1

[2] Herbert Jaeger. Towards a generalized theory comprising digital, neuromorphic and unconventional computing. *Neuromorphic Computing and Engineering*, **2021**.
DOI: 10.1088/2634-4386/abf151 1

[3] Don Monroe. Neuromorphic computing gets ready for the (really) big time. *Commun. ACM*, **2014**.
DOI: 10.1145/2601069 1

[4] Erik Bollt. On explaining the surprising success of reservoir computing forecaster of chaos? The universal machine learning dynamical system with contrast to VAR and DMD. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, **2021**.
DOI: 10.1063/5.0024890 1

[5] Reservoir computing with memristors. *Nature Electronics*, **2022**.
DOI: 10.1038/s41928-022-00867-y 1

[6] Peng Yao, Huaqiang Wu, Bin Gao, Jianshi Tang, Qingtian Zhang, Wenqiang Zhang, J. Joshua Yang, and He Qian. Fully hardware-implemented memristor convolutional neural network. *Nature*, **2020**.
DOI: 10.1038/s41586-020-1942-4 1, 7

[7] Teresa Serrano-Gotarredona, Timothée Masquelier, Themistoklis Prodromakis, Giacomo Indiveri, and Bernabe Linares-Barranco. STDP and STDP variations with memristors for spiking neuromorphic learning systems. *Frontiers in Neuroscience*, **2013**.
DOI: 10.3389/fnins.2013.00002 1, 7

[8] Chao Du, Fuxi Cai, Mohammed A. Zidan, Wen Ma, Seung Hwan Lee, and Wei D. Lu. Reservoir computing using dynamic memristors for temporal information processing. *Nature Communications*, **2017**.
DOI: 10.1038/s41467-017-02337-y 1, 7

[9] John Moon, Wen Ma, Jong Hoon Shin, Fuxi Cai, Chao Du, Seung Hwan Lee, and Wei D. Lu. Temporal data classification and forecasting using a memristor-based reservoir computing system. *Nature Electronics*, **2019**.
DOI: 10.1038/s41928-019-0313-3 1, 7

[10] 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*, **2021**.
DOI: 10.1002/qute.202100027 1

[11] Giampaolo Pitruzzello. Neuromorphic photonics for efficient computing. *Nature Photonics*, **2025**.
DOI: 10.1038/s41566-025-01654-9 1

[12] Daniel Brunner et al. Roadmap on Neuromorphic Photonics. **2025**.
DOI: 10.48550/arXiv.2501.07917 arXiv: 2501.07917[cs] 1

[13] Guy Van der Sande, Daniel Brunner, and Miguel C. Soriano. Advances in photonic reservoir computing. *Nanophotonics*, **2017**.
DOI: 10.1515/nanoph-2016-0132 1, 6

[14] Iris Paparelle, Johan Henaff, Jorge García-Beni, Émilie Gillet, Daniel Montesinos, Gian Luca Giorgi, Miguel C. Soriano, Roberta Zambrini, and Valentina Parigi. Experimental memory control in continuous-variable optical quantum reservoir computing. *Nature Photonics*, **2026**.
DOI: 10.1038/s41566-026-01880-9 1

[15] Danilo Zia, Luca Innocenti, Giorgio Minati, Salvatore Lorenzo, Alessia Suprano, Rosario Di Bartolo, Nicolò Spagnolo, Taira Giordani, Valeria Cimini, G. Massimo Palma, Alessandro Ferraro, Fabio Sciarrino, and Mauro Paternostro. Quantum reservoir computing for photonic entanglement witnessing. *Science Advances*, **2025**.
DOI: 10.1126/sciadv.ady7987 1

[16] Alessia Suprano, Danilo Zia, Luca Innocenti, Salvatore Lorenzo, Valeria Cimini, Taira Giordani, Ivan Palmisano, Emanuele Polino, Nicolò Spagnolo, Fabio Sciarrino, G. Massimo Palma, Alessandro Ferraro, and Mauro Paternostro. Experimental property-reconstruction in a photonic quantum extreme learning machine. *Physical Review Letters*, **2024**.
DOI: 10.1103/PhysRevLett.132.160802 arXiv: 2308.04543[quant-ph] 1

[17] Emanuele Pelucchi, Giorgos Fagas, Igor Aharonovich, Dirk Englund, Eden Figueroa, Qihuang Gong, Hübel Hannes, Jin Liu, Chao-Yang Lu, Nobuyuki Matsuda, Jian-Wei Pan, Florian Schreck, Fabio Sciarrino, Christine Silberhorn, Jianwei Wang, and Klaus D. Jöns. The potential and global outlook of integrated photonics for quantum technologies. *Nature Reviews Physics*, **2022**.
DOI: 10.1038/s42254-021-00398-z 1, 12

[18] Michele Spagnolo, Joshua Morris, Simone Piacentini, Michael Antesberger, Francesco Massa, Andrea Crespi, Francesco Ceccarelli, Roberto Osellame,

and Philip Walther. Experimental photonic quantum memristor. *Nature Photonics*, **2022**.
DOI: 10.1038/s41566-022-00973-5 2, 7, 8, 13, 18, 29, 37

[19] Mirela Selimović, Iris Agresti, Michał Siemaszko, Joshua Morris, Borivoje Dakić, Riccardo Albiero, Andrea Crespi, Francesco Ceccarelli, Roberto Osellame, Magdalena Stobińska, and Philip Walther. Experimental neuromorphic computing based on quantum memristor. **2025**.
DOI: 10.48550/arXiv.2504.18694 ARXIV: 2504.18694[quant-ph] 2, 17, 19, 29, 37

[20] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and Policy Considerations for Deep Learning in NLP. **2019**.
DOI: 10.48550/arXiv.1906.02243 ARXIV: 1906.02243[cs] 3

[21] Catherine D. Schuman, Shruti R. Kulkarni, Maryam Parsa, J. Parker Mitchell, Prasanna Date, and Bill Kay. Opportunities for neuromorphic computing algorithms and applications. *Nature Computational Science*, **2022**.
DOI: 10.1038/s43588-021-00184-y 3

[22] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, **1986**.
DOI: 10.1038/323533a0 4

[23] H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks, **2001**.
DOI: 10.24406/publica-fhg-291111 5

[24] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-Time Computing Without Stable States: A New Framework for Neural Computation Based on Perturbations. *Neural Computation*, **2002**.
DOI: 10.1162/089976602760407955 5

[25] Gouhei Tanaka, Toshiyuki Yamane, Jean Benoit Héroux, Ryosho Nakane, Naoki Kanazawa, Seiji Takeda, Hidetoshi Numata, Daiju Nakano, and Akira Hirose. Recent advances in physical reservoir computing: A review. *Neural Networks*, **2019**.
DOI: 10.1016/j.neunet.2019.03.005 5, 6

[26] Matteo Cucchi, Steven Abreu, Giuseppe Ciccone, Daniel Brunner, and Hans Kleemann. Hands-on reservoir computing: a tutorial for practical implementation. *Neuromorphic Computing and Engineering*, **2022**.
DOI: 10.1088/2634-4386/ac7db7 5

[27] Peter L. McMahon. The physics of optical computing. *Nature Reviews Physics*, **2023**.
DOI: 10.1038/s42254-023-00645-5 6

[28] L. Chua. Memristor-The missing circuit element. *IEEE Transactions on Circuit Theory*, **1971**.

DOI: 10.1109/TCT.1971.1083337 7

[29] L.O. Chua and Sung Mo Kang. Memristive devices and systems. *Proceedings of the IEEE*, **1976**.
DOI: 10.1109/PROC.1976.10092 7

[30] Dmitri B. Strukov, Gregory S. Snider, Duncan R. Stewart, and R. Stanley Williams. The missing memristor found. *Nature*, **2008**.
DOI: 10.1038/nature06932 7, 8

[31] Massimiliano Di Ventra, Yuriy V. Pershin, and Leon O. Chua. Circuit Elements With Memory: Memristors, Memcapacitors, and Meminductors. *Proceedings of the IEEE*, **2009**.
DOI: 10.1109/JPROC.2009.2021077 7

[32] Yenpo Ho, Garng M. Huang, and Peng Li. Nonvolatile memristor memory: device characteristics and design implications. In: *Proceedings of the 2009 International Conference on Computer-Aided Design*. Association for Computing Machinery, **2009**.
DOI: 10.1145/1687399.1687491 7

[33] Julien Borghetti, Gregory S. Snider, Philip J. Kuekes, J. Joshua Yang, Duncan R. Stewart, and R. Stanley Williams. 'Memristive' switches enable 'stateful' logic operations via material implication. *Nature*, **2010**.
DOI: 10.1038/nature08940 7

[34] Sung Hyun Jo, Ting Chang, Idongesit Ebong, Bhavitavya B. Bhadviya, Pinaki Mazumder, and Wei Lu. Nanoscale Memristor Device as Synapse in Neuromorphic Systems. *Nano Letters*, **2010**.
DOI: 10.1021/nl904092h 7

[35] P. Pfeiffer, I. L. Egusquiza, M. Di Ventra, M. Sanz, and E. Solano. Quantum memristors. *Scientific Reports*, **2016**.
DOI: 10.1038/srep29507 8

[36] J. Salmilehto, F. Deppe, M. Di Ventra, M. Sanz, and E. Solano. Quantum Memristors with Superconducting Circuits. *Scientific Reports*, **2017**.
DOI: 10.1038/srep42044 8

[37] M. Sanz, L. Lamata, and E. Solano. Invited Article: Quantum memristors in quantum photonics. *APL Photonics*, **2018**.
DOI: 10.1063/1.5036596 8

[38] K. M. Davis, K. Miura, N. Sugimoto, and K. Hirao. Writing waveguides in glass with a femtosecond laser. *Optics Letters*, **1996**.
DOI: 10.1364/OL.21.001729 12

[39] G Della Valle, R Osellame, and P Laporta. Micromachining of photonic devices by femtosecond laser pulses. *Journal of Optics A: Pure and Applied Optics*, **2008**.

DOI: 10.1088/1464-4258/11/1/013001 12

[40] Michael Reck, Anton Zeilinger, Herbert J. Bernstein, and Philip Bertani. Experimental realization of any discrete unitary operator. *Physical Review Letters*, **1994**.
DOI: 10.1103/PhysRevLett.73.58 13

[41] William R. Clements, Peter C. Humphreys, Benjamin J. Metcalf, W. Steven Kolthammer, and Ian A. Walmsley. Optimal design for universal multiport interferometers. *Optica*, **2016**.
DOI: 10.1364/OPTICA.3.001460 13, 25

[42] Nicholas C. Harris, Jacques Carolan, Darius Bunandar, Mihika Prabhu, Michael Hochberg, Tom Baehr-Jones, Michael L. Fanto, A. Matthew Smith, Christopher C. Tison, Paul M. Alsing, and Dirk Englund. Linear programmable nanophotonic processors. *Optica*, **2018**.
DOI: 10.1364/OPTICA.5.001623 13

[43] Ciro Pentangelo, Niki Di Giano, Simone Piacentini, Riccardo Arpe, Francesco Ceccarelli, Andrea Crespi, and Roberto Osellame. High-fidelity and polarization-insensitive universal photonic processors fabricated by femtosecond laser writing. *Nanophotonics*, **2024**.
DOI: doi:10.1515/nanoph-2023-0636 25

[44] Giacomo Corrielli, Andrea Crespi, and Roberto Osellame. Femtosecond laser micromachining for integrated quantum photonics. *Nanophotonics*, **2021**.
DOI: 10.1515/nanoph-2021-0419 25

[45] H. Jaeger. Short term memory in echo state networks, **2001**.
DOI: 10.24406/publica-fhg-291107 29

[46] Daniel Fry, Amol Deshmukh, Samuel Yen-Chi Chen, Vladimir Rastunkov, and Vanio Markov. Optimizing quantum noise-induced reservoir computing for nonlinear and chaotic time series prediction. *Scientific Reports*, **2023**.
DOI: 10.1038/s41598-023-45015-4 31

[47] Edward N. Lorenz. Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences*, **1963**.
DOI: 10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2 31