Master Memory Function for Delay-Based Reservoir Computers With Single-Variable Dynamics

Felix Köster, Serhiy Yanchuk, and Kathy Lüdge

***Abstract*— We show that many delay-based reservoir computers considered in the literature can be characterized by a universal master memory function (MMF). Once computed for two independent parameters, this function provides linear memory capacity for any delay-based single-variable reservoir with small inputs. Moreover, we propose an analytical description of the MMF that enables its efficient and fast computation. Our approach can be applied not only to single-variable delay-based reservoirs governed by known dynamical rules, such as the Mackey–Glass or Stuart–Landau-like systems, but also to reservoirs whose dynamical model is not available.**

***Index Terms*— Machine learning, nonlinear dynamics, reservoir computing.**

I. Introduction

RESERVOIR computing is a neuromorphic-inspired machine learning paradigm, which enables high-speed training of recurrent neural networks and is capable of solving highly complex time-dependent tasks. First proposed by Jaeger [1] and inspired by the human brain [2], it utilizes the inherent computational capabilities of dynamical systems. Very recently, the universal approximation property has also been shown for a wide range of reservoir computers, which solidifies the concept as a broadly applicable scheme [3]. Bollt et al. [4], Gauthier et al. [5], and Jaurigue and Lüdge, [6] pointed out a connection between reservoir computers and vector autoregressive (VAR) and nonlinear VAR machines, which may be one of the reasons behind the surprising efficiency of reservoir computers for time-dependent tasks. Realizations range from electrical and optoelectrical [7], [8], [9] up to optical setups [10], [11], [12], [13], [14] and have shown the relevance of reservoir computing to practical applications like human action recognition [15]. Additionally, analytical and numerical analyses [16], [17], [18], [19] help in building an understanding of its working principles and improve its performance while pinpointing to easily implementable realizations [20], [21]. Utilizing reservoir computers as a handy supporting tool for predicting difficult spatiotemporal patterns has shown promising results in increasing the accuracy for predictions of chaotic dynamics like weather forecasting [22], [23]. Many groups endeavor to optimize delay-based reservoir computing, e.g., through the utilization of Taken's embedding theorem [24], [25], [26], [27]. This article provides additional analytical insight into the general computation performance of delay-based reservoir computers and with it introduces new possibilities to optimize reservoirs.

Originally, reservoir computing is performed with a network of nonlinear nodes, which projects the input information into a high-dimensional phase space, allowing a linear fit to regress or to linearly separate features [1]. In time-delayed reservoir computing, a single-dynamical node with delayed feedback is employed as a reservoir instead of the network [28]. The time-multiplexing procedure allows for such a single-element system to implement a recurrent ring network [28], [29], [30], [31], see Fig. 1. The absence of the need for a large number of nonlinear elements significantly reduces the complexity of the reservoir hardware implementation. Promising realizations with a single-delay-based reservoir [32], [33], [34], [35] give a first glimpse over the potential of this idea for, e.g., time-series predictions [36], [37], [38], [39], equalization tasks on nonlinearly distorted signals [40], and fast word recognition [41]. For a general overview, we refer to [42], [43], [44], and [45].

Often reservoirs are optimized to a specific task by hyperparameter tuning, which defeats the purpose of reservoir computing as a fast trainable machine learning scheme. Dambre et al. [46] introduced a task-independent quantification of a reservoir computer, building on the memory capacity notion already introduced in [1], whereas a high memory capacity pinpoints to generally well-performing reservoirs.

In this article, we provide an analytical tool for finding promising reservoir setups by introducing a master memory function (MMF) for delay-based reservoir computing with small inputs. The MMF allows for fast computable predictions of the linear memory capacity and it indicates that the linear memory capacity of reservoirs is similar for systems with similar linearizations.

The main idea behind our method can be outlined as follows. Consider a delay-based reservoir described by a general

Manuscript received 13 September 2021; revised 2 June 2022 and 6 September 2022; accepted 3 November 2022. Date of publication 18 November 2022; date of current version 4 June 2024. This study was supported by the "Deutsche Forschungsgemeinschaft" (DFG) in the framework of SFB910 and Project 411803875. *(Corresponding author: Felix Köster.)*

Felix Köster is with the Institut für Theoretische Physik, Technische Universität, 10623 Berlin, Germany (e-mail: f.koester@tu-berlin.de).

Serhiy Yanchuk is with the Potsdam Institute for Climate Impact Research, 14473 Potsdam, Germany, and also with the Institut für Mathematik, Humboldt University of Berlin, 10117 Berlin, Germany.

Kathy Lüdge is with the Institute of Physics, Ilmenau University of Technology, 98693 Ilmenau, Germany.

Digital Object Identifier 10.1109/TNNLS.2022.3220532

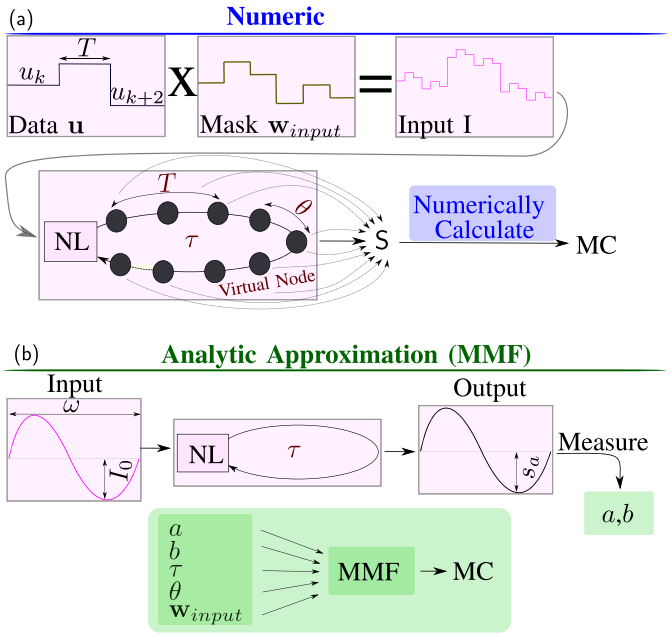

Fig. 1. Scheme of delay-based reservoir computing and the new analytic approximation method called MMF. Important timescales are marked in purple: T clock cycle, τ delay, and θ virtual node separation time. The upper panel depicts the direct computation method of the MC via numerical integration of the delay differential equation, and thus, the state matrix S. The lower panel shows the newly proposed method (MMF) which can be obtained from the measured small-signal parameters a and b, time timescales, and the input weights $\mathbf{w}_{\text{input}}$. (a) Numeric. (b) Analytic approximation (MMF).

nonlinear system $\dot{s}(t) = f(s(t), s(t-\tau), I(t))$, where $I(t)$ is an input signal, which is "small" in a certain sense, $s(t)$ determines the state of the reservoir, and τ is a time delay [see Fig. 1(a)]. The response $s(t)$ of the reservoir must be independent (at least to some extend) of its initial state, and the property is known as the echo state. Such a situation occurs when the reservoir is operating near an equilibrium state s^* that is stable in the absence of the input signal. Therefore, all reservoir dynamics take place in a neighborhood of this equilibrium and as a result, the reservoir linearization $\delta\dot{s}(t) = a\delta s(t) + b\delta s(t-\tau) + cI(t)$ approximates these dynamics. Here, δs is the deviation from the equilibrium. In the considered case of the single-variable reservoir, the scalar parameters a and b are the only determining quantities. The relatively simple form of the linearized system allows us to obtain an analytical expression for the linear memory capacity, which depends on the parameters a and b, and thus, parametrically determines the linear memory capacity of any reservoir with the above properties [see Fig. 1(b)]. We call the obtained function MMF due to its universal features, i.e., different reservoir computing setups, which possess the same linearizations (a, b), and yield the same linear memory capacity given by the MMF. Our results, thus, extend the recent results that calculate the (linear) memory capacity for either resonant setups [17], [47], i.e., setups where the clock cycle T equals the time-delay τ, or reservoirs with $\tau \neq T$ for special cases of differential equations [48], by utilizing an analyzing formula for arbitrary single-variable delay-based reservoir setups with small inputs.

This article is structured as follows. First, we will briefly revise the concept of time-delay-based reservoir computing and the concept of linear memory capacity. We will then present our main analytical result while additionally presenting an example code for an efficient evaluation of the obtained expression; the derivation is given in the Appendix. Finally, comparisons of numerically simulated reservoir computer performance with the semianalytical approach are provided. We also show in Section IV-F the applications of our results to reservoirs with an unknown dynamical model, where the parameters a and b are evaluated using the system response to external small periodic stimuli.

II. Time-Delay-Based Reservoir Computing

Reservoir computing utilizes the intrinsic abilities of dynamical systems to project the input information into a high-dimensional phase space [1]. By linearly combining the responses of the dynamical reservoir, a specific task is approximated. In the classical reservoir computing scheme, often, a so-called echo state network is used by feeding the input into a spatially extended network of nonlinear nodes. Linear regression is then applied to minimize the Euclidean distance between the output and a target. This approach is particularly resourceful for time-dependent tasks because the dynamical system that forms the reservoir acts as a memory kernel.

In the time-delay-based reservoir computing scheme [28], the spatially extended network is replaced by a single-nonlinear node with a time-delayed feedback loop. A time-multiplexing procedure with a periodic mask function g is applied to translate the input data to a temporal input signal. Similarly, the time-multiplexing procedure translates the single-temporal high-dimensional reservoir response to the spatiotemporal responses of virtual nodes. The virtual nodes play the same role as the spatial nodes in echo state networks.

A sketch of the delay-based reservoir computing setup is shown in Fig. 1(a). In the following, we will give a short overview of the quantities and notations used in this article. We also refer to our previous works [27], [48], [49], [50], [51] for a detailed explanation of how the reservoir setup is operated and task-independent memory capacities are computed.

Let us briefly remind the main ingredients of the time-multiplexed reservoir computing scheme [28], [48], [49], [50]. We apply an input vector $\mathbf{u} \in \mathbb{R}^K$ componentwise at times $t \in [t_{k-1}, t_k)$, $t_k = kT$, $k = 1, \ldots, K$, K being the number of sample points. The administration time for each input $t_{k+1} - t_k = T$ is the same and it is called the clock cycle T. To achieve a high-dimensional response to the same input, a T-periodic mask function g multiplies the input and the resulting signal enters the system (see Figs. 1 and 2). The mask g is a piecewise-constant function on N_V intervals, each of length $\theta = T/N_V$ corresponding to N_V virtual nodes with the corresponding constant values $\mathbf{w}_{\text{input}}$ called the input weights. The values of the mask function g play the same role as the input weights $\mathbf{w}_{\text{input}}$ in spatially extended reservoirs, with the difference that time-multiplexing distributes the weights $\mathbf{w}_{\text{input}}$ over time. The responses of the reservoir are collected in the

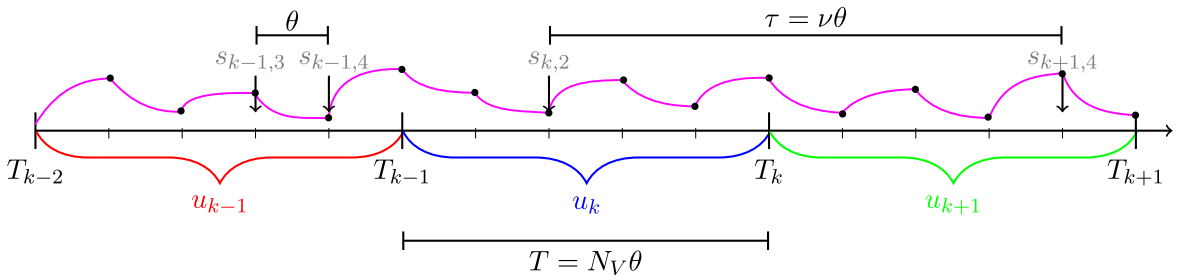

Fig. 2. Exemplary timeline sketch for time-delay-based reservoir computing. Three input intervals of length T are shown for the inputs u_{k-1}, u_k, and u_{k+1} in red, blue, and green, respectively. The delay time τ is $\tau = 7\nu$. The number of virtual nodes is $N_V = 5$, and thus, $\tau > T$. Four system states are indicated in gray ($s_{k-1,3}$, $s_{k-1,4}$, $s_{k,2}$ and $s_{k+1,4}$), where $s_{k,2}$ influences $s_{k+1,4}$ directly via the delay time τ. The pink line indicates an example trajectory, with black dots showing the measured system states, i.e., the virtual nodes.

state matrix $\mathsf{S} \in \mathbb{R}^K \times \mathbb{R}^{N_V}$, see Fig. 3. The elements of the state matrix are $[\mathsf{S}]_{kn} = \hat{s}(kT + n\theta)$ with $n = 1, \ldots, N_V$, and $k = 1, \ldots, K$, where $\hat{s}(kT + n\theta) \in \mathbb{R}$ is the state of the dynamical element of the reservoir at time $(kT + n\theta)$ shifted by the mean over all clock cycles $\hat{s}(kT + n\theta) = s(kT + n\theta) - \langle s(\cdot\, T + n\theta)\rangle$, see [46]. The average $\langle s(\cdot\, T + n\theta)\rangle$ can be understood as the averaging over the column elements, where the $\cdot$ denotes the variable over which the averaging is done.

A linear combination of the state matrix is given by $\mathsf{S}\mathbf{w}_{\text{reg}}$, where $\mathbf{w}_{\text{reg}} \in \mathbf{R}^{N_V}$ is a vector of weights. Such a combination is trained by ridge regression, i.e., the least square approximation to some target vector $\hat{\mathbf{y}}$

$$\arg\min_{\mathbf{w}_{\text{reg}}}\left[\left\|\mathsf{S}\mathbf{w}_{\text{reg}} - \hat{\mathbf{y}}\right\|_2^2 + \lambda\left\|\mathbf{w}_{\text{reg}}\right\|_2^2\right] \tag{1}$$

where $\|\cdot\|_2$ is the Euclidean norm, and λ is a Tikhonov regularization parameter. The solution to this problem is

$$\mathbf{w}_{\text{reg}} = \left(\mathsf{S}^T\mathsf{S} + \lambda\mathrm{I}\right)^{-1}\mathsf{S}^T\hat{\mathbf{y}}. \tag{2}$$

In the case of invertible $\mathsf{S}^T\mathsf{S}$, the matrix $(\mathsf{S}^T\mathsf{S})^{-1}\mathsf{S}^T$ is the Moore–Penrose pseudoinverse. We set $\lambda = 10^{-6} \cdot \max_s(\mathsf{S})$, where $\max_s(\mathsf{S})$ is the largest state response in the state matrix S. To quantify the system's performance, we use the capacity $\mathrm{C}_{\hat{\mathbf{y}}}$ (see [46], [49]) to approximate a specific task which is given by

$$\mathrm{C}_{\hat{\mathbf{y}}} = 1 - \mathrm{NRMSE} \tag{3}$$

where NRMSE [52] is the normalized root-mean-square error between the approximation $\mathbf{y} = \mathsf{S}\mathbf{w}$ and the target $\hat{\mathbf{y}}$

$$\mathrm{NRMSE} = \left(\frac{\sum_{k=1}^{K}(\hat{y}_k - y_k)^2}{K \cdot \mathrm{var}(\hat{\mathbf{y}})}\right)^{1/2} \tag{4}$$

where $\mathrm{var}(\hat{\mathbf{y}})$ is the variance of the target values $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_K)$.

III. Reservoir Computers and Their Quantification

Here, we introduce the linear memory capacity as a quantitative measure for the memory kernel of a dynamical system and give an overview of all used reservoir computer systems in this article.

$$\mathsf{S} = \overbrace{\begin{pmatrix} \vdots & & & & \\ \hat{s}_{k-1,1} & \hat{s}_{k-1,2} & \hat{s}_{k-1,3} & \hat{s}_{k-1,4} & \hat{s}_{k-1,5} \\ \hat{s}_{k,1} & \hat{s}_{k,2} & \hat{s}_{k,3} & \hat{s}_{k,4} & \hat{s}_{k,5} \\ \hat{s}_{k+1,1} & \hat{s}_{k+1,2} & \hat{s}_{k+1,3} & \hat{s}_{k+1,4} & \hat{s}_{k+1,5} \\ \vdots & & & & \end{pmatrix}}^{\text{Virtual Nodes } N_V} \Bigg\} \begin{matrix}\text{Number of} \\ \text{Inputs } K\end{matrix} \tag{5}$$

Fig. 3. State matrix S corresponding to the timeline shown in Fig. 2 with $N_V = 5$.

A. Memory Capacity

The central task-independent quantification was introduced by Jaeger [1] and refined by Dambre et al. [46], which yields that the computational capability of a reservoir system can be quantified via an orthonormal set of basic functions on a sequence of inputs. Here, we give a recap of the quantities introduced in [27], [49], [50], and [51] and focus on the linear memory capacity.

In particular, the capacity to fulfill a specific task is given by

$$\mathrm{C}_{\hat{\mathbf{y}}} = \frac{\hat{\mathbf{y}}^T\mathsf{S}\left(\mathsf{S}^T\mathsf{S} + \lambda\mathrm{I}\right)^{-1}\mathsf{S}^T\hat{\mathbf{y}}}{\|\hat{\mathbf{y}}\|^2} = \frac{\hat{\mathbf{y}}^T\mathbf{y}}{\|\hat{\mathbf{y}}\|^2} \tag{6}$$

which can be derived from (3) (see [46], [49]). The capacity equals 1 if the reservoir computer computes the task perfectly, and thus, $\mathbf{y} = \hat{\mathbf{y}}$; and it equals $C = 0$ if the prediction is not correlated with the target. In between 0 and 1 if it is partially capable of fulfilling the task. To quantify the system's capability for approximating linear recalls of inputs, an input sequence $\{u\} = \{u_{-K}, \ldots, u_{-3}, u_{-2}, u_{-1}\}$ is applied, where u_k are uniformly distributed random numbers, independent and identically drawn in $[-1, 1]$. With the input sequence $\{u\}$ of random numbers, the reservoir response is collected in the state matrix S.

To describe a linear recall task of l steps into the past, the target vector $\hat{\mathbf{y}}$ is defined as

$$\hat{\mathbf{y}}_l = \{\ldots, u_{-3-l}, u_{-2-l}, u_{-1-l}\} \tag{7}$$

which is the linear recall l steps into the past. Formally, one considers an infinitely long sequence $K \to \infty$. To approximate it numerically, we use $K = 75\,000$.

The *linear memory capacity* MC is defined as the sum of the capacities of all possible linear recall tasks

$$\mathrm{MC} = \sum_{l=0}^{\infty} \mathrm{C}_l \tag{8}$$

where $C_l = C_{\hat{\mathbf{y}}_l}$ is the capacity of the lth recall into the past. This quantification is task-independent, and thus, implications for specific applications cannot be given. Different tasks may need different specific capacities. The measure MC thus only gives a hint for well-performing reservoirs in the context of using the full scope of the given reservoirs, rather than a direct task-specific estimate. We have to point out, that the linear-nonlinear tradeoff is a well-known effect [46], [53], and thus, a system with high linear memory capacity can yield a low nonlinear transformation capability. Nevertheless, we believe predicting a well-performing linear memory kernel reservoir is beneficial for a general reservoir computer setup, as higher nonlinear memory transformation can be utilized by adding additional reservoir systems with increased perturbations.

B. Reservoir Systems

The delay-based reservoir computer systems used are as follows.

1) Stuart–Landau Oscillator [54], [55] With Delayed Feedback:

$$\frac{ds(t)}{dt} = \big(p_{\mathrm{SL}} + \eta I(t) + \gamma_{\mathrm{SL}} s(t)^2\big) s(t) + \kappa s(t-\tau). \tag{9}$$

Here, $s(t)$ describes the real-valued amplitude of the system, κ is the feedback strength, τ the delay time, p_{SL} is a system parameter, and η the input strength of the information fed into the system.

For $p_{\mathrm{SL}} + \kappa > 0$ and $\eta = 0$, (9) has only the trivial equilibrium $s^* = 0$, and for $p_{\mathrm{SL}} + \kappa < 0$, additionally, the nontrivial equilibria $(s^*)^2 = -(p_{\mathrm{SL}} + \kappa)/\gamma$ exist, which appear in a pitchfork bifurcation at $p_{\mathrm{SL}} + \kappa = 0$. The linearization at the nontrivial equilibria (taking into account the input term) reads

$$\frac{\delta s(t)}{dt} = a\delta s(t) + b\delta s(t-\tau) + cI(t) \tag{10}$$

where $a = -2p_{\mathrm{SL}} - 3\kappa = p_{\mathrm{SL}} + 3\gamma\,(A^*)^2$, $b = \kappa$, and $c = \eta s^*$.

2) Mackey–Glass System [56]:

$$\frac{ds(t)}{dt} = (p_{\mathrm{MG}} + \eta I(t)) s(t) + \frac{\alpha s(t-\tau)}{1 + s^q(t-\tau)} \tag{11}$$

where $s(t)$ is the dynamical variable, $s(t-\tau)$ is the delayed variable, and p_{MG}, q, and α are control parameters. The reservoir input is fed into the system via the term $\eta I(t)$. We set $q = 1$, for which the system possesses a stable nontrivial equilibrium $s^* = -(p_{\mathrm{MG}} + \alpha)/p_{\mathrm{MG}}$ (for $\eta = 0$). The corresponding linearization at this equilibrium is (10) with $a = p_{\mathrm{MG}}$, $b = \alpha/(1+s^*)^2$, and $c = \eta s^*$.

TABLE I
OVERVIEW OF USED PARAMETERS AND VARIABLES

Parameter	Description
	Sec. I and Sec. II
T	Clock cycle/Input interval time
$\mathbf{w}_{input}$	Input vector weights
w_n	components of the input vector $\mathbf{w}_{input}$
θ	Time interval between two virtual nodes
$\mathbf{u}$	Input for the reservoir
u_k	k-th input for the reservoir
τ	Time-delay of feedback
$s(t)$	System state variable
ν	Time-delay in virtual node distances $\nu = \tau/\theta$
N_V	Number of virtual nodes. $N_V \cdot \theta = T$
n	Running index for virtual nodes
K	Number of training samples applied to the reservoir
S	State matrix filled with responses of the reservoir state $s(t)$
$\mathbf{w}_{reg}$	Weight vector that minimizes Eq. (1). Solved via Eq. (2)
λ	Tikhonov regularization parameter
$\hat{\mathbf{y}}$	The target task given to the input $\mathbf{u}$
$\mathrm{C}_{\hat{\mathbf{y}}}$	Capacity to approximate task given by $\hat{\mathbf{y}}$
NRMSE	Normalized root mean square error given by Eq. (4)
	Sec. (III-A)
$\hat{\mathbf{y}}_l$	The input vector $\mathbf{u}$ shifted by l entries as task target
C_l	Capacity to approximate the linear recall task l steps into the past
MC	Total linear memory capacity $\mathrm{MC} = \sum_0^\infty \mathrm{C}_l$
	Sec. III-B
p_{SL}	Control parameter of the Stuart-Landau oscillator
η	Input strength of reservoir inputs
$I(t)$	Reservoir computer input
γ_{SL}	Nonlinearity of Stuart-Landau oscillator
κ	Feedback strength of Stuart-Landau oscillator
p_{MG}	Control parameter of the Mackey-Glass system
α	Control parameter of the Mackey-Glass system
q	Control parameter of the Mackey-Glass system
μ	Control parameter of the Ikeda nonlinearity
s_0	Control parameter of the Ikeda phase
a	Linearization parameter for local dynamics
b	Linearization parameter for delayed dynamics
c	Linearization parameter for input dynamics
s^*	Fixed point of the reservoir computer
	Sec. IV-A
$\mathrm{MC}_{\mathrm{MMF}}$	Total linear memory capacity via the master memory function
p	Local linear answer $p = e^{a\theta}$
m	Delay linear answer $m = -(b/a)(1-p)$
γ	Input linear answer $\gamma = -(c/a)(1-p)$

3) Ikeda-System [57], [58]:

$$\frac{ds(t)}{dt} = -s(t) + \mu \sin(s(t-\tau) - s_0) + \eta I(t) \tag{12}$$

where μ is a control parameter modeling the influence of the nonlinearity, s_0 is a constant phase, τ is the time-delay, and $\eta I(t)$ is the reservoir computer input. The fixed point of the system for $\eta = 0$ is given by $s^* = \mu \sin(s^* - s_0)$. Being a transcendental equation, there is no closed-form solution, and thus, we numerically calculate s^*. The linearization values are then given by $a = -1$, $b = \mu\cos(s^* - s_0)$, and $c = \eta$. The Ikeda-system is used in Section IV-F to show the application of the MMF for unknown systems.

C. Overview of Used Parameters

An overview of all used parameters is given in Table I. Table I is divided into the sections of the first parameter appearance.

IV. Results

A. Analytic Description of Memory Capacity

From (6), we see that the capacity to approximate a specific input is given by the inverse of the covariance matrix $(\mathsf{S}^T\mathsf{S})^{-1}$ (corrected by $\lambda\mathrm{I}$), also called the concentration matrix [59], and the matrix multiplication of the state matrix and the target $\mathsf{S}^T\hat{\mathbf{y}}$. Thus, it is necessary to derive the state matrix S from the responses to the small perturbations of the system. This has already been done for 1-D reservoirs with $\tau = T$ by an Euler step scheme [17], [47], and for 1-D reservoirs with $\tau \neq T$ for special cases of differential equations [48]. We would like to extend this knowledge by analyzing arbitrary single-variable systems and $\tau \neq T$. We assume the virtual node distance θ to be small and $\tau = \nu\theta$, with $\nu \in \mathbb{N}^+$. We also assume the operation point of the reservoir to be a stable equilibrium. We will exemplarily validate our analysis on the three 1-dimensional nonlinear reservoirs given by (9), (11), and (12).

Our main analytic result is the modified state matrix $\tilde{\mathsf{S}}$, which we can use to determine the MC, while we can calculate it solely from the linearized system. The mathematical derivation is given in Appendix A. The entries $[\tilde{\mathsf{S}}]_{jn}$ are given by

$$[\tilde{\mathsf{S}}]_{\ln} = \frac{\gamma}{\sqrt{3}} \sum_{n+lN_V \le i+r\nu}^{i+r\nu<n+(l+1)N_V} \binom{r+i}{i} p^i m^r w_{(n-i-r\nu) \bmod N_V} \tag{13}$$

where $p = e^{a\theta}$, $m = -(b/a)(1-p)$, $\gamma = -(c/a)(1-p)$, the parameters a, b, and c are given by the linearization (10), w_n are the weights of the mask function $g(t)$, and thus, the time-multiplexing. The index l corresponds to the lth linear recall, and n to the nth virtual node. The columns of the modified state matrix contain entries in the direction of the lth shifted input. As we show in Appendix A, the covariance of the modified state matrix approximates the original state matrix

$$\tilde{\mathsf{S}}^T\tilde{\mathsf{S}} = \mathsf{S}^T\mathsf{S}. \tag{14}$$

Moreover, we also show in Appendix A that the full linear memory capacity can be calculated by using solely the modified state matrix

$$\mathrm{MC} \approx \mathrm{MC}_{\mathrm{MMF}} = \mathrm{tr}\left(\tilde{\mathsf{S}}^T\left(\tilde{\mathsf{S}}^T\tilde{\mathsf{S}} + \lambda\mathrm{I}\right)^{-1}\tilde{\mathsf{S}}\right) \tag{15}$$

and the capacity of the lth recall is given by

$$C_l \approx \tilde{\mathsf{S}}_l^T\left(\tilde{\mathsf{S}}^T\tilde{\mathsf{S}} + \lambda\mathrm{I}\right)^{-1}\tilde{\mathsf{S}}_l \tag{16}$$

where $\tilde{\mathsf{S}}_l$ is the lth row of $\tilde{\mathsf{S}}$.

We call the memory capacity given by (15) the MMF. For given parameters of the linearization a, b, and c, as well as the mask coefficients w_j, this function can be evaluated in a much more efficient way than the direct evaluation of the linear memory capacity via a stepwise integration of the differential equation. A speed comparison is given in Appendix B and shows improvement of at least by a factor of 25. The new approach does not require calculating the reservoir, and it does not involve the input sequence u_k. We believe that such an analytical approach to a general computation performance like the linear memory capacity can help to build insight into delay-based reservoir computers and give a handy tool for new optimization approaches. We also point out that the MMF opens the possibility to use measurements of the small signal response of unknown dynamical systems to evaluate the linear memory capacity.

B. Efficient Numerical Evaluation of the Memory Capacity and the Modified State Matrix

The obtained approximations of the modified state matrix (13) and memory capacity function (15) allow for efficient numerical evaluation. For this, we propose the following scheme, which we also show as a pseudocode in Algorithm 1.

First, we iterate over all entries of a modified Pascal's triangle (see Fig. 7 in the appendix for more information on that), which can be done by two nested loops r and i. We do this until all entries in a row q are below a given threshold ϵ for $i + r = q$ for $i, r \in \mathbb{N}$ (see Fig. 7). The threshold ϵ ensures that we cut unnecessary terms smaller than the regularization parameter λ. A third loop n goes over all virtual nodes N_V adding the result $\binom{r+i}{i} p^i m^r$ multiplied with the corresponding input weights $w_{n+i+r\nu \bmod N_V}$ to all corresponding entries $\tilde{s}_{\lfloor (n+i+r\nu)/N_V \rfloor, n}$, that and thus, lie in the same input interval l. See Appendix A for more information. The algorithm to compute the modified state matrix $\tilde{\mathsf{S}}$ is given in the following, where $\lfloor y \rfloor$ is the floor function rounding y down to the greatest integer less than or equal to y, get BinomialTerm (i,l,p) returns $\binom{r+i}{i} p^i m^r$ and $\mathbf{w}_{\mathrm{input}}$ is the mask weight vector of length N_V. The implemented C++ code can be found in a linked GitHub repository and can be used to calculate the memory capacity for given a,b, ν, θ, and N_V.

Algorithm 1 Calculate Modified State Matrix

```
State ∈ ℝ^{N_V × K};
for RowsInPascalsTriangle < maxRow do
    for ColumnsInRow < RowNumber + 1 do
        for VirtualNeuron < NumberVirtualNeurons do
            n = VirtualNeuron;
            i = RowsInPascalsTriangle;
            r = ColumnsInRow;
            State (⌊(n + i + rν)/N_V⌋,n) +=
            getBinomialTerm(i, r, p) · w_{n+i+rν mod N_V};
            if getBinomialTerm(i, r, p) < σ for all
            ColumnsInRow then
                return State ;
        end for
    end for
end for
```

C. Direct Simulation of the Reservoir and Memory Capacity

Simulations have been performed in standard C++. For linear algebra calculations, the linear algebra library "Armadillo" [60] was used. To numerically integrate the delay-differential equations, a Runge–Kutta fourth-order method was applied, with integration step $\Delta t = 0.01$ in dimensionless

time units. First, the system is simulated without reservoir inputs, thus letting transients decay. After that, a buffer time of 10 000 inputs was applied (this is excluded from the training process). In the training process, $K = 75\,000$ inputs were used to have sufficient statistics. Afterward, the memory capacities C_l of linear recalls were calculated with (6), whereby a testing phase is not necessary. The linear memory capacity MC was calculated by summing the obtained capacities C_l. For the piecewise-constant T-periodic mask function $g(t)$ independent and identically distributed random numbers between [0, 1] were used.

For all simulations, the input strength η was fixed to 10^{-3}. The small input strength was used to guarantee linear answers of the reservoir and, hence, the relevance of the approximation. In the appendix we show a detailed analysis for increased input strength.

D. Comparison of MMF and Direct Numeric Calculations of the Memory Capacity

In this section, we illustrate the MMF's effectiveness. First, we show that the MMF provides a very good approximation of MC using the reservoir given by (9). The analytical approximation of the MC works quite well as long as θ is small enough compared with the linear answer time scale of the system, given by the largest Lyapunov exponent of the local dynamics. This is fulfilled for typical reservoir computing setups, as one would, otherwise, lose computation speed. In the second part, we show how MMF provides a universal, system-independent characteristic. For this, we compare MMF with the memory capacities of different reservoirs. Each particular reservoir realization is described by one parameter combination of linearization. In the last part, we describe how MMF can be computed for reservoirs with the unknown dynamical rule. For this, the parameters a and b of the linearization are measured from the system's response to a small periodic input.

Fig. 4 shows the memory recall capacity C_l obtained from direct simulations and via the MMF for four different cases of the Stuart–Landau system, given by (9). The exact parameters are given in the caption of Fig. 4. The directly simulated results are shown by blue solid lines and blue markers, whereby green dashed lines and green markers show the MMF. For a small virtual node distance $\theta = 0.5$ in Fig. 4(a) and (c), the MMF predicts the linear memory capacity very accurately. For a higher value of $\theta = 1.6$ [Fig. 4(b) and (d)], the accuracy drops, though the results are still accurate for qualitative predictions and describe the general trend of the system's memory capacity.

The scans in Fig. 4(c) and (d) were done with a higher delay time $\tau = 3.06$ T, which induces memory gaps [50]. Even though the memory capacity has a complex dependency on l at these parameter values, the prediction for the two different virtual node distances $\theta = 0.5$ and $\theta = 1.6$ is still accurate.

A 2-D parameter scan shown in Appendix E indicates that the predictions for the MC made by the MMF work for all system parameters investigated there. Thus, the predictive power of the new scheme is very promising.

Comparing the computation speed of the classical numerical integration and the new proposed scheme shows an increase in two to three orders of magnitude, depending on the operation point, the number of training steps K, and the value of the clock cycle T. A higher clock cycle T and more training steps K increase the simulation time for the direct numerical integration, whereas the new proposed scheme is independent of that. If the operation point is close to a bifurcation, the convergence of the new proposed scheme is slower, increasing the computation time needed. See Appendix B for a plot of the computation speed comparison in dependence of the linearization parameter b close to a bifurcation. Still, even very close to the bifurcation line, the computation speed is significantly higher (with a factor of about 100) making the MMF a valuable tool.

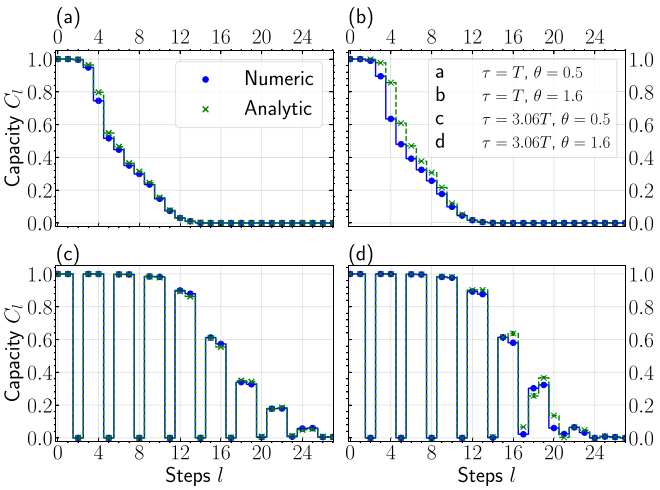

Fig. 4. Linear memory capacity C_l computed directly (blue dots) and via MMF by (16) (green crosses) for the Stuart–Landau reservoir computer, plotted as a function of the recall steps l. (a) and (c) $\theta = 0.5$, (b) and (d) $\theta = 1.6$, and (a) and (b) $\tau = T$, (c,d) (green) $\tau = 3.06T$. The values are averaged over 100 different masks. The parameters for all four setups are $T = 80$, $p_{\mathrm{SL}} = -0.05$, $\kappa = 0.06$, $\eta = 10^{-3}$, and $\gamma = 0.1$.

We also performed an analysis of the valid approximation range for the input strength η and virtual node distance θ. Both results are shown in Appendix C. Fig. 9 shows the results for the Stuart–Landau delay-based reservoir computer for different input strengths η, for which η of up to 10^{-2} yield comparable good predictions of the MC via the MMF. Higher values start to induce nonlinear answers in the system, thus inducing the well-known linear–nonlinear tradeoff.

Fig. 10 in Appendix C shows the results for the same system over the virtual node distance θ, ranging from $\theta = 0.05$ up to $\theta = 5$. The system was simulated with different p_{SL}, thus changing the linear answer time-scale a from $a = -0.11$ up to $a = -0.29$. The faster the system reacts the more inaccurate the predictions of the MMF are for higher values of θ. An explanation for this can be given via the assumption used in Section A. Because θ is always small in reservoir computing, we assumed a constant value of $s(t)$ on one θ-interval in (27). This approximation only holds for θ, which is small relative to the linear answer time-scale a. The rule of thumb used in [28] is given by $\theta = 0.2a$. Looking at Fig. 10, we see that our approximation holds for values of up to $\theta = 3a$ for a relative error of $\Delta\mathrm{MC} \approx 0.1$. We can, thus, conclude that

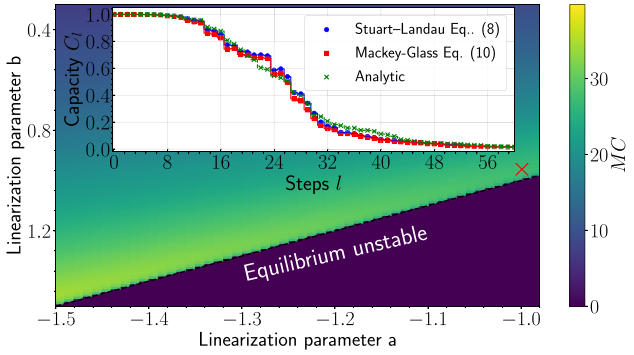

Fig. 5. Memory capacity computed by the MMF in the 2-D parameter plane of the linearization parameters a and b, with $N_V = 100$, $\tau = 72$, and $\theta = 0.5$. At the edge of instability, the performance is highest. Inset: MC over the recall steps l of the MMF for the Stuart–Landau and the Mackey–Glass system at the parameter point indicated at the red cross.

the assumption of a constant value on one θ interval for a delay-based reservoir computer is justified.

E. Universality

An exciting result that follows from the MMF concept is the possibility to generalize to any 1-D reservoir with one time-delay. Every 1-D reservoir with one time-delay that has the same linearization yields the same linear memory capacity. To illustrate this, we compare the Stuart–Landau reservoir system given by (9) and the Mackey–Glass reservoir system given by (11). The inset of Fig. 5 shows the capacity to recall the lth step into the past C_l as a function of l for the Stuart–Landau (blue), the Mackey–Glass (red), and the MMF given by (16) (green). Both systems are operated such that their linearization parameters a and b are equal. It can be seen that in this case the C_l's have the same values.

This underlines that it is enough to compute the linearization parameters a and b to predict the MC of a single-variable delay-based reservoir computer. The color plot in Fig. 5 shows the MMF given by (15) for different parameter values a and b. A well-performing operation point seems to be the edge to instability, agreeing with the known rule of thumb from the literature [61], [62].

It, thus, follows that analyzing the Jacobian [linearization given by (10)] for fixed delay τ, virtual node distance θ, and the number of virtual node N_V is sufficient to predict the linear memory capacity of a single-variable time-delay-based reservoir computer, and this memory capacity is given by the MMF via (15) and (16).

F. Systems With Unknown Dynamics—Small Signal Response Approach

In this chapter, we show an experimentally accessible approach for measuring the parameters a and b for a delay system whose dynamical equations of motion are not known and which can be described by a single variable. The linearized dynamical system was introduced in (10). To measure a and b, we perturb the system with a harmonic periodic signal $I(t) = I_0 e^{i\omega t}$. When this signal is small, we can consider the perturbed linearized system

$$\frac{d\delta s(t)}{dt} = a\delta s(t) + b\delta s(t-\tau) + cI_0 e^{i\omega t} \tag{17}$$

where the complex form is chosen out of convenience. Due to linearity, the real solution is obtained simply by taking the real part. We consider the case of real a and b, which always holds when the reservoir variables are real.

Since the homogeneous solution decays to the stable equilibrium (we assume its exponential stability), the solution of (17) converges to the particular solution, given by

$$\delta s_a(t) = cI_0 H^{-1}(\omega) e^{i\omega t} \tag{18}$$

with $H^{-1}(\omega) = i\omega - a - be^{-i\omega\tau}$. The ratio of the output to the input amplitude equals the transfer function

$$\frac{|\text{Output}|}{|\text{Input}|} = \frac{|\delta s_a(t)|}{|cI_0|} = |H^{-1}(\omega)| \tag{19}$$

where $|H(\omega)|$ can be measured.

To determine the parameters a and b, it is sufficient to measure the transfer function at two frequencies, for example, at $\omega_R = 2\pi/\tau$ and $\omega_A = \pi/\tau$. The first frequency is resonant to the delay while the second is in "antiphase" to the delay τ. We assume τ to be known, even though an extension of the scheme to unknown τ is possible. It holds

$$F(\omega_R) := |H(\omega_R)|^2 = \omega_R^2 + (a+b)^2$$
$$F(\omega_A) := |H(\omega_A)|^2 = \omega_R^2 + (a-b)^2.$$

From the above, we can obtain the values for a and b

$$a = -\frac{1}{2}\left(\sqrt{F(\omega_R) - \omega_R^2} + \sqrt{F(\omega_A) - \omega_A^2}\right) \tag{20}$$

$$b = -\frac{1}{2}\left(\sqrt{F(\omega_A) - \omega_A^2} - \sqrt{F(\omega_R) - \omega_R^2}\right) \tag{21}$$

where the values of $F(\omega_A)$ and $F(\omega_R)$ can be obtained experimentally or numerically by perturbing and measuring the response of the reservoir.

We remark that the choices of the resonant and antiphase perturbation frequencies are convenient but not unique. Clearly, one can perturb at other frequencies to obtain a and b. Moreover, the above-mentioned idea can be generalized to the case of complex-valued parameters a and b and to unknown time-delays τ, whereby more frequencies must be tested.

The measured values of the parameters a and b for a reservoir with unknown dynamics can be then simply used in MMF to determine the linear memory capacities.

To demonstrate the experimental procedure of measuring a and b via a small signal response, we use the Ikeda-system introduced in Section III-B in (12) with $x_0 = 1.5$, $\tau = 72$, and varying nonlinearity parameter μ ranging from $\mu_{\min} = 0.1$ up to $\mu_{\max} = 1.2$. In this range, the system has one stable fixed-point solution. We choose $I(t) = \cos(\omega t)$. With $\omega_R = 2\pi/\tau$ or $\omega_R = \pi/\tau$ and $\eta = 0.01$. We simulate the systems response and measure $|s_{a,R/A}(t)| = \max|s(t) - s^*|$, thus $|s_{a,R/A}(t)|$ is the induced linear response oscillation amplitude superimposed on the fixed point s^* for the in-phase $|s_{a,R}(t)|$ or antiphase $|s_{a,A}(t)|$ case.

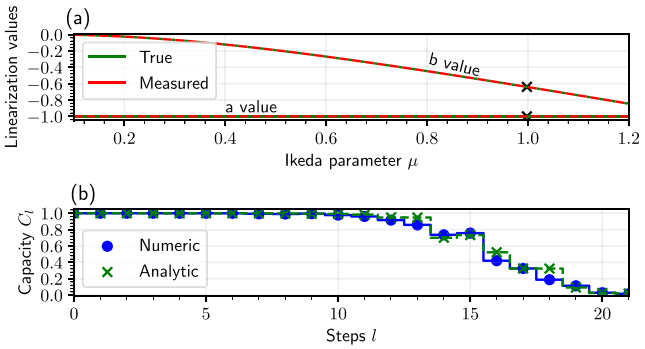

Fig. 6. Measurement of the Ikeda-system via small signal response and application of the MMF. (a) Linearization parameters a and b calculated (green) and measured (red) plotted over the Ikeda nonlinearity parameter μ. The black cross indicates the simulation parameters chosen for (b). (b) Linear recall capacities C_l over the recall steps l. Blue solid lines with circle markers show the fully numerical calculated capacities, while green dashed with cross markers show the ones calculated via the MMF. We chose $\tau = 72$, $\theta = 0.5$, and $N_V = 100$.

The results are shown in Fig. 6(a). The red curve shows the measured values of a and b over the Ikeda parameter μ. We additionally calculated the true linearization parameters via the linearization that was derived in Section III-B. The result is shown as a green solid line in Fig. 6(a). We see that the measured and the true values are nearly identical, and thus, the measuring method is working as expected. We then used one example parameter pair shown as a black cross in Fig. 6(a) and simulated the full Ikeda-system as a delay-based reservoir computer at this parameter setup, numerically evaluated the linear memory capacities C_l, and plotted them over the recall steps l in Fig. 6(b) as a blue curve. Subsequently, the MMF was used to calculate C_l for the measured $a = -1$ and $b \approx -0.6169$ over the same recall steps l. The results are shown in Fig. 6(b) as a green dashed line. The fully numerically simulated and calculated capacities C_l and the MMF values obtained after measure a and b show very well the potential of the MMF.

V. Conclusion

We have developed a simple and fast method for calculating the linear memory capacity for time-delay-based reservoirs with single-variable dynamics, which we call the MMF. Our results can be used to predict reservoir computing setups with high linear memory capacities independent of the specific reservoir computer. We have shown universality properties for our MMF for any single-variable delay-based reservoir computer with small inputs, i.e., our approach predicts the linear memory capacity of the set of all single-variable delay-based reservoirs with small inputs. We additionally provided a method for measuring the linearization parameters a and b for dynamical systems with unknown models, therefore, enabling the possibility for experimental evaluation of the MMF. An example approach for the Ikeda-system was given.

One of the advantages of the delay-based reservoir, which allows the introduction of the MMF, is that it contains a small number of system parameters, while the dynamics remain infinite-dimensional. In the case of a small input signal and single-variable dynamics, these are only the linearization parameters a and b, the time-delay τ, the virtual node distance θ, and the number of virtual nodes N_V. Thus, if the linear memory capacity is computed for all possible values of these parameters, it covers the case of all possible reservoirs. This procedure could be difficult, if not impossible, for network-based reservoirs, where the system's parameters may include, e.g., multiple coupling weights.

Appendix A
Derivation of the Modified State Matrix and Reduced Formula for Memory Capacity

Consider a single-variable delay-differential equation, which describes the dynamics of the reservoir

$$\dot{s}(t) = F(s(t), s(t-\tau), I(t)) \tag{22}$$

where $I(t)$ stands for an external input. We assume that s^* is an equilibrium of this system, i.e., $F(s^*, s^*, 0) = 0$, and $I(t)$ is small. In the case when the dynamics of (22) takes place near the equilibrium point s^*, we can introduce the perturbative ansatz $s(t) = s^* + \delta s(t)$. Then, the linearized system for the perturbation $\delta s(t)$ reads

$$\delta\dot{s}(t) = a\delta s(t) + b\delta s(t-\tau) + cI(t) \tag{23}$$

where $a = \partial_1 F(s^*, s^*, 0)$, $b = \partial_2 F(s^*, s^*, 0)$, and $c = \partial_3 F(s^*, s^*, 0)$.

Consider θ to be the temporal node spacing of the reservoir, which is typically $\theta < \tau$. Then, (23) on any interval $[j\theta, (j+1)\theta]$ can be considered as the simple scalar ordinary differential equation (ODE) $\delta\dot{s}(t) = a\delta s(t)$ with the inhomogeneity $b\delta s(t-\tau) + cI(t)$. Moreover, according to the reservoir setup, the input is constant on this interval and equals $I(t) = I_j$. By variation of constants formula, we obtain the solution of (23) for $t \in [(j-1)\theta, j\theta]$

$$\delta s(t) = -\frac{cI_j}{a}\left(1 - e^{a(t-t_{j-1})}\right) + \delta s(t_{j-1})e^{a(t-t_{j-1})} + b\int_{t_{j-1}}^{t} e^{a(t-\xi)}\delta s(\xi-\tau)d\xi \tag{24}$$

where $t_{j-1} = (j-1)\theta$ is the left endpoint of the interval.

Denoting $\delta s_j(t) = \delta s(t_{j-1} + t)$ to be the function on the interval $[t_{j-1}, t_{j-1} + \theta]$, with $t \in (0, \theta)$, we rewrite (24) as

$$\delta s_j(t) = -\frac{cI_j}{a}\left(1 - e^{at}\right) + \delta s_{j-1}(\theta)e^{at} + b\int_0^t e^{a(t-\xi)}\delta s_{j-\nu}(\xi)d\xi, \quad t \in [0, \theta] \tag{25}$$

where we additionally used the relation $\delta s_{j-1}(\theta) = \delta s_j(0)$ and $\delta s_{j-\nu} = \delta s_j(t-\tau)$, $\nu = \tau/\theta$. By evaluating (25) at $t = \theta$, we obtain

$$\delta s_j(\theta) = -\frac{cI_j}{a}(1-p) + \delta s_{j-1}(\theta)p + bp\int_0^\theta e^{-a\xi}\delta s_{j-\nu}(\xi)d\xi, \quad p = e^{a\theta}. \tag{26}$$

Denote $s_j := s_j(\theta) = s^* + \delta s_j(\theta)$, which is the approximation for state of the reservoir (22) at the virtual node $s(j\theta)$.

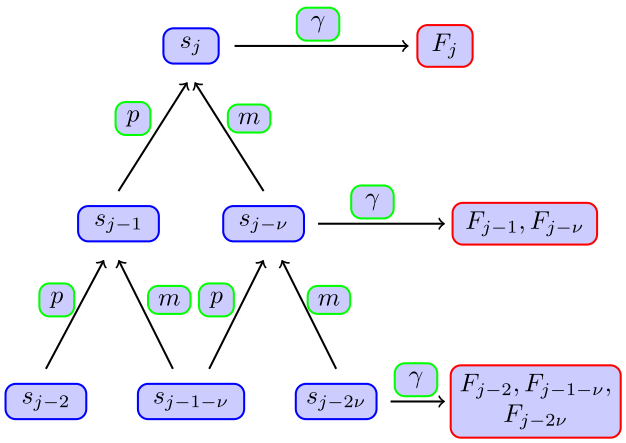

Fig. 7. Pascal's Triangle showing the series contributions for s_j, given by all the participating equilibria $s^*_{j-i-r\nu}$. Blue boxes show participating timesteps, green boxes show multiplications by time propagating factors and red boxes give equilibria contributions for the specific timesteps. Out of convenience, we denote $m = -(b/a)(1-p)$ and $\gamma = (1+(b/a))(1-p)$. F_j denotes the fixpoint factor contributions given by $\gamma \binom{r+i}{i} p^i m^r I_{j-i-r\nu}$ for a specific i, r.

From (26), we obtain

$$s_j = \left(1 - \frac{b}{a}p\right)(1-p)s^* - \frac{cI_j}{a}(1-p) + ps_{j-1} + bp \int_0^\theta e^{-a\xi} s_{j-\nu}(\xi) d\xi. \tag{27}$$

Furthermore, we approximate the integral from (27) by assuming $s_{j-\nu}(\xi) \approx s_{j-\nu}(\theta) = s_{j-\nu}$. The approximation holds, in particular, when θ is small. The obtained expression

$$s_j = \hat{s}^* + \gamma\, I_j + ps_{j-1} + ms_{j-\nu} \tag{28}$$

represents a discrete map (coupled map lattice) for approximating the state matrix S. Here, $\hat{s}^* = (1-(b/a)p)(1-p)s^*$, $m = -(b/a)(1-p)$, and $\gamma = -(c/a)(1-p)$. If considering it as a corresponding network with the nodes s_j, see [29], [31], [63], we see that the node s_j is coupled with the two nodes s_{j-1} and $s_{j-\nu}$ in a feed-forward manner with the coupling weights p and m, respectively. The schematic representation of such a coupling structure leads to Pascal's triangle shown in Fig. 7. The first row of Pascal's triangle from Fig. 7 shows the dependence on I_j, which is simply the multiplication by γ. In the second row, the contributions of I_{j-1} and $I_{j-\nu}$ are shown. To obtain these dependencies explicitly, we insert s_{j-1} and $s_{j-\nu}$ recursively in (28)

$$s_j = (1+p+m)\hat{s}^* + \gamma\left(I_j + pI_{j-1} + mI_{j-\nu}\right) + p^2 s_{j-2} + 2pms_{j-\nu-1} + m^2 s_{j-2\nu} \tag{29}$$

that is, we obtain the terms $\gamma\, pI_{j-1}$ and $\gamma\, mI_{j-\nu}$. To build up further intuition about the dependence of the state matrix on the input, we show here the third level by substituting recursively s_{j-2}, $s_{j-\nu-1}$, and $s_{j-2\nu}$ into (29)

$$\begin{aligned} s_j = {} & \left(1+p+p^2+2pm+m+m^2\right)\hat{s}^* \\ & + \gamma\left(I_j + pI_{j-1} + mI_{j-\nu} + p^2 I_{j-2} \right. \\ & \left. + 2pmI_{j-\nu-1} + m^2 I_{j-2\nu}\right) \\ & + p^3 s_{j-3} + 3p^2 ms_{j-2-\nu} + 3pm^2 s_{j-2\nu-1} + m^3 s_{j-3\nu}. \end{aligned} \tag{30}$$

To obtain a general recursive formula, we need to split the index into the appearing terms as $j - i - k\nu$, where k corresponds to the delayed ("right," m) and i to the "left" (p) connections in the coupling network in Fig. 7

$$s_j = A_1 s^* + \gamma \sum_{i,r=0} \binom{r+i}{i} p^i m^r I_{j-i-r\nu} \tag{31}$$

where A_1 is a constant depending only on p and m. For an infinitely long input sequence, the sum in (31) goes for all i, r from 0 to ∞. Practically, the sum is considered for the available data I_j. As a result, the reservoir states s_j are composed of a linear combination of the inputs with corresponding coefficients given in (31). We can drop the j-independent constant term $A_1 s^*$, because it only induces a constant shift of the state matrix S, and thus, can be absorbed in either a bias term or a centering preprocessing, one of the two is always done when calculating the memory capacity.

The elements of the state matrix S used in the reservoir computing setup are

$$[\mathsf{S}]_{kn} = \hat{s}_{kN_V+n} = s_{kN_V+n} - \langle s_{\cdot\, N_V+n}\rangle$$

where $\langle s_{\cdot\, N_V+n}\rangle$ is the average over the input intervals $\langle s_{\cdot\, N_V+n}\rangle = (1/K)\sum_{k=1}^{K} s_{kN_V+n}$. Here and later, the dot denotes the index, over which the averaging is performed. Taking into account (31), we obtain

$$\hat{s}_{kN_V+n} = \gamma \sum_{i,r=0} \binom{r+i}{i} p^i m^r \times \left(I_{kN_V+n-i-r\nu} - \langle I_{\cdot\, N_V+n-i-r\nu}\rangle\right). \tag{32}$$

The input I_k of the reservoir computer is given by the discrete input sequence $\mathbf{u}$ multiplied by the input weights

$$I_k = u_{\lfloor k/N_V \rfloor} w_{k \bmod N_V}.$$

Therefore, we obtain

$$\langle I_{\cdot\, N_V+n-i-r\nu}\rangle = w_{n-i-r\nu \bmod N_V}\left\langle u_{\cdot\, +\lfloor (n-i-r\nu)/N_V \rfloor}\right\rangle \approx 0 \tag{33}$$

since $\mathbf{u}$ has zero mean. Hence, we have the elements of the state matrix

$$\hat{s}_{kN_V+n} = \gamma \sum_{i,r=0} \binom{r+i}{i} p^i m^r I_{jN_V+n-i-r\nu}. \tag{34}$$

Correspondingly, the elements of the covariance matrix $\mathsf{S}^T\mathsf{S}$ from (6) are

$$[\mathsf{S}^T\mathsf{S}]_{nn'} = \sum_k [\mathsf{S}]_{kn}[\mathsf{S}]_{kn'} = \sum_k \hat{s}_{kN_V+n}\hat{s}_{kN_V+n'} \tag{35}$$

and they describe the covariance of the virtual node n with virtual node n' over all clock cycles k.

By substituting (34) into (35), we obtain

$$[\mathsf{S}^T\mathsf{S}]_{nn'} = \gamma^2 \sum_k \left(\sum_{r,i} \binom{r+i}{i} p^i m^r I_{kN_V+n-i-r\nu}\right) \times \left(\sum_{r',i'} \binom{r'+i'}{i'} p^{i'} m^{r'} I_{kN_V+n'-i'-r'\nu}\right). \tag{36}$$

One can show that the elements from $[\mathsf{S}^T\mathsf{S}]_{nn'}$ containing mixed terms of the form $u_i u_j$, $i \neq j$ can be approximated by zero since the random variable u_j is independently distributed with zero mean. The only nonzero second moment, which matters in $[\mathsf{S}^T\mathsf{S}]_{nn'}$, is the mean square

$$\sum_{k=0}^{\infty} u_k^2 = \frac{1}{3}. \tag{37}$$

Hence, for further simplification of $[\mathsf{S}^T\mathsf{S}]_{nn'}$, we keep only terms of the form u_k^2. The following calculations formalize these arguments, equation (38), as shown at the bottom of the page, where the second summation range (*) is taken over all values of r, i, r', i' such that $n + lN_V \leq i + r\nu < n + (l+1)N_V$ and $n' + lN_V \leq i' + r'\nu < n' + (l+1)N_V$. The obtained expression (38) does not depend on the sequence u_k, and hence, provides a significant simplification for calculating the covariance matrix. We may further notice that the same covariance (38) can be obtained by defining the modified state matrix $\tilde{\mathsf{S}} = \tilde{s}_{ln}$, where l is the lth interval of the shifted input (the lth recall) and n the nth virtual node. $\tilde{s}_{ln}$ is given by the sum over all combinations i, r that fall into the same shifted input interval j, that is,

$$\tilde{s}_{ln} = \frac{\gamma}{\sqrt{3}} \sum_{n+lN_V \leq i+r\nu}^{i+r\nu<n+(l+1)N_V} \binom{r+i}{i} p^i m^r w_{(n-i-r\nu) \bmod N_V}. \tag{39}$$

This is our main result because (39) defines the modified state matrix $\tilde{\mathsf{S}}$ from which all capacities C_l are derivable. More specifically, we have shown

$$\mathsf{S}^T\mathsf{S} \approx \tilde{\mathsf{S}}^T\tilde{\mathsf{S}}. \tag{40}$$

Furthermore, for the lth recall, where the target is the shifted input sequence $\hat{\mathbf{y}} = \{u_{k-l}\}_{k=1}^{\infty}$, we have, equation (41), as shown at the bottom of the page, therefore, it holds

$$\mathsf{S}^T\hat{\mathbf{y}}_l \approx \frac{1}{\sqrt{3}}\tilde{\mathsf{S}}_l$$

where $\tilde{\mathsf{S}}_l$ is the lth row of $\tilde{\mathsf{S}}$. Furthermore, we notice that

$$\hat{\mathbf{y}}_l^T\mathsf{S} = \left(\mathsf{S}^T\hat{\mathbf{y}}_l\right)^T \approx \frac{1}{\sqrt{3}}\tilde{\mathsf{S}}_l^T$$

and $\|\hat{\mathbf{y}}_l\|^2 \approx 1/3$. As a result, taking into account the definition of the memory capacity (6), we obtain the approximation for the capacity of the lth recall C_l by

$$C_l \approx \tilde{\mathsf{S}}_l^T\left(\tilde{\mathsf{S}}^T\tilde{\mathsf{S}} + \lambda\mathrm{I}\right)^{-1}\tilde{\mathsf{S}}_l. \tag{42}$$

The results can be understood in such a way that we constructed the modified state matrix $\tilde{\mathsf{S}}$, such that every column has entries in the statistical direction of the lth shifted input recall.

The full linear memory capacity is then given by the trace

$$\mathrm{MC}_{\mathrm{MMF}} = \mathrm{tr}\left(\tilde{\mathsf{S}}^T\left(\tilde{\mathsf{S}}^T\tilde{\mathsf{S}} + \lambda\mathrm{I}\right)^{-1}\tilde{\mathsf{S}}\right). \tag{43}$$

Appendix B
Computation Time

To compare the computation speed of the full numerically simulated differential equation and our new analytic approach, we simulated both systems. The full system with a timestep of $dt = 0.01$, buffer samples of 10 000, i.e., that 10 000 clock cycles were simulated and discarded, and 50 000 training

$$\begin{aligned}
[\mathsf{S}^T\mathsf{S}]_{nn'} &= \gamma^2 \sum_k \left(\sum_{r,i} \binom{r+i}{i} p^i m^r u_{k+\lfloor (n-i-r\nu)/N_V \rfloor} w_{(n-i-r\nu) \bmod N_V} \right) \\
&\quad \times \left(\sum_{r',i'} \binom{r'+i'}{i'} p^{i'} m^{r'} u_{k+\lfloor (n'-i'-r'\nu)/N_V \rfloor} w_{(n'-i'-r'\nu) \bmod N_V} \right) \\
&\approx \gamma^2 \sum_{l=1} \left(\sum_{*} \binom{r+i}{i} p^i m^r \binom{r'+i'}{i'} p^{i'} m^{r'} w_{(n-i-r\nu) \bmod N_V} w_{(n'-i'-r'\nu) \bmod N_V} \sum_k u_{k+l-1}^2 \right) \\
&\approx \gamma^2 \frac{1}{3} \sum_{l=1} \left(\sum_{*} \binom{r+i}{i} p^i m^r \binom{r'+i'}{i'} p^{i'} m^{r'} w_{(n-i-r\nu) \bmod N_V} w_{(n'-i'-r'\nu) \bmod N_V} \right)
\end{aligned} \tag{38}$$

$$\begin{aligned}
[\mathsf{S}^T\hat{\mathbf{y}}_l]_n &= \sum_k [\mathsf{S}]_{kn} [\hat{\mathbf{y}}_l]_k = \sum_k \hat{s}_{kN_V+n} u_{k-l} \\
&= \sum_k \left(\gamma \sum_{r,i} \binom{r+i}{i} p^i m^r u_{k+\lfloor (n-i-r\nu)/N_V \rfloor} w_{n-i-r\nu \bmod N_V} \right) u_{k-l} \\
&= \gamma \sum_{r,i} \binom{r+i}{i} p^i m^r w_{n-i-r\nu \bmod N_V} \sum_k u_{k+\lfloor (n-i-r\nu)/N_V \rfloor} u_{k-l} \\
&\approx \frac{\gamma}{3} \sum_{n+lN_V < i+r\nu}^{i+r\nu \leq n+(l+1)N_V} \binom{r+i}{i} p^i m^r w_{n-i-r\nu \bmod N_V} = \frac{1}{\sqrt{3}} \tilde{s}_{ln}
\end{aligned} \tag{41}$$

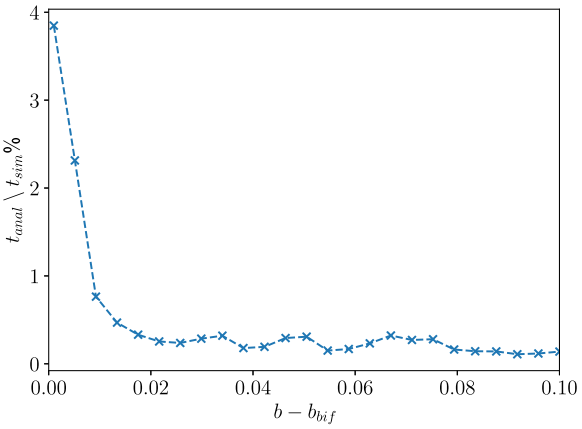

Fig. 8. Procentual change in computation speed between the full simulated system [Fig. 1(a)] and the analytic approach [Fig. 1(b)] as a function of the distance to the bifurcation point b_{bif}. t_{anal} is the needed computation time for the analytic and t_{sim} for the fully numerically simulated system. The simulations were done with a Runge–Kutta-4 method of timestep $dt = 0.01$, a buffer sequence of 10 000 clock cycles, and training samples of 50 000. The analytic approach computed all values in Pascal's triangle up to $10^{-6} \cdot \max_s(\mathsf{S})$. Parameters are $\tau = 141$, $T = 100$, and $N_V = 100$ (i.e., $\theta = 1$ and $a = -0.503$).

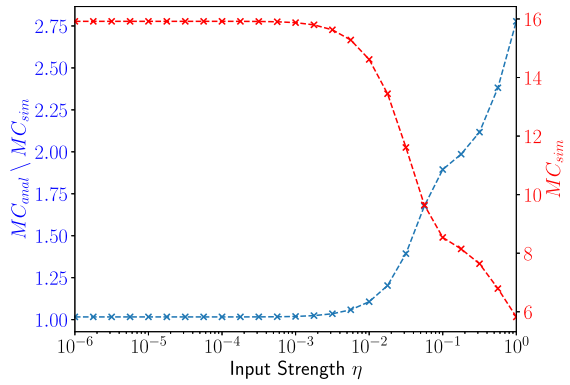

Fig. 9. Comparison of memory capacity MC of the fully simulated system MC_{sim} and the analytic approach MC_{anal} over the input strength η. The left axis shows the percentage of the analytically derived MC_{anal} to the fully numerically simulated MC_{sim}, while the right axis shows the MC_{sim} directly. The system was simulated with Runge–Kutta-4 method with timestep $dt = 0.01$, a buffer sequence of 10 000 clock cycles, and training samples of 50 000. The analytic approach computed all values in Pascal's triangle up to $10^{-6} \cdot \max_s(\mathsf{S})$. Parameters are $\tau = 141$, $T = 100$, and $N_V = 100$ (i.e., $\theta = 1$, $a = -0.503$, and $b = 0.201$).

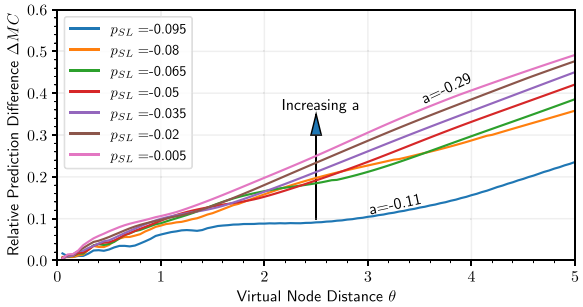

Fig. 10. Comparison of memory capacity MC of the fully simulated system MC_{sim} and the analytic approach MC_{anal} over the virtual node distance θ. The quantity ΔMC describes the relative MC difference [see (44)]. The system was simulated with Runge–Kutta-4 method of timestep $dt = 0.01$, a buffer sequence of 10 000 clock cycles, and training samples of 50 000. The analytic approach computed all values in Pascal's triangle up to $10^{-6} \cdot \max_s(\mathsf{S})$. Parameters are $\tau = N_V \cdot \theta$, $T = N_V \cdot \theta$, and $N_V = 50$.

samples to get high accuracy on the memory capacity. The analytic program was calculated until all values in a row in Pascal's triangle were below $10^{-6} \cdot \max_s(\mathsf{S})$. We compared the simulation speeds of both approaches on the same hardware on a parameter line scan of the linearization parameter b, scanning from values close to the bifurcation value b_{bif} in which the linearized system destabilizes up to values of about 0.1 greater than the bifurcation value b_{bif}. We show the percentage of the simulation time for the analytic approach t_{anal} in comparison with the simulation time t_{sim}, i.e., $t_{\text{anal}} \setminus t_{\text{sim}}\%$ in Fig. 8. We see that close to the bifurcation the analytic approach increases in computation time. This comes from the fact that the convergence of Pascal's triangle close to the bifurcation is slower. Still, the simulation time is at maximum 4% of the fully simulated system, showing at least a 25-fold increase in computation speed.

Appendix C
Range of Approximation

We would like to show the range of approximation for the new analytic approach for both the input strength η and the virtual node distance θ. For the input strength η, we compute the memory capacity MC of the fully simulated system MC_{sim} and the analytic approach MC_{anal} and show the relative memory capacity of the analytic approach to the full system, i.e., $\text{MC}_{\text{anal}} \setminus \text{MC}_{\text{sim}}$. The results are shown in Fig. 9 plotted over the input strength η for six orders of magnitude. A result close to 1 indicates a good agreement between the simulation and the analytic approach. For high input strengths, starting at around $\eta = 10^{-2}$, the analytic approach overestimates the real memory capacity, because high values of η induce nonlinear answers in the system, and thus, increase the nonlinear transformations of the reservoir in exchange for linear memory, see [46], [49], [50] for more information on that effect. This can be seen when the MC of the fully simulated system MC_{sim} is shown, plotted for the second y-axis in red.

As a second quantity, we plot the relative difference of the MC, that is,

$$\Delta\text{MC} = \frac{|\text{MC}_{\text{MMF}} - \text{MC}_{\text{direct}}|}{\text{MC}_{\text{direct}}} \tag{44}$$

over the virtual node distance θ for different values of the Stuart–Landau control parameter p_{SL}, where MC_{MMF} is the prediction of the MC for the MMF and $\text{MC}_{\text{direct}}$ is the true MC computed via a fully numerical simulation. The results are shown in Fig. 10. The different colored graphs depict different operation points of the Stuart–Landau delay-based reservoir computer p_{SL} ranging from $p_{\text{SL}} = -0.005$ up to $p_{\text{SL}} = -0.095$, which corresponds to linearization parameters of $a = -0.11$ up to $a = -0.29$, respectively. With increasing θ the prediction of the MMF becomes less accurate. This effect is intensified if the linearized local dynamics are faster, i.e., for more negative a. Faster local dynamics result in faster changing dynamics, and thus, the assumption of a constant

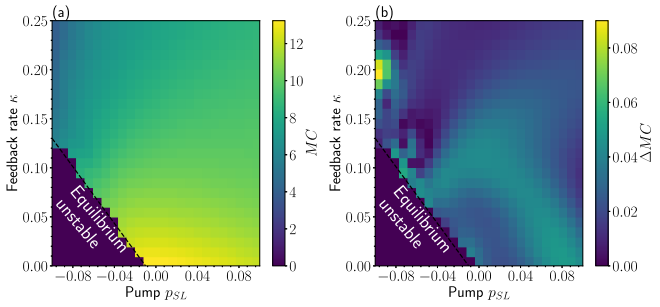

Fig. 11. Two-parameter characterization of the memory capacity of system (9) with respect the pump p_{SL} and feedback rate κ. (a) Total linear memory capacity of the directly simulated $\mathrm{MC}_{\mathrm{direct}}$. (b) Relative difference $\Delta\mathrm{MC}$ of the MMF $\mathrm{MC}_{\mathrm{MMF}}$ and the directly simulated $\mathrm{MC}_{\mathrm{direct}}$ value. The black dashed line shows the threshold of stabilization of the nontrivial equilibrium. The RV coefficient is $\mathrm{RV}(\mathrm{MC}_{\mathrm{direct}}, \mathrm{MC}_{\mathrm{MMF}}) = 0.99925$. The parameters are $N_V = 100$, $T = 100$ (corresponding to $\theta = 1$), $\tau = 1.41T$, $\eta = 10^{-3}$, and $\gamma = 0.1$.

$\delta s(t)$ on one virtual node interval θ becomes increasingly inaccurate.

Appendix D
χ_k^2 Estimation

We give a short insight into the χ_k^2 estimation introduced in [46]. When calculating capacities C_l, all below a fixed value r^* were excluded because of finite statistics, where r^* is given by the following relation. $\mathrm{CDF}(\chi^2(N_V, r^*))$ is the cumulative distribution function of the χ^2 function, and r^* is chosen such that $1 - \mathrm{CDF}(\chi^2(N_V, r^*))$ yields a probability $p_{\chi^2} = 10^{-6}$, i.e., the probability of a capacity having a value greater than r^* even though with infinite statistics ($K \to \infty$), it would have a value less than r^*. χ^2 is the probability density function of the sum of squared independent, standard normal random variables

$$\chi_k^2 = \sum_{i=1}^{k} Z_i^2. \tag{45}$$

See [46] for more information.

Appendix E
Broader Parameter Range Check

A two-parameter characterization of the memory capacity of the Stuart–Landau system (9) is shown in Fig. 11. The parameter space is spanned by the pump p_{SL} and the feedback rate κ. Fig. 11(a) shows the linear memory capacity, while Fig. 11(b) shows the relative difference $\Delta\mathrm{MC}$ of the MMF and the direct numerics [see (44)]. Small relative differences of up to 0.08 are seen for the simulations presented here for $\theta = 1$. One has to remember that reservoir computing is usually done with very small θ. The work [1] introduced a rough estimate of the optimal value for θ as $\theta \approx 0.2\lambda_{\mathrm{ans}}$, where λ_{ans} is the linear answer timescale of the system. In the case of the Stuart–Landau system, this is given by $\lambda_{\mathrm{ans}} = -2p_{\mathrm{SL}} - 3\kappa$. For the parameter space in Fig. 11, the highest value of the linear answer timescale $\lambda_{\mathrm{ans}} = -0.95$, and thus, θ is by a factor of five bigger than the proposed value given in [1] for optimal virtual node distance. In our approximation, we assume a constant state value on one θ-interval, and thus, $\theta = 1$ is a very high value, which is one of the reasons for the deviations.

To underline that the MMF (15) gives a reasonable estimation of the memory capacity, we also calculated the 2-D correlation coefficient $\mathrm{RV}(X, Y)$ between the directly simulated total linear memory capacity $\mathrm{MC}_{\mathrm{direct}}$ and the linear memory capacity $\mathrm{MC}_{\mathrm{MMF}}$ given by MMF in the 2-D plane of the pump p_{SL} and the feedback rate κ parameters. $\mathrm{RV}(X, Y)$ is the generalization of the squared Pearson coefficient for two dimensions and is calculated via

$$\mathrm{RV}(X, Y) = \frac{\mathrm{COVV}(X, Y)}{\sqrt{\mathrm{VAV}(X)\mathrm{VAV}(Y)}} \tag{46}$$

with

$$\Sigma_{XY} = E\left(X^T Y\right) \tag{47}$$

$$\mathrm{COVV}(X, Y) = \mathrm{tr}(\Sigma_{XY}\Sigma_{YX}) \tag{48}$$

$$\mathrm{VAV}(X) = \mathrm{tr}\left(\Sigma_{XX}^2\right). \tag{49}$$

Here, $E()$ is the expectation value, Σ_{XY} denotes the centered covariance matrix of the matrices X and Y, $\mathrm{COVV}(X, Y)$ denotes the trace of the matrix multiplication of $\Sigma_{XY}\Sigma_{YX}$ and $\mathrm{VAV}(X)$ the trace of the matrix multiplication Σ_{XX}^2. Calculating RV over the parameter range shown in Fig. 11 yields a value of $RV(\mathrm{MC}_{\mathrm{direct}}, \mathrm{MC}_{\mathrm{MMF}}) \approx 0.99925$. The correlation is close to the maximum of 1, allowing us to make accurate predictions of high-performing reservoirs with the MMF.

Acknowledgment

The authors thank David Hering, Lina Jaurigue, and Joscha Matysiak for fruitful discussions.

References

[1] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks," German Nat. Res. Inst. Comput. Sci., Berlin, Germany, GMD Rep. 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, no. 11, pp. 2531–2560, 2002.

[3] 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, Jan. 2020.

[4] E. 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, Interdiscipl. J. Nonlinear Sci.*, vol. 31, no. 1, Jan. 2021, Art. no. 013108.

[5] D. J. Gauthier, E. Bollt, A. Griffith, and W. A. S. Barbosa, "Next generation reservoir computing," *Nature Commun.*, vol. 12, no. 1, p. 5564, Sep. 2021.

[6] L. Jaurigue and K. Lüdge, "Connecting reservoir computing with statistical forecasting and deep neural networks," *Nature Commun.*, vol. 13, no. 1, p. 227, Dec. 2022.

[7] 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, Nov. 2017.

[8] L. Millet, H. Jeon, B. Kim, B. Bhoi, and S.-K. Kim, "Reservoir computing using photon-magnon coupling," *Appl. Phys. Lett.*, vol. 119, no. 18, Nov. 2021, Art. no. 182405.

[9] C. Du, F. Cai, M. A. Zidan, W. Ma, S. H. Lee, and W. D. Lu, "Reservoir computing using dynamic memristors for temporal information processing," *Nature Commun.*, vol. 8, no. 1, p. 2204, Dec. 2017.

[10] B. Shi, N. Calabretta, and R. Stabile, "InP photonic integrated multilayer neural networks: Architecture and performance analysis," *APL Photon.*, vol. 7, no. 1, Jan. 2022, Art. no. 010801.

[11] 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, Oct. 2018, Art. no. 152004.

[12] K. Takano et al., "Compact reservoir computing with a photonic integrated circuit," *Opt. Exp.*, vol. 26, no. 22, pp. 29424–29439, 2018.

[13] X. Porte, A. Skalli, N. Haghighi, S. Reitzenstein, J. A. Lott, and D. Brunner, "A complete," parallel and autonomous photonic neural network in a semiconductor multimode laser," *J. Phys. Photon.*, vol. 3, Apr. 2021, Art. no. 024017.

[14] J. Yang, H. Cho, H. Ryu, M. Ismail, C. Mahata, and S. Kim, "Tunable synaptic characteristics of a Ti/TiO_2/Si memory device for reservoir computing," *ACS Appl. Mater. Interfaces*, vol. 13, no. 28, pp. 33244–33252, Jul. 2021.

[15] P. Antonik, N. Marsal, D. Brunner, and D. Rontani, "Human action recognition with a large-scale brain-inspired photonic computer," *Nat. Mach. Intell.*, vol. 1, pp. 530–537, Nov. 2019.

[16] C. Gallicchio and A. Micheli, *Richness of Deep Echo State Network Dynamics* (Lecture Notes in Computer Science), vol. 11506. Cham, Switzerland: Springer, 2019.

[17] L. Grigoryeva, J. Henriques, L. Larger, and J.-P. Ortega, "Time-delay reservoir computers and high-speed information processing capacity," in *Proc. IEEE Intl Conf. Comput. Sci. Eng. (CSE) IEEE Intl Conf. Embedded Ubiquitous Comput. (EUC) 15th Intl Symp. Distrib. Comput. Appl. Bus. Eng. (DCABES)*, Aug. 2016, pp. 492–495.

[18] Z. Tong and G. Tanaka, "Reservoir computing with untrained convolutional neural networks for image recognition," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 1289–1294.

[19] A. Röhm, L. C. Jaurigue, and K. Lüdge, "Reservoir computing using laser networks," *IEEE J. Sel. Topics Quantum Electron.*, vol. 26, no. 1, Jan. 2020, Art. no. 7700108.

[20] A. Röhm, D. J. Gauthier, and I. Fischer, "Model-free inference of unseen attractors: Reconstructing phase space features from a single noisy trajectory using reservoir computing," *Chaos, Interdiscipl. J. Nonlinear Sci.*, vol. 31, no. 10, Oct. 2021, Art. no. 103127.

[21] M. Goldmann, F. Köster, K. Lüdge, and S. Yanchuk, "Deep time-delay reservoir computing: Dynamics and memory capacity," *Chaos, Interdiscipl. J. Nonlinear Sci.*, vol. 30, no. 9, Sep. 2020, Art. no. 093124.

[22] J. Pathak et al., "Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model," *Chaos, Interdiscipl. J. Nonlinear Sci.*, vol. 28, no. 4, Apr. 2018, Art. no. 041101.

[23] A. Wikner et al., "Combining machine learning with knowledge-based modeling for scalable forecasting and subgrid-scale closure of large, complex, spatiotemporal systems," *Chaos, Interdiscipl. J. Nonlinear Sci.*, vol. 30, no. 5, May 2020, Art. no. 053111.

[24] 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.

[25] B. A. Marquez, J. Suarez-Vargas, and B. J. Shastri, "Takens-inspired neuromorphic processor: A downsizing tool for random recurrent neural networks via feature extraction," *Phys. Rev. Res.*, vol. 1, no. 3, Oct. 2019, Art. no. 033030.

[26] S. Deligiannidis, C. Mesaritakis, and A. Bogris, "Performance and complexity analysis of bi-directional recurrent neural network models versus Volterra nonlinear equalizers in digital coherent systems," *J. Lightw. Technol.*, vol. 39, no. 18, pp. 5791–5798, Jun. 25, 2021.

[27] T. Hülser, F. Köster, L. Jaurigue, and K. Lüdge, "Role of delay-times in delay-based photonic reservoir computing," *Opt. Mater. Exp.*, vol. 12, no. 3, pp. 1214–1231, 2022.

[28] L. Appeltant et al., "Information processing using a single dynamical node as complex system," *Nature Commun.*, vol. 2, no. 13, p. 468, Sep. 2011.

[29] J. D. Hart, D. C. Schmadel, T. E. Murphy, and R. Roy, "Experiments with arbitrary networks in time-multiplexed delay systems," *Chaos, Interdiscipl. J. Nonlinear Sci.*, vol. 27, no. 12, Dec. 2017, Art. no. 121103.

[30] 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, 2019, Art. no. 20180123.

[31] F. Stelzer and S. Yanchuk, "Emulating complex networks with a single delay differential equation," *Eur. Phys. J. Special Topics*, vol. 230, nos. 14–15, pp. 2865–2874, Jun. 2021.

[32] G. Dion, S. Mejaouri, and J. Sylvestre, "Reservoir computing with a single delay-coupled non-linear mechanical oscillator," *J. Appl. Phys.*, vol. 124, no. 15, Oct. 2018, Art. no. 152132.

[33] Y. Chen et al., "Reservoir computing system with double optoelectronic feedback loops," *Opt. Exp.*, vol. 27, no. 20, pp. 27431–27440, Sep. 2019.

[34] J. Schumacher, H. Toutounji, and G. Pipa, "An analytical approach to single node delay-coupled reservoir computing," in *Proc. 23rd Int. Conf. Artif. Neural Netw.*, Jan. 2013, pp. 26–33.

[35] C. Sugano, K. Kanno, and A. Uchida, "Reservoir computing using multiple lasers with feedback on a photonic integrated circuit," *IEEE J. Sel. Topics Quantum Electron.*, vol. 26, no. 1, pp. 1–9, Jan. 2020.

[36] J. Bueno, D. Brunner, M. C. Soriano, and L. Fischer, "Conditions for reservoir computing performance using semiconductor lasers with delayed optical feedback," *Opt. Exp.*, vol. 25, no. 3, pp. 2401–2412, Feb. 2017.

[37] H. Toutounji, J. Schumacher, and G. Pipa, "Homeostatic plasticity for single node delay-coupled reservoir computing," *Neural Comput.*, vol. 27, no. 6, pp. 1159–1185, Jun. 2015.

[38] 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, Dec. 2019.

[39] Y. Kuriki, J. Nakayama, K. Takano, and A. Uchida, "Impact of input mask signals on delay-based photonic reservoir computing with semiconductor lasers," *Opt. Exp.*, vol. 26, no. 5, pp. 5777–5788, Mar. 2018.

[40] A. Argyris et al., "Comparison of photonic reservoir computing systems for fiber transmission equalization," *IEEE J. Sel. Topics Quantum Electron.*, vol. 26, no. 1, pp. 1–9, Jan. 2020, Art. no. 5100309.

[41] 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, no. 1, Feb. 2017, Art. no. 011015.

[42] D. Brunner, M. C. Soriano, and G. Van Der Sande, *Photonic Reservoir Computing, Optical Recurrent Neural Networks*. Berlin, Germany: De Gruyter, 2019.

[43] G. Van Der Sande, D. Brunner, and M. C. Soriano, "Advances in photonic reservoir computing," *Nanophotonics*, vol. 6, p. 561, May 2017.

[44] G. Tanaka et al., "Recent advances in physical reservoir computing: A review," *Neural Netw.*, vol. 115, pp. 100–123, Jul. 2019.

[45] K. Nakajima and I. Fischer, *Reservoir Computing: Theory, Physical Implementations, and Applications Natural Computing Series*. Singapore: Springer, 2021.

[46] J. Dambre, D. Verstraeten, B. Schrauwen, and S. Massar, "Information processing capacity of dynamical systems," *Sci. Rep.*, vol. 2, p. 514, Apr. 2012.

[47] L. Grigoryeva, J. Henriques, L. Larger, and J.-P. Ortega, "Optimal nonlinear information processing capacity in delay-based reservoir computers," *Sci. Rep.*, vol. 5, no. 1, p. 12858, Oct. 2015.

[48] F. Stelzer, A. Röhm, K. Lüdge, and S. Yanchuk, "Performance boost of time-delay reservoir computing by non-resonant clock cycle," *Neural Netw.*, vol. 124, pp. 158–169, Apr. 2020.

[49] F. Köster, S. Yanchuk, and K. Lüdge, "Insight into delay based reservoir computing via eigenvalue analysis," *J. Phys., Photon.*, vol. 3, no. 2, Apr. 2021, Art. no. 024011.

[50] F. Köster, D. Ehlert, and K. Lüdge, "Limitations of the recall capabilities in delay based reservoir computing systems," *Cogn. Comput.* Cham, Switzerland: Springer, 2020, pp. 1–8.

[51] T. Hülser, F. Köster, K. Lüdge, and L. Jaurigue, "Deriving task specific performance from the information processing capacity of a reservoir computer," *Nanophotonics*, pp. 1–11, Oct. 2022.

[52] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *Int. J. Forecasting*, vol. 22, no. 4, pp. 679–688, Oct./Dec. 2006.

[53] M. Inubushi and K. Yoshimura, "Reservoir computing beyond memory-nonlinearity trade-off," *Sci. Rep.*, vol. 7, no. 1, p. 10199, Dec. 2017.

[54] L. D. Landau, "On the problem of turbulence," *C. R. Acad. Sci. UESS*, vol. 44, p. 311, 1944.

[55] J. T. Stuart, "On the non-linear mechanics of hydrodynamic stability," *J. Fluid Mech.*, vol. 4, no. 1, pp. 1–21, May 1958.

[56] M. C. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science*, vol. 197, p. 287, Jul. 1977.

[57] K. Ikeda, "Multiple-valued stationary state and its instability of the transmitted light by a ring cavity system," *Opt. Commun.*, vol. 30, no. 2, pp. 257–261, 1979.

[58] K. Ikeda and K. Matsumoto, "High-dimensional chaotic behavior in systems with time-delayed feedback," *Phys. D*, vol. 29, pp. 223–235, Dec. 1987.
[59] M. H. DeGroot, *Optimal Statistical Decisions*. New York, NY, USA: McGraw-Hill, 1970.
[60] C. Sanderson and R. Curtin, "Armadillo: A template-based C++ library for linear algebra," *J. Open Source Softw.*, vol. 1, p. 26, Jun. 2016.
[61] R. Legenstein and W. Maass, "Edge of chaos and prediction of computational performance for neural circuit models," *Neural Netw.*, vol. 20, no. 3, pp. 323–334, 2007.
[62] L. Büsing, B. Schrauwen, and R. Legenstein, "Connectivity," dynamics, and memory in reservoir computing with binary and analog neurons," *Neural Comput.*, vol. 22, no. 5, pp. 1272–1311, 2010.
[63] F. Stelzer, A. Röhm, R. Vicente, I. Fischer, and S. Yanchuk, "Deep neural networks using a single neuron: Folded-in-time architecture using feedback-modulated delay loops," *Nature Commun.*, vol. 12, no. 1, p. 5164, Aug. 2021.

Felix Köster received the M.Sc. degree in physics from the Technische Universität Berlin, Berlin, Germany, in 2018.

He has worked on the modeling of optical neurons, laser networks, and reservoir computing.

Serhiy Yanchuk received the Diploma degree in physics from the Moscow Engineering Physics Institute (Technical University), Moscow, Russia, and the Ph.D. and Dr.Sc. degrees in mathematics from the National Academy of Sciences of Ukraine, Kyiv, Ukraine.

He was a Senior Researcher with the Institute of Mathematics, National Academy of Sciences of Ukraine, a Post-Doctoral Fellow with the Weierstrass Institute, Berlin, Germany, a Junior Research Group Leader with the Humboldt University of Berlin, Berlin, and a Visiting Professor with the Berlin Institute of Technology, Berlin. He is currently a Project Leader with the Potsdam Institute for Climate Impact Research and Privatdozent, Humboldt University of Berlin. His current research interests include nonlinear dynamics of interacting and forced systems, dynamical and adaptive networks, spatiotemporal behavior of distributed systems, systems with time delays, and reservoir computing.

Kathy Lüdge was born in Berlin, Germany, in 1976. She received the Diploma and Dr.rer.nat. degrees in physics and the Habilitation (Venia Legendi) degree from the Berlin Institute of Technology (TU Berlin), Berlin, in 2000, 2003, and 2011, respectively.

Since 2021, she has been a Professor with the Ilmenau University of Technology, Ilmenau, Germany, where she is the Head of the Department of Theoretical Physics II. During her scientific career, she was a Visiting Scientist with the University of Minnesota, Minneapolis, MN, USA, in 2002, a Visiting Professor with Freie Universität Berlin, Berlin, in 2015, and a Humboldt Feodor-Lynen Fellow with The University of Auckland, Auckland, New Zealand, in 2016. From 2016 to 2021, she was a University Professor with TU Berlin, where she was the Head of the Department of Nonlinear Laser Dynamics. She is known to the scientific community through more than 100 articles in renowned journals. She investigates the emission properties of semiconductor devices using numerical methods and develops new approaches for their optimization and innovative applications, with a focus on simulations of spatiotemporal dynamics and machine learning methods.