Research Article

Tobias Hülser, Felix Köster, Kathy Lüdge and Lina Jaurigue*

Deriving task specific performance from the information processing capacity of a reservoir computer

https://doi.org/10.1515/nanoph-2022-0415
Received July 19, 2022; accepted September 19, 2022;
published online October 3, 2022

Abstract: In the reservoir computing literature, the information processing capacity is frequently used to characterize the computing capabilities of a reservoir. However, it remains unclear how the information processing capacity connects to the performance on specific tasks. We demonstrate on a set of standard benchmark tasks that the total information processing capacity correlates poorly with task specific performance. Further, we derive an expression for the normalized mean square error of a task as a weighted function of the individual information processing capacities. Mathematically, the derivation requires the task to have the same input distribution as used to calculate the information processing capacities. We test our method on a range of tasks that violate this requirement and find good qualitative agreement between the predicted and the actual errors as long as the task input sequences do not have long autocorrelation times. Our method offers deeper insight into the principles governing reservoir computing performance. It also increases the utility of the evaluation of information processing capacities, which are typically defined on i.i.d. input, even if specific tasks deliver inputs stemming from different distributions. Moreover, it offers the possibility of reducing the experimental cost of optimizing physical reservoirs, such as those implemented in photonic systems.

***Corresponding author: Lina Jaurigue**, Technische Universität Ilmenau, Institute of Physics, Ilmenau, Germany, E-mail: lina.jaurigue@tu-ilmenau.de. https://orcid.org/0000-0002-5104-7111
Tobias Hülser and Felix Köster, Institut für Theoretische Physik, Technische Universität Berlin, Berlin, Germany, E-mail: huelser@fhi.mpg.de (T. Hülser), f.koester@tu-berlin.de (F. Köster). https://orcid.org/0000-0003-3645-9609 (T. Hülser). https://orcid.org/0000-0002-3577-3690 (F. Köster)
Kathy Lüdge, Technische Universität Ilmenau, Institute of Physics, Ilmenau, Germany, E-mail: kathy.luedge@tu-ilmenau.de. https://orcid.org/0000-0002-4831-8910

Keywords: information processing capacity; memory capacity; nonlinear oscillator; reservoir computing.

1 Introduction

Reservoir computing is a versatile, fast-trainable machine learning scheme inspired by the human brain [1, 2]. It avoids difficulties in the training of recurrent neural networks, like the vanishing gradient in time [3] by using a high-dimensional dynamical system, instead of a network with optimized weights, and only training the linear output weights. Recently, it was shown that the universal approximation property holds for a wide range of reservoir computers [4], demonstrating the generality of the reservoir computing approach. Furthermore, because there is no need to train weights within a reservoir, a wide range of hardware implementations have been shown to be feasible [5–9]. Of particular interest are optical implementations, due to the potential speed up in computation times [10, 11].

Generally, two different approaches to reservoir computing exist. First, the so-called echo state approach, which typically uses a reservoir constructed out of randomly connected nonlinear nodes [1] and has been implemented both experimentally [6, 7, 9, 12] and computationally [13–15]. Second, the alternative delay-based approach introduced in [16], where a single dynamical node, e.g., a laser subjected to external time delayed feedback, serves as a time-multiplexed reservoir. This approach has the benefit of relatively simple implementation and uses the dynamic complexity of time delayed systems [17], introducing so-called virtual nodes. There have been various experimental realizations of the delay-based scheme, including optoelectronic [8, 16, 18, 19], optical [20–22] and electrical [23] ones. Potential applications are time-series-predictions [24, 25], fast word recognition [10], signal conditioning [26] and optical communication [27].

Aside from speed and power consumption issues, one wants reservoir computers (RCs), which perform well on various regression or classification tasks. In order to

evaluate the performance of a reservoir computer (RC), there are various benchmark measures, e.g., the very commonly used NARMA10 task. A task-independent measure is the (linear) memory capacity [14], which measures the capability of the reservoir to reconstruct previous inputs. The linear memory capacity was generalized to the information processing capacity by Dambre et al. [28] in 2012, which measures the capability of a reservoir to memorize previous inputs and perform nonlinear calculations on them. The information processing capacities (IPCs) are also referred to as nonlinear memory capacities in the literature. This measure has been used in a number of experimental and theoretical publications [29–32] as a classification of the computing abilities of a reservoir. Besides measuring the IPC in theoretical or experimental frameworks, there are multiple recent advances in calculating and manipulating the IPC. For instance, it was recently possible to calculate the linear part of the IPC (the linear memory capacity) through a linearization of the operating reservoir [33], to systematically manipulate the memorizable inputs via delay-time tuning in a delay-based approach [34] and to manipulate the orders of nonlinear transformation performed via manipulating the input gain [29, 35]. Very recently, the measure of IPC was generalized to systems that are not time invariant [36]. The corresponding measure was introduced as the temporal information processing capacity (TIPC) and has potential relevance for biological systems, as it could be measured in neural cortices [36].

However, despite the extensive research that has been carried out on the IPC, the general connection between the IPC and task-specific performance remains unclear. As an additional challenge, the IPC is typically defined on i.i.d. input; however, tasks often require a different input distribution. Moreover, to the authors' knowledge, there is no known measure that characterizes a reservoir and at the same time strongly correlates to the performance on specific tasks. It is the aim of this paper to work towards addressing these issues by presenting a method to explicitly relate the IPCs to task-specific performances by providing estimates of a RCs performance on specific tasks using its IPCs. The relevance of this work for hardware-implemented reservoirs, such as photonic reservoirs, is that if the IPC is experimentally determined, our approach would allow for an efficient optimization of the performance of the reservoir on a range of tasks.

This work is structured as follows. First, we shortly explain the concept of reservoir computing, using the example of delay-based reservoir computing, and introduce the information processing capacity, as well as the typical benchmarking tasks used in this work. Second, we motivate our new approach by showing that the commonly used sums of IPCs are in general only weakly correlated to task-specific performance. Third, we analytically derive an explicit relation between weighted sums of IPCs and task-specific performance for the case that the task has the same input distribution as used to calculate the IPCs, thereby obtaining an estimate of a tasks error out of its IPCs. We then analyse the validity of our method when the input deviates from the mathematical constraints.

2 Methods

In reservoir computing, a dynamical system, called a reservoir, is fed with input information and the nonlinear response of the reservoir is used to perform a linear approximation of an input-dependent specific task. In the original approach, the reservoir consists of many randomly coupled nonlinear nodes with, e.g., tanh-function dynamics [1]. The input enters into the system via a weighted input matrix. The nonlinear response to the input is then read out via a linear combination of the internal-nodes states. The output weights are trained to minimize the Euclidean distance between the generated output and the target. We refer the reader to the literature [1, 37–40] for more in-depth discussions of the concepts and mathematical foundations. In the alternative delay-based approach [16], instead of multiplexing in space, the system is multiplexed in time via measuring the systems' response at multiple times. In the simplest case, one has a system with one dynamical variable $x(t)$ ("real node") subjected to a linear delayed feedback term [16]. Several expansions to the original concept with more than one real node were discussed in the literature [41–44].

In this manuscript, we use the delay-based approach, which is shortly introduced in the following. For a more thorough discussion, consider, e.g., [16, 39]. However, our analytical derivations (Section 3) make no reference to what system is used as a reservoir. All of the results are expected to hold in the Echo State Network approach as well.

2.1 Time-delayed reservoir computing

In the delay-based approach of reservoir computing [16, 27, 34, 39, 40, 45], a nonlinear system subjected to one or multiple delays is fed by an input series $(u_1, u_2, \dots, u_M)$ and the response of the system is measured multiple times during each input interval. The latter procedure is called time-multiplexing. A corresponding setup is shown in Figure 1. Using a so-called sample-and-hold procedure, each input is fed into the system for an interval T, called the input clock cycle. Inside each input interval, a T-periodic mask function g is applied on the inputs. The mask is typically a stepwise constant function with random step heights. Applying it reduces the linear dependency of the virtual nodes. If the system contains multiple real nonlinear nodes, e.g., a laser network with time-delayed coupling, each node obtains its own input with its own mask function.

The virtual nodes are the states $x(t)$ evaluated at different times $t = (m-1)T + j\theta;\ m \in [1, 2, \dots, M],\ j \in [1, 2, \dots, N_v]$. The separation

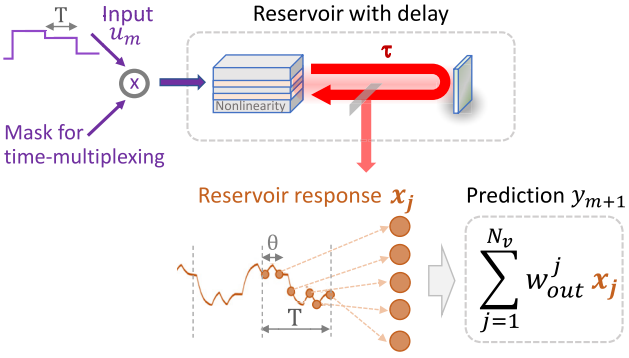

Figure 1: Scheme of a time multiplexed reservoir computer with a nonlinearity (e.g., a laser) subjected to delayed feedback.

between the virtual nodes is denoted as $\theta = \frac{T}{N_v}$, where N_v is the number of virtual nodes. The output y_m is constructed as a linear combination of the virtual nodes:

$$y_m = \sum_{j=1}^{N_v} w_{\text{out}}^j x\big((m-1)T + \theta j\big) + b,$$

with a constant bias b. In the training phase, the reservoir is fed M_{tr} successive inputs and the response of the reservoir is sampled N_v times for each input. These responses are written into a $M_{\text{tr}} \times (N_v + 1)$-dimensional state matrix $\mathbf{S}$, the last column of which is filled with a bias term of one. The linear weights $\mathbf{w}_{\text{out}}$ are trained to minimize a suitable loss function. We choose the Euclidean difference between output $\mathbf{y} = \mathbf{S}\mathbf{w}^{\text{out}}$ and target output $\hat{\mathbf{y}}$ and train via ridge regression. The ridge regression is the solution to

$$\min_{\mathbf{w}_{\text{out}}} \big(\|\mathbf{S}\mathbf{w}_{\text{out}} - \hat{\mathbf{y}}\|_2^2 + \zeta \|\mathbf{w}_{\text{out}}\|_2^2\big). \tag{2}$$

Here, ζ denotes the Tikhonov regularization parameter, introduced to avoid overfitting. Using the Moore–Penrose pseudoinverse, the solution to Eq. (2) is given by

$$\mathbf{w}_{\text{out}} = (\mathbf{S}^T\mathbf{S} + \zeta\mathbf{I})^{-1}\mathbf{S}^T\hat{\mathbf{y}}. \tag{3}$$

To quantify the quality of the prediction, we use the normalized mean-square error (NMSE). It is defined as

$$\text{NMSE} = \frac{\sum_{m=1}^{M} (y_m - \hat{y}_m)^2}{M \cdot \text{var}(\hat{\mathbf{y}})}. \tag{4}$$

here $y_m(\hat{y}_m)$ denotes the mth prediction (target) value, M denotes the number of samples and $1/\text{var}(\hat{\mathbf{y}})$ is a normalization factor. Zero NMSE indicates a perfect prediction, and one corresponds to $y_m = \text{mean}(\hat{\mathbf{y}})$.

When the reservoir contains multiple real nodes $x_1(t), x_2(t), \ldots$, each real node contains its own virtual nodes and the state matrix $\mathbf{S}$ is expanded accordingly. The reservoir used in this manuscript is described in Section 4.

2.2 Information processing capacity

The information processing capacity is a task-independent method of characterizing the performance of a reservoir computer. It quantifies the capability of the reservoir to reconstruct a set of basis functions in the Hilbert space of fading memory functions [28] and is a generalization of the concept of linear memory capacity [14]. In the following, we give a brief introduction to the IPC, using definitions suitable for the later calculations. As mentioned before, the corresponding measures are also referred to as nonlinear memory capacities. Generally, the capacity to reconstruct a certain task $\hat{y}$ is defined as [28]

$$C_{\hat{y}} := 1 - \frac{\text{MSE}(\mathbf{y})}{1/M\|\hat{\mathbf{y}}\|^2}, \tag{5}$$

where $\mathbf{y} = (y_1, y_2, \ldots y_M)^T = \mathbf{S}\mathbf{w}$ is the best linear approximation of a target vector $\hat{\mathbf{y}}$ and MSE denotes the mean square error. Note, that the capacity is evaluated without a regularization, i.e., $\zeta = 0$. In [46], it is shown that the capacity is more conveniently given by

$$C_{\hat{y}} = \frac{\hat{\mathbf{y}}^{\mathbf{T}}\mathbf{y}}{\|\hat{\mathbf{y}}\|^2}. \tag{6}$$

If the capacity equals 1, the approximation is perfect, i.e., $\mathbf{y} = \hat{\mathbf{y}}$. If the capacity is zero, the system is not at all capable of linearly estimating $\mathbf{y}$. In between, a partial reconstruction is possible. The IPCs are the capacities, described by Eq. (6), to reconstruct individual basis functions $\mathbf{P}_n$ on the Hilbert space of fading memory functions, i.e., $\hat{\mathbf{y}} = \mathbf{P}_n$.

In order to construct a basis on the Hilbert space of fading memory functions, first consider the sequence $\{u^{-h}\} = \{u_0, u_{-1}, \ldots, u_{-h}\}$ of a current input u_0 and h previous inputs prior to u_0. Then, one can construct a basis out of finite products of Legendre polynomials $p_d(u_{-i})$ [28], where u_{-i} denotes the input i time steps into the past and $p_d(u_{-i})$ denotes the Legendre polynomial of order d corresponding to input i time steps into the past. The resulting basis functions are $P_n(u^{-\infty})$, with $P_n(u^{-\infty}) = \Pi_i p_{d_n^i}(u_{-i})$. Here, the index $n = \{d^i\} = \{d^1, d^2, \ldots, \}$ serves as a multi-index corresponding to a set of degrees d^i, only finite of which are non-zero. Formally, the memory capacity is defined for an infinitely long input sequence, i.e., $h \to \infty$, but for numerical evaluation, a suitable cut-off has to be chosen (see Supplementary material for details). As input sequence, we use independent and identically distributed (i.i.d.) random numbers in $[-1, 1]$. In [28], it is shown that the so defined collection of P_n indeed form an orthogonal basis on the Hilbert space of fading memory functions. The IPCs to reconstruct a collection of P_n are now given via Eq. (6):

$$IPC_n := C_{\hat{y}=P_n} = \frac{\mathbf{P}_n^T\mathbf{y}}{\|\mathbf{P}_n\|^2}$$
$$P_n(u^{-\infty}) = \Pi_i p_{d_n^i}(u_{-i}) \tag{7}$$

We can define the order $d(P_n)$ of a basis element P_n as the sum of the orders of the individual capacities in the sequence $\{d^i\}$ yielding $d(P_n) = \sum_i d^i$. The degree is used to define linear, quadratic, and higher order capacities as the sum of all capacities corresponding to a certain degree. Using Eq. (7), the IPC of degree d is defined as

$$IPC^d = \sum_{n:\ d(P_n)=d} IPC_n, \tag{8}$$

where the summation takes place over all capacities corresponding to basis elements P_n with order $d(P_n) = d$.

Finally, the total IPC is defined as the sum

$$\text{IPC}^T = \sum_{d>0} IPC^d.$$

In [28], it was shown that the total information processing capacity is always smaller than or equal to the output dimension, i.e., the number of virtual nodes.

2.3 Benchmark tasks

Four standard benchmark tasks are considered in the study: NARMA10 [47], one-step-ahead predictions of the x variable of the Lorenz 63 system [48], nonlinear channel equalization and one-step-ahead prediction of the Mackey–Glass system [49]. Descriptions of all four tasks can be found in Section 1 of the Supplementary material.

2.4 Reservoir – network of ring coupled Stuart–Landau oscillators

As a numerical testing setup, we use a system of delay coupled Stuart–Landau oscillators in a delayed ring topology modeled by

$$\dot{x}_j = (\lambda_j + \eta_j g_j(t)u(t) + i\omega_j + (\gamma_j + i\alpha_j)|x_j|^2)x_j + \kappa_j e^{i\phi_j} x_{j+1}(t-\tau_j) + D_{\text{noise}}\xi(t),$$

where λ_j denotes the pump rate of oscillator j, ω_j the frequency, γ_j the nonlinearity parameter, α_j denotes the re-scaled sheer parameter, κ_j the feedback strength, ϕ_j the feedback phase, τ_j the delay time and $\xi(t)$ models Gaussian white noise with amplitude $D_{\text{noise}} = 10^{-8}$. If the number of oscillators is N, then the ring topology implies $x_{N+1} \equiv x_1$. The input is denoted as $\eta_j g_j(t)u(t)$, where $u(t)$ is the original (unmasked) input, $g_j(t)$ denotes the jth mask function and η_j the input strength. The mask functions were chosen to be piecewise constant random binary. It is important to note that each oscillator j has a distinct mask. Figure 2a illustrates the oscillator setup with two nodes ($j \in \{1, 2\}$) and Figure 2b with one node ($j = 1$). The system is similar to a system of coupled lasers, as the Stuart–Landau oscillator is the generic form of a Hopf bifurcation and behaves like a laser near the lasing threshold [31]. The parameters chosen for the simulations are as in Table 1, unless stated otherwise.

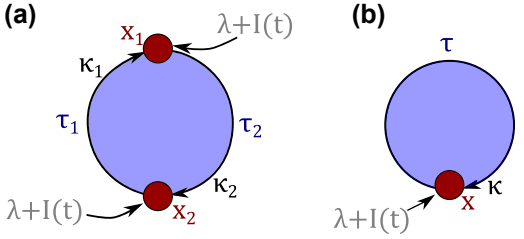

Figure 2: Delay-coupling schemes used for the reservoir: (a) Two oscillators x_j arranged in a delay-coupled ring topology, corresponding to Eq. (9) where each oscillator is unidirectionally coupled to the next oscillator with delays τ_j. Every node is externally driven with pump rate λ_j. The input $I(t) = g_j(t)u(t)$ is fed in via the pumping. (b) Same setup as in (a), but with one oscillator subjected to delayed feedback.

Table 1: Default parameter values for numerical simulations.

Parameter	Description	Value
τ	Feedback delay time	425
T	Clock cycle	200
Λ	Pump rate	0.01
κ	Feedback strength	0.18
Φ	Feedback phase	0
η	Input strength	0.06
α	Sheer parameter	0
ω	Frequency	0
γ	Nonlinearity	−0.1
N_v	Virtual nodes per oscillator	100

3 Connection between information processing capacity and task-specific performance

In the literature, in the context of the IPC, mostly measures consisting of sums of capacities of a certain order are investigated (IPC^d in Eq. (8)). However, the summed IPC^ds are in general only weakly correlated to the error of a specific task and are, therefore, a poor measure for predicting task-specific performance. As an example, we consider a system of two delay-coupled Stuart–Landau oscillators in a ring coupling configuration (see Section 2.4 for details). In this setup, we find that the ring delays have a strong influence on the IPCs and task performance, but the parameter dependencies of these quantities vary strongly.

In Figure 3, a scan over the two ring delays τ_1, τ_2 is performed and the linear, quadratic, cubic and total IPC are depicted. Figure 4 shows the corresponding plots for the performance on various regression benchmark tasks. For all tasks, a strong dependence on the delay times is evident and various resonance lines can be found, as explained in [34]. However, important for the predictive power of the summed IPC^ds is that the delay dependence is different for each of the tasks.

To quantify the predictive power of the summed IPC^ds, we calculate a Pearson correlation coefficient between the depicted IPC^ds and the NMSE of the benchmark tasks. A negative value close to −1 indicates a low NMSE at values of high IPC, whereas a high positive value indicates a counterproductive effect between the memory capacity and the task NMSE. Zero indicates no linear statistical correlation between both quantities. As one can see in Table 2, the obtained correlation coefficients are generally low. Furthermore, some values are positive, meaning that high values of the corresponding sum of IPCs tend to

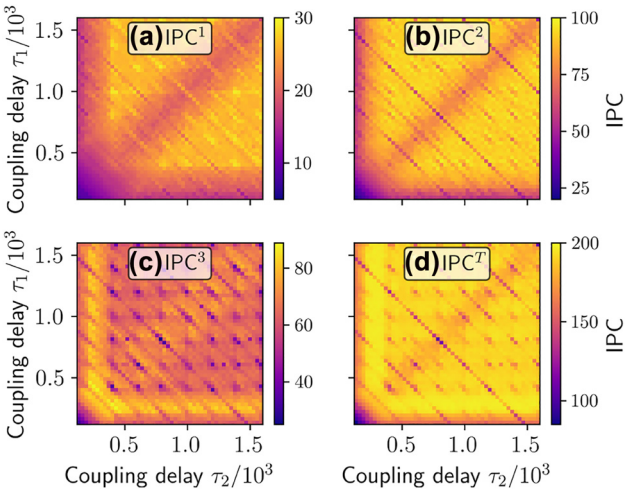

Figure 3: Information processing capacities: (a) IPC^1, (b) IPC^2, (c) IPC^3 and (d) IPC^T, as defined in Eq. (8), plotted colour coded as a function of τ_1 and τ_2 in the ring-coupled system Eq. (9) with two oscillators. Pearson correlation coefficients between the IPCs and the task NMSEs shown in Figure 4 are given in Table 2.

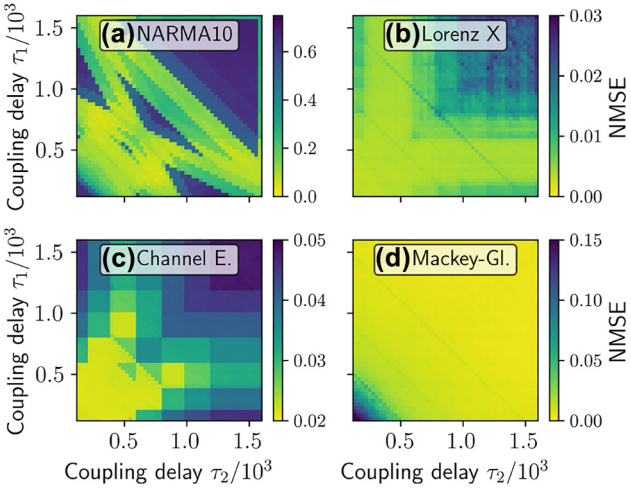

Figure 4: Performance on different tasks: (a) NARMA10, (b) Lorenz x, (c) channel equalization and (d) Mackey–Glass task errors (NMSE) plotted colour coded as a function of τ_1 and τ_2 in the ring-coupled system Eq. (9) with two oscillators. Pearson correlation coefficients between the task NMSEs and the IPCs shown in Figure 3 are given in Table 2.

correspond to low task performance. These results demonstrate that choosing reservoir hyperparameters based on summed capacities, such as the total IPC, is not helpful for optimizing task-specific performance. The underlying reason is that most tasks have specific IPC requirements and do not need all available ones. Different past inputs are in general not equally relevant for the performance of a specific tasks. A simple sum of all IPCs is, therefore, a very limited, yet often considered, measure.

Table 2: Pearson correlation coefficients between the information processing capacities (IPC) (plotted in Figure 3) and the benchmark task NMSE errors (plotted in Figure 4).

Task	IPC^1	IPC^2	IPC^3	Total IPC
Lorenz X	0.18	0.12	−0.20	−0.07
NARMA10	0.07	0.03	−0.13	−0.08
Channel equal.	0.14	0.08	−0.12	−0.02
Mackey–Glass	−0.62	−0.63	−0.02	−0.53

3.1 Analytic derivation of an explicit relationship between IPCs and NMSE

The appeal of the information processing capacity is that it quantifies the computational capabilities of a reservoir in a task-independent manner. In this section, we introduce a method of connecting the IPCs with task-specific performance. The motivation for this is that once the IPCs of a reservoir are determined, these can be used to predict the performance of the reservoir on a multitude of tasks. This can reduce the hyperparameter optimization, which would otherwise need to be carried out individually for each task.

To generate a more task relevant measure than simple sums of information processing capacities, we build a weighted sum of IPCs instead of the unweighted sums usually considered. The corresponding ansatz is

$$IPC_{\text{task}} = 1 - \sum_n c_n \cdot IPC_n, \tag{10}$$

where $c_n \in \mathbb{R}$ denotes the weight of the nth capacity and $n \in \mathbb{N}$ serves as a (multi-)index, indexing all possible capacities, as in Section 2.2. The above approach is justified analytically in the following. The values of c_n, that are derived, give the information, which capacities are relevant for a given task. Note that until now, there only exist some approaches to directly evaluate relevant IPCs of the NARMA10 task [36] or indirectly evaluate the previous inputs that are relevant to specific tasks [50]. However, a direct connection between IPC and RC performance was not described yet.

The exact calculations are shown in the Supplementary material, here only the most important steps are given. Consider the capacity to compute an arbitrary task (Eq. (6)) [46]:

$$C_{\hat{\mathbf{y}}_n} = \frac{\hat{\mathbf{y}}_n^T \mathbf{y}}{\|\hat{\mathbf{y}}\|^2} \tag{11}$$

The idea is to develop the task $\hat{y}$ into a basis $\{P_n\}$ on the Hilbert space of fading memory functions and simultaneously develop the prediction into a series of predictions of the basis elements P_n with identical development coefficients. Note, if one instead directly develops the prediction into a basis on the Hilbert space of fading memory functions, one obtains a direct relationship between reservoir computing and nonlinear vector autoregressive models (see Supplementary material).

It is known that any time invariant dynamical system with fading memory can be approximated by a finite discrete Volterra series [51]. The discrete Volterra series is a series of products of monomials of previous inputs. We, therefore, assume that the task $\hat{y}$ is time invariant and has fading memory property. For the prediction, this is ensured by the fading memory property of reservoir computing. The Volterra series is non-orthogonal and, therefore, not suitable for the purpose of this paper. However, it is possible to choose an orthogonal basis dependent on the input distribution, as it is done with products of Legendre polynomial in the definition of the IPC Eq. (6). This connects to a technique called polynomial chaos expansion, where a random variable is expanded in a polynomial basis orthogonal on a possibly arbitrary input distribution [52]. Polynomial chaos expansions have, e.g., been successfully used in a context of uncertainty quantification [52, 53], they are not limited to fading memory systems and converge under general conditions [52, 54].

We can, therefore, expand the target function $\hat{y}$ as

$$\hat{y} = \sum_n a_n P_n + \xi, \tag{12}$$

where P_n are the basis functions, a_n the corresponding coefficients, and ξ is a noise term independent of the input. We set $P_0 = 1$ and, therefore, a_0 is the constant bias term. The choice of (useful) basis functions is dependent on the probability distribution of the input series. In the context of IPC, often identically drawn independent random numbers in $[-1, 1]$ are chosen for the input. In this case, an orthogonal set of basis functions are products of Legendre polynomials formed by different past input steps (see Section 2.2). We use this basis in Section 4, but the analytical calculations are generalizable and do not refer to a specific basis. We emphasize that other orthogonal bases could be used as well, see e.g., [36], for use of different bases in an IPC context.

While we developed the target into the above-mentioned series (Eq. (12)) and obtain the $a_n's$, we can develop the prediction into predictions of the corresponding basis functions via

$$y = \sum_n a_n p_n + h(\xi), \tag{13}$$

where p_n is the prediction of basis function P_n within a given input series and $h(\xi)$ is a noise term, stemming from the input-independent part of the target series. It will be neglected in the following. Note that the development coefficients a_n are the same as in the series expansion Eq. (12) (see Supplementary material).

We further assume that the input series for the task comes from the same probability distribution as in the definition of the IPC, for example i.d.d. random numbers in $[-1, 1]$. If one then puts Eqs. (12) and (13) into Eq. (11), we obtain a formula for the NMSE for a given task:

$$NMSE_{\text{pred}} \approx 1 - \frac{\sum_n a_n^2 \|\mathbf{P}_n\|^2 IPC_n}{N\text{var}(\hat{\mathbf{y}})}. \tag{14}$$

Here, a_n are the coefficients of the series expansion of the task (Eq. (12)), $\|\mathbf{P}_n\|^2$ the squared norms of the basis polynomials and $N\text{var}(\hat{\mathbf{y}})$ the variance of the target $\hat{y}$ multiplied with the number of training samples N. IPC_n denotes the nth information processing capacity corresponding to basis polynomial P_n. For simplicity, it is assumed that the mean of the target equals zero; however, non-zero mean values are taken care of via the bias term. Eq. (14) is, therefore, to be evaluated without the constant bias term, i.e., $a_0 = 0$.

In order for Eq. (14) to be used, the expansion coefficients a_n must be determined. To evaluate the coefficients a_n, we use a linear regression approach, constructing a linear model out of the basis elements $P_n(u^{-\infty})$. The corresponding model is

$$\mathbf{y} = \sum_n \tilde{a}_n \mathbf{P}_n(u^{-\infty}). \tag{15}$$

Where $\mathbf{y}$ denotes the prediction of the target vector $\hat{\mathbf{y}}$. The coefficients $\tilde{a}_n$ are evaluated via ridge regression and serve as an estimate for the searched coefficients a_n. Details can be found in the Supplementary material. This is a nonlinear vector regressive model and similar to a recent approach to predict the Lorenz task [55]. Note, that if the task is an auto-prediction task, e.g., in the Lorenz X task example, the model is a nonlinear vector autoregressive model (NVAR). This approach works well if the regressors $P_n(u^{-\infty})$ are weakly correlated, i.e., the auto-correlation of the inputs is low, otherwise the model suffers from multicollinearity and the a_n no longer uniquely identifiable. Details on the model can be found in the Supplementary material. It is to be noted that generally, the number of coefficients to be evaluated grows exponentially with increasing order of basis functions and steps into the past considered.

Suitable cut-offs, therefore, have to be chosen to limit computational cost and avoid overfitting issues (details are given in Section 4 and in the Supplementary material).

Our approach predicts the NMSE corresponding to a task $\hat{\mathbf{y}}$ out of the *IPC*s of the RC. It is important to note that the prediction does not need any knowledge about the RC system other than its IPCs. It is, therefore, not limited to a specific reservoir computing scheme. The main assumption of our derivation of Eq. (14) is that the input for a specific task comes from the same probability distribution as in the evaluation of the IPCs. This assumption does not hold for most real tasks. Additionally, the method used to evaluate the coefficients a_n is best suited for inputs, which are only weakly correlated. Therefore, in Section 4, we test the extent to which Eq. (14) can provide accurate error estimates when these assumptions are not fulfilled.

4 Numerical evaluation

In this section, we check the validity of Eq. (14) derived in the previous section.

4.1 Impact of deviating from an i.i.d. input distribution

In order to derive the explicit expression relating the NMSE of a task to the IPCs, the assumption was made that the task input distribution equals the IPC input distribution. It is possible to evaluate IPCs for arbitrary input distributions [36] and, therefore, matching arbitrary tasks. However, the IPC input distribution has to be predefined when measuring the IPCs. Therefore, we investigate how well Eq. (14) performs under systematic deviations of this assumption. To do this, we require a task where the input distribution can gradually be altered from i.i.d. random inputs in $[-1, 1]$. We use the NARMA10 task (given in the Supplementary material). It can be operated with non i.d.d. random inputs, linearly transformed to be in $[-1, 1]$, without having to change the equation that defines the target series.

There are two main factors that are to be considered. First, it is assumed that different inputs $u_i, u_j, i \neq j$ are not correlated, and therefore, the autocorrelation function of the input series $\{u_0, u_1, \dots\}$ is zero. This is a severe restriction, since most realistic input series will have an auto-correlation function that is non-zero. Second, it is assumed that the inputs u_i are drawn from a uniform probability distribution $p(u_i)$. Deviating from a uniform distribution, even in the absence of a non-zero auto-correlation, could also degrade the predictive power of Eq. (14). We analyse both effects with input defined via the following equations:

$$u_i = \rho u_{i-1} + (\rho - 1)\chi_i, \tag{16}$$

where ρ denotes a correlation parameter and χ_i denotes independently drawn random numbers from a probability distribution $p(\chi)$. With this construction, $\rho = 0$ corresponds to uncorrelated fully independent u_i with zero auto-correlation and $\rho = 1$ corresponds to $u_i = u_{i-1}$ and thereby the auto-correlation equals one for every time lag. We can feed the above defined input series u_i into the iterative formula for the NARMA10 task.

To test the predictive power of Eq. (14), we consider the system of one Stuart–Landau oscillator with delayed feedback (see Figure 2b). While fixing the input clock cycle, we scan over the delay time for 50 linearly distributed delays, in a range where the NARMA10 NMSE varies strongly ($\tau \in [20, 1600]$). This means, we investigate 50 different reservoirs and evaluate the performance of our new method for the NARMA task with different input distributions. We evaluate the Pearson correlation coefficient as well as the mean square error between the predicted NMSE from Eq. (14) and the simulated NMSE.

Figure 5a shows these quantities in a scan over the correlation parameter ρ for different distributions $p(\chi)$ (different symbols). The line without symbols is the uniform distribution in $[-1, 1]$, where $\rho = 0$ corresponds to i.i.d. inputs and, therefore, the assumptions made in the derivation of Eq. (14) hold (a correlation parameter of $\rho = 1$ corresponds to identical inputs and, therefore, an infinite auto-correlation time). It can be seen that for low ρ, Eq. (14) works well. For increasing ρ, the mean squared deviation between predicted and measured NMSE becomes increasingly large and the Pearson correlation coefficient decreases. The lines with symbols in Figure 5a show the results for binary, gamma and Gaussian probability distributions $p(\chi)$. We chose a binary distribution between 0 and 0.42, a gamma distribution with shape 1.5 and scale 0.1, and a Gaussian distribution with zero mean and standard deviation 0.3. Note, after the target series is created, the input is linearly normalized such that it approximately fits in the interval $[-1, 1]$, see Supplementary material for further details. Even though these distributions strongly violate the assumption of uniform input, the method works well if ρ does not become too high. As a rule of thumb, if one sets a threshold of a Pearson correlation coefficient of at least 0.5, this roughly corresponds to values of $\rho \lessapprox 0.8$ (which corresponds to an auto-correlation time $t_a \lessapprox 4.9$, using the definition given below in Eq. (17)).

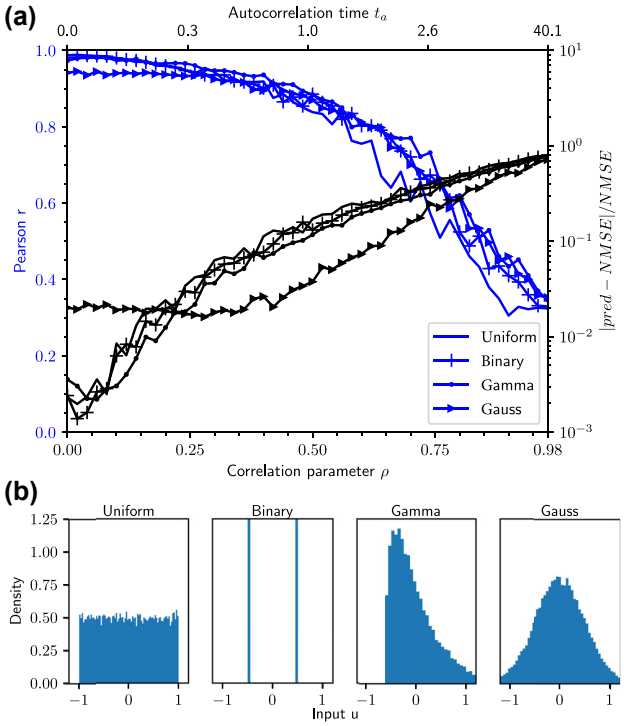

Figure 5: Impact of correlated inputs for performance predictions: (a) Blue lines: Pearson correlation coefficient r between predicted and measured NMSE as a function of the input correlation parameter (see Eq. (16)) for four different random distributions $p(\chi)$ (different line styles). r is determined from a scan over 50 distinct delay times $\tau \in [20, 1600]$ in the setup corresponding to Figure 2b. Black lines: Mean square error between predicted and measured NMSE. (b) Histograms of uniform, binary, gamma and Gaussian distribution chosen in (a).

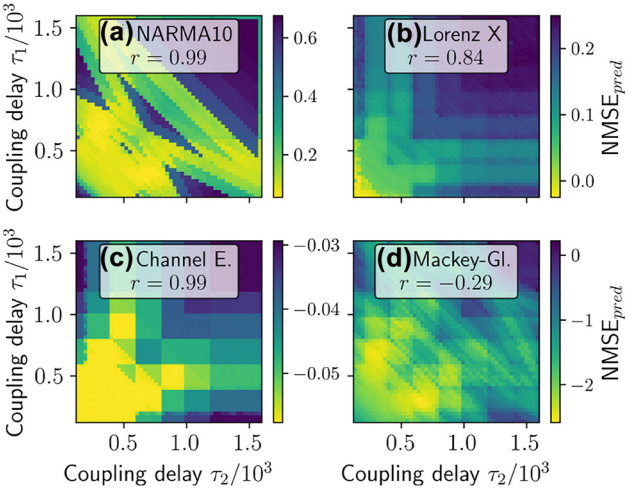

Figure 6: Predicted performance (NMSE) in parameter space for different tasks: (a) NARMA10, (b) Lorenz X, (c) Channel equalization and (d) Mackey–Glass tasks as a scan over the delays in the ring-coupled system of Figure 2a as determined using Eq. (14). (a–c) Insets give Pearson correlation coefficients r to the computed errors in Figure 4.

The results indicate that Eq. (14) is able to qualitatively predict a reservoir computer's performance on a task, even if the input distribution is not uniform and uncorrelated. For high auto-correlation times, however, Eq. (14) with weights evaluated with the multilinear model given by Eq. (15) performs insufficiently.

4.2 Numerical results for benchmark tasks

In this section, we test our method on a set of benchmarking tasks. Here, we once again use the system of two delay-coupled Stuart–Landau oscillators in a ring coupling configuration (see Section 2.4 for details) as the test reservoir. In this setup, we obtain complex patterns for the NMSE of the benchmark tasks, if we scan over the two delay times τ_1, τ_2 (Figure 4a–d). In Figure 6a–d, we apply our new method (Eq. (14)) for the NARMA10, Lorenz X, channel equalization and Mackey–Glass tasks. The prediction of the NARMA10 NMSE nearly perfectly reproduces the structure of the simulated NARMA10 NMSE without any knowledge of the system other than the IPCs, which is to be expected in this case as the assumptions made to derive Eq. (14) hold for the NARMA10 input. The Pearson correlation coefficient between the predicted and simulations NMSE is depicted in the respective insets of Figure 6. It reaches a value of 0.99 for NARMA10. In this case, we evaluated capacities up to 2nd order and up to 20 previous inputs. See Supplementary material for a discussion of the cut-off orders.

For the other benchmark tasks evaluated in Figure 6b–d, the assumption of i.i.d. distributed input does not hold. That is first because these tasks have non-zero auto-correlation time (Figure 7a) and second, the distribution of the inputs strongly deviates from an uniform input distribution (Figure 7b). However, if we apply Eq. (14) for the Lorenz task, the structure is still well reproduced (Figure 6b), with a high correlation coefficient of 0.84. For the Lorenz task, we evaluated capacities up to 5th order and up to 5 steps into the past. For the channel equalization task (Figure 6c), the absolute values of the predicted NMSE are inaccurate, in this case even negative. However, the correlation between the predicted and true NMSE is 0.99, a near maximum value. Therefore, for parameter optimization purposes, the predicted NMSE is usable. For the channel equalization task, we evaluated capacities up to third order and up to 10 steps into the past. Both, the Lorenz and the channel equalization task indicate that Eq. (14) is useful even in the case the assumption of i.i.d. input is not fulfilled. These results show that evaluating

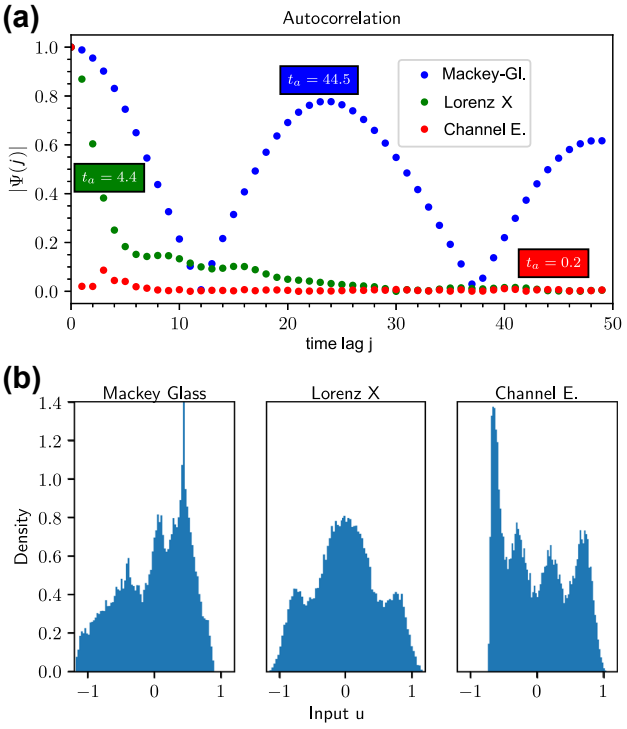

Figure 7: Properties of the input fed into the reservoir: (a) Modulus of the auto-correlation function $\Psi(j)$ for Lorenz X, Mackey–Glass and channel equalization task, respectively, as a function of the time lag j (colored boxes give the auto-correlation times t_a according to Eq. (17)). (b) Histograms of the input distributions for three different tasks.

IPCs for an i.i.d. input distribution still gives information on the performance of a RC fed with inputs from a different distribution.

If we consider the Mackey–Glass task (Figure 6d), the NMSE is poorly approximated using Eq. (14) with coefficients a_n corresponding to Eq. (15). This is because, in comparison with the other tasks, the Mackey–Glass input shows a high input auto-correlation (Figure 7a), which leads to an ambiguous development of the coefficients a_n, as well as a larger deviation from the i.i.d. input assumption. For the Mackey–Glass task, we evaluated capacities up to 20 previous inputs and up to 3 orders.

To compare the timescales over which the various inputs are auto-correlated, we calculate the auto-correlation time t_a of the input series according to

$$t_a = \sum_{j=1}^{j_{\max}} |\psi(j)|, \tag{17}$$

where $\psi(j)$ is the value of the normalized auto-correlation function at time lag j (see Supplementary material for details). To avoid summing up noise, we only evaluated auto-correlation values greater than a threshold of 0.02 and evaluated $j_{\max} = 100$ time lags. For the NARMA10 task, we obtain a value of $t_a \approx 0$ (i.i.d input), for the Lorenz task $t_a \approx 4.4$, for the channel task $t_a \approx 0.2$ and for the Mackey–Glass task a value of $t_a \approx 44.5$. This indicates a great difference in auto-correlation times t_a between the Mackey–Glass task and the other considered tasks. In Section 4.1, we systematically tuned the auto-correlation of a NAMRA10 input in order to investigate how deviations of i.i.d. input harm the performance of Eq. (14). There, the Pearson correlation between the estimated and the actual NMSE was above 0.5 for ρ up to 0.8, which corresponded to an auto-correlation time $t_a \lessapprox 4.9$. This auto-correlation time is similar to the Lorenz task example.

5 Conclusions

We have investigated the relationship between the commonly used information processing capacity and the performance on regression tasks, e.g., time series prediction tasks, in reservoir computing. To characterize the computational capabilities of a reservoir, the total information processing capacity or the IPCs summed over each order are commonly considered. However, these simple sums of IPCs are an insufficient measure for predicting task-specific performances. We analytically derived an explicit relation between weighted sums of individual information processing capacities and the expected computing error for specific tasks. We have further tested the extent to which the expression for the NMSE that we have derived deviates from the true NMSE when the problem deviates from the strict mathematical assumptions, i.e., the input distribution of the task varies from that used to characterize the IPC. We found high correlation between the predicted and the actual NMSE for NARMA10, channel equalization and Lorenz time series prediction tasks. For the Mackey–Glass task, we found poor agreement. Our results indicate that the auto-correlation time of the input sequence is crucial in determining the accuracy of the trends of the predicted NMSE, with longer auto-correlation times reducing the applicability of our proposed method.

The above derived approach can be exploited to reduce the experimental cost of optimizing a reservoir computing setup, such as a photonic reservoir, for multiple tasks. Before, in order to obtain measures of the performance for different tasks, the input series for each task had to be fed into the reservoir to measure the responses. With Eq. (14), however, one obtains a direct link between the performance of a task and the individual IPCs of a reservoir computer, and the IPCs can be measured via only one sufficiently large input series and post-processing. By

establishing a link between the IPC and the performance for specific tasks, we have demonstrated the utility of the IPC as a means of characterizing physically implemented reservoir computing setups.

Author contributions: All the authors have accepted responsibility for the entire content of this submitted manuscript and approved submission.
Research funding: This research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) in the framework of the SFB910 (project B9) and Projektnummer 445183921 (project LU 1729/3-1).
Conflict of interest statement: The authors declare no conflicts of interest regarding this article.

References

[1] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks," GMD – German National Research Institute for Computer Science, GMD Rep., vol. 148, 2001.
[2] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: a new framework for neural computation based on perturbations," *Neural Comput.*, vol. 14, pp. 2531–2560, 2002.
[3] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *Int. J. Uncertain. Fuzziness Knowledge based Syst.*, vol. 6, pp. 107–115, 1998.
[4] L. Gonon and J. P. Ortega, "Reservoir computing universality with stochastic inputs," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 1, pp. 100–112, 2020.
[5] P. Antonik, F. Duport, M. Hermans, A. Smerieri, M. Haelterman, and S. Massar, "Online training of an opto-electronic reservoir computer applied to real-time channel equalization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 11, pp. 2686–2698, 2016.
[6] K. Dockendorf, I. Park, P. He, J. C. Principe, and T. B. DeMarse, "Liquid state machines and cultured cortical networks: the separation property," *Biosystems*, vol. 95, no. 2, pp. 90–97, 2009.
[7] C. Fernando and S. Sojakka, "Pattern recognition in a bucket," *Advances in Artificial Life*, pp. 588–597, 2003, https://doi.org/10.1007/978-3-540-39432-7_63.
[8] L. Larger, M. C. Soriano, D. Brunner, et al., "Photonic information processing beyond turing: an optoelectronic implementation of reservoir computing," *Opt. Express*, vol. 20, no. 3, pp. 3241–3249, 2012.
[9] K. Vandoorne, P. Mechet, T. Van Vaerenbergh, et al., "Experimental demonstration of reservoir computing on a silicon photonics chip," *Nat. Commun.*, vol. 5, p. 3541, 2014.
[10] L. Larger, A. Baylón-Fuentes, R. Martinenghi, V. S. Udaltsov, Y. K. Chembo, and M. Jacquot, "High-speed photonic reservoir computing using a time-delay-based architecture: million words per second classification," *Phys. Rev. X*, vol. 7, p. 011015, 2017.
[11] M. Nakajima, K. Tanaka, and T. Hashimoto, "Scalable reservoir computing on coherent linear photonic processor," *Commun. Phys.*, vol. 4, p. 20, 2021.
[12] S. Sackesyn, C. Ma, J. Dambre, and P. Bienstman, "Experimental realization of integrated photonic reservoir computing for nonlinear fiber distortion compensation," *Opt. Express*, vol. 29, no. 20, pp. 30991–30997, 2021.
[13] M. Bauduin, A. Smerieri, S. Massar, and F. Horlin, "Equalization of the non-linear satellite communication channel with an echo state network," in *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, 2015.
[14] H. Jaeger, "Short term memory in echo state networks," GMD – Forschungszentrum Informationstechnik GmbH, GMD Rep., vol. 152, 2002.
[15] M. Sorokina, S. Sergeyev, and S. Turitsyn, "Fiber echo state network analogue for high-bandwidth dual-quadrature signal processing," *Opt. Express*, vol. 27, pp. 2387–2395, 2019.
[16] L. Appeltant, M. C. Soriano, G. Van der Sande, et al., "Information processing using a single dynamical node as complex system," *Nat. Commun.*, vol. 2, p. 468, 2011.
[17] J. D. Hart, L. Larger, T. E. Murphy, and R. Roy, "Delayed dynamical systems: networks, chimeras and reservoir computing," *Philos. Trans. R. Soc. A*, vol. 377, no. 2153, p. 20180123, 2019.
[18] Y. Chen, L. Yi, J. Ke, et al., "Reservoir computing system with double optoelectronic feedback loops," *Opt. Express*, vol. 27, no. 20, pp. 27431–27440, 2019.
[19] Y. Paquot, F. Duport, A. Smerieri, et al., "Optoelectronic reservoir computing," *Sci. Rep.*, vol. 2, p. 287, 2012. https://doi.org/10.1038/srep00287.
[20] D. Brunner, M. C. Soriano, C. R. Mirasso, and I. Fischer, "Parallel photonic information processing at gigabyte per second data rates using transient states," *Nat. Commun.*, vol. 4, p. 1364, 2013.
[21] Y. S. Hou, G. Q. Xia, W. Y. Yang, et al., "Prediction performance of reservoir computing system based on a semiconductor laser subject to double optical feedback and optical injection," *Opt. Express*, vol. 26, no. 8, pp. 10211–10219, 2018.
[22] Q. Vinckier, F. Duport, A. Smerieri, et al., "High-performance photonic reservoir computer based on a coherently driven passive cavity," *Optica*, vol. 2, no. 5, pp. 438–446, 2015.
[23] Z. Q. Zhong, D. Chang, W. Jin, et al., "Intermittent dynamical state switching in discrete-mode semiconductor lasers subject to optical feedback," *Photon. Res.*, vol. 9, no. 7, pp. 1336–1342, 2021.
[24] J. Bueno, D. Brunner, M. C. Soriano, and I. Fischer, "Conditions for reservoir computing performance using semiconductor lasers with delayed optical feedback," *Opt. Express*, vol. 25, no. 3, pp. 2401–2412, 2017.
[25] Y. Kuriki, J. Nakayama, K. Takano, and A. Uchida, "Impact of input mask signals on delay-based photonic reservoir computing with semiconductor lasers," *Opt. Express*, vol. 26, no. 5, pp. 5777–5788, 2018.

[26] A. Argyris, J. Cantero, M. Galletero, et al., "Comparison of photonic reservoir computing systems for fiber transmission equalization," *IEEE J. Sel. Top. Quantum Electron.*, vol. 26, no. 1, p. 5100309, 2020.
[27] A. Argyris, "Photonic neuromorphic technologies in optical communications," *Nanophotonics*, vol. 11, no. 5, pp. 897–916, 2022.
[28] J. Dambre, D. Verstraeten, B. Schrauwen, and S. Massar, "Information processing capacity of dynamical systems," *Sci. Rep.*, vol. 2, p. 514, 2012.
[29] M. Goldmann, C. R. Mirasso, I. Fischer, and M. C. Soriano, "Exploiting transient dynamics of a time-multiplexed reservoir to boost the system performance," in *2021 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2021, pp. 1–8. Available at: https://ieeexplore.ieee.org/document/9534333.
[30] K. Harkhoe and G. Van der Sande, "Task-independent computational abilities of semiconductor lasers with delayed optical feedback for reservoir computing," *Photonics*, vol. 6, no. 4, p. 124, 2019.
[31] F. Köster, D. Ehlert, and K. Lüdge, "Limitations of the recall capabilities in delay based reservoir computing systems," *Cogn. Comput.*, vol. 2020, pp. 1–8, 2020.
[32] S. Ortín and L. Pesquera, "Delay-based reservoir computing: tackling performance degradation due to system response time," *Opt. Lett.*, vol. 45, no. 4, pp. 905–908, 2020.
[33] F. Köster, S. Yanchuk, and K. Lüdge, "Master memory function for delay-based reservoir computers with single-variable dynamics," 2021 [Online]. Available at: https://arxiv.org/abs/2108.12643.
[34] T. Hülser, F. Köster, L. C. Jaurigue, and K. Lüdge, "Role of delay-times in delay-based photonic reservoir computing," *Opt. Mater. Express*, vol. 12, no. 3, pp. 1214–1231, 2022.
[35] B. Vettelschoss, A. Röhm, and M. C. Soriano, "Information processing capacity of a single-node reservoir computer: an experimental evaluation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 6, pp. 2714–2725, 2021.
[36] T. Kubota, H. Takahashi, and K. Nakajima, "Unifying framework for information processing in stochastically driven dynamical systems," *Phys. Rev. Res.*, vol. 3, no. 4, p. 043135, 2021.
[37] D. Brunner, B. Penkovsky, B. A. Marquez, M. Jacquot, I. Fischer, and L. Larger, "Tutorial: photonic neural networks in delay systems," *J. Appl. Phys.*, vol. 124, no. 15, p. 152004, 2018.
[38] M. Lukosevicius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Comput. Sci. Rev.*, vol. 3, no. 3, pp. 127–149, 2009.
[39] G. Tanaka, T. Yamane, J. B. Héroux, et al., "Recent advances in physical reservoir computing: a review," *Neural Netw.*, vol. 115, pp. 100–123, 2019.
[40] G. Van der Sande, D. Brunner, and M. C. Soriano, "Advances in photonic reservoir computing," *Nanophotonics*, vol. 6, no. 3, p. 561, 2017.
[41] M. Goldmann, F. Köster, K. Lüdge, and S. Yanchuk, "Deep time-delay reservoir computing: dynamics and memory capacity," *Chaos*, vol. 30, no. 9, p. 093124, 2020.
[42] S. Ortín and L. Pesquera, "Reservoir computing with an ensemble of time-delay reservoirs," *Cogn. Comput.*, vol. 9, no. 3, pp. 327–336, 2017.
[43] A. Röhm and K. Lüdge, "Multiplexed networks: reservoir computing with virtual and real nodes," *J. Phys. Commun.*, vol. 2, p. 085007, 2018.
[44] C. Sugano, K. Kanno, and A. Uchida, "Reservoir computing using multiple lasers with feedback on a photonic integrated circuit," *IEEE J. Sel. Top. Quantum Electron.*, vol. 26, no. 1, p. 1500409, 2020.
[45] Y. K. Chembo, "Machine learning based on reservoir computing with time-delayed optoelectronic and photonic systems," *Chaos*, vol. 30, no. 1, p. 013111, 2020.
[46] F. Köster, S. Yanchuk, and K. Lüdge, "Insight into delay based reservoir computing via eigenvalue analysis," *J. Phys. Photonics*, vol. 3, no. 2, p. 024011, 2021.
[47] A. F. Atiya and A. G. Parlos, "New results on recurrent network training: unifying the algorithms and accelerating convergence," *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 697–709, 2000.
[48] E. N. Lorenz, "Deterministic nonperiodic flow," *J. Atmos. Sci.*, vol. 20, p. 130, 1963.
[49] M. C. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science*, vol. 197, p. 287, 1977.
[50] L. C. Jaurigue, E. Robertson, J. Wolters, and K. Lüdge, "Reservoir computing with delayed input for fast and easy optimization," *Entropy*, vol. 23, no. 12, p. 1560, 2021.
[51] S. Boyd and L. O. Chua, "Fading memory and the problem of approximating nonlinear operators with volterra series," *IEEE Trans. Circuits Syst.*, vol. CAS-32, p. 1150, 1985.
[52] S. Oladyshkin and W. Nowak, "Data-driven uncertainty quantification using the arbitrary polynomial chaos expansion," *Reliab. Eng. Syst. Saf.*, vol. 106, pp. 179–190, 2012.
[53] D. Zhang, L. Lu, L. Guo, and G. E. Karniadakis, "Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems," *J. Comput. Phys.*, vol. 397, p. 108850, 2019.
[54] O. G. Ernst, A. Mugler, H. J. Starkloff, and E. Ullmann, "On the convergence of generalized polynomial chaos expansions," *ESAIM Math. Model. Numer. Anal.*, vol. 46, no. 2, pp. 317–339, 2012.
[55] D. J. Gauthier, E. M. Bollt, A. Griffith, and W. A. S. Barbosa, "Next generation reservoir computing," *Nat. Commun.*, vol. 12, no. 1, p. 5564, 2021.

Supplementary Material: The online version of this article offers supplementary material (https://doi.org/10.1515/nanoph-2022-0415).